Oriented Particles: A Tool for Shape Memory Objects Modelling

Jean-Christophe Lombardo
Jean-Christophe.Lombardo@imag.fr

Claude Puech
Université Joseph Fourier, Grenoble
Claude.Puech@imag.fr

iMAGIS[†]/ IMAG & INRIA
BP 53, F-38041 Grenoble cedex 09, France
fax +33 76 44 66 75

Abstract

Originally introduced to render probabilistic objects, particle systems are now used in a wide range of applications. Oriented particle systems were designed as a dynamic modelling tool for describing 3D free-form objects. We propose a new method using oriented particle systems to dynamically simulate 3D *deformable* objects that may either come back to their initial shape or break during animations. This method can be used in a physically based animation system. We also propose a new more robust expression for the attraction/repulsion interaction between particles.

Résumé

Introduits en informatique graphique pour pallier aux lacunes de la géométrie classique, les systèmes de particules sont maintenant utilisés dans de nombreux types d'applications. Les systèmes de particules orientées ont été crées pour modéliser des surfaces tri-dimensionnelles de forme libre. Nous proposons une nouvelle méthode utilsant des systèmes de particules orientées pour simuler dynamiquement des objets déformables. Cette méthode peut être utilisée pour l'animation par ordinateur avec des modèles physiques. Nous présentons aussi une nouvelle expression pour les interactions d'attraction/répulsion utile pour tous les systèmes de particules (orientées ou non).

Keywords: particle systems, deformable models, dynamic simulation, computer animation

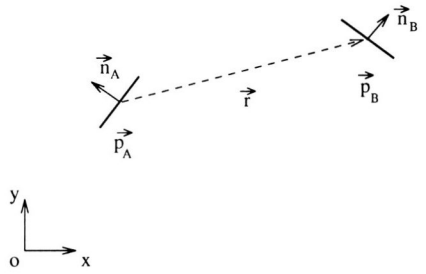
1 Introduction

Originally, particle systems were designed by W. T. Reeves to render fuzzy objects like fire [10] or trees and grass [11]. These first particle systems where initialized by stochastic processes and driven by a very rough approximation of the dynamic laws. There was no interaction between particles. Spectacular movies such as *Star Trek II: The Wrath of Khan* - scene of the *Genesis Demo* (June 1982 - Paramount) show the power of particle modelling. Further research focused on the control of movements [12, 15, 5], as well as physical simulation of fluids and deformable bodies [8, 14, 7, 6]. In 1992, Richard Szeliski and David Tonnesen presented another improvement of particle systems [13]. They used anisotropic basic elements, called *oriented particles*, to model complex 3D surfaces.

A *particle system* can be seen as a set of point masses called particles moving under external actions. Such a simple system can be improved in several ways. The first one is to add interactions between particles such as attraction/repulsion forces. These forces maintain a distance between two interacting particles by applying a repulsion force when particles are too close and an attraction force when they are too far away.

The paper proposes a new approach, based on oriented particles, for modeling and animating shape memory objects. The next section is an introduction to the notion of oriented particle. Section 3 presents a new attraction-repulsion force called *cohesion force*. This force gives a better control on the oscillations of (oriented or not) particle systems. Section 4 describes interaction laws, called *form interactions*, that impose shape constraints on the objects geometry. As a result, particles can be used to simulate 3D free-form deformable surfaces that recover their original shape after deformation. This is described in section 5. A different application is given in section 6 where form interactions are used to impose shape memory not to the surface of objects but to skeletons that define an implicit surface. This leads to a new kind of shape memory deformable objects based on oriented particles and implicit surfaces. The last section is devoted to ongoing research and future work.

[†]iMAGIS is a joint project between CNRS, INRIA, Institut National Polytechnique de Grenoble and Université Joseph Fourier



$$\vec{r} = \vec{P}_B - \vec{P}_A$$

Figure 1: Oriented particles - Definition

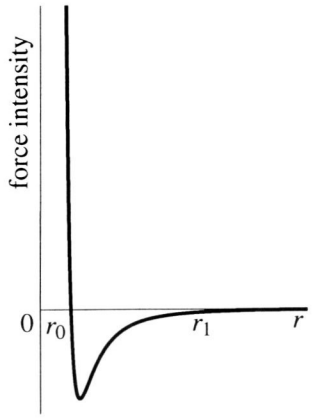


Figure 2: Lennard-Jones intermolecular force function

2 Oriented particles

Oriented particles prove to be useful for modelling 3D shapes. The following definitions are taken from [13] and will be used throughout the paper. The original paper [13] also describes some derived tools for modelling.

