Computational morphology of curves

LUIZ HENRIQUE DE FIGUEIREDO
JONAS DE MIRANDA GOMES

IMPA–Instituto de Matemática Pura e Aplicada
Estrada Dona Castorina, 110
22460-030 Rio de Janeiro, RJ, Brazil
{lhf,jonas}@visgraf.impa.br

Abstract. We prove that Euclidean minimal spanning trees correctly reconstruct differentiable arcs from sufficiently dense samples. The proof is based on a combinatorial characterization of minimal spanning paths and on a description of the local geometry of arcs inside tubular neighbourhoods. We also present simple heuristics for reconstructing more general curves.
KEYWORDS: Computational morphology — Curve reconstruction — Minimal spanning trees

Introduction

Suppose a finite set of points on a plane curve is given. If the sample is large enough and is well distributed, then it is an easy task for a human being to perceive the shape of the curve (Fig. 1). Human perception of curve shape includes not only topological aspects, such as the identification of connected components and the differentiation between open and closed pieces, but also geometrical aspects, such as qualitative measures of curvature and winding.

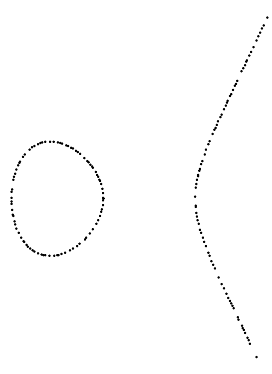

Figure 1: easy perception of the shape of a curve.

A computer, in contrast, has no such natural perception: the sample has no *a priori* structure that can be exploited to provide a computational description of the curve. To the machine, the sample is merely a list of coordinates. A fundamental problem, which we would expect to solve using such computational descriptions, is exactly how to sort the points in an order compatible with the natural trace of the curve, as perceived by a human. This order may be used to structure the sample into a polygonal approximation of the curve, thus reconstructing the curve from the sample.

Reproducing human perception of shapes from *dot patterns* (dense uniform samples of "fat" points on plane regions) is a classical problem in low-level computer vision, pattern recognition, and cluster analysis (Zahn 1971; Toussaint 1980). Much of the work in computational geometry was originally motivated by problems in these fields. Toussaint (1980) coined the term *computational morphology* for the use of computational geometry for extracting shape information.

Previous work on computational morphology was aimed mainly at reconstructing plane regions; these objects have co-dimension 0, in the sense that they have the same intrinsic dimension as the ambient space. In this paper, we are interested in plane curves, which are objects of co-dimension 1. We prove that arcs can be correctly reconstructed by computing Euclidean minimal spanning trees on sufficiently dense samples. Based on this result, we describe simple heuristics for reconstructing more general curves. For an approach to the important problem of reconstructing surfaces in three-dimensional space, see Hoppe et al. (1992).

As mentioned, we assume that the curves are to be reconstructed from a sample of points on it. How this sample is obtained is not our concern here; the only assumption is that the points lie on a curve. Nevertheless, this work arose in connection with implicit objects. For such objects, many sampling methods are known (Figueiredo 1992).

Minimal spanning trees

We structure samples of points on a curve by means of graph models. A *graph model* for a curve is a graph having the same topology as the curve. However, the graph models that we shall describe are not only topological structures, but also *geometric*, in the sense that the vertices are points in the plane lying on the curve, and the edges are straight-line segments connecting the points into a polygonal approximation of the curve. For such graph models, a homeomorphism from the model to the curve is given by a linear homotopy.

A geometric graph model can be easily simplified into a minimal graph model by contracting edges incident on vertices of degree 2. Minimal graph models have been used for obtaining robust plotting algorithms for plane algebraic curves (Arnon 1983).

The various graphs used to describe shape differ, of course, in their choice of edges. This choice depends on the notion of proximity that one intends to capture in the graph; such shape descriptors are accordingly called *proximity graphs*. (Boissonnat (1984) says that "a structure is needed to make proximity relationships explicit". Kirkpatrick and Radke (1985) say that describing shape needs the identification of "essential neighbours".)

Proximity varies with the type of description sought: describing curves is different from describing plane regions of co-dimension 0. Several criteria have been proposed for defining the proximity of two points; for each criterion, a corresponding graph is defined. Well known proximity graphs include: nearest neighbour graphs, minimal spanning trees, relative neighbourhood graphs, Gabriel graphs, Delaunay triangulations, sphere of influence graphs, α-shapes, β-skeletons and γ-graphs (Kirkpatrick–Radke 1985; Veltkamp 1992).

