Physically-based sampling of implicit objects

LUIZ HENRIQUE DE FIGUEIREDO
JONAS DE MIRANDA GOMES

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

Abstract. After reviewing three classical sampling methods for implicit objects, we describe a new sampling method that is not based on scanning. In this method, samples are "randomly" generated using physically-based particle systems.

Introduction

Polygonal approximations are frequently used to represent discrete versions of geometric objects inside computers. Such approximations are useful in practice not only for solving numerical problems, such as partial differential equations arising in engineering, but also for rendering, because many graphics systems have special hardware for handling polygons.

The computation of polygonal approximations of geometric objects can be conceptually divided into two phases: **sampling**, which is the computation of points on the object, and **structuring**, which is the creation of a data structure representing a polygonal approximation interpolating these points (see Figure 1).

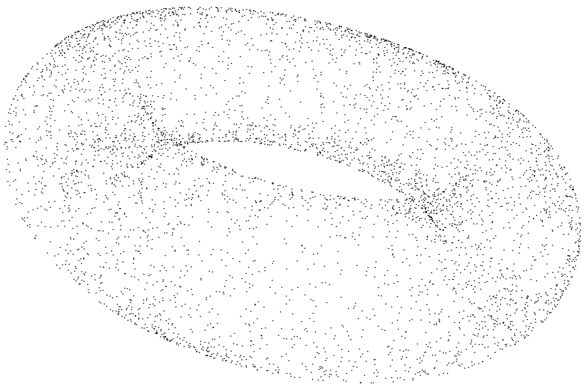

Figure 1: sampling.

The way these sub-problems are solved depends on how geometric objects are defined: the two most common ways of defining geometric objects—parametric and implicit equations—require very different methods for sampling and structuring.

Sampling parametric objects is easy because it reduces to sampling the parameter domain; this is usually done on a mesh so that structuring is immediate. Therefore, constructing polygonal approximations for parametric objects is easy not only because it is easy to generate points on them, but also because it is easy to structure mesh samples: they have *a priori* structure. On the other hand, constructing polygonal approximations for objects defined implicitly is hard because both sampling and structuring are hard: sampling requires the solution of many non-linear equations, and structuring is a difficult problem because there is no guiding mesh [Allgower–Schmidt (1985)].

Polygonal approximations for implicit objects are classically computed with methods that combine sampling and structuring (see below). In this article, we study sampling as a problem independent from structuring. This separation aims to identify the problems which are particular to each phase. When structuring is done concurrently with sampling, these problems tend to lose their original source and merge into a set of problems that is characteristic of the combined method used. The goal in separating structuring from sampling is not to claim that the computation of polygonal approximations of geometric objects should be done in two phases, although such a method has been proposed for curves [Figueiredo (1992)].

Sampling implicit objects

Let h be a differentiable real function on $\mathbf{R}^n$ defining an implicit object $V = h^{-1}(0)$. Sampling points on V means finding solutions of the equation $h(x) = 0$. In practice, sampling means finding enough solutions so that the topology of V can be reconstructed and the geometry can be approximated.

A sampling method can impose a structure that does not correspond to the geometry of the object being sampled [Boissonnat (1984)]. For instance, sampling an object by slices, as done in computer-aided tomography, impose a sweep structure on the object; accordingly, reconstructing a solid from a series of slices is a difficult problem [Giertsen–Halvorsen–Flood (1990)].

Classical methods compute polygonal approximations for implicit objects by performing structuring concurrently with sampling. As mentioned in the Introduction, we prefer to study sampling and structuring separately. We shall now review the sampling technique of three classical polygonization methods and then describe a new, physically-based approach to sampling. The problems of structuring are discussed elsewhere [Figueiredo (1992)].

Sampling by ray-casting

A naive way of finding solutions of the equation

$$h(x_1, \ldots, x_n) = 0$$

is to reduce it to single-variable equations by computing the intersection of V with a family $\mathcal{R}$ of straight lines, which we shall call **rays**.

