Sampling implicit objects with physically-based particle systems

Luiz Henrique de Figueiredo, Jonas Gomes

IMPA–Instituto de Matemática Pura e Aplicada
Estrada Dona Castorina 110, 22460-320 Rio de Janeiro, RJ, Brazil
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 the ambient space. In this method, samples are "randomly" generated using physically-based particle systems.

Introduction

In computer graphics, an object is described either by a set of sample points or by an analytic scheme that uses mathematical equations to define its geometry and topology. Descriptions based on samples occur in areas such as medical images and terrain models. Analytical descriptions are usually found in applications of geometric modeling, such computer-aided design and manufacture.

When an object is described by samples, a reconstruction scheme is needed to recover its geometry and topology from the samples. This problem, called *structuring*, consists of providing a combinatorial structure to the samples in order to (ideally) recover the exact topology of the object and an approximation of its geometry.

When the object is described analytically, we must find a finite, discrete representation of its geometry and topology so that the object can be represented in the computer. Usually, this is accomplished by *sampling* points on the object. Again, these samples must be structured to provide an exact reconstruction of the topology and a robust approximation of its geometry.

A typical example of this process is the polygonization of objects usually found in geometric modeling software. These polygonal approximations are discrete representations 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.

Therefore, the computation of polygonal approximations of geometric objects defined analytically 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 (Fig. 1).

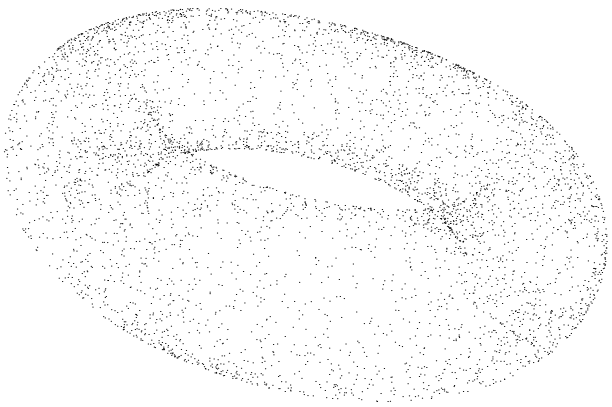

Figure 1: A sample of points on a torus.

The way these sub-problems are solved depends on how geometric objects are defined: the two most common ways of defining geometric objects—parametrically and implicitly—require very different methods for sampling and structuring. In this paper, we discuss methods for sampling implicit objects.

Sampling parametric objects is easy because it reduces to sampling the parameter domain; this is usually performed 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 conceptually requires the solution of many non-linear equations; structuring is difficult because there is no guiding mesh.

Classical methods for computing polygonal approximations for implicit objects 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 that 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.

We start by reviewing the sampling technique of three classical polygonization methods for implicit objects. We then describe in detail and discuss a new approach to sampling that uses physically-based particle systems.

Sampling implicit objects

Let $h: \mathbf{R}^n \to \mathbf{R}$ be a differentiable real function defining an implicit object $V = h^{-1}(0)$. For $n = 2$, the object V is a curve; for $n = 3$, it is a surface; in general, V is a manifold of co-dimension 1, i.e., of dimension $n - 1$.

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. In this sense, sampling implicit objects conceptually requires the solution of many non-linear equations.

A sampling method can impose a structure that does not correspond to the geometry of the object being sampled. 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 (Fig. 2).

Classical polygonization methods for implicit objects perform structuring concurrently with sampling. As mentioned before, we prefer to study sampling and structuring separately. In this section, we review the sampling technique of three classical methods: ray-casting, continuation, and enumeration.

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 call *rays*.

The simplest rays correspond to fixing some of the variables. For instance, we can 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 $\mathbf{R}^{n-1}$. In other words, we compute the intersection of V with the "vertical" rays $\{a\} \times \mathbf{R}$; accordingly, we call this method *vertical ray-casting* (Fig. 2).

