Volume construction from moving unorganised points

Claire Guilbaud, Annie Luciani, Ambroise Leclerc

Volume Construction from Moving Unorganised Points

Claire Guilbaud
INPG
Grenoble, France

Annie Luciani
INPG
Grenoble, France

Ambroise Leclerc
INPG
Grenoble, France

Abstract

We present a recursive method to construct a volume from particles generated by a physically based simulation of a flow field. The particles are scattered in the body of the simulated object giving no explicit information about its surface.

We show how to build a surface representation of the non-convex "hull" of a scattered points set. The surface is supposed to enclose all the points and have a minimal volume. The surface that we obtain must maintain the main characteristics of accurate geometrical singularities of the phenomenon during its evolution.

We have considered the case of toothpaste running out of a tube. Our method uses a density map to construct an enclosing volume of the generated particles, taking care of local singularities.

Keywords: *volume construction, particles models, visualisation.*

1. INTRODUCTION

Animation is a main topic of image synthesis. Since several years, scientists are looking for methods that ease production of realistic animations. These animations must show realistic collisions between objects and good animation of characters or natural phenomena.

Some years ago, animation was a series of pictures made by hand. To realize a movie, the work was hard and long, hence the necessity to automate the task. Three main methods have been developed: cinematic functions applied on geometrical objects, physically based models and artificial life algorithms. With the first one, we cannot easily obtain complex deformations such as in natural phenomena. With the others, we can perform very complex motions (displacements, deformations, transformations), but it is quite difficult to match complex shapes on these complex motions.

Usually, the physically based simulations of natural phenomena are displayed with points, lines and simple geometric objects. Therefore, to obtain a well and aesthetic visualisation, we must construct a volume, or extract a surface. In this volume, we must be able to show the pertinent morphological features of the modelled phenomenon.

When the movies produced by the simulation are visualised by points, human observers are able to recognize precisely the represented phenomenon: we have the feeling that the points are in the adequate volume (points are inside or at the borders of the fluid flow, if we modelled a fluid), even if we are not able to discern all the details.

2. MODELLING AND SIMULATION

2.1 Particle Systems

Many methods to simulate natural phenomena and deformable objects are based upon physically based particles systems. Physically based particles models are generic and simple. Moreover, all deformations and topological changes can be modelled. Terzopoulos and al. ([9]) linked particles with non-linear springs to simulate thermo-conductor behaviour, while G. Miller and A. Pearce ([7]) used Lennard-Jones forces between particles. In 1991, D. Tonnesen ([10]) designed a particle system in which the interaction law depends on the thermal energy. At the same time, the CORDIS-ANIMA system developed by A. Luciani and al. ([5]) implements generic non-linear interaction components to simulate unstructured materials for modelling natural phenomena such as granular materials ([5]) or fluids and smoke ([6]).

2.2 CORDIS-ANIMA: A Physical Modeller-Simulator ([2],[3])

We want to model a flow field using a physically based model. In this section, we explain how we can obtain a large variety of natural phenomena using the CORDIS-ANIMA library.

To construct a model with this method, we assemble a great number of very simple automata. These automata are divided in two types: mass elements and interaction elements. The input of the former is a force value and the output is two opposite forces. The first type is characterized by only one algorithm (Newton's law). In the second type, the elasticity and viscosity of the interaction function can be controlled by linear or piecewise linear memory less functions, or in the more general case by finite state automata.

All of CORDIS-ANIMA models are built by assembling these automata in networks in which the nodes are mass automata and arcs are interaction automata.

In a more simple way, we can see a model like a network of masses linked by viscoelastic elements.

2.3 A CORDIS-ANIMA Model of Generic Pastes

By assembling all the particles with an elementary interaction: a viscoelasticity link with a single threshold, we obtained granular material dynamics ([5]) and fluids dynamics ([6]).

We noticed that the dynamical behaviour of objects such as pastes, creams, foams seems to be an intermediate state between granular and fluids effects, we assume that it would be possible to

obtain granular, fluids and pastes effects with the same CORDIS-ANIMA model.