Besides classical mass and position, an oriented particle is defined by a normal vector[1] (Figure 1). Each particle can be considered as a surface element. The system obeys solid dynamic laws. Each interaction law is modelled by a force and a torque. Linear acceleration can be computed from applied forces and particle mass, and angular acceleration from applied torques and inertia matrix with the standard Newtonian equations of motion [1].

The cohesion of the set of particles is ensured by the use of a long-range attraction / short-range repulsion force. As in almost every interacting particle system, the force $\vec{\mathcal{F}}_{LJ}$ used by Szeliski and Tonnesen derives from the Lennard-Jones model of intermolecular potential function Φ_{LJ} [1] (Figure 2).

$$\begin{aligned} \Phi_{LJ}(r) &= \frac{\beta}{r^n} - \frac{\alpha}{r^m} \\ \vec{\mathcal{F}}_{LJ}(\vec{r}) &= -\vec{\text{grad}}(\Phi_{LJ}(r)) \end{aligned}$$

where r is the distance between the two interacting particles, α, β, n and m are constants[2]. With r less than r_0 this is a repulsion force, when r is equal to r_0 particles are in a rest state. With r between r_0 and r_1, the force applied is an attractive force. When r is greater than r_1, there is no interaction between the particles. r_0 is called the rest distance, and r_1 the influence radius.
This kind of force is isotropic; it depends only on the distance between interacting particles. The torque generated by this interaction law is identically null.

In order to force the particles to group themselves into surface-like shapes, a complex interaction law deriving from the weighted sum of three potentials is added. The force and torque derived from each potential try to place particles in specific relative positions. The geometric constraints enforced by the potentials emitted by particle A applied to B, as presented in [13] are the following:

- co-planarity: $\Phi_P = (\vec{n_A}.\vec{r})^2\psi(\|\vec{r}\|)$
  The force and the torque deriving from this potential act to place particles in the same plane.
- co-normality: $\Phi_N = \|\vec{n_A} - \vec{n_B}\|^2\psi(\|\vec{r}\|)$
  The co-normality potential has been added to the set of interaction forces to control twist.
- co-circularity: $\Phi_C = ((\vec{n_A} + \vec{n_B}).\vec{r})^2\psi(\|\vec{r}\|)$
  The co-circularity force and torque translate and rotate particles to place them along a circle.

The weighting function $\psi(r)$ is a monotone decreasing function used to limit the range of interactions. We refer to [13] for a discussion of these potentials.

3 Keeping the particle set cohesion

In particle systems, the standard attraction/repulsion force model is the Lennard-Jones one. This function issued from gas molecular dynamics, although being commonly used to model interactions between particles, presents two main drawbacks. First, the parameter set is not intuitive at all. It is quite hard to anticipate the results of any change. Secondly, the system oscillates. Attraction/repulsion force is a conservative force, so oscillation is a natural phenomenon. Damping is introduced to dissipate energy. Problems come from the amplitude of these oscillations and the use of a discrete integration of dynamic laws. Their summed effect leads to a slow convergence for the particle system. It can reach a rest state

[1] The system described in this section is three-dimensional, figures are drawn in 2D for a better understanding.

[2] We can find in [6] $\alpha = Er_0^{2n+1}/n$, $\beta = Er_0^{n+1}/n$ and $m = 2n$; where E is a scaling factor, r_0 is the rest distance and n remains a non intuitive constant.

only if the time step is very small. A Lennard-Jones function has a steep slope near the equilibrium position $r = r_0$. When discrete integration methods such as the Euler or the Newton-Cotes [9] technique are used, if the time step is too large, particles may need "infinite" time to reach an equilibrium state. For example, if at time t distance r between A and B is too small, a repulsion force is applied during next dt. At $t+dt$, r is greater than r_0 and an attraction force is applied. At $t+2dt$, particles are too close, and so on. If dt is much too large, the modulus of applied force grows and the object explodes. If dt is too large (the most often), it takes a long time for the system to reach its rest state. During this interval of time, particles movement may cause a change in neighborhood, thus a change in object behavior. A trivial method to limit oscillations is to give the medium in which particles evolve an important damping effect. This method does not give the expected results. Such a damping effect is global to the scene. Movements due to external forces are damped too. So external forces amplitude must be increased. Such a solution is not acceptable because manipulating high amplitude forces requires the use of a small time integration step. Another solution is to add a friction interaction between particles. It means that the damping effect depends on local particles density. Although giving better results than global damping, this approach of local damping isn't satisfactory. Theses approaches don't solve the oscillation problem, they try to mask it.