The simplest rays correspond to fixing some of the variables. For instance, we could fix the first $n-1$ variables and solve $h(a,t) = 0$ for a sample of points $a = (a_1, \ldots, a_{n-1})$ in $\mathbf{R}^{n-1}$. This computes the intersection of V with the "vertical" rays $\{a\} \times \mathbf{R}$; accordingly, we call this method **vertical ray-casting** (see Figure 2).

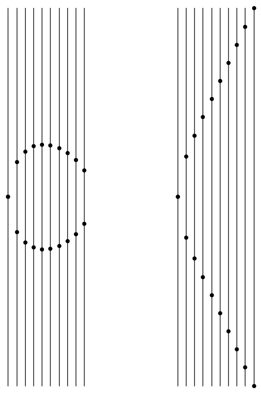

Figure 2: sampling by ray-casting.

Sampling by ray-casting needs the solution of many single-variable equations. Even if a good equation solver is available, there are several problems with this approach, showing how the sample obtained on V will depend on the sample of rays $\mathcal{R}$ (see Figure 2):

- because there is no *a priori* criterion for choosing rays intersecting V, a majority of the rays in $\mathcal{R}$ may not contribute sample points on V;
- even when a ray intersects V, the equation solver may not find all intersections, resulting in a partial sample of V;
- several rays in $\mathcal{R}$ may intersect V at the same point (this is is not a problem if the rays in $\mathcal{R}$ are parallel, as in vertical ray-casting).
- a common choice for vertical ray-casting is to sample $\mathbf{R}^{n-1}$ on a uniform mesh, as in parametric sampling. The resulting sample on V is biased against regions where the tangent space of V is vertical.

In general, it is too hard to relate the size and density a sample of rays $\mathcal{R}$ to the size and density of the corresponding sample of V. Nevertheless, ray-casting is useful for rendering implicit surfaces.

Sampling by continuation

The **gradient** of h,

$$\nabla h = (\frac{\partial h}{\partial x_1}, \ldots, \frac{\partial h}{\partial x_n}),$$

is the Jacobian matrix of h in co-dimension 1. The gradient is a vector field on the ambient space $\mathbf{R}^n$, which, at the regular points of h, points in the direction of local growth of h. Moreover, ∇h is orthogonal to the level sets of h.

In dimension 2, the associated **Hamiltonian** vector field $\mathcal{H}(h) = (-\partial h/\partial y, \partial h/\partial x)$ is orthogonal to the gradient $\nabla h = (\partial h/\partial x, \partial h/\partial y)$, and is therefore tangent to the level curves of h. Thus, the level sets of h are the integral curves of $\mathcal{H}(h)$, and hence can be traced by solving an ordinary differential equation (see Figure 3). The integration of ordinary differential equations related to the gradient is also the main theme of the physically-based sampling method described later in this article.

To cast the computation of a level curve as an initial value problem, a point on the curve is needed. Actually, a point on each connected component of the level curve is needed, if all components are to be found. In this case, for each such point (x_0, y_0), the corresponding connected component is the solution of the following Cauchy problem:

$$\begin{aligned} \frac{dx}{dt} &= -\frac{\partial h}{\partial y} & \qquad x(0) &= x_0, \\ \frac{dy}{dt} &= \frac{\partial h}{\partial x} & \qquad y(0) &= y_0. \end{aligned}$$

Several classical numerical methods exist for solving initial value problems [Lambert (1973)]. Although they are well-known and easily implemented, these general methods cannot exploit the fact that the solution of the Cauchy problem above is a level curve of a smooth function. One way to use this additional information to help stabilize the integration is to do a few iterations of Newton's method for the solution of non-linear equations as a correction after each prediction [Allgower–Georg (1990)]: if p is the current point on the curve, then Euler's prediction of the next point is $p \leftarrow p + \delta\mathcal{H}(h)(p)$, which is possibly on another level curve; we use Newton's method to bring the point p back to the correct level along the straight line orthogonal to $\nabla h(p)$ (see Figure 3).