Despite the success of other graphs in describing plane dot patterns of co-dimension 0, our basic graph model is the Euclidean minimal spanning tree, because it proved to be well suited to describing one-dimensional objects, while being well understood and easily computed. Indeed, we shall see that minimal spanning trees provide a way of constructing geometric graph models of curves.

A *spanning graph* for a graph is a subgraph passing through all vertices of the graph. A *minimal spanning tree* for a weighted graph is a spanning tree for which the sum of edge weights is minimal. The geometric version of a minimal spanning tree is called a *Euclidean minimal spanning tree*: the graph is a complete geometric graph and the weight of an edge is its length. The terminology for geometric graphs is adopted for non-geometric weighted graphs, even though the weights are not distances; thus, for instance, edges of least weight are called"shortest" edges.

The fundamental property of minimal spanning trees, on which most algorithms for their computation are based, is due to Prim (1957); we shall call it the *bridge theorem*:

THEOREM 1. *Let $A \cup B$ be a partition of the set of vertices of a connected weighted graph G. Then any shortest edge in G connecting a vertex of A and a vertex of B is an edge of a minimal spanning tree for G.*

PROOF: Let T be a minimal spanning tree for G, and let e be a shortest edge in G connecting A and B. If e is not an edge of T, let f be an edge of T connecting A and B. Then the graph $T' = T - f + e$, obtained by replacing f by e in T, is still a spanning tree for G. Since e is not longer than f, the total edge length of T' is not greater than that of T. Therefore, the tree T' is a minimal spanning tree containing the bridge e between A and B. ■

Reconstructing arcs

A tree cannot model curves with multiple or closed components; a tree can only model the simplest curves, i.e., those diffeomorphic to the closed unit interval (we avoid the unpleasantness of wild embeddings by requiring diffeomorphisms instead of homeomorphisms). Such curves are called *arcs*, and tree models for arcs must be *paths*: every vertex has degree at most 2. Accordingly, if we want to use minimal spanning trees to reconstruct arcs, we must investigate when a path is a minimal spanning tree.

A *chord* in a path is an edge joining non-consecutive vertices. A *short chord* is a chord that is shorter than some edge of the subpath connecting its extremes; such an edge is called a *witness* for the short chord (Fig. 2).

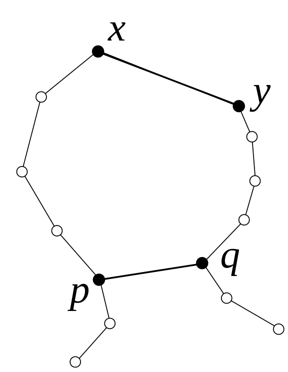

Figure 2: pq is a short chord with xy as a witness.

The absence of short chords characterize the paths that are minimal spanning trees, as the next theorem shows.

THEOREM 2. *A spanning path is a minimal spanning tree if and only if it has no short chords.*

PROOF: If a path has a short chord, then it cannot be a minimal spanning tree because a shorter spanning tree is obtained by replacing any witness by the chord.

Conversely, we shall show that if a path is not a minimal spanning tree, then it has short chord.

The proof is by induction on the number of vertices in the graph spanned by the path. For graphs with 1 or 2 vertices, there is nothing to prove, because every path is minimal.

Otherwise, let P be a non-minimal spanning path for a graph G. Then, there is an edge e of P and an edge f of G such that the graph $T = P - e + f$, obtained by replacing e by f in P, is a tree shorter than P. Note that this implies that f is shorter than e.

Let u be an extreme of P, and let $P' = P - u$. We shall prove that either P has a short chord or P' has a short chord. Of course, any short chord in P' is also a short chord in P.

If u is an extreme of f, then f is a chord of P. Moreover, since T is a tree, the edge e must be in the subpath of P joining u and w, where w is the other extreme of f. Since f is shorter than e, this means that the edge f is a short chord in P, witnessed by e.