Defining a new attraction/repulsion force

As oscillations result from the steep slope near $r = r_0$, we define new attraction/repulsion forces with a null slope at equilibrium point. First of all, we must study what are the requirements for a function to model an attraction/repulsion force.

Mathematical definition

The mathematic properties that a function $\vec{\mathcal{F}}(\vec{r})$ should have to model an attraction-repulsion force are the following:

- $\vec{\mathcal{F}}(\vec{r})$ only depends on inter-particle distance. Its action is to move particles farther or closer to one another: $\vec{\mathcal{F}}(\vec{r}) = f(r)\vec{r}/r$ where $r = \|\vec{r}\|$
- two particles cannot be at the same location:
$$\lim_{r \to 0^+} f(r) = +\infty$$
- There is one and only one rest distance:
$$\exists! r_0, f(r_0) = 0$$
- A repulsive force is applied when particles are too close: $\forall r < r_0, f(r) > 0$

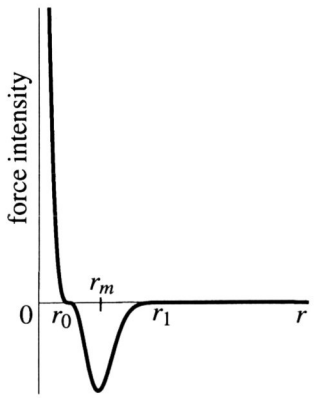


Figure 3: Cohesion force

- An attractive force is applied when particles are too distant: $\forall r > r_0, f(r) < 0$
- Repulsion forces grow as particles move nearer: $\forall r, 0 < r < r_0, f'(r) < 0$
- As particles move away from one another, attraction forces grow, reach a maximum and then decrease to be negligible as soon as distance becomes too large:
$$\exists r_m > r_0, f'(r_m) = 0$$
$$\forall r_0 < r < r_m, f'(r) < 0$$
$$\forall r > r_m, f'(r) > 0$$
$$\lim_{r \to +\infty} f(r) = 0$$

Cohesion force

We propose the function

$$f_{AR}(r) = E\frac{(r_0 - r)^3}{r} e^{-\alpha(r_0 - r)^2}$$

to model the interaction. It verify all the "natural" mathematical conditions required described above and have the desired property (Figure 3):

$$f'(r_0) = 0$$

Computing this expression (we call it *cohesion force*) at each time step is time consuming, but it can be easily tabulated. This force function is also conservative. Damping is used, but low coefficients are sufficient to limit oscillations. Experiments show that oscillations are drastically cut down.

The parameter set is more intuitive than Lennard-Jones one. r_0 appears explicitly. E is a scaling factor, as E gets larger the objects becomes stiffer. As in Lennard-Jones expression, parameters are global[3]. Parameter α

[3]Acting on a parameter modifies all the curve, so it is quite hard to adjust the parameters. It should be better to specify properties for each part of the curve.

is not easy to manipulate. Increasing α squashes all the curve. It allows to shorten the influence radius but the scaling factor must be used to restore the maximum attraction force.

Experiments and results

To compare cohesion and Lennard-Jones forces, a set of tests was performed. Two (non-oriented) particle systems are initialized with each type of force (with parameters chosen in order to get similar behaviors). Each system is given the same damping, each particle is given the same mass. Simulations are computed using the same time step. Particles are placed near their equilibrium position and released. While they reach the equilibrium state, oscillations are measured and plotted versus time in both cases. Let's call $\vec{\delta x_i}$ the displacement of particle i during one time step. We choose:

$$O(t) = \sqrt{\sum_i (\|\vec{\delta x_i}\|^2) - \|\sum_i \vec{\delta x_i}\|^2}$$

as a measure of oscillations. $O(t)$ is a measure of the movement of the particles around the mass center of the system. Plots made for systems with various number of particles show that cohesion forces always make the system converge[4] faster than Lennard-Jones forces.

