COMPUTING ARRANGEMENTS OF IMPLICIT CURVES

PAULO CEZAR CARVALHO
IMPA–Instituto de Matemática Pura e Aplicada

LUIZ HENRIQUE DE FIGUEIREDO
LNCC–Laboratório Nacional de Computação Científica

PAULO ROMA CAVALCANTI
UFRJ–Universidade Federal do Rio de Janeiro

We examine the problem of determining, when possible, the exact topological adjacency structure of the planar subdivision induced on a rectangle by a set of curves given in implicit form. This means finding the number of points of intersection of the curves and the way curve segments are linked to determine regions. We use a recursive method based on estimates given by interval arithmetic, together with conditions under which topology can be exactly determined.

1 Introduction

An *arrangement* of a set of curves in the plane is the planar subdivision induced by these curves. The problem of *computing* such an arrangement is finding and representing the complete topology of the induced planar subdivision, i.e., finding all vertices, edges and faces, and the adjacencies between them.

Representing the topology is easy, with a suitable data structure [13]. Finding the topology of an arrangement requires two geometrical steps: finding the vertices, which are the points where the curves intersect each other, and finding geometrical representations for the edges.

The simplest arrangements are *line arrangements*, in which the curves are infinite straight lines. Much is known about line arrangements, including many efficient algorithms for computing them [1, 2, 7]. For example, a topological sweep [1] can compute an arrangement of n lines in general position in $O(n^2)$ time using $O(n)$ working space. It is thus a linear algorithm, since such an arrangement has size $O(n^2)$: there are $O(n^2)$ vertices, and Euler's theorem applies.

To handle arrangements defined by more general curves, the simplest approach is to use polygonal approximations to represent the curves. We are thus naturally led to consider *polygonal arrangements*, in which the curves are polygonal lines. In a previous paper [3], we studied such arrangements, and described EDP, a system for computing them. Another implementation is described in [8].

We may use polygonal approximations for representing implicit curves in arrangements, but these approximations are hard to compute. To begin with, it is much harder to sample points on implicit curves [4]. Also, finding the topology of the curve between two points on the curve is not straightforward. In fact, the points may even belong to different connected components of the curve.

Our goal in this paper is to find, whenever possible, the exact topology determined by a set of implicit curves, without having to resort to polygonal approximations. It is important to distinguish exact topology from exact geometry. In general, we cannot expect to obtain exact geometric results, such as vertex coordinates, since we must rely on numerical methods for curve sampling and intersection. Nevertheless, in many cases we can determine the exact topology of the arrangement. In this paper, we use *range analysis* [9] to find regions where the arrangement is locally simple. Even when it is not possible to find exactly the complete topology, we try to distinguish mathematically provable properties from those that depend on the adopted tolerance level for numerical approximations.

2 Range analysis

Range analysis is the study of the behavioru of real functions based on estimates for their set of values. More precisely, given a function $f\colon \Omega \to \mathbf{R}$, range analysis methods provide an *inclusion function* for f, that is, a function F defined on the subsets of Ω such that $F(R)$ is an interval containing $f(R)$, for all $R \subseteq \Omega$. Thus, range analysis provides robust estimates for the maximum and minimum values of f in R. However, these estimates are not required to be tight, and $F(R)$ may be larger than $f(R)$.

Interval arithmetic (IA) is the natural technique for range analysis; it was invented by Moore [9] with the explicit goal of improving the reliability of numerical computation. IA provides robust estimates for the results of numerical computations by representing numbers as intervals. Moore [9] showed that every computable function f has an natural interval extension F, which is an inclusion function for f. Moreover, any algorithm for computing f can be automatically interpreted as an algorithm for computing F.

3 Solutions based on polygonization

An *implicit curve* is the set of zeros of a function $f\colon \Omega \subseteq \mathbf{R}^2 \to \mathbf{R}$. The simplest method for computing polygonal approximations of an implicit curve $C = f^{-1}(0)$ is *full enumeration*: The domain Ω is subdivided uniformly into a mesh of small rectangles, and C is approximated by a linear segment within each rectangle for which the sign of f at its vertices is not the same. The segments thus obtained can then be fed into a system for computing polygonal arrangements [3, 8].

Full enumeration is simple but not efficient, because very fine meshes are needed to avoid missing parts of the curve. An efficient and robust approximation method combines *adaptive enumeration* with range analysis, and recursively subdivides Ω, using small rectangles only near C. Range analysis allows us to discard large portions of Ω that are *guaranteed* not to contain any portion of the curve [5, 6, 10, 12].

More precisely, given a function f and a subrectangle R of Ω, we estimate $f(R)$ using an inclusion function F. If $0 \notin F(R)$, then one can be sure that the curve $C = f^{-1}(0)$ does not intersect R and R can be discarded. Otherwise, the curve may intersect R (recall that range estimates are not required to be tight). In this case, unless R is too small, it is subdivided and the same test is recursively applied to each sub-rectangle. When the diameter of R is smaller than a specified tolerance, a polygonal approximation for C in R is computed by linear interpolation.

The topology generated using this method cannot be guaranteed to be correct, because the method will neither detect closed loops completely contained in a rectangle that is too small nor distinguish almost touching different branches from singularities. The problem here is not that the method has trouble finding the correct topology near singularities or almost singularities; any numerical method will have the same kind of difficulty. The real problem is that the method is unable to recognize that, in many cases, the topology can be *guaranteed* to be correct [10].