If u is not an extreme of f, then u is not an extreme of e: otherwise, $T = P - e + f = P' + f$ would not be a tree, because f is a chord of P'. Hence, the edge e is an edge of P', and $P - e + f = (P' - e + f) + uv$, where v is the (only) neighbour of u in P. Now, T is a tree shorter than P if and only if $P' - e + f$ is a tree shorter than P'. Thus, P' is not a minimal tree, and hence, by induction, P' has a short chord. As already remarked, this short chord is also a short chord in P. ■

This theorem implies that if a sample on an arc is dense enough that the induced polygonal path through the sample points has no short chords, then we can recover the polygonal path by computing a minimal spanning tree for the sample. It is striking that a global algorithm, without any knowledge of the curve being sampled, is able to recover the local information needed to visit the sample points in an order compatible with the trace of the curve, thus constructing a correct polygonal approximation for a curve it does not know. This is the kind of result that we would like to classify as computational morphology for non-parametric scattered data interpolation. As far as we know—and despite the work of Hoppe et al (1992)—there is no analogous result for the reconstruction of surfaces.

A *tubular neighbourhood* for a curve C is a set T containing C such that every point of T belongs to exactly one line segment totally contained in T and normal to C (Fig. 3). Tubular neighbourhoods always exist for regular curves, such as arcs. (Hirsh 1976).

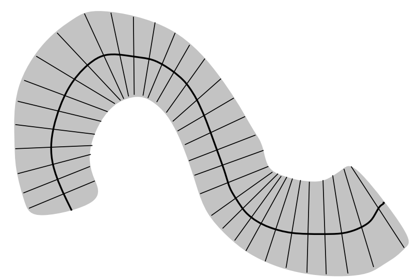

Figure 3: tubular neighbourhood.

Tubular neighbourhoods are closely related to the notion of sample density that we need for reconstruction. We say that a sample of a curve C is *dense* if there is a real number $\varepsilon > 0$ such that no two consecutive sample points are more than ε apart, and the closed disks of radius ε, centered at the sample points, form a tubular neighbourhood of C. A disk centered on the curve and contained in a tubular neighbourhood of the curve will be called a *tubular disk* (Fig. 4).

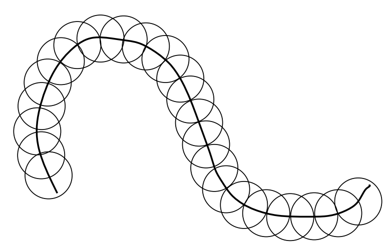

Figure 4: tubular disks on a dense sample.

Note that dense samples always exist for arcs, because arcs are compact: for a compact curve C, it is always possible to choose $\varepsilon > 0$ such that the closed disks of radius ε, centered at points on C, form a tubular neighbourhood of C.

An arc does not wander too much inside a tubular disk, as the following theorem shows.

THEOREM 3. *Let p and q be points on an arc C, such that q is inside a tubular disk D centered at p. Then the subarc of C between p and q is completely inside the disk with diameter pq.*

PROOF: Let A be the subarc of C between p and q. First, we show that A is completely inside the tubular disk D. In fact, we show that $C \cap D$ is connected. Indeed, let K be a connected component of $C \cap D$ (Fig. 5). If K is not the component of p, then there is a point $k \in K$ closest to p. The line pk is normal to C at k, but it meets the normal to C at p; this cannot happen inside the tubular disk D.

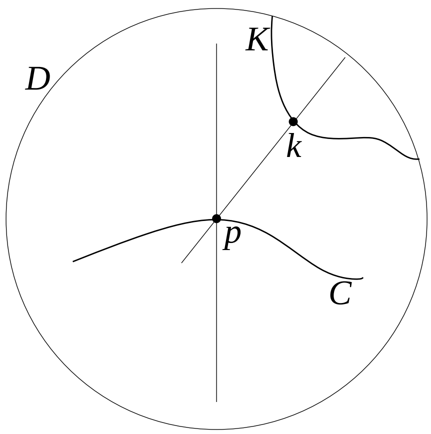

Figure 5: $C \cap D$ is connected.

To show that the arc A is completely inside the disk D' with diameter pq, let m be the midpoint of pq (Fig. 6). If A leaves D', then there is a point v on A but outside D' that is farthest from m. The circle with center m and radius $R = d(m, v)$ is tangent to A at v, and completely contains A. Hence, the curvature of A at v is at least $1/R$. Thus, the radius of curvature of A at v is at most equal to R. Therefore, the center of curvature at v lies on the segment $rm \subset D$. However, centers of curvature cannot lie inside a tubular disk, because they are the limit of intersecting normals, which is a contradiction. ■

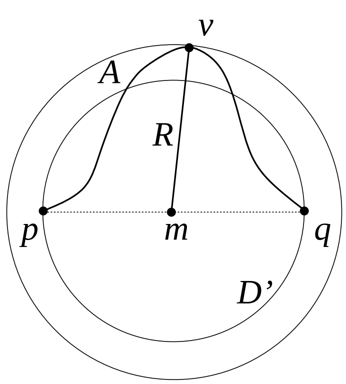

Figure 6: arc remains inside smaller disk.

