A Fast Level-set Approach to 2D and 3D Reconstruction from Unorganized Sample Points

Marco Marcon, Luca Picarreta, Augusto Sarti, Stefano Tubaro
Dip. di Elettronica e Informazione – Politecnico di Milano
Piazza L. Da Vinci 32 – 20133 Milano – Italy
marcon/piccarre/sarti/tubaro@elet.polimi.it

Abstract

In this paper we propose a novel algorithm for the reconstruction of surfaces from sets of unorganized sample points, based on the temporal evolution of a volumetric function's level-set. The evolving front can be thought of as the surface that separates two different fluids that obey specific laws of fluid dynamics. One remarkable feature of this approach is its ability to model complex topologies thanks to a novel strategy that allows us to steer the front evolution using Voronoi surfaces in 3D space. Another remarkable feature of this algorithm is its computational efficiency, which proved to between one and two orders of magnitude better than traditional level-set approaches [1]

1. Introduction

Modeling surfaces from unorganized sets of points, i.e., retrieving surface topology from surface geometry, is a long-debated problem in the computer vision community. When the point-based data-set is very dense and the surface topology is not so complex to generate topological ambiguities, the solution to this problem is provided by a simple Delaunay triangulation equipped with appropriate distance-based criteria. Point-connection ambiguities, however, are easy to arise even with dense data-sets, and this is confirmed by a very rich literature on the topic.

In general, the solutions to the considered problem can be classified into two broad categories: those that directly construct the surface (boundary representation), and those that define the surface as a constraint in 3D space (volumetric representation). Working with boundary representations has the advantage of speed and allows us to control shape in a very straightforward fashion. Surface-based solutions, however, are difficult to use when dealing with complex topologies. Conversely, volumetric solutions tend to be quite insensitive to topological complexity (they may accommodate self-occluding surfaces, concavities, surfaces of volumes with holes, or even multiple objects), but they require a more redundant (volumetric) data structure, and a much heavier computational load.

One example of surface-based solution, proposed in [2,3], is based on the computation of the signed Euclidean distance between each sample point and a linearly regressed plane that approximates the local tangent plane. The final surface is then obtained by interpolating this distance function with a *marching cubes* algorithm. Curless and Levoy [4] developed an algorithm tuned for laser range data, which is able guarantee a good rejection to point misalignments using the deviation from the local tangent plane. Another well-known approach is that of the α-shape [5,6], which associates a polyhedral shape to an unorganized set of points through a parametrized construction. Bajaj, Bernardini, and Xu [7] recently used the α-shape approach as a first step in a complete reconstruction pipeline. Finally, algorithms based on "Delaunay sculpting" are often used (see, for example, Boissonnat [8] and Amenta ed al. [9]). Such solutions progressively eliminate tetrahedra from the Delaunay triangulation based on their circumspheres.

Algorithms based on the temporal evolution of a level-set of a 3D function belong to the volumetric category of surface modeling solutions [1]. Such methods require a volumetric function to be updated at every time step until the evolving front (level-set) reaches the desired configuration. If the volumetric function is defined on a voxset of N voxels per side, in principle the evolution requires an order of N^3 voxels to be updated for a number of iterations that is proportional N. This number of updates can be reduced from an order of N^4 to an order of N^3 by restricting the volume of interest to a narrow band sorrounding the evolving front [1]. More recently, a multi-resolution approach to level-set evolution was proposed in order to further reduce the number of updates to an order of $N^2 \log N$ [10]. Still, all such solutions need further steps to sufficiently reduce the computational cost and bring volumetric methods to practical usability.

In this paper we propose a novel approach to level-set evolution that dramatically reduces the computational cost of the method down to the level of surface-based solutions. In addition, the method that we propose further improves the ability of level-set methods to adapt to complex topological configurations using Voronoi point-sets.

The time-space evolution model of the level-set is based on the Navier-Stokes equations [11], which give the most general description of a fluid flow. The physical model of the system gives us a set of

parameters that allow us to accurately calibrate the front evolution.