A robustness test was performed too, using systems composed of 3 particles placed on an equilateral triangle with side length equal to $p\%$ of the theoretical rest length. Simulations were run with p ranging from 100% to 60% (Figure 4). With a length below 60% of the rest length both systems diverge. We notice that cohesion forces are much more robust than Lennard-Jones ones. Systems initialized with Lennard-Jones forces diverge with an initial distance between particles equal to 80% of the rest length while cohesion forces lead to convergence for an initial length less than 60% of the rest length (Figure 4d). Cohesion forces in this extreme case are not worse than Lennard-Jones forces in a good case ($p = 95\%$).
During a simulation, except in case of collisions, shape changes are slow. It means that, if an object is in its rest state a time t, at time $t + dt$ it will be near its rest state. Top curves (Figures 4a and 4b) show that oscillations amplitude stays low.

4 Keeping the object form

We now turn to the main topic of this paper. Our goal is to simulate deformable objects. Theses objects have their own shape. An object, under a fair external force field[5], should be distorted. When the force field disappears, the object is expected to take back its original shape (or a "close" approximation of the latter).

The object shape is defined through local interactions of particles. The set of interaction laws given by R. Szeliski and D. Tonnesen is not suitable for simulating deformable objects. The main drawback is related to the co-circularity potential. This potential forces particles to be co-cyclic, but the circle isn't defined (the particles being oriented particles have an associated normal vector, so that two arbitrary particles are not trivially co-cyclic!). Consider two particles initially on a circle. If an external event, such as a collision, occurs, submitting only one of these particles to an external force field, this particle moves. Because of the co-circularity interaction law, both particles will move and rotate to be co-cyclic. When both particles get on a circle (or near a circle), there is no reason for this circle to be the same as the initial one. As we want to model memory shape objects, we want these objects to restore their initial shape after deformation. The constraint imposed by the co-circularity potential isn't strong enough. Thus, we must use a stronger one. A good potential is a potential with only a few minima corresponding to a small number of particles relative positions.

A way to define the geometry of a surface is to give the local curvature of each point. This information is sufficient for rebuilding the object. So we define an interaction which acts to maintain this local curvature. An easy way to obtain adequate forces and torques is to derive them from an adequate potential. Such a potential should be minimal at rest. Any distance function between an "arbitrary" state and the rest state is a candidate. We now describe our choice.

The inverse of the radius of the tangent sphere is a measure of the curvature. In order to "restore" the object we try to restore this radius. An oriented particle A and a radius R define an unique circle. The center of this circle is the point $\vec{P_A} - R\vec{n_A}$. Thus, two particles A and B are on the same circle of radius R if and only if: $\vec{P_A} - R\vec{n_A} = \vec{P_B} - R\vec{n_B}$ or, equivalently, if and only if: $\|\vec{r} + R(\vec{n_A} - \vec{n_B})\| = 0$. To avoid computing a square root at each time step, we choose

$$\Phi_F = \|\vec{r} + R(\vec{n_A} - \vec{n_B})\|^2$$

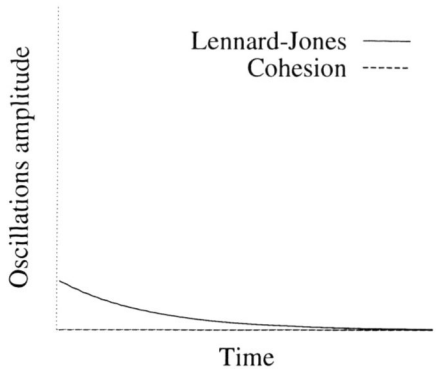
for the geometrical part of the potential, and get the following interactions laws:

$$\begin{aligned} \vec{\tau_F} &= 2R[\vec{n_A} \times (\vec{r} - R\vec{n_B})]\Psi(\|\vec{r}\|) \\ \vec{\mathcal{F}_F} &= 2[\vec{r} + R(\vec{n_A} - \vec{n_B})]\Psi(\|\vec{r}\|) \end{aligned}$$

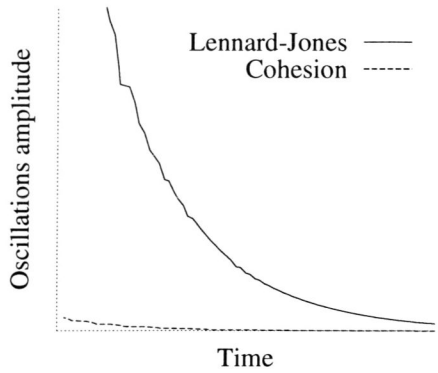
Applied torque and force are respectively computed by multiplying $\vec{\tau_F}$ and $\vec{\mathcal{F}_F}$ by a decreasing function of the dis-

[4] We say that a system converges when it reaches an equilibrium state. We say that it diverges when the object explode under the action of too large interaction forces.

