# Particle-based viscoplastic fluid/solid simulation

Afonso Paiva, Fabiano Petronetto, Thomas Lewiner and Geovan Tavares

Department of Mathematics — Pontifícia Universidade Católica — Rio de Janeiro — Brazil
{apneto, fbipetro, tomlew, tavares}@mat.puc--rio.br.

**Abstract.** Simulations of viscoplastic materials are traditionally governed by continuum mechanics. The viscous behavior is typically modeled as an internal force, defined by diverse quantities. This work introduces a fluid model to simulate the viscoplastic effect of solid materials, such as plastic, wax, clay and polymer. Our method consists in modeling a solid object through a non-Newtonian fluid with high viscosity. This fluid simulation uses the Smoothed Particle Hydrodynamics method and the viscosity is formulated by using the General Newtonian Fluid model. This model concentrates the viscoplasticity in a single parameter. Our results show clear effects of creep, melting, hardening and flowing.

**Keywords:** ***Viscoplastic fluid. Solid deformation. Smoothed Particle Hydrodynamics. Non–Newtonian Fluid. Heat Equation. Physically Based Animation. Computational Fluid Dynamics.***

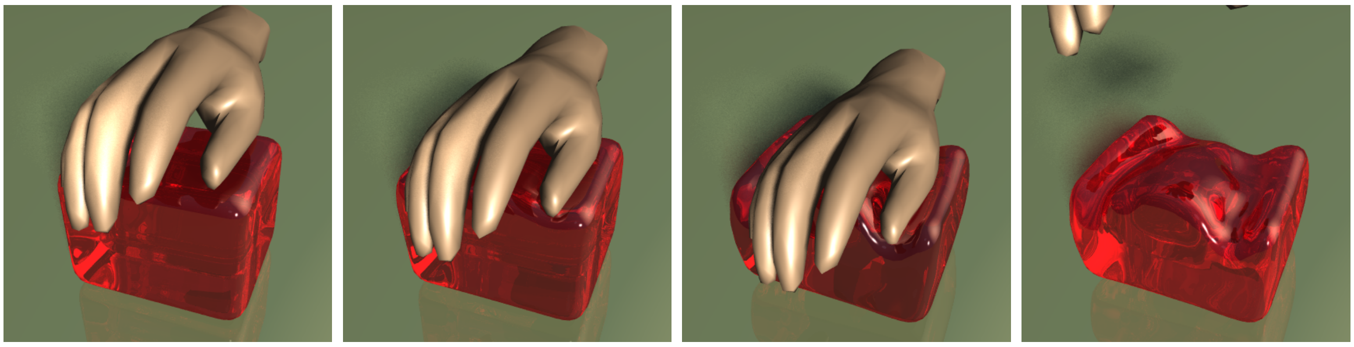


**Figure 1:** *Flow behavior of viscoplastic material.*

## 1 Introduction

Physical simulations entered the Computer Graphics community to produce visually realistic animations. In particular, fluid objects are frequently used for sensational effects, such as water, fire or gas evolutions, large continuous deformations, suspenseful object ruptures among many others. However, the delicate point for animation purposes remains formulating the underlying physics simple enough to produce a sequence efficiently, and rich enough for the simulation to remain realistic. Moreover, an animation artist should be able to control material behavior through a reduced set of intuitive parameters.

This involves finding an adequate formulation for the physical laws, which are usually formulated through differential equations, and a stable numerical approximation scheme for their discretization. Furthermore, the main parameters can be freely set while always generating a visually realistic simulation. This paper proposes a simplified formulation for viscoplastic objects simulation (see the example of Figure 1) by keeping the physical intuition behind a single parameter and avoiding the artist to cope with technical parts.