The CORDIS-ANIMA model of the toothpaste consists of:

- N masses linked together by two non-linear elementary interactions
- An elasticity K with its Ks threshold
- A viscosity Z with its Zs threshold

We obtained generic models of paste (toothpaste, creams, foams) with non-mixing piling and circumvolutions, if Ks << Zs, whatever K and Z are (see fig. 1).

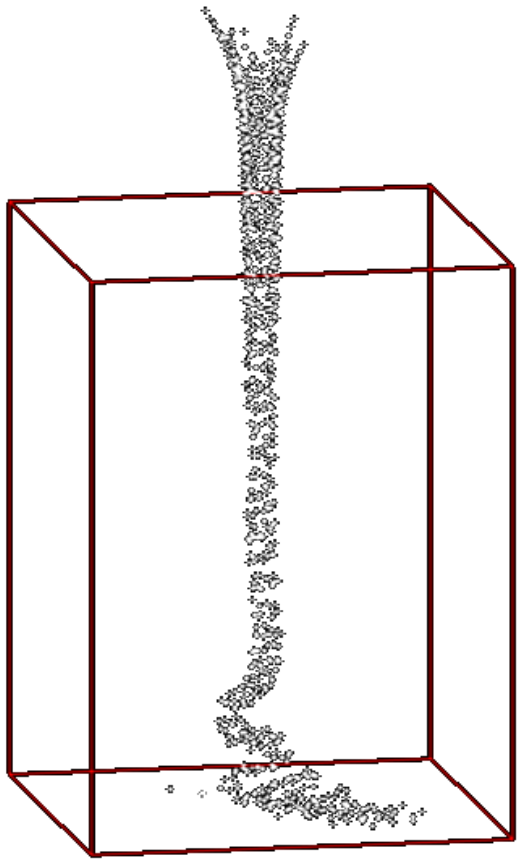


Figure 1.1 paste with N=900 masses (~400000 interactions)

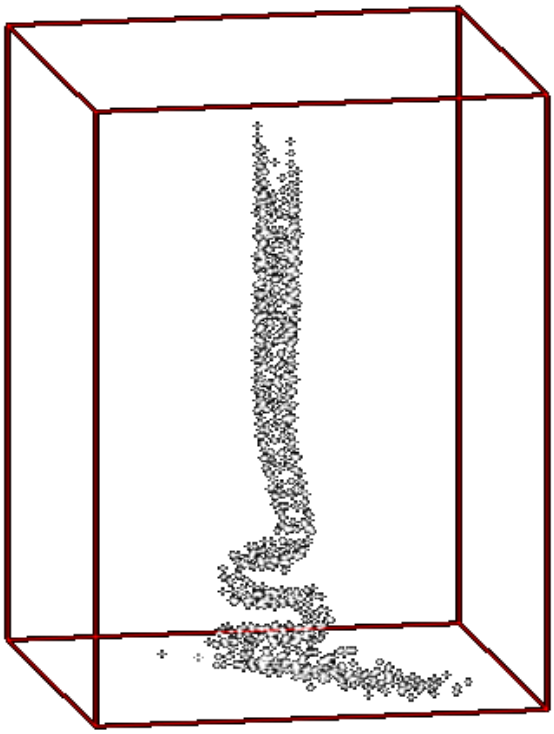


Figure 1.2 the same paste 4 seconds later

3. VISUALISATION OF PARTICLES

3.1 Related Works

The surface or volume representation of particles systems represents a big challenge. Since the discrete masses are scattered in space without any explicit information about the simulated object surface (figures 1.1 – 2.2), the visualisation algorithm must be able to extract all the surface related information in order to obtain a geometrically realistic image or animation.

The most used method is implicit surfaces ([1], [11]) with which we can obtain smooth surfaces. The implicit surfaces are often used to visualize highly deformable objects ([8], [9], [7], [10]). But because of the smoothness, this method is not able to render singularities such as breakpoints, dissymmetric fractures, bifurcations, forth order extremum, points of return, and their dynamical evolution (for example, evolution from forth order extremum to turn-back points). Then they cannot be used to display pastes behaviours without morphological pre-processing. The figures 2.1-2.2 show some singularities of the typical behaviour of paste flows.

The Engraved Screen ([6]) is a method that produces highly aesthetic visualization of 2D fluids. It is also an implicit method but thanks to the fact that the density shape is deformable according to the dynamics of the particles, it is able to render some refined features but remains limited to 2D visualisation.