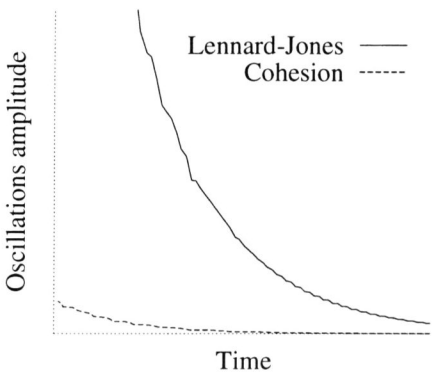
[5] Obviously, due to particle model, unreasonable force will cause uncontrolled permanent deformation or even object explosion.



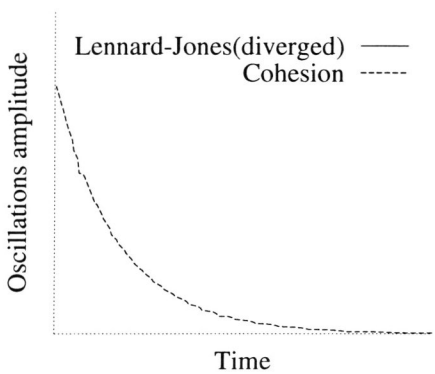
a) length = 99% of the rest length



b) length = 90% of the rest length



c) length = 85% of the rest length



d) length = 60% of the rest length

Figure 4: Distance between initial and rest states influence upon oscillation damping for both attraction forces

tance between particles. We call this interaction *form interaction*. $\vec{\mathcal{F}}_{\mathcal{F}}$ is null only at the surface of the sphere and elsewhere attracts particles towards the surface. $\vec{\tau}_F$ is null when the particle normal goes through the center of the sphere and elsewhere tries to enforce this property. So this interaction is null if particles are on the same sphere and else acts to place them on it.

5 Modelling shape memory surfaces

Cohesion forces and form interaction allow modelling and simulation of free-form deformable 3D surfaces. Each particle is given an interaction law which is a weighted sum of all the interactions[6] described above. The particle simulation algorithm is a standard one:

```
As long as the simulation is running
  For each particle in the scene
    Compute interactions with each
      particle in the neighborhood
  For each particle
    Sum interactions and external actions
    Integrate dynamics laws to compute
      acceleration, speed and position
```

The only difference with a non-oriented particle system is that interaction involves torques besides forces. Thus, acceleration, speed and position have a linear and a rotational component.

We introduce a typing mechanism to model more complex behaviors, with two kinds of particle types (*hinge* and *standard*). We introduce also two kinds of interactions: a *complete interaction* computed with an interaction law composed of a weighted sum of all the interactions previously described in this paper, and a *repulsion interaction* modelled with an exponential function of the distance. To handle this, we use a set of simple rules:

- a hinge has a complete interaction with every standard particle.

[6] To make them roughly scale independent, we use normalized interactions. ϕ_P and ϕ_C are divided by $\|\vec{r}\|^2$ and ϕ_F by $(R\|\vec{r}\|)^2$.

- a standard particle has a complete interaction with all instances of the same standard type and with hinges.
- particles with no complete interaction repel themselves.

This feature allows to subdivide the object into as many independents parts (with a standard type per part) as needed and connect them with hinges.
As in most particle systems that model deformable objects, we introduce another rule to specify object membership:

- a particle interacts (as described above) with each particle belonging to the same object and repels others.

The use of such a rule allows detection and treatment of collisions between objects at no extra cost.

Szeliski and Tonnesen oriented particles define only a normal and a tangent plane. We choose to model an oriented particle as a complete referential, with an origin position (particle position) and three axes (particle orientation). The normal vector $\vec{n}$ used in interactions is an arbitrary normed vector in particle referential. In this way we can model more complex behaviors and avoid using twist control interaction.