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 depends on the sample of rays $\mathcal{R}$ (Fig. 2):

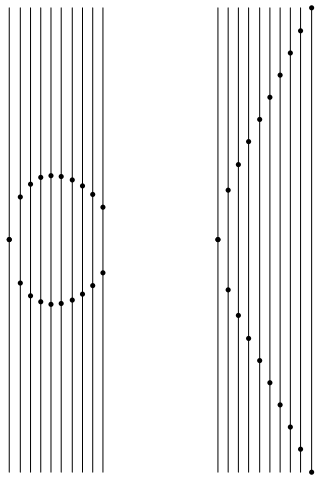

Figure 2: Sampling by ray-casting.

- because there is no *a priori* criterion for choosing rays intersecting V, a majority of the rays in $\mathcal{R}$ may not contribute with 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 (but this 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 use a uniform mesh to sample the "horizontal plane" $\mathbf{R}^{n-1}$, as in parametric sampling. In this case, the resulting sample on V is then biased against regions where the tangent space of V is vertical.

In general, it is too hard to relate the size and density of 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,

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

is the Jacobian matrix of h in co-dimension 1. The gradient is a vector field defined 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 (Fig. 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:

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

Several classical numerical methods exist for solving initial value problems. 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 nonlinear equations as a correction after each prediction [1]: if p is the current point on the curve, then Euler's prediction of the next point is $p + \delta\mathcal{H}(h)(p)$, which is possibly on a different level curve; we use Newton's method to bring the point p back to the correct level along the straight line orthogonal to $\nabla h(p)$ (Fig. 3).

The algorithm for this Euler–Newton continuation method can be written:

$p \leftarrow p + \delta\mathcal{H}(h)(p)$ — *Euler predictor*
$u \leftarrow \nabla h(p)$ — *correction direction*
while $|h(p)| > \varepsilon$
  $p \leftarrow p - \dfrac{h(p)}{\langle \nabla h(p), u \rangle} u$ — *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 component;
- it needs special care with closed components.

This continuation method can be extended to higher-dimensional manifolds by starting with an orthogonal complement of the gradient and carefully integrating along each vector in this complement; this is the *moving frame* method [1].

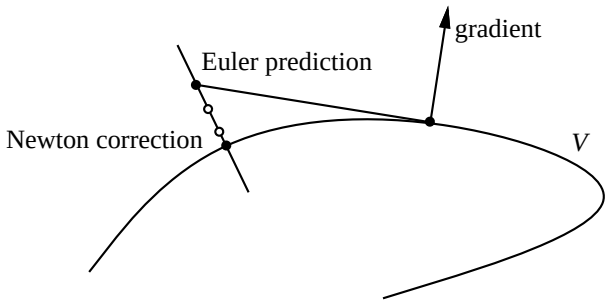

Figure 3: Sampling by continuation.

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 [2]. 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.

The most common approximations are piecewise linear, obtained by scanning the cellular 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 (Fig. 4). A very common choice for cellular decompositions are *simplicial* decompositions in which the cells are simplices (i.e., triangles, tetrahedra, etc.).

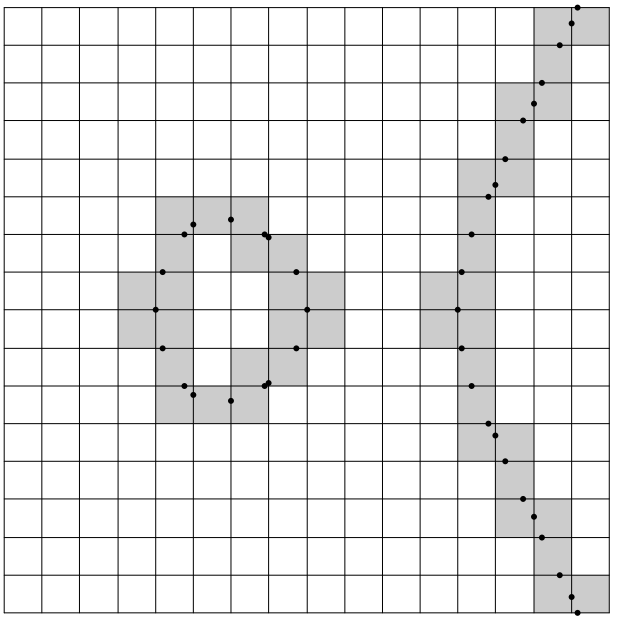

Figure 4: Sampling by enumeration.