Figure 2.1

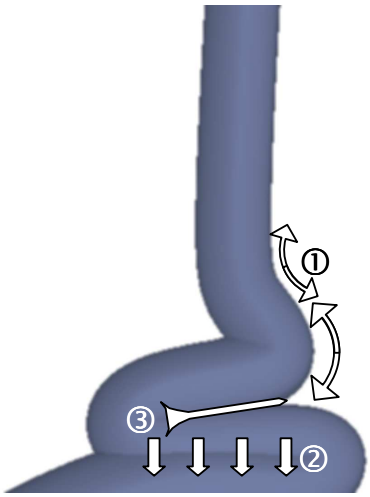
Figure 2.2

3.2 Geometrical Singularities

During the simulation, the toothpaste shows some evolving geometrical singularities that can't be efficiently rendered by classical methods.

For example, when the paste flow forms multiple layers, the contact between them shows some evolving singularities (cf. *Evolution of the geometrical singularities*):

- Simple contact between two layers (Figure 3 ③) (can't be constructed and rendered with implicit surfaces since such a method may blend the layers)
- Packing with fusion of the layers (Figure 3 ②) (which can't be rendered with generalised cylinders – implicit surfaces may produce an aesthetic (but not necessarily realistic) result)

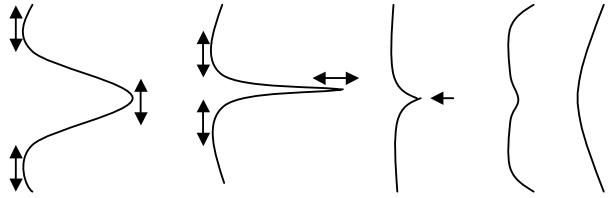


①point of inflection

②packing

③point of return

Figure 3



Evolution of the geometrical singularities

3.3 Volume Construction

To explain our method, we explain the terms used in our algorithm while we notice them for the first time.

To construct the volume, we chose a semi-automatic method in which the user remains master of the manner to construct the volume. The user specifies three parameters: the discrete size of the volume (number of voxels making up the volume), the neighbourhood size to consider to compute the density map (number of voxels that contains simulation's points in a given area), and the average number of voxels which separate two simulation's points in the discrete volume. The last parameter allows the algorithm to "construct" (fill in a voxel) in a pertinent region.

The volume construction algorithm is composed of three steps. We firstly discretised the space in a volume made up out of

voxels. This discrete volume encloses all simulation's points, which are all "put" in a voxel. A voxel that contains that kind of point is marked full. We worked in the discrete space.

Secondly, we compute the density map (which has the same size as the discrete volume) by using a convolution mask, where the closest neighbour is more important than the furthest. This step is in O(N), where N is the number of voxels in the volume.

Now, we have all the elements to construct the volume. We looked for connecting all the full voxels (voxels that contain simulation's points). Connecting two voxels is the same action as to fill in the voxels that separate them. We chose the voxels to fill in according to the density map. We had to do this step for each pair of simulation's points. To do that, we use a recursive method explained in the algorithm 1. The complexity is in O(N), where N represents the number of voxels in the discrete volume.

We said in previous paragraph that we chose the voxels to fill in according to the density map. In fact, in the density map, we were looking for the voxel with the highest density value. If we found several voxels with a highest density, we chose the closest compared to the position of the current_voxel and of the voxel with highest density. It is possible to have several voxels with the same density and the same distance. The distance between these two voxels is less or equal to the third parameter specified by the user.

With the highest density voxel, we determine the voxels to fill in. We fill in the voxels that lie on the discrete paths between the highest density voxel and the current_voxel (only the first voxel of the path are chose – see the figure below).