Figure 5 shows a spiral modelled with ten oriented particles. Each particle's interaction law is a weighted sum of cohesion, form, co-planarity, and friction interactions. Each particle has a specific local curvature (used for the form interaction) that grows as the particle gets closer to the inside of the spiral. The co-planarity interaction is used to keep all the particles in the same plane. Figure 5a) shows the object in its rest state. Opposing external forces are applied to the two extremity of the spiral. Their action is to uncoil it (Figure 5b and Figure 5c), until interactions between particles (due to their relatives positions) generate compensating internal forces (Figure 5d). This state is a rest state. The sum of external plus internal forces is zero. During the second phase, external forces are released. The spiral progressively recovers its shape (Figure 5e and Figure 5f).

6 Modelling implicit objects with shape memory skeleton

Implicit surfaces have been used by Marie-Paule Gascuel to model deformable objects [4]. An object is defined by a set of skeletons and a set of associated potential functions. The object surface is an isosurface of the sum of all emitted potentials. The slope of the potential function around this constant defines the stiffness of the object. Each implicit surface is sampled. Collision detection is computed by testing sample points of an implicit surface against the potential emitted by the other object. Potentials are decreasing functions of the distance of the emitting skeleton. Therefore it is straightforward to know if a point is inside or outside the object. If the value of the potential in the tested point is higher than the considered constant, the point is inside the object, otherwise it is outside. Exact contact surfaces are modelled by adding negative terms to the potentials. The exact contact surfaces allows to compute reaction forces in the collision area. These reaction forces are then expressed at the skeleton center of mass as a pair (force, torque) and integrated during the following time step.

Recent work uses a non-oriented particle system as skeleton for this kind of object to model highly deformable objects. As shown in [2], using a particle system as a set of skeletons for implicit objects means being able to dispatch reaction forces between particles, being able to re-sample efficiently implicit surfaces after a change of topology.

We propose the use of an oriented particles system to handle skeletons. This allows modelling a new kind of objects: implicit defined deformable objects with *shape memory skeletons*. Such objects combine features of the two models: shape memory and adaptable topology, precise contact processing and high quality rendering. As oriented particles are referentials, using them to handle skeletons of the implicit surface allows many improvements in this kind of modelling. Anisotropic potential functions can be used instead of distance functions used in [4, 2]. Each skeleton can be a complex object defined as a collection of primitives placed in the oriented particle referential.

Figure 6 shows a simulation of a collision between implicitly defined objects. The ball is a deformable object [4]. To each particle of the spiral from Figure 5 we attached a square skeleton. Deformations of the resultant implicit surface are only due to the movements of the skeletons. Initially the spiral is in a rest state an the ball is given a linear speed to collide the spiral. Collision deforms the spiral which recovers its original shape.

7 Conclusions and future work

As the implemented system was designed to test whether dynamic simulation with oriented particles is feasible or not, there was no optimization (neither in time nor in memory) done up to now. However, it is important to note that simulations are computed at interactive rate (ratio between computing and simulation time is about 3). This feature is important for such a tool. Modifications on parameters can be validated immediately by their influence on simulation. New animations can easily be designed.

Let us emphasize a few advantages of the proposed methods.