***Viscoplastic materials.*** The viscoplastic objects are non-Newtonian *fluids*, i.e. their flow behavior is viscous when weak forces are applied. However,when under significant forces, the material starts to flow like a liquid (toothpaste effect): it changes phase from non-Newtonian viscous but almost solid behavior to Newtonian liquid. Unlike viscoelastic objects, viscoplastic materials have no memory: a rupture does not induce a “spring effect” of any part. This work is based on the recent advances of Mendes *et al.* [17] in formulating a viscosity function for viscoplastic objects that encompasses both viscous and liquid phases. The conciseness and generality of this formulation provides an efficient control of the viscosity, since it mainly depends on one physical parameter, named the *jump number*, which replaces multiple parameters, such as stiffness, compressibility, plasticity, viscosity, cohesion… This rheological model suits better for plastic deformation and it has been recently introduced in

Preprint MAT. 28/06, communicated on November 19th, 2006 to the Department of Mathematics, Pontifícia Universidade Católica — Rio de Janeiro, Brazil. The corresponding work was published in Computer-Aided Design, volume 41, number 4, pp. 306-314, Elsevier 2009.

melting simulation [27].

Previous techniques for animating viscoplastic objects rely on continuous mechanics formulation, although those materials are essentially fluids. The approach of this work follows directly their physical models, which allows a realistic behavior for almost any material parameter. The jump number can be set to an arbitrary value, generating creep and work hardening effects. It can also be linked to other variable parameters such as temperature. This allows straightforward simulations of freezing and melting behaviors, when parts of the objects alternate continuously between solid and liquid phases, or lava flow simulations. This occurs discontinuously, depending on the terrain's topography.

***Computational approximation.*** Simulating the fluid behavior of a viscoplastic object in its liquid phase requires a computational fluid dynamics (CFD) framework. In the Computer Graphics literature, the most common CFD model relies on Eulerian formulation where physical quantities are sampled on a regular grid. This suits well for classical Newtonian fluids like water for instance. However, in order to control grid-based methods, the fluid free surface must be tracked, which remains a laborious task in free flow simulations.

In this work, we use a Lagrangian formulation on a particle-based representation, called *Smoothed Particle Hydrodynamics* (SPH). The SPH method was introduced in 1977 by Gingold and Monaghan [18] and Lucy [15] simulate compressible fluids in astrophysics. Each particle represents a small volume of fluid subjected to natural forces such as gravity, pressure and viscosity. SPH fluid frameworks are simple to implement, easy to capture the fluid free surface and its accuracy compares nicely to grid-based methods in several instances. Recently SPH simulation became very popular in the special effects industry where it was used for the water simulations in *Poseidon* and *300* (`http://www.nextlimit.com`).

## Related works

This work uses a variation of the SPH framework to simulate viscoplastic objects as a non-Newtonian fluid. We will quickly summarize the most relevant works on these three topics.

***Smoothed Particle Hydrodynamics.*** The SPH method was introduced in the Computer Graphics community by Desbrun and Cani in [8], where they used a simplified version of SPH to simulate deformable bodies. Stora *et al.* [28] proposed a simplified model for lava flow, using only artificial viscosity instead of a physical model for viscous effects and the terrain is defined by a digital elevation model and the collision between particles and terrain reduces only the computation of the terrain's altitude at the particle's projection onto the horizontal plane. Müller *et al.* [24] proposed a SPH approach for simulating fluids with surface tension using a spike kernel exclusively to compute the pressure force and avoiding particle inter-penetration problem, which may result in stable clusters of particles. Furthermore, they introduced point-splatting to capture the fluid free surface. In several applications using SPH, the wall boundary of the scene model (e.g. terrains or walls) are modeled by using virtual boundary particles [20], which impose repulsive forces on the fluid particles. Although it is elegant, this method increases the particle resolution, computationally intensive and it requires high memory storage. Müller *et al.* in [23] created a method which SPH fluid particles interact with deformable object represented by a triangle mesh. The main contribution of their paper is to place temporary boundary particle onto each triangle according to Gaussian quadrature rules. In order to avoid the use of boundary particles, Harada *et al.* in [3] developed a method to simulate particle-based fluid flows in complex mesh models by using additional repulsive forces in each term of the SPH fluid equations. These repulsive forces are computed by wall weight distance functions between particles and the mesh polygons, which can lead to small particle penetration into the mesh.