We are now ready to prove our main theorem.

THEOREM 4. *Let S be a dense sample on an arc C containing the extremes of C. Then the Euclidean minimal spanning tree of S coincides with the polygonal path induced by C on S, and is therefore a geometric graph model for C.*

PROOF: By Theorem 2, we must show that the polygonal path P induced by C on S has no short cords. Suppose that pq is a chord shorter than an edge xy of the subpath pq. Let $\varepsilon > 0$ be the radius of the tubular disks centered on S and covering C. Then $d(x, y) < \varepsilon$ because the sample is dense. Hence, if $d(p, q) < d(x, y)$, then q is inside the tubular disk D centered at p. However, Theorem 3 says that the arc pq is completely inside a disk with diameter pq; no chord of the arc can be longer than pq. This contradicts $d(p, q) < d(x, y)$. ■

Both the statement and the proof of Theorem 4 strongly rely on the differentiability of the curve. In fact, the theorem is false for curves that are continuous, but not differentiable, such as polygons. For instance, take a polygonal line abc such that the sides ab and bc have the same length and make an acute angle. Take a sample on ab containing b and replicate it on bc. Then, no matter how dense the sample is, the point b will always be a leaf of any minimal spanning tree (Fig. 7).

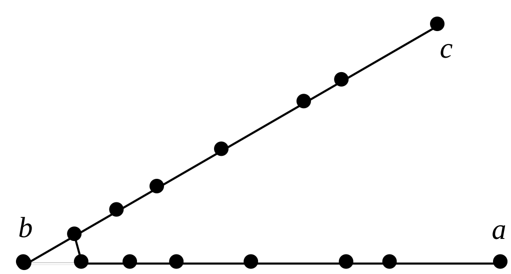

Figure 7: Theorem 4 needs differentiability.

Reconstructing loops

After arcs, the next simplest kind of curves are *loops*: they are diffeomorphic to the unit circle. A correct structure for a dense sample on a loop is obtained by computing a minimal spanning tree and then adding an edge joining the two vertices of degree 1 (called *leaves*). As we have already seen, a minimal spanning tree for the sample models an arc of the loop, not the complete loop (Fig. 8). This arc is defined by removing the longest segment of the loop determined by the sample.

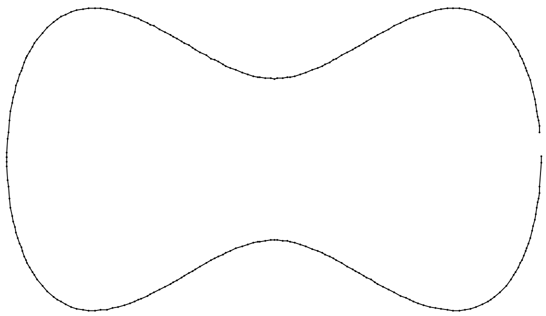

Figure 8: a minimal spanning tree on a loop.

Thus, assuming that the sample is on a loop, a correct geometric graph model is given by the addition of a single, well defined edge to a minimal spanning tree on the sample.

Reconstructing regular curves

The connected components of regular curves are either arcs or loops. Accordingly, to structure dense samples on regular curves, it is sufficient to identify their connected components. Each component can then be reconstructed separately, as we have shown.

The identification of connected components is a basic problem in the description of dot patterns. Cluster analysis deals with this problem and motivated our choice of minimal spanning trees as shape descriptors for curves (Zahn 1971; Duda–Hart 1973). A sample can be separated into clusters by removing atypically long edges from minimal spanning trees. Zahn describes such edges as *inconsistent*, and suggests two natural heuristic criteria for deciding when an edge is too long. One criterion is global: remove the edges in order of decreasing length. The other criterion is local: remove edges that are much longer than the average length of neighbour edges, e.g., at least twice as long. The problem with both approaches is very familiar: the presence of parameters that have to be tuned. With the global criterion, one must decide when to stop removing edges, unless the number of clusters is known beforehand. We have seen that this is not a realistic assumption. With the local criterion, one must define what is meant by "much longer". Nevertheless, inconsistent edge removal is adequate for identifying the connected components of curves.