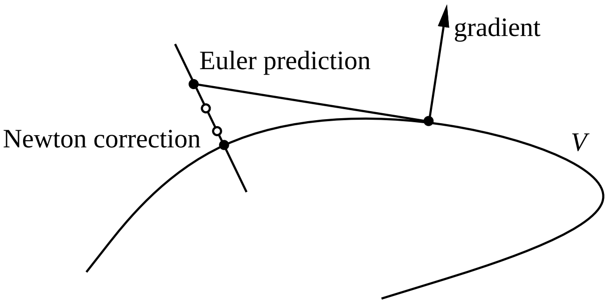

Figure 3: sampling by continuation.

Thus, we have the following **continuation** algorithm:

$p \leftarrow p + \delta\mathcal{H}(h)(p)$ *Euler predictor*
$u \leftarrow \nabla h(p)$ *correction direction*
while $|h(p)| > \varepsilon$
$\quad p \leftarrow p - \dfrac{h(p)}{\langle \nabla h(p), u\rangle} u$ *Newton corrector*

The combination of single-step predictors (such as Euler's) and Newton correctors provides a robust method for tracing level curves in several circumstances, despite the following restrictions:

- it is only applicable to plane curves;
- it needs starting points on each connected component;
- it needs special care with closed orbits.

It is possible to extend the continuation method described above to higher-dimensional manifolds by starting with an orthogonal complement of the gradient and carefully integrating along each vector in this complement; this is called the **moving frame** method [Allgower–Georg (1990)].

Sampling by enumeration

Instead of computing an approximation of the level sets of the exact h, we can compute the exact level sets of an approximation of h. If this is to be easier than dealing with the exact h, then the approximation should be sufficiently simple so that its exact level sets are easy to compute. This is achieved by taking a piecewise smooth approximation that is very simple on each piece: the pieces are the cells of a cellular decomposition of the ambient space, and the most common approximations are piecewise linear, obtained by scanning the decomposition and computing the intersection of the level set with each cell. The need for starting points is thus avoided and all connected components are found with no special processing. Methods that solve equations by scanning cellular decompositions are called **enumeration** methods; when the cells are simplices—a very common choice—these methods are called **simplicial** methods (see Figure 4).

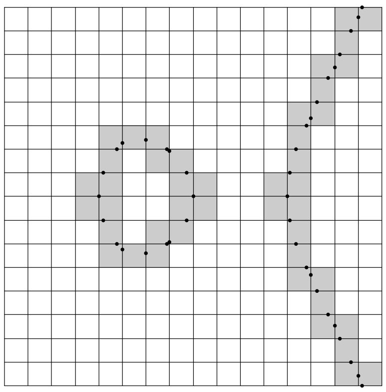

Figure 4: sampling by enumeration.

The simplest decompositions are regular, obtained by translating and scaling a single prototype cell, e.g., a unit hypercube aligned with the coordinate axes. The next level of complexity allows rotations of the prototype cell, as in regular triangular decompositions. In principle, the geometry of the cells could be arbitrary, but if a cell has many facets, then there are many possibilities for its intersection with the level set. Moreover, if the cells have complex geometry, then the decomposition has a complex topology, making it harder to coordinate the scan. In practice, only hypercubical or simplicial cells are used, arranged in cellular decompositions having well understood combinatorics [Allgower–Schmidt (1985), Hall–Warren (1990)].

As usual, tolerance is a problem and the size of the cells must be carefully chosen to avoid missing features because of undersampling. However, choosing a very small cell size will greatly increase the number of cells to be scanned. Thus, regular cellular decompositions rapidly increase in complexity under the demands of precision. One attempt to overcome this problem is to exploit the geometry of the object and make adaptive cellular decompositions, which reduce the total number of cells to be scanned by concentrating small cells around features [Bloomenthal (1988), Velho (1990), Hall–Warren (1990)].