In order to speed up the convergence of the system the fronts of the fluids, instead of evolving freely, are oriented towards the nearest sample point. In addition, in order to guarantee a fast evolution and a smooth convergence, their speed is proportional to the distance from the nearest point.

2. A two-fluid evolution paradigm

Our level-set evolution corresponds to that of two fluids of opposite mass that evolve in the volume. A volumetric function F is defined in such a way to describe the content of the two fluids in each voxel and we define as 'internal' voxels with a negative value of F and as 'external' the ones with a positive value. The resulting surface is then represented from the zero level surface of the F function that represents the interface between the two fluids. At the beginning of the evolution of the system all the space is filled with the internal fluid setting each voxel to a conventional value of -1. The external fluid sources are placed only on the boundaries of the whole space setting the value of these voxels to $+1$.

From this initial condition the system is left free to evolve following an equation based on the Navier-Stokes model for the conservation of the mass with a redefinition of the speed vector v as defined below.

The law of mass conservation is a general statement of kinematic nature, that is, independent of the nature of the fluid or of the forces acting on it. It express the empirical fact that, in a fluid system, mass cannot disappear from the system nor be created.

The quantity F is, in our case, the specific mass. The general form of conservation law for a system bounded by a closed surface S can be expressed in terms of variations of F due to fluxes that express the contributions from the surrounding points to the local value and through sources Q. The flux vector G contains two components, a diffusive contribution G_D and a convective part G_C. In its general form, a conservation law states that the variation per unit time of the quantity F within the volume Ω, i.e.

$$\frac{\partial}{\partial t}\int_\Omega F d\Omega \quad ,$$

should be equal to the net contribution from the incoming fluxes through the surface S, with the surface element vector dS pointing outward:

$$-\int_S G \cdot dS$$

plus contributions from the sources of the quantity F. These sources can be divided into volume and surface sources, Q_v and Q_S, and the total contribution is:

$$\int_\Omega Q_V \cdot d\Omega + \int_S Q_S \cdot dS$$

Hence the general form for the conservation equation for the quantity F is

$$\frac{\partial}{\partial t}\int_\Omega F d\Omega + \int_S G \cdot dS = \int_\Omega Q_V \cdot d\Omega + \int_S Q_S \cdot dS$$

or, with Gauss's theorem, for continuous fluxes and surface sources:

$$\frac{\partial}{\partial t}\int_\Omega F d\Omega + \int_\Omega \nabla \cdot G d\Omega = \int_\Omega Q_V d\Omega + \int_S \nabla \cdot Q_S dS$$

This last leads to the differential form of the conservation law, since the last equation is written for an arbitrary volume Ω:

$$\frac{\partial F}{\partial t} + \nabla \cdot G = Q_V + \nabla \cdot Q_S$$

An essential aspect of the conservation law lies in the fact that the internal variations of F, in the absence of volume sources, depend only on the flux contribution through the surface S and not on the flux values inside the volume Ω.

Separating the flux vector into its two components G_C and G_D we obtain a more precise form of the equation. Indeed the convective part of the flux vector G_C, attached to the quantity F in a flow of velocity v is the amount of F transported with the motion, and is given by $G_C = vF$.

The diffusive flow is defined as the contribution present in fluids at rest, due to the molecular, thermal agitation and is usually proportional to the gradient of F, i.e. $G_D = \gamma \nabla F$, where γ is the diffusivity constant.

In our algorithm we oriented the velocity v for the propagation of both fluxes towards the nearest sampled point. Each point, therefore, represents an attractor for both the internal and external fluids. The modulus of the speed vector is proportional to the distance from the nearest point allowing the fluids to *gently* converge to the desired surface. A further diffusive behaviour is also taken into account to obtain a more natural flux and a smooth interpolating surface.

We joined together the two contributions of the flux defining the convective part as above and implementing the diffusive part weighting the contributions from the near points with a Gaussian: with this formulation the standard deviation σ plays a role similar to the diffusivity term γ accounting the contribution of the whole region.