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 [2].

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 greatly increases 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 use adaptive cellular decompositions, which reduce the total number of cells to be scanned by concentrating small cells around features [3,4,5] (see below for an alternative method).

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 [6].

Independently of how the cells are chosen, the sample provided by the full scan described above 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.

Some variants of the enumeration method perform global structuring concurrently with sampling by using continuation: they follow the solution at each of the intersecting cells and only at those, by using "pivoting" procedures to choose the cells to be scanned [1,2,3]. However, this brings back the need for starting points on each connected component; it is also necessary to keep track of all visited cells, in order to identify closed components. Despite these minor restrictions, such continuation methods are very successful, and widely used in practice [1].

Some recent enumeration methods perform adaptive enumeration, using small cells only where they are needed, i.e., near the object. These methods search for the object by recursively exploring a cellular decomposition of the ambient space. The recursion criterion is "does this cell contain solutions of $h(x) = 0$?". If a cell contains solutions, then it is subdivided into smaller subcells, which are explored recursively, until they are small enough. The meaning of "small enough" depends on the application: for rendering, it might mean "smaller than a pixel"; for other applications, it might mean $|h(x)| \leq \varepsilon$ for all vertices x of the cell, where ε is a pre-defined tolerance.

To decide whether a cell contains solutions of $h(x) = 0$, an estimate of the image of the cell under h is computed. This estimate is an interval; if it does not contain zero, then the cell does not contain zeros of h. Interval arithmetic is the natural technique for computing such estimates [7] (Fig. 5).

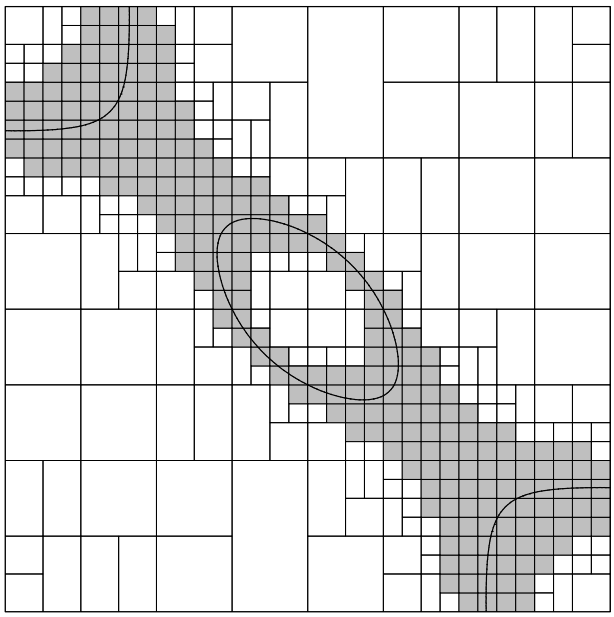

Figure 5: Adaptive enumeration with interval arithmetic; $h(x, y) = x^2 + y^2 + xy - (xy)^2/2 - 1/4$.

The estimates provided by interval arithmetic are too conservative; in complex expressions having many coupled sub-expressions, these estimates can quickly become useless. An alternative to interval arithmetic, called affine arithmetic, has been proposed to overcome this problem [8]. Affine arithmetic does handle coupling in expressions and is therefore able to provide better estimates. Adaptive enumeration of implicit objects using affine arithmetic is a promising technique, specially for rendering [9] (Fig. 6).