Independently of how the cells are chosen, the sample provided by a scan is only structured locally at each cell; global structuring, such as the correct glueing of pieces and the identification of connected components, must be done *a posteriori*. This is an additional reason for requiring simple decomposition topology.

Enumeration methods can be very expensive because only a few cells intersect the object, especially in high co-dimension. Nevertheless, enumeration methods are applicable not only to hypersurfaces but to submanifolds of all co-dimensions. However, the higher the co-dimension, the fewer the cells intersected by the object. This is a serious problem because, when the cells are simplices, the total number of cells grows exponentially with the dimension: a simplicial decomposition of a hypercube in dimension n needs $\Omega(c^n\sqrt{n!})$ simplices [Haiman (1991)].

Piecewise linear approximations for implicit objects can be found by a continuation variant of the enumeration method that follows the solution at each of the intersecting cells and only at those, by using "pivoting" procedures to choose the cells to be scanned. This method provides global structuring concurrently with sampling; however, it brings back the need for starting points on each connected component. Moreover, it is now necessary to keep track of all visited cells, in order to identify closed components. Despite these restrictions, piecewise linear continuation methods are very successful, and widely used in practice [Allgower–Georg (1990)].

Physically-based sampling

The main theme of our physically-based method for sampling an object V given implicitly by a function h is the integration of ordinary differential equations related to the gradient vector field ∇h.

Unlike continuation methods, which integrate orthogonal complements of the gradient, and need starting points correctly placed on each connected component of V, our method integrates differential equations based on a modified gradient field, and can start at an arbitrary sample of points on the ambient space: the ω-limit of the orbit of each point will be on V, and will provide the desired sample.

The modified gradient vector field we shall use is $F = -\text{sign}(h)\nabla h$, obtained by reversing the sense of ∇h when h is positive: this provides a vector field pointing locally in the direction of V (see Figure 5). The attractors of F are the set V and the points where h has a positive local minimum. Because we are mainly interested in V, we shall call these latter points **spurious attractors**. Note that spurious attractors need not be isolated points: in general, they will be a submanifold of V. Although some sample points can land on spurious attractors, this does not seriously affect the sampling because we can discard points where h is not zero.

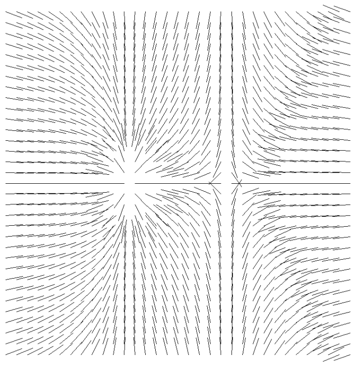

Figure 5: modified gradient vector field.

The field F corresponds to the potential function $U = |h|$. The points of global minimum potential energy are exactly those on V. However, the points where h has a positive local minimum are also local minima of the potential energy; spurious attractors correspond to particles that get trapped at these local minima.

The potential function U is not smooth on V, where $h = 0$. This implies that the field F is not continuous at regular points of V; in fact, the field F is continuous on V only at singular points. One way to avoid this discontinuity is to consider h^2 instead of h: the two functions have the same set of zeros but the modified gradient for h^2 is $-\text{sign}(h^2)\nabla h^2 = -2h\nabla h$, which is continuous everywhere. However, the convergence of the numerical methods used for integration is slower for h^2 than it is for h. Moreover, the field for h^2 has more spurious attractors than the field for h because all local extrema of h^2 are positive.

We consider two physical interpretations for the vector field F; they provide autonomous dynamical systems that are integrated to simulate Newtonian mechanics. Discrete models for physical systems are well suited for computer simulation [Greenspan (1973)] and have recently been successfully used in geometric modeling [Terzopoulos–Fleischer (1988), Szeliski–Tonnesen (1991)] and mesh generation for finite-element analysis [Gossard (1991)].