We found experimentally that a good choice for the evolution equation is (1)

$$\frac{\partial F}{\partial t} = \frac{\int_\Omega F(x)|v(x)|e^{-\frac{x^2}{2\sigma^2}}dx}{\int_\Omega |v(x)|e^{-\frac{x^2}{2\sigma^2}}dx} - F(x) \quad (1)$$

Experimentally we found that a sharp square function that extends only to the nearest neighbours works as well as the gaussian, and this is particularly useful for discretizing eq. (1).

In order to speed up computations, only the voxels where the derivative of the $F(x)$ exceeds a threshold are

recomputed at every time step, while all the others are kept fixed. In (1) $|\mathbf{v(x)}|$ is defined as $|\mathbf{x}-\mathbf{p}|^{\alpha}$ where $\mathbf{p}$ is a vector indicating the nearest point to $\mathbf{x}$ while α regulates the speed of convergence: good values have been found in the range between 1.6 and 2.4. Unlike traditional level-set algorithms, our approach defines the distance as a positive number ($\alpha = 2$), which results in a faster evolution (farther points are assigned a greater speed) and a more accurate convergence near the desired surface configuration.

3. Medial axis steering

Conventional level-set methods have a hard time producing a correct front evolution when dealing with thin blades, sharp spikes or deep and narrow holes, as they tend to round off sharp details, and level out holes. In our two-fluid model we can decide to look at the system from the inside or from the outside. From the inside, spikes and thin blades appear as narrow holes and grooves, respectively. Smoothing corners and spikes can thus be interpreted as a failure of the internal fluid to flow through such features. In order to get the fluids to penetrate into this type of features, a *medial axis* analysis, derived from the *Voronoi diagram*, is performed. In volume space the *medial axis* is the closure of the set of points with more than one closest point on the surface and this set of points constitutes surfaces that never meet the object but stay completely inside or outside. As a consequence, if any *medial axis* meets the level-set zero surface, then it means that we are in the presence of one of the features described above and that the fluid failed to flow inside it and stopped too early. This stagnation of the wrong fluid inside the feature can be solved by selecting the voxels that lie on the *medial axis* and setting them as sources of correct fluid. This allows us to locally restart the evolution of the system.

In order to choose the correct type of source for each voxel on the medial axis we have to consider that each Voronoi surface outside the object reaches the volume boundary so the external fluid sources belong to this type of *medial axis*.

A 2D example of this situation is show in figure 1. The extension to the 3D case is quite easy and guarantees a correct identification of the points inside convex hulls.

An example of the usage of the Voronoi surfaces on a 3D object is shown in Fig. 2, where the image on the left-hand side is correctly reconstructed after having inserted "external" sources inside the dragon's mouth along the Voronoi vertices. The image on the right-hand side, on the other hand, is based on a simple fluid dynamic model therefore the mouth ends up not being correctly modeled.
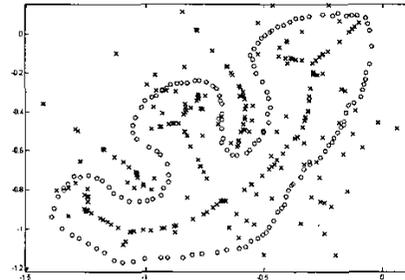

Figure 1: Discrimination of internal and external points using Vonoroi vertices, circles represent a non-convex discrete curve and crosses are the Voronoi vertices that seep into hulls.
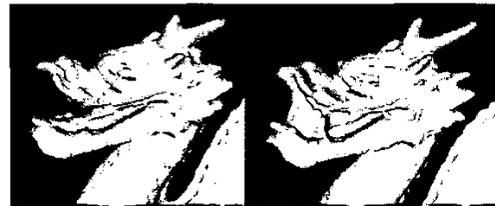

Figure 2: Impact of Voronoi-set steering on front evolution. Carving voxels out of the mouth of the dragon (left) is enabled by Voronoi diagrams. Without this mechanism the results would be as in the right image.

4. Improving the accuracy

As already explained above, in order to avoid the definition of internal sources the whole volume is initialized with internal fluid with the exclusion of the boundary of the defined space. In this area we place the sources of the external fluid. This approach gives a good approximation of the desired surface as the fluids move towards it but the initial excess of the internal fluid may give rise to a lumpy surface.