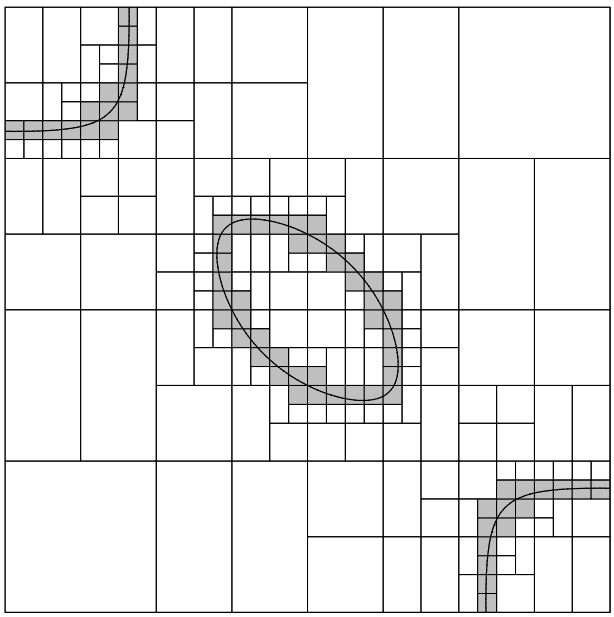

Figure 6: Adaptive enumeration with affine arithmetic; $h(x,y) = x^2 + y^2 + xy - (xy)^2/2 - 1/4$.

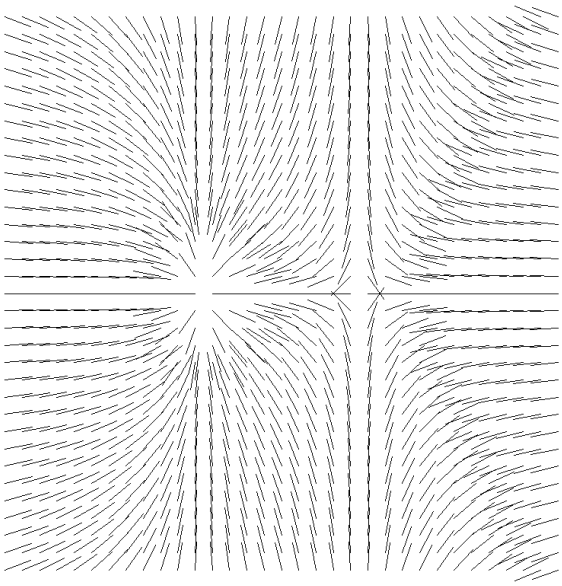

Figure 7: Modified gradient vector field for $h(x,y) = y^2 - x^3 + x$.

Physically-based sampling

Let V be an object given implicitly in $\mathbf{R}^n$ by a function h. The main theme of the physically-based method for sampling V that we now describe 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, this method integrates differential equations based on a modified gradient field, and can start at an arbitrary sample of points on the ambient space: the limit of the orbit of each point is on V; together, they provide the desired sample (Fig. 8).

The modified gradient vector field we 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 (Fig. 7). The attractors of F are the set V and the points where h has either a positive local minimum or a negative local maximum. Because we are mainly interested in V, we call these latter points *spurious attractors*. Note that spurious attractors need not be isolated points: in general, they form 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 (of course, in practice, "not zero" means "above a sampling tolerance ε").

Physical interpretations

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 or a negative local maximum 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 $h = 0$. Hence, 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 because the field for h^2 decreases in magnitude near V, where it is zero.

We consider two physical interpretations for the vector field F: one kinematic and one dynamical. Both interpretations provide autonomous dynamical systems that are integrated to simulate Newtonian mechanics. Discrete models for physical systems are well suited for computer simulation and have recently been successfully used in geometric modeling [10,11].