The first physical interpretation is **kinematic**: the field F describes velocity in terms of position. The equation of motion corresponding to this interpretation is

$$\frac{dx}{dt} + \mathrm{sign}(h)\nabla h = 0.$$

The other interpretation is **dynamical**: F is a force field in a dissipative medium. A particle released at rest into this medium is subjected to forces induced by F which make it move towards V and then oscillate around it. Adding friction to the movement guarantees that the particle will tend to equilibrium at a point on V. The resulting equation of motion for a unit mass particle is then

$$\frac{d^2x}{dt^2} + \gamma\frac{dx}{dt} + \mathrm{sign}(h)\nabla h = 0,$$

where γ is a positive real number representing friction proportional to the velocity. The presence of friction is important because, when $\gamma = 0$, this second-order dynamical system can have additional spurious attractors, other than the ones described above. Incerti, Parisi and Zirilli (1979) have proposed a similar differential equation for finding zeros of functions $\mathbf{R}^n \to \mathbf{R}^n$; however, they were interested in finding any one solution, not many, as required in sampling for geometric modeling.

By releasing a large set of randomly placed particles into either field and simulating the corresponding physics, we can generate a "random" sample of points on V (see Figure 6). However, there seems to be no clear relation between the initial position of the particles and the final distribution of points on V, in the sense that we do not know how to predict where a particle will land on V, even though we are dealing with deterministic systems. Nevertheless, points tend to concentrate near high curvature and singularities. On the other hand, no portion of V is consistently missed by this process, in the sense that every open subset of V is contained in the ω-limit of some open subset of $\mathbf{R}^n$.

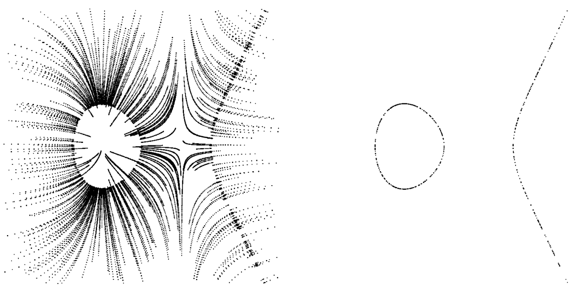

Figure 6: orbits and final sample.

The equations of motion for each particle are solved numerically, using one of the classical numerical methods, such as those of Euler or Runge–Kutta. However, if particles are to approach V steadily, adaptive variations of these methods are required, in which step control is used to avoid divergent oscillations. Such oscillations are mainly due to the discretizations used in the numerical methods, although they are also inherent to the physics in the dynamical case: indeed, whereas the orbits in the kinematic case do not cross V and oscillations are the artifact of numerical methods, in the dynamical case, particles approach V with non-zero velocity, causing them to cross V and only then being forced back to it because the force field changes sign. Controlling integration step size causes such particles to approach V more carefully each time they cross it, thus guaranteeing convergence. On the other hand, approaching attractors with non-zero velocity can actually help a particle to avoid spurious attractors.

Note that the equations of motion are uncoupled and hence can be solved in parallel. In particular, it is not necessary to keep all particles in a single time frame, and we may use a different step size for each particle. This contrasts with other particle systems used in computer graphics and animation, where the joint movement of the particles is important for realism [Reeves (1983)]. Therefore, we can implement independent step control by making the step size of a particle depend on its position; we do this by halving the step size every time the particle crosses V. Moreover, we can stop the simulation of a particle's movement as soon as it has reached a desired level of equilibrium (measured, for instance, by the value of h at the position of the particle or by the value of its the step size).

The following algorithm combines a classical single-step Euler integration with step control as described above, to provide a simple and robust method for the integration of the equations of motion in the dynamical case. If x is the position of a particle, v is its velocity, and δ is its current step size, then its next position, velocity and step size are computed as follows:

$y \leftarrow h(x)$	*remember current level*
$v \leftarrow v + \delta(F(x) - \gamma v)$	*Euler predictor*
$x \leftarrow x + \delta v$	*modified Euler predictor*
if $\mathrm{sign}(y) \neq \mathrm{sign}(h(x))$	*check crossing*
$\quad \delta \leftarrow \delta/2$	*step control*
$\quad v \leftarrow 0$	*re-start from rest*

Note that this algorithm is a variant of the strict Euler method, in the sense that the predicted value of v is used to predict x, instead of predicting both in parallel. Step control is done when h changes sign at two consecutive positions: it is assumed that the particle has crossed V at this time. As a further refinement, we re-start a particle from rest after such crossings; this allows oscillating particles to approach V very carefully.