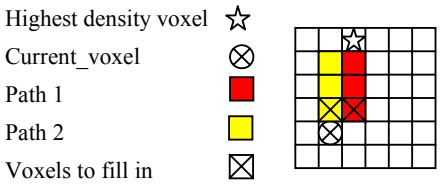


Figure 4: Determination of voxels to fill in

Algorithm 1: the volume reconstruction phase

For each point from data called `current_point` **do**

`current_voxel` = voxel which contains `current_point`

Detect(`current_voxel`)

End for

Detect(`current_voxel`)

For each region surrounding `current_voxel` **do**

With each of the highest density voxels

Determine the voxels to fill in (compared to the position of the highest density voxel)

For each `new_voxel` determinate by the previous step **do**

If the new_voxel is empty

Then fill in ; Detect(`new_voxel`)

End for

End for

3.4 Rendering Method

The constructed volume is made up out of voxels. The aim is not to obtain a smooth and aesthetic volume, but render the relevant volume in a simple way. We use the Marching Cubes method.

The Marching Cubes algorithm was designed by William E. Lorensen and Harvey E. Cline ([4]) to extract surface information from a 3D field of values. We explain the algorithm in 2d space. For the Marching Cubes algorithm to work we need to provide some basic information, the question, we will need to ask of out data in order to reconstruct the surface is "Is the point at (x,y,z) inside or outside of the object?".

The basic principle behind the Marching Cubes algorithm is to subdivide space into a series of small cubes. The algorithm then instructs us to "march" through each of the cubes testing the corner points and replacing the cubes with an appropriate set of polygons.

The first step is to calculate the corners that are inside the volume. We can now insert some vertices, since we know which points are inside and which are outside we can guess that a vertex should be positioned approximately halfway between an inside corner and any outside corners that are connected by the edge of a cell.

The method usually used in the Marching Cubes algorithm is not suitable for our method. For us, the space is already subdivided and we know only if a voxel is inside or outside the volume, but no if a voxel vertex is inside or outside the volume.

We have to find the conditions to determine if a voxel vertex is inside or outside the volume. We make a distinction between a full voxel and an empty voxel. The positions of full voxels compared to the current_voxel are important: a neighbour by the

faces is more important than a neighbour by the edges, which is itself more important than a neighbour by the vertices. From these observations, we defined conditions for knowing the "border" voxels.