In the kinematic interpretation, the field F describes velocity in terms of position. The equation of motion corresponding to this interpretation is

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

In the dynamical interpretation, F is a force field in a dissipative medium. A particle released at rest into this medium is subjected to forces induced by F that 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

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

where γ is a positive real number representing friction proportional to the velocity. Friction is important because, when $\gamma = 0$, this second-order dynamical system can have additional spurious attractors, other than the ones described above. Friction also provides additional control over the sampling process. Incerti, Parisi and Zirilli [12] 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 until equilibrium, we can generate a "random" sample of points on V (Fig. 8).

Motion simulation

The equations of motion for each particle are solved numerically, using one of the classical integration 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, in the kinematic case, the orbits 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. 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 step size).

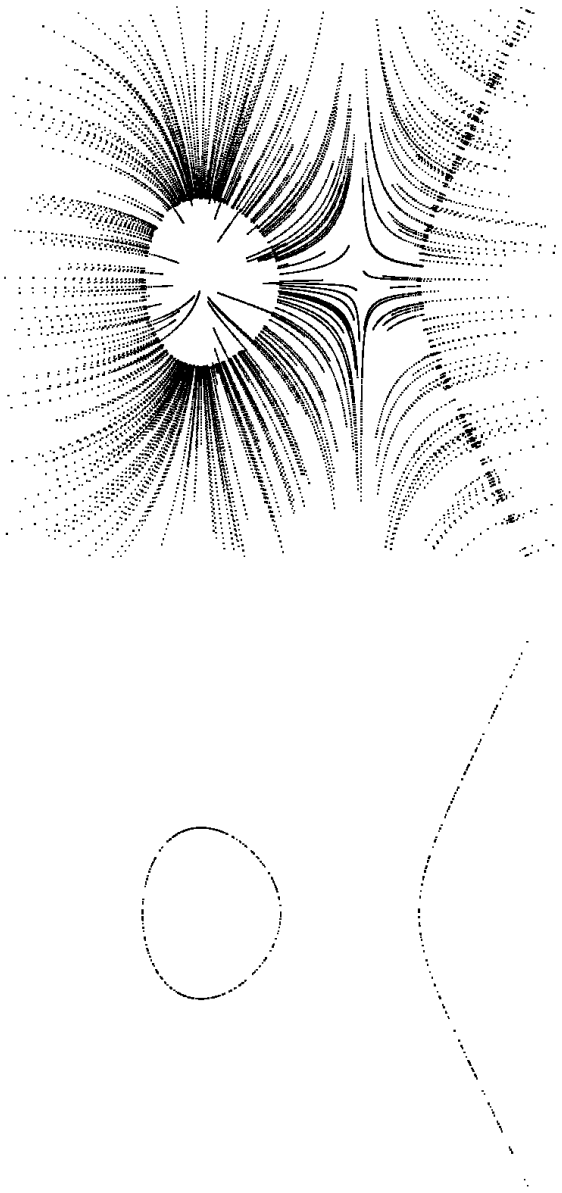

Figure 8: Orbits and final sample.

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 $\text{sign}(y) \neq \text{sign}(h(x))$ *check crossing*
  $\delta \leftarrow \delta/2$ *step control*
  $v \leftarrow 0$ *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 $\text{sign}(y) \neq \text{sign}(h(x))$ *check crossing*
  $\delta \leftarrow \delta/2$ *step control*

The two physical interpretations 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 whenever they seem not to be converging to equilibrium.

Qualitative distribution of sample

There is no obvious relation between the initial position of the particles and the final distribution of points on V. Although the dynamical systems are deterministic, it is difficult to predict precisely where a particle will land on V. Nevertheless, the final distribution can be described qualitatively: no portion of V is consistently missed, and samples concentrate near high curvature and near spurious attractors.

To justify these claims, we consider the *reverse flow*, i.e., the flow associated to the vector field $-F$. This flow runs away from V and orthogonally to the level sets of h (Fig. 9).