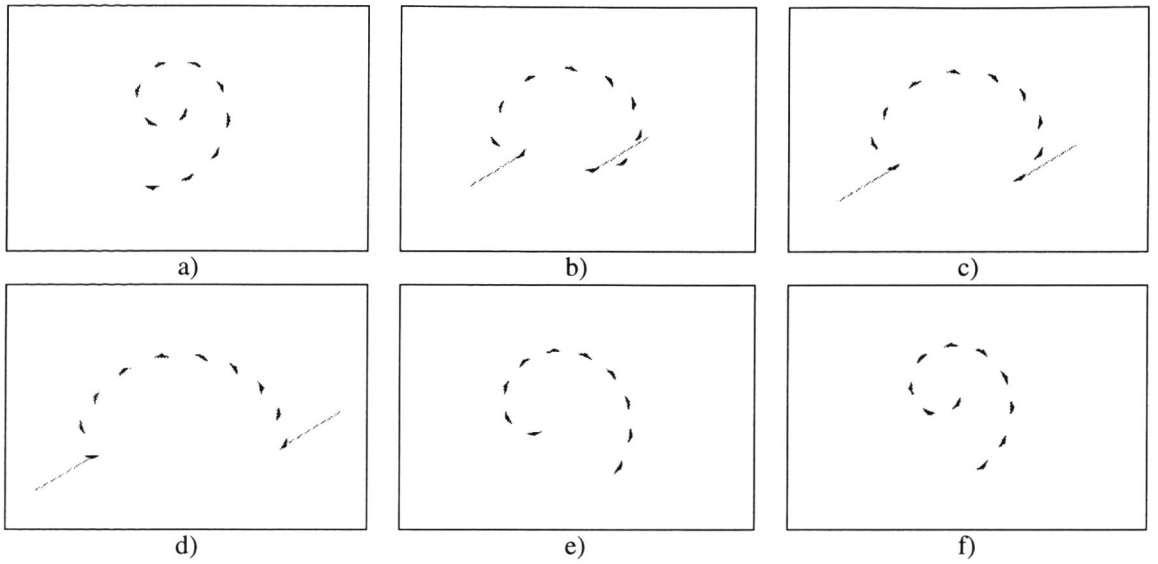


Figure 5: Recovering shape after a distortion

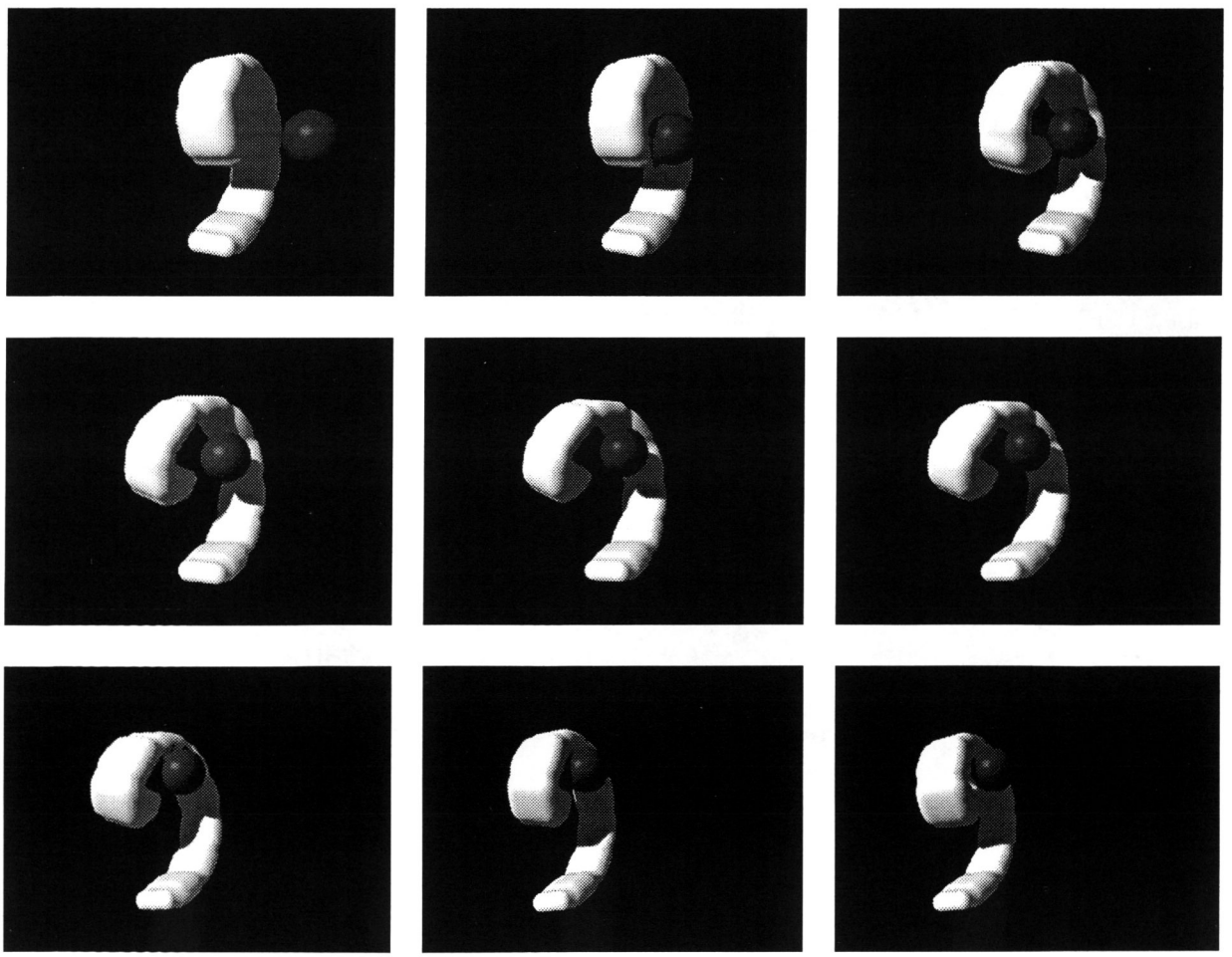


Figure 6: 3D simulation with implicit surfaces

- As with all particle-based modelling systems, the topology of the modeled objects can very easily change through time.
- Representing objects by a discrete sample of their surfaces can lead to significant savings in memory.
- The use of oriented particles to model shape memory skeletons as presented in the last section is new and offers a wide range of modeling new options.