The corresponding algorithm for the kinematic case is simply:

$y \leftarrow h(x)$	*remember current level*
$x \leftarrow x + \delta F(x)$	*Euler predictor*
if $\mathrm{sign}(y) \neq \mathrm{sign}(h(x))$	*check crossing*
$\quad \delta \leftarrow \delta/2$	*step control*

Although both methods integrate ordinary differential equations, the sampling method we have described is more robust than the continuation methods described earlier because:

- continuation methods need starting points on V, whereas arbitrary samples are adequate as initial conditions for our method;
- care must be taken when integrating the Hamiltonian in order to stay on level 0: single-step numerical methods without correction tend to jump to other levels. On the other hand, our method starts at arbitrary positions in the ambient space and aims for a submanifold—an easier task than having to start on a submanifold and trying to stay on it. Level jumping is no longer a problem because all trajectories lead to V; accordingly, single-step integration methods are adequate.
- our method is naturally parallel;
- our method does not need a predefined region of interest, because the particles will track V wherever it is in the ambient space.

The two interpretations used in our sampling method have complementary advantages. In the kinematic case, the differential equation is simpler, and the convergence is faster. Moreover, particles approach V orthogonally. On the other hand, in the dynamical case, spurious attractors are more easily avoided, making the sampling more robust. Moreover, convergence can be controlled by modifying the value of friction or by re-starting particles from rest when they seem not to be converging to equilibrium.

Practical issues

The sampling method described above can be summarized in the following algorithm:

· select initial sample randomly;
· select sampling tolerance $\varepsilon > 0$;
· simulate physical motions;
· stop simulation when $|h| < \varepsilon$ on the sample;

The main practical problems with this method are the selection of the initial sample and the selection of the sampling tolerance ε. If the initial sample is too far away from $h^{-1}(0)$, then the sample after equilibrium might not cover all of V. We may therefore view the sampling step as a preliminary search for V. After V is located, we can then sample it more thoroughly by selecting another set of initial conditions near V, e.g., by sampling a bounding box twice as large as the bounding box of the preliminary sample. A few iterations of this process are usually enough to locate and sample all components of V without bias.

Although prediction and step control are very simple, the integration algorithms perform well in practice. One minor problem is the choice of initial step size. In our explorative implementation, this choice is made by trial and error, but it should be possible to estimate a good step size based on the initial position. A naive estimate would be the value of h at the point, but this is not a good estimate if h is flat near this point (see Figure 7). Better estimates would take the gradient of h into account, in a way similar to Newton's method.

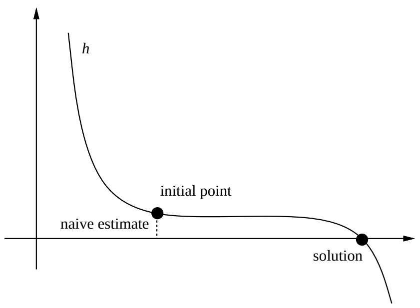

Figure 7: naive estimate of initial step size.

To get a good final sample with our physically-based method, we usually need many points, and this means thousands instead of hundreds, especially for higher dimensional objects. (The sample in Figure 6 has 500 points; this many points are usually enough for curves. On the other hand, the sample in Figure 1 has 5000 points; surfaces cannot usually be adequately sampled with less that 1000 points.) Whereas the simulation of uncoupled physical systems with thousands of particles is not too expensive in a workstation, geometric computations, such as the computation of polygonal approximations, are usually tightly coupled. Moreover, the samples provided by the physically-based method usually contain many almost coincident points; such concentrations occur at singularities and points of high curvature, as mentioned above. We are thus faced with the problem of extracting a representative sample from the equilibrium sample.