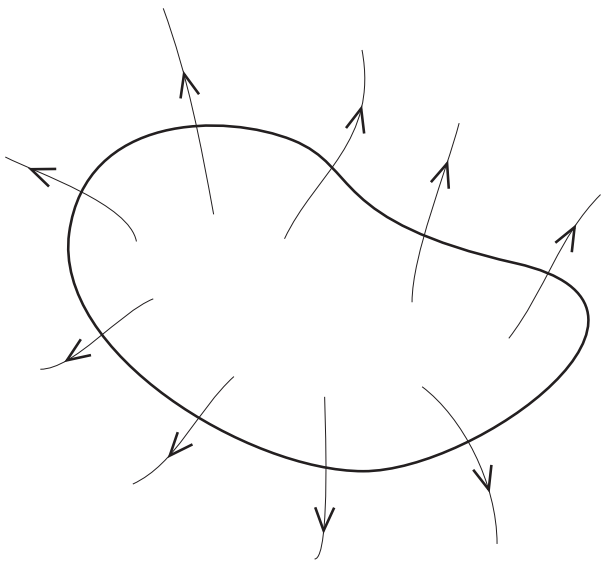

Figure 9: Reverse flow.

By considering the orbits of points near V along the reverse flow, we can describe where initial samples land on V. More precisely, take a small $\varepsilon > 0$ such that the set $\{x \in \mathbf{R}^n : |h(x)| < \varepsilon\}$ is a tubular neighborhood $\mathcal{T}$ of V, and consider an open set $\mathcal{U}$ of $\mathcal{T}$. By moving $\mathcal{U}$ through the reverse flow, we obtain a continuous family of open sets of the ambient space, called the *open family* associated to $\mathcal{U}$. The main property of this family is that an initial sample point chosen in any set of the family converges to a point in $\mathcal{U}$, i.e., to a sample point very near V. We can now justify the qualitative description given above:

1. *no portion of V is consistently missed.*
More precisely, every open set of $\mathcal{T}$ is the limit of an open set of $\mathbf{R}^n$. Indeed, take an open set $\mathcal{U}$ of $\mathcal{T}$. Then, all initial samples in the open family associated to $\mathcal{U}$ land on $\mathcal{T}$. Therefore, if the initial sample covers the ambient space without gaps, then the final sample covers all of V without gaps.

Although V is completely sampled, the points in the final sample are not evenly distributed: they tend to concentrate near high curvature and spurious attractors. This can be seen in Fig. 8: more particles converge to the top and bottom of the closed component than to anywhere else, although the initial sample was random and uniformly distributed.

2. *samples concentrate near high curvature.*
By definition of curvature, the direction of the normal vector varies faster in regions of high curvature than in regions of low curvature. Since the normal vector of V is the gradient of h, the integral curves of the reverse flow spread out more in regions of high curvature than in regions of low curvature. Now take

two open sets $\mathcal{U}$ and $\mathcal{W}$ of $\mathcal{T}$ of the same volume in V (i.e., segments of the same length for curves; regions of the same area for surfaces). If the curvature of V in $\mathcal{W}$ is higher than in $\mathcal{U}$, then the volume of the sets in the associated open families grows faster for the family associated to $\mathcal{W}$. Thus, the probability of an initial sample point landing on $\mathcal{W}$ is higher than the probability of it landing on $\mathcal{U}$ (Fig. 10).

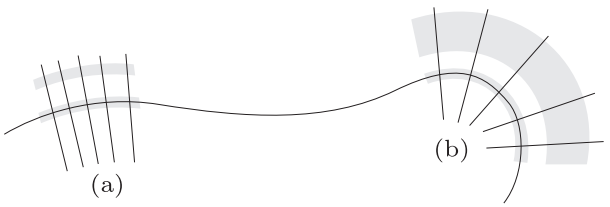

Figure 10: Reverse flow on regions of low curvature (a) and high curvature (b).