In order to overcome this problem, after the convergence and the Voronoi surface determination, a narrow band is defined around the surface and each voxel inside this band is set to zero defining an equilibrated presence of both fluids. At the same time all the other voxels outside this band are saturated to +1 or -1 accordingly to their sign. After this step the evolution is restarted but now with a substantial equilibrium for the two fluids which converge from the same distance towards the surface. Now, however, both fluids come from opposite sides, giving a better and balanced reconstruction for the interface.

5. Near-source local relaxation

Sources that are placed too close to sample points can prevent a correct reconstruction of the surface. In fact, an excessive flow of fluid could result in an overflow through the desired surface. In order to overcome this problem, we monitor the gradient of the interface between the two fluids: if it exceeds an

assigned threshold, then the intensity of the source is reduced in such a way to improve the match with the sample points location. This problem occurs, in particular, when modeling thin blades whose *medial axis* runs very close to the surface and to the volume boundary. One way to avoid this problem is to use a larger voxset at the cost of a heavier resource usage.

In Fig. 3 we show the evolution of the system in a 1D environment, we placed two points, at x=16.3 and x=86.4, placed in a space of 100 points. Their position is correctly honored from the level-set evolution as lon as they are far from the boundary. In Fig. 4 we can see the evolution of a level-set when the point is very close to the volume boundary and the external fluid overflows pushing the level-set zero inside (the point is placed at x =4.2).
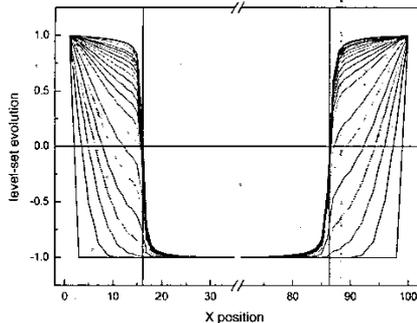

Figure 3 Temporal evolution of a 1D level-set.

6. Implementational issues

The new fluid dynamic model for the level set evolution has been implemented in 2D and in 3D. One remarkable feature of the algorithm is its speed. This is proved by the rendering times for some well-known data sets, which are reported in Table 1 for several resolution levels. Various implementational solutions have been adopted in order to boost the performance of the proposed algorithm. For example, we perform a pre-computation of the speed vector field for each voxel and we update the volume following a spiral path. This way the updating process is oriented from the outside towards the inside following the evolution of the front.

As far as the 3D algorithm is concerned the updating is obtained moving from the most external box towards the central point of the space; the update is then performed in concentric boxes each side of which is updated with a spiral path.

In figure 5 there is the representation of the level sets for a circle made of 50 point with a radius of 30 points, The time required for the convergence is 3 s.
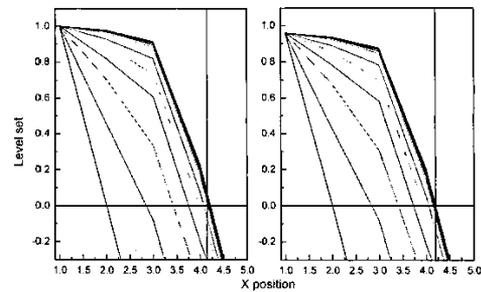

Figure 4: Undesired overflow of the external source (left) and its correction reducing the source strenght (right).
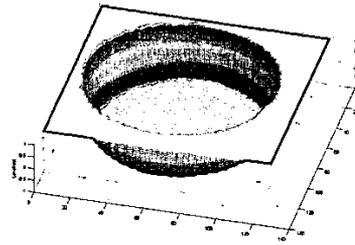

Figure 5 2D rendering of a circle (initial pixel-set of 50 pixels per side, radius of 30 pixels). Synthetic point data-set.

The level set evolution in 3D is reported for the different sets of points. In Table 1 we list the computational time required for rendering the sets of points represented in Figs. 6,7 and 8. The rendering was performed on a AMD Athlon™ XP at 2.1 GHz with 512 MB RAM under Windows™ 2000.