One good way of doing this extraction is to select a real number $\delta > 0$ and collapse all δ-cliques: a **δ-clique** is a set of points such that the distance between any two points in the set is at most δ. The obvious brute-force algorithm computes δ-cliques in cubic time. A good approximation is given by the following bucketing algorithm, which runs in linear time: divide the bounding box of the sample into square buckets of size δ, and collapse the points in each bucket. This collapsing can be done either by electing representative or by using the barycenter of the population of each bucket.

Conclusion

We have described a new method for sampling implicit objects that uses equilibrium configurations of simulated physical motions, and which is more robust than classical continuation methods.

Although a sample of points is not a complete geometric model, dense samples have strong perceptual meaning (see Figure 1), especially if adequate visualization tools are available. It would be easier to use implicit objects in geometric modeling if samples on such objects could be computed fast. The method we have presented is an example of such a tool.

Much like splines, which are used not only for approximation, but also for free-form modeling, our sampling method provides not only an approximation tool, but also a dynamic modeling tool: by varying implicit equations or by adding local potential fields, it is possible to dynamically control the shape of the model. After a satisfactory shape is found, a complete geometric model can then be constructed by using structuring methods [Figueiredo (1992)].

Acknowledgements

Demetri Terzopoulos generously shared with us his thoughts on physically-based surface reconstruction, in an informal conversation during the Workshop on Geometric Modeling at IMPA, in January 1991. Experimentation with some of his ideas later resulted in the physically-based sampling method presented in this article.

The research reported here 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), FAPERJ (Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro), and IBM Brasil.

References

E. L. Allgower and K. Georg, *Numerical Continuation Methods: An Introduction*, Springer-Verlag, 1990.

E. L. Allgower and P. H. Schmidt, An algorithm for piecewise-linear approximation of an implicitly defined manifold, *SIAM Journal on Numerical Analysis* **22** (1985) 322–346.

J. Bloomenthal, Polygonization of implicit surfaces, *Computer Aided Geometric Design* **5** (1988) 341–355.

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

L. H. de Figueiredo, *Computational Morphology of Implicit Curves*, doctoral thesis, IMPA, 1992.

C. Giertsen, A. Halvorsen and P. R. Flood, Graph directed modelling from serial sections, *The Visual Computer* **6** (1990) 284–290.

D. Gossard, *Product models with built-in physics*, unpublished manuscript presented at the Second SIAM Conference on Geometric Design, 1991.

D. Greenspan, *Discrete Models*, Addison-Wesley, 1973.

M. Haiman, A simple and relatively efficient triangulation of the n-cube, *Discrete and Computational Geometry* **6** (1991) 287–289.

M. Hall and J. Warren, Adaptive polygonization of implicitly defined surfaces, *IEEE Computer Graphics & Applications* **10** (1990) 33–42.

S. Incerti, V. Parisi and F. Zirilli, A new method for solving nonlinear simultaneous equations, *SIAM Journal on Numerical Analysis* **16** (1979) 779–789.

J. D. Lambert, *Computational Methods in Ordinary Differential Equations*, Wiley, 1973.

W. T. Reeves, Particle systems—a technique for modeling a class of fuzzy objects, *Computer Graphics* **17** (1983) 359–376 (Proceedings of SIGGRAPH '83).

D. Terzopoulos and K. Fleischer, Deformable models, *The Visual Computer* **4** (1988) 306–331.

R. Szeliski and D. Tonnesen, *Surface modeling with oriented particle systems*, *Computer Graphics* **26** (1992) 185–194. (Proceedings of SIGGRAPH '92).

L. C. Velho, Adaptive polygonization of implicit surfaces using simplicial decomposition and boundary constraints, *Proceedings of Eurographics '90*, September 1990, 125–136.

V. Vlassopoulos, Adaptive polygonization of parametric surfaces, *The Visual Computer* **6** (1990) 291–298.