3. *samples concentrate near spurious attractors.* The spurious attractors of F are the points where h has either a positive local minimum or a negative local maximum. On these points, the reverse flow spreads out in all directions. By the preceding argument, this implies that samples concentrate near spurious attractors.

Practical issues

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

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

The main practical issues in this method are the selection of the initial sample, and the selection of the control parameters: the sampling tolerance ε, the initial step size δ, and the friction coefficient γ.

If the initial sample is too far away from V, 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 gaps. Alternatively, we can perform a coarse enumeration using interval or affine arithmetic to locate regions guaranteed to contain pieces of V; the initial sample can then be taken in these regions.

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 is the value of h at the point, but this is not a good estimate if h is flat near this point (Fig. 11). Better estimates can be obtained by taking the gradient of h into account, in a way similar to Newton's method.

The sampling tolerance ε depends on the application. The friction coefficient γ depends on the field F. They are best controlled interactively.

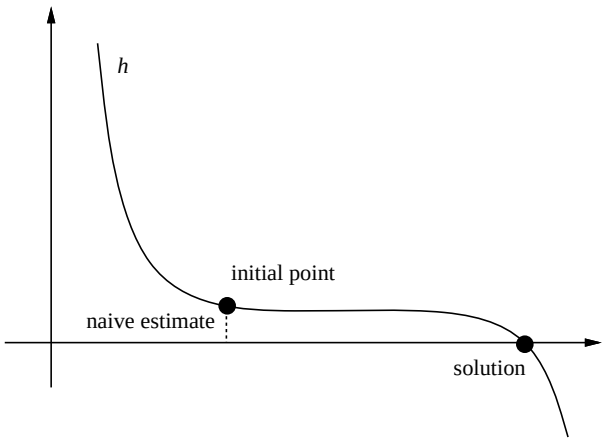

Figure 11: Naive estimate of initial step size.

Post-processing

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. For instance, the initial sample in Fig. 8 has 500 points; this many points are usually enough for curves. On the other hand, the sample in Fig. 1 has 5000 points; surfaces cannot usually be adequately sampled with less that 1000 points with our method because of concentration.

Whereas the simulation of uncoupled physical systems with thousands of particles is not too expensive in a workstation, the geometric computations needed for structuring are usually tightly coupled (we have proposed a structuring method for curves that uses minimal spanning trees [13,14]). Moreover, the samples provided by the physically-based method usually contain many almost coincident points; such concentrations occur at regions of high curvature and on spurious attractors, as mentioned above (Fig. 8). 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 expected time: divide the bounding box of the sample into regular buckets of size δ, and collapse the points in each bucket. This collapsing can be done either by electing a representative or by using the barycenter of the population in each bucket.

Comparison with continuation methods

Although both methods integrate ordinary differential equations, the physically-based sampling method we have described in this section is more robust than the continuation methods described earlier, in the following aspects:

- continuation methods need starting points on V, whereas arbitrary samples are adequate as initial conditions for our method;
- single-step Euler integration, without Newton correction, is adequate for solving motion equations, because all trajectories lead to V;
- the method is naturally parallel;
- the method does not need a predefined region of interest because the particles will track V wherever it is in the ambient space.

Related work

The *scattering* method briefly described by Bloomenthal and Wyvill [15] also uses particles to sample implicit surfaces defined by skeletons. This method is essentially our kinematic sampling method, except that their initial samples lie on the skeleton and on its cross sections. However, they do not describe the integration algorithm in any detail.

More recently, Witkin and Heckbert described a different method for sampling and modeling implicit surfaces with particles [11]. They use a much more complete physical model that constrains particles to move on the surface once they reach it, and thus can perform adaptive sampling by uniformly distributing particles on the surface using relaxation. Our method uses a simpler physical model that is faster to simulate but does not allow particle interaction and does not provide uniform sample distribution. On the other hand, particle concentration near high curvature zones can be exploited for enhancing visual perception of surface geometry.