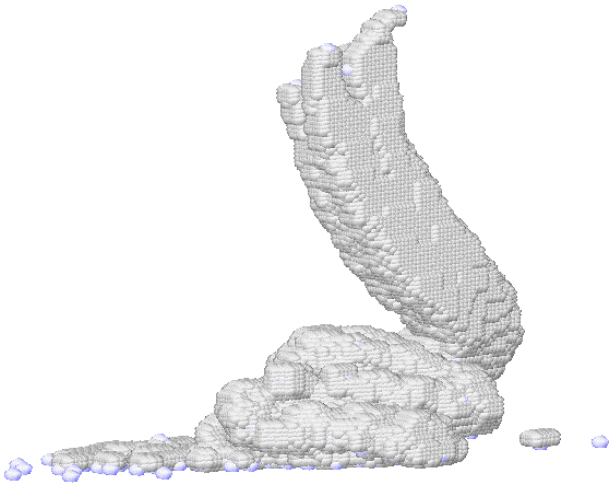


Figure 5.1 Constructed paste rendered with a simple voxel-sphere association

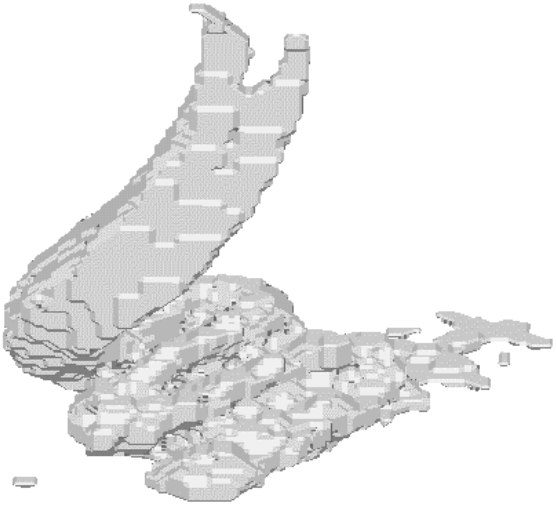


Figure 5.2 Constructed paste rendered with the marching cubes algorithm

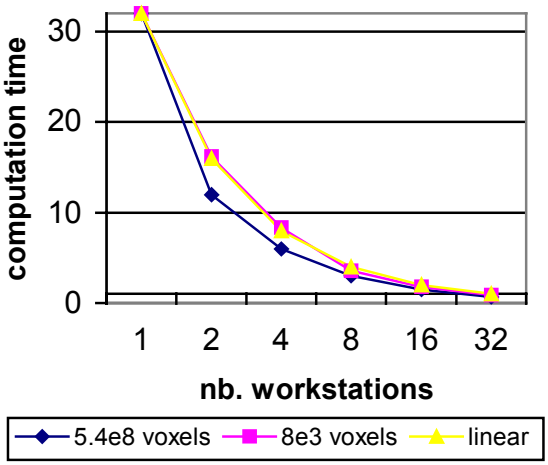
4. CONCLUSION

In this paper, we have presented a recursive method for the volume construction from particles generated by a physically based simulation of a flow field. Here our main focus has been to extract the pertinent morphological features of the toothpaste simulation.

The method is efficient to visualise scattered particles without information on the volume: we can display geometrical singularities such as points of return, points of inflection, packing, …

Being given that each element of the voxel space can be computed independently from the others, the parallel version of the algorithm is rather obvious to implement.

- On a multiple-processor workstation, the parallel implementation of the algorithm shows an almost linear speedup.
- On a network of workstations with distributed shared memory, the speedup can be made super-linear, depending on the size of the voxel space.



The apparent super-linear acceleration is only due to the size of the voxel space. With one workstation, the voxel space can't fit entirely in memory. As the number of workstation increases, the ratio of local/non-local memory becomes better. An important speedup occurs when the used part of the voxel space can fit entirely in the local memory of the workstation.

5. REFERENCES

[1] J. Blinn. *A generalization of algebraic surfaces drawing*. ACM Transactions on Graphics, pages 235-256, July 1982.

[2] A. Luciani, S. Jimenez, C. Cadoz, J.L. Florens and O.Raoult. *Computational Physics: A modeler-simulator for animated physical objects*. Proceedings of Eurographics Conference'91, Elsevier Ed., 1991.

[3] A. Luciani, S. Jimenez, O. Raoult, C. Cadoz, and J.L. Florens. *An unified view of multiple behaviour, flexibility, plasticity and fractures: balls, bubbles and agglomerates*. Modeling in Computer Graphics, Springer Verlag Ed., pages 54-74, 1991.

[4] W. Lorensen and H. Cline. *Marching cubes: A high-resolution 3d surface construction algorithm*. Computer Graphics, 21(4):163--169, July 1987.

[5] A. Luciani, A. Habibi, and E. Manzotti. *A multi-scale physical model of granular materials*. Proceedings of Graphics Interface'95, 1995.

[6] A. Luciani, A. Habibi, A. Vapillon, and Y. Duroc. *A physical model of turbulent fluids*. Computer Animation and Simulation, Eurographics'95, pages 16--29, September 1995. Maastricht, The Netherlands.

[7] G. Miller and A. Pearce. *Globular dynamics: a connected particle system for animating viscous fluids*. Computers and Graphics, 13(3):305--309, 1989. Also in SIGGRAPH'89 Course notes number 30.

[8] J. Stam. *A general animation framework for gaseous phenomena*. ERCIM Research Reports ERCIM-01/97-R047, ERCIM, VTT, January 1997.

[9] D. Terzopoulos, J. Platt, and K. Fleisher. *Heating and melting deformable models (from goop to glop)*. Graphics Interface'89, pages 219-226, June 1989, London, Ontario.

[10] D. Tonnesen. *Modelling liquids and solids using thermal particles*. Graphics Interface'91, pages 255--262, June 1991. Calgary, AL

[11] G. Wyvill, C. McPheeters, and B. Wyvill. *Data structures for soft objects*. The Visual Computer, 2(4): 227--234, August 1986.

About the authors

Claire Guilbaud

INPG
46 av. Felix Viallet
38031, Grenoble Cedex, France
E-mail: **Claire.Guilbaud@imag.fr**

Annie Luciani

INPG
46 av. Felix Viallet
38031, Grenoble Cedex, France
33 4 76 57 48 48
E-mail: **Annie.Luciani@imag.fr**

Ambroise Leclerc

INPG
46 av. Felix Viallet
38031, Grenoble Cedex, France
E-mail: **Ambroise.Leclerc@imag.fr**