In figure 8 we tested the algorithm on a teapot dataset acquired with an image-based method from a real object. The resulting mesh is obtained by the marching cube algorithm. This data-set is interesting because it allows us to test the method on slightly noisy data.

Set	Resolution •	Points	Time (s)
Bunny	180	35780	40
Bunny	100	35780	4
Happy buddha	350	3836	105
Teapot	256	33061	110

Table 1 Computational time of the 3D algorithm, the resolution in the maximum number of points along the larger dimension of the object.

6. Implementational issues

In this paper we proposed a novel volumetric approach to surface modeling from unorganized sets of points, which is able to overcome the typical problem of computational efficiency that is typical of level-set methods. In addition, we gave the algorithm the ability

to model complex topologies by steering the front evolution using the medial axes of the data-set.

The results in terms of both computational efficiency and topological flexibility are very encouraging, and make the approach extremely usable.
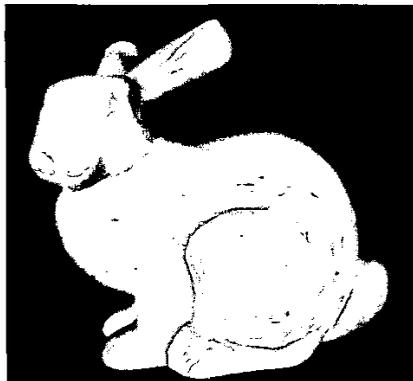

Figure 6: "Bunny" model (initial voxset of 180 voxels per side), obtained by wrapping a cloud of points.
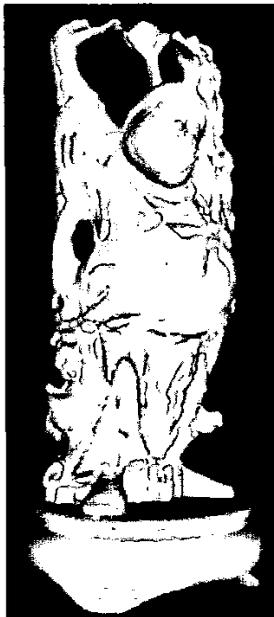

Figure 7: "Happy Buddha" model (initial voxset of 350 voxels per side) obtained by wrapping a cloud of points.

Figure 8: Teapot model (initial voxset of 110 voxels per side) obtained by wrapping a cloud of points. The data-set was acquired with an image-based method applied to a real object.

7. References

[1] J.A. Sethian, Level Set Methods and Fast Marching Methods Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science, Cambridge University Press, 1999

[2] H.Hoppe, T.DeRose, T.Duchamp, J.McDonald, W.Stuetzle. Surface Reconstruction from Unorganized points in SIGGRAPH'92 Proceedings, pages 71-78, July 1992

[3] H. Hoppe. Surface Reconstruction from Unorganized Points. Ph.D. Thesis, Computer Science and Engineering, University of Washington, 1994.

[4] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH '96 Proceedings*, pages 303-312, July 1996.

[5] H. Edelsbrunner, D.G. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane, *IEEE Transactions on Information Theory* 29:551-559, (1983).

[6] H. Edelsbrunner and E. P. Mücke. Three-dimensional Alpha Shapes. *ACM Transactions on Graphics* 13:43-72, 1994.

[7] C. Bajaj, F. Bernardini, and G. Xu. Automatic reconstruction of Surfaces and Scalar Fields from 3D Scans. *SIGGRAPH '95 Proceedings*, pages 109-118, July 1995.

[8] J-D. Boissonnat. Geometric structures for three-dimensional shape reconstruction, *ACM Transactions on Graphics* 3: 266-286, 1984.

[9] Nina Amenta, Marshall Bern and David Eppstein. The Crust and the Skeleton: Combinatorial Curve Reconstruction. To appear in *Graphical Models and Image Processing*.

[10] A. Sarti, S. Tubaro: "Image-Based Multiresolution Implicit Object Modeling". *J. on Applied Signal Processing*, Vol. 2002, No. 10, Oct. 2002, pp. 1053-1066.

[11] C.Hirsh, Numerical computation of internal and external flows, (Wiley interscience series in numerical methods in engineering;1) ISBN 0471917621