A brief description of the sampling method presented here appeared in a previous paper [16].

Conclusion

We have described a new method for sampling implicit objects that uses equilibrium configurations of simulated physical motions of particles. This method is in some aspects more robust than classical continuation methods.

Although a sample of points is not a complete geometric model, dense samples have strong perceptual meaning (Fig. 1), especially if adequate visualization tools are available. We found that a simple interactive "fly-about" tool with depth cueing was very effective for visualizing clouds of three-dimensional particles generated by our sampling method. Bloomenthal and Wyvill report a similar observation [15]. It is easier to use implicit objects in geometric modeling if samples on such objects can be computed fast for faster feedback. The method we have presented is an example of such a tool; we think it would be a useful in an implicit modeler (see also [11,15]).

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 [15]. After a satisfactory shape is found, a complete geometric model can then be constructed by using structuring methods [13,14].

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 these ideas later [7,16] resulted in the sampling method presented here. We thank the referees whose careful reading and suggestions helped improved this paper. Figures 5 and 6 were kindly provided by J. Stolfi. This research was done in the VisGraf computer graphics laboratory at IMPA, and partially supported by CNPq, FAPERJ, and IBM Brasil.

References

1. E. Allgower, K. Georg, *Numerical Continuation Methods: An Introduction*, Springer (1990).
2. D. Dobkin, S. Levy, W. Thurston, A. Wilks, Contour tracing by piecewise linear approximations, *ACM Trans. on Graphics* **9** (1990) 389–423.
3. J. Bloomenthal, Polygonization of implicit surfaces, *Comp. Aided Geometric Design* **5** (1988) 341–355.
4. M. Hall, J. Warren, Adaptive polygonization of implicitly defined surfaces, *IEEE Computer*

Graphics & Applications **10** (1990) 33–42.
5. L. Velho, Adaptive polygonization of implicit surfaces using simplicial decomposition and boundary constraints, *Proc. Eurographics '90*, 125–136.
6. M. Haiman, A simple and relatively efficient triangulation of the n-cube, *Discrete and Computational Geometry* **6** (1991) 287–289.
7. T. Duff, Interval arithmetic and recursive subdivision for implicit functions and constructive solid geometry, *Proc. SIGGRAPH '92*, 131–138.
8. J. Comba, J. Stolfi, Affine arithmetic and its applications to computer graphics, *Proc. VI SIBGRAPI* (1993) 9–18. (Brazilian Symposium on Computer Graphics and Image Processing).
9. L. H. de Figueiredo, J. Stolfi, Adaptive enumeration of implicit surfaces with affine arithmetic, *Proc. Implicit Surfaces '95*, 161–170.
10. R. Szeliski, D. Tonnesen, Surface modeling with oriented particle systems, *Proc. SIGGRAPH '92*, 185–194.
11. A. Witkin, P. Heckbert, Using particles to sample and control implicit surfaces, *Proc. SIGGRAPH '94*, 269–278.
12. S. Incerti, V. Parisi, F. Zirilli, A new method for solving nonlinear simultaneous equations, *SIAM J. on Numerical Analysis* **16** (1979) 779–789.
13. L. H. de Figueiredo, *Computational Morphology of Implicit Curves*, doctoral thesis, IMPA, Rio de Janeiro, Brazil, April 1992.
14. L. H. de Figueiredo, J. Gomes, Computational morphology of curves, *The Visual Computer* **11** (1995) 105–112.
15. J. Bloomenthal, B. Wyvill, Interactive techniques for implicit modeling, *Computer Graphics* **24** #2 (Mar 1990) 109–116 (Proc. 1990 Symposium on Interactive 3D Graphics).
16. L. H. de Figueiredo, J. Gomes, D. Terzopoulos, L. Velho, Physically-based methods for polygonization of implicit surfaces, *Proc. of Graphics Interface '92*, 250–257.