Connected components are identified by finding and removing inconsistent edges from minimal spanning trees. The assumption that the sample lies on a regular curve allows the selection of inconsistent edges by performing simple topological tests instead of using the heuristic criteria suggested by Zahn. More precisely, since a graph model for a regular curve cannot have vertices of degree greater than 2 (called *branching points*), the edges joining two such vertices in a minimal spanning tree are exactly the edges to be removed: they bridge different connected components (Fig. 9). There is no need for an inconsistency analysis *à la* Zahn.

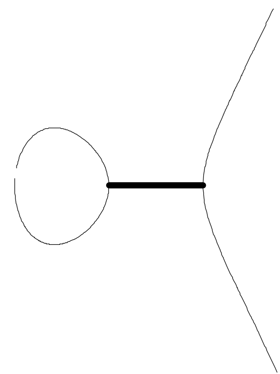

Figure 9: bridge in minimal spanning tree.

Once the connected components have been identified, we need to identify the closed components. After removing the bridges from a minimal spanning tree, we are left with a set of paths. To identify which paths should be closed, we must decide to which paths we should add an edge joining their leaves. This edge is called the *closing edge*. Unlike the previous cases, however, this step must be done heuristically.

We suggest the following heuristic criteria for closing a path:

- the angles defined by the closing edge at the leaves are both close to π;
- the closing edge is not much greater than others in the same component, being consistent in the sense of Zahn;
- the closing edge is an edge of the relative neighbourhood graph.

For the last criterion—which was suggested by experiments with alternative graph models—we can take either the whole sample or the sample corresponding to the connected component being considered. The first choice is more consistent with the philosophy of local structuring through global structuring.

Reconstructing implicit curves

An implicit curve is the set of zeros of a differentiable real function $h: \mathbf{R}^2 \to \mathbf{R}$. General implicit curves can be very complex. They can be singular, and they can have multiple connected components, some of which may be closed.

We shall keep minimal spanning trees as our basic structuring tool. The additional knowledge provided by the simplifying assumptions made in the previous sections will now be provided by the values of the function h that defines the curve implicitly.

The structuring method is essentially the same as the one for regular curves: identify connected components by identifying and removing bridges, and handle each component separately. However, we cannot immediately classify as bridges all edges joining branching points: such vertices may correspond to singular points of the curve. Nevertheless, all edges incident on branching points are considered potential bridges. A confirmation can be obtained by sampling the function h on such edges: by definition, the value of h on true bridges is non-zero.

After removing the bridges, we close the remaining paths in exactly the same way as described for regular curves, except that we must consider all pairs of leaves. Singular components will several leaves, some of which should be joined, some of which should not. The function h also provides a new criterion for the selection of closing edges: they should be approximately orthogonal to ∇h. This same criterion can be used for the identification of bridges: in this case, bridges are not only far from orthogonal to ∇h, but are actually approximately parallel to ∇h.

Reconstruction from noisy data

Our main assumption in this work has been that the sample points lie *on* a curve. However, practical applications frequently deal with noisy data. Nevertheless, it is still possible to use minimum spanning trees for curve reconstruction from noisy data, provided the sample is "representative" of the curve (a density notion based on tubular disks is probably still adequate here).

A minimal spanning tree on a noisy sample will have edges dangling from the longest path in the tree; this path is called the *diameter path*, and reconstructs the main trace of the curve. Thus, curves can be reconstructed in the presence of noise by computing a diameter path of a minimal spanning tree (Fig. 10). The distance of vertices from the diameter path can be taken as a measure of noise present in the data (Duda–Hart 1973).

The diameter path of a tree T can be easily computed in linear time. Start from an arbitrary vertex u of T; using depth-first search, find a vertex v farthest from u (i.e., v has maximum depth with respect to u); use depth-first search again to find a vertex w farthest from v. Then the path vw is a diameter path.

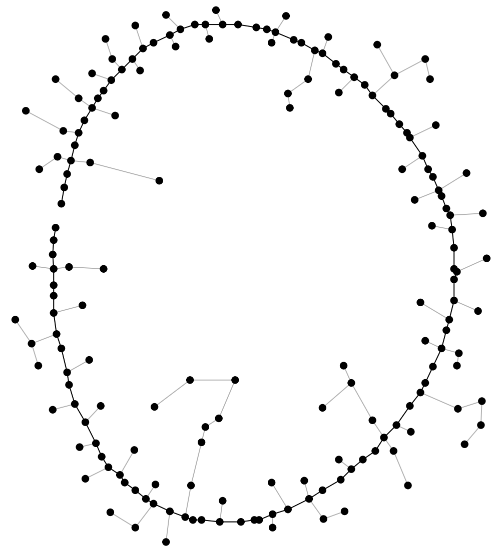

Figure 10: reconstruction with diameter path.