A 2D implementation was done with a X windows interface. A 3D implementation was done within *Fabule* [3], a dynamic animation system developed in our research group.

The use of oriented particle systems to automatically reconstruct and simulate deformable objects from 3D data such as data from medical imagery is currently being investigated. Further work to be done includes:

- Re-writing the cohesion force to obtain a scale independent interaction. This will allow to write a cohesion force depending on the direction of the interacting particle.
- Writing a new form interaction with curvature function of direction of the interacting particle. This feature will allow to reconstruct object with less error.
- Finally, using oriented particles to handle skeletons of an implicitly defined object allows to define anisotropic potentials.

Acknowledgements

We wish to thank M.-P. Gascuel for discussions about implicit modelling of objects, M. Desbrun for allowing us to use his work on interface between particle systems and implicit objects, J.-D. Gascuel and N. Tsingos who helped for the integration of oriented particles within *Fabule*. Our thanks also go to M. Depey for carefully re-reading this paper.

References

[1] Marcello Alonso and Edward J. Finn. *Fundamental University Physics*, volume 1. Addison-Wesley, 1967.

[2] Mathieu Desbrun and Marie-Paule Gascuel. Highly deformable material for animation and collision processing. In *5th Eurographics Workshop on Animation and Simulation*, Oslo, Norway, September 1994.

[3] Jean-Dominique Gascuel. Fabule: une environnement de recherche pour l'animation et la simulation. In *Troisième séminaire du groupe de travail Français "Animation et simulation" - Les simulateurs*, pages 31–40, Lille, France, October 1994.

[4] Marie-Paule Gascuel. An implicit formulation for precise contact modeling between flexible solids. *Computer Graphics*, pages 313–320, August 1993. Proceedings of SIGGRAPH'93 (Anaheim, California, August 1993).

[5] David Haumann, Jakub Wejchert, Kavi Arya, Bob Bacon, Al Khorasani, Alan Norton, and Paula Sweeney. An application of motion design and control for physically-based animation. *ACM SIGGRAPH'92 Course Notes*, 16:22–29, July 1992.

[6] Stéphane Jimenez. *Modélisation et simulation physique d'objets volumiques déformables complexes*. PhD thesis, Institut National Polytechnique de Grenoble, 1993.

[7] Annie Luciani, Stéphane Jimenez, Olivier Raoult, Claude Cadoz, and Jean-Loup Florens. An unified view of multitude behaviour, flexibility, plasticity, and fractures: balls, bubbles and agglomerates. In *IFIP WG 5.10 Working Conference*, Tokyo, Japan, April 1991.

[8] Gavin Miller and Andrew Pearce. Globular dynamics: A connected particle system for animating viscous fluids. *SIGGRAPH '89 Courses 30 notes*, pages 305–309, August 89.

[9] W. H. Press and al. *Numerical Recipes in C*. Cambridge University Press, second edition, 1992.

[10] William T. Reeves. Particle systems - a technique for modelling a class of fuzzy objects. *Computer Graphics*, 17(3), July 1983.

[11] William T. Reeves and Ricki Blau. Approximate and probabilistic algorithms for shading and rendering structured particles systems. *Computer Graphics*, 19(3), July 1985.

[12] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *Computer Graphics*, 21(4), July 1987.

[13] Richard Szelisky and David Tonnesen. Surface modelling with oriented particle systems. *Computer Graphics*, 26(2):185–194, July 1992.

[14] Demetri Terzopoulos and Kurt Fleischer. Modeling inelastic deformations: Viscoelasticity, plasticity, fracture. *Computer Graphics*, 22(4):269–278, August 1988.

[15] Jakub Wejchert and David Haumann. Animation aerodynamics. *Computer Graphics*, 25(4):19–22, July 1991.