4 Proving topology correctness

We now give a set of sufficient conditions under which the topology of an arrangement of n implicit curves $C_i = f_i^{-1}(0)$ in a subrectangle R of Ω can be precisely determined. Moreover, in the spirit of CSG modeling, we want to be able to locate any point $p \in \Omega$ with respect to this subdivision based solely on the *sign signature* of p, that is, on the n-tuple $(\mathrm{sign} f_1(p), \ldots, \mathrm{sign} f_n(p))$.

Sign signatures alone are not enough to classify the regions in an arrangement. For example, take the hyperbola C given implicitly by $f(x, y) = xy - 1$ on $\Omega = [-1, 1] \times [-1, 1]$. Then, C divides Ω into three regions, but two of them correspond to the same sign.

The following conditions ensure that the arrangement restricted to a subrectangle R is completely determined by sign signatures. If one of them holds, we stop the recursive exploration of Ω at R.

1. No curves intersect R.
2. Exactly one curve intersects R and that intersection is a simple arc.
3. Exactly two curves intersect R, each them along a simple arc, and they cross each other exactly once in R.

In these cases, the curves subdivide R into 1, 2 or 4 connected regions, respectively. Moreover, sign signatures are sufficient to locate points in R.

We check these conditions using range analysis. Case 1 is simple: if $0 \notin F_i(R)$, then C_i does not intersect R. For Case 2, the following conditions ensure that $C = C_i$ intersects R along a simple arc:

2a. C crosses the boundary of R exactly twice.
2b. the gradient of f does not vanish over R.

Sufficient conditions for 2a and 2b can be obtained by computing range estimates for $f = f_i$ and its partial derivatives over R. For 2a, we compute estimates for the range of f over each side of R. We must have 0 in exactly two of these estimates (this implies that f may cross the boundary only at those sides). If, furthermore, the directional derivatives of f along those sides do not change signs, then we can guarantee that 2a holds. For 2b, we may, for example, test the signs of $\partial f/\partial x$ and $\partial f/\partial y$ over R. If at least one of them does not vanish in R, then the gradient of f does not vanish in R.

Now, let f and g satisfy the conditions in Case 3. Then, the following conditions ensure that the corresponding curves cross exactly once in R:

3a. the curves cross the boundary of R in alternating points
3b. the sets of directions of the gradients of f and g over R are disjoint.

Condition 3a is easily verified when the intersections of the curves with the boundary of R occur on different sides. If the curves intersect the same side, then one can use bisection over that side to properly order those points. For 3b, range estimates for the gradients can be turned into arc estimates for their directions; we only need to check whether these arcs intersect.

5 Conclusion

This work is still in progress. We are currently working on efficient data structures and algorithms for computing the topology of the arrangement from quadtree decompositions and also for point location. We are also interested in more efficient tests, using affine arithmetic [11], and in extensions to 3D.

References

[1] P. K. Agarwal. *Intersection and decomposition algorithms for planar arrangements*. Cambridge University Press, New York, NY, 1991.

[2] Te. Asano, L. J. Guibas, and T. Tokuyama. Walking on an arrangement topologically. *Internat. J. Comput. Geom. Appl.*, 4:123–151, 1994.

[3] P. C. P. Carvalho, M. Gattass, and L. F. Martha. A software tool which allows interactive creation of planar subdivisions, and applications to educational programs. In E. Oñate, editor, *CATS'90 - Proceedings of International Conference on Computer Aided Training in Science and Technology*, pages 201–207, Barcelona, Spain, July 1990. CIMNE Pineridge Press.

[4] L. H. de Figueiredo and J. Gomes. Sampling implicit objects with physically-based particle systems. *Computers & Graphics*, 20(3):365–375, 1996.

[5] L. H. de Figueiredo and J. Stolfi. Adaptive enumeration of implicit surfaces with affine arithmetic. *Computer Graphics Forum*, 15(5):287–296, 1996.

[6] T. Duff. Interval arithmetic and recursive subdivision for implicit functions and constructive solid geometry. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):131–138, July 1992.

[7] H. Edelsbrunner and L. J. Guibas. Topologically sweeping an arrangement. *J. Comput. Syst. Sci.*, 38:165–194, 1989. Corrigendum in 42 (1991), 249–251.

[8] M. Goldwasser. An implementation for maintaining arrangements of polygons. In *Proceedings of 11th Annual ACM Symposium on Computational Geometry*, pages C32–C33, 1995.

[9] R. E. Moore. *Interval Analysis*. Prentice-Hall, 1966.

[10] J. M. Snyder. Interval analysis for computer graphics. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):121–130, July 1992.

[11] J. Stolfi and L. H. de Figueiredo. *Self-Validated Numerical Methods and Applications*. Monograph for 21st Brazilian Mathematics Colloquium, IMPA, 1997. Available at `ftp://ftp.tecgraf.puc-rio.br/pub/lhf/doc/cbm97.ps.gz`.

[12] K. G. Suffern and E. D. Fackerell. Interval methods in computer graphics. *Computers & Graphics*, 15:331–340, 1991.

[13] K. Weiler. Edge-based data structures for solid modeling in curved-surface environments. *IEEE Computer Graphics and Applications*, 5(1):21–40, 1985.