Conclusion

We have proved that it is possible to construct a correct polygonal approximation for an arc, given a dense sample of points on it (and a suitable definition of density). This was done by computing a Euclidean minimum spanning tree on the sample. We have also described simple heuristics for reconstructing more general curves and for dealing with noisy data.

Although other graphs can be used for curve reconstruction (e.g., relative neighbourhood graphs and γ-neighbourhood graphs), the Euclidean minimal spanning tree is well understood and easily computed (there are optimal $\Theta(n \log n)$ algorithms and also fast quadratic algorithms).

The definition we gave for sample density is somewhat strong because it forces dense samples to be fairly uniform. In practice, however, minimum spanning trees are able to reconstruct arcs from samples that are not so uniform. Of course, the absence of short chords is still an obstruction to reconstruction. A question poses itself: what is the "correct" definition of density for reconstruction using minimum spanning trees? Answers to this question will probably shed some light on the nature of good graph models for curve reconstruction.

The reconstruction method we have described here has been combined with a physically-based sampling method and proposed as a modeling method for implicit curves (Figueiredo 1992).

Acknowledgments

We are grateful to Paulo Cezar Pinto Carvalho and Carlos Gustavo Tamm de Araújo Moreira for useful discussions on the theorems presented here. Part of the proof of Theorem 3 was supplied by Carlos Gustavo. The algorithm for computing the diameter path was provided by Jayme Luiz Szwarcfiter.

This research was done in the VisGraf computer graphics laboratory at IMPA. The VisGraf project is partially supported by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnólogico) and IBM Brasil.

References

Arnon DS (1983) Topologically reliable display of algebraic curves. *Computer Graphics* **17**:219–227 (Proceedings of SIGGRAPH '83).

Boissonnat J-D (1984) Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics* **3**:266–286.

Duda RO, Hart PE (1973) *Pattern Classification and Scene Analysis*. Wiley.

Figueiredo LH (1992) *Computational Morphology of Implicit Curves*. doctoral thesis, IMPA, Rio de Janeiro.

Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W (1992) Surface reconstruction from unorganized points. *Computer Graphics* **26**:71–78 (Proceedings of SIGGRAPH '92).

Hirsh MW (1976) *Differential Topology*, Graduate Texts in Mathematics **33**. Springer.

Kirkpatrick DG, Radke JD (1985) A framework for computational morphology. In: Toussaint G (ed) *Computational Geometry*. North-Holland, pp. 217–248.

Prim RC (1957) Shortest connection matrix network and some generalizations. *Bell Systems Technical Journal* **36**:1389–1401.

Toussaint G (1980) Pattern recognition and computational complexity. In: *Proceedings of the Fifth International Conference on Pattern Recognition*, December 1980, pp. 1324–1347.

Veltkamp RC (1992) The γ-neighborhood graph. *Computational Geometry: Theory and Applications* **1**:227–246.

Zahn CT (1971) Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers* **C-20**:68–86.

Luiz Henrique de Figueiredo graduated in Mathematics in 1982 from PUC-Rio, the Catholic University of Rio de Janeiro, where he also obtained a M.Sc. in Mathematics in 1984. He received his Ph.D. in Mathematics in 1992 from IMPA. His current research interests include computational geometry, geometric modeling and software tools for the development of interactive graphical applications. He is currently a Research Associate at TeCGraf, the Computer Graphics Technology Group of PUC-Rio. He is also a member of the VisGraf project at IMPA.

Jonas de Miranda Gomes received his Ph.D. in Mathematics in 1984 from the Institute of Pure and Applied Mathematics (IMPA) in Rio de Janeiro. He worked as the R&D manager of the computer graphics group at Globo TV network from 1984 to 1988. Since 1989, he is a Research Associate at IMPA, where he started the VisGraf project, a computer graphics research group. His current research interests include modeling, digital publishing and conceptual issues in graphics. He has published several papers in these areas, and was the organizer of a course on Modeling at SIGGRAPH '93.