***Non-Newtonian fluids.*** There are few works in Computer Graphics for non-Newtonian fluids. Goktekin *et al.* [9] proposed a grid-based method to compute the stress tensor of these fluids. They use a linear Maxwell model with von Misses plastic yield condition. Clavet *et al.* [7] used SPH with a linear combination of elastic springs between particles driven plastic yield condition. Another method using SPH was introduced by Mao and Yang [16], where the stress tensor derives from a co-rotational Maxwell model.

***Plastic deformation.*** Plastic deformations are usually simulated by using finite elements [6, 14]. O'Brien *et al.* [26] simulated several materials with a simple plasticity model. However, the numerical calculation becomes ill-conditioned when the material undergoes substantial elastic or plastic deformation, which results in unstable simulations. Bargteil *et al.* [5] proposed a remeshing based Delaunay triangulation improving the quality of the tetrahedral elements guaranteed a stable simulation. Terzopoulos *et al.* [29] proposed a melting model based on particles. Müller *et al.* [25] created a point-based framework to simulate elastoplastic objects using continuum mechanics model with von Misses plastic yield condition. They used moving least squares (MLS) for approximating the velocity tensor field. Keiser *et al.* in [11] replaced MLS approach by SPH method. Finally, Solenthaler *et al.* in [4] introduced rigid body dynamics in the previous framework. However, their interaction model may also allow particle penetration in solids when strong forces are involved.

The models used in these works involve many parameters for the plasticity, which turns the viscosity control delicate.

## Contributions

This paper proposes a new meshless animation framework to simulate viscoplastic materials through a simplified physical formulation. This formulation is based on the viscous model for General Newtonian Fluid model of [17], allowing a proper fluid representation of the materials as opposed to usual solid mechanics approximations. The main contributions are:

- *Conciseness and efficiency*: our framework concentrates the viscoplasticity in mainly one parameter, which represents explicitly the dependency to the stress tension applied. This provides a very intuitive control of the material behavior, producing a wide variety of realistic animations that illustrate the generality of the proposed method in a simple and efficient manner.
- *Complex solid-fluid interactions*: we implement an explicit (geometric) collision handling to simulate fluid interactions with complex scene models. Unlike the previous works, the collision test is made directly involving particles (spheres) and triangles of the scene model, avoiding particle penetration. Moreover, this allows to add physical properties to the scene, e.g. increasing their roughness.
- *Accuracy and stability*: to avoid unstable particle clusters formation in SPH frameworks, we use a particle velocity correction called XSPH *method* [19]. The numerical stability of our framework is improved by using a SPH version of artificial viscosity and an adaptive time step based on the Courant-Friedrichs-Lewy (CFL) condition.

The present work is an expanded version of [27], refining on the viscosity model and introducing an efficient collision detection and response. While [27] was limited to melting objects, the present contribution allows more complex simulations involving viscosity such as lava flow and complex fluid/solid interactions leading to large deformations.

**Paper outline**

We introduce the physical model of the proposed framework in the next section. Then, we give a brief overview of the SPH method in Section 3. Section 4 presents the numerical improvements of our animation framework. A wide variety of results and the details of the implementation are given in Section 5. Finally, we conclude the paper with a discussion of results and glimpse on future works.

## 2 Formulation of the physical laws

This section focuses on the three main aspects of our physical model: the fluid equations, the viscosity model and the phase transition. Computational fluid dynamics CFD aims at prediction fluid behavior through Navier-Stokes equations. These equations are commonly solved by using conventional Eulerian formulation with grid-based methods such as finite differences and finite elements. In this work, we chose an alternative method driven by the Lagrangian formulation. Lagrange's approach describes physical laws from the viewpoint of a moving particle, which suits perfectly for meshless methods such as SPH. The reader will find further details on CFD in Anderson's book [1].

In this work, the viscoplastic materials are modeled using Mendes' formulation for General Newtonian Fluid [17], which allowed accurate and efficient viscosity computations.

In our simulations of melt and lava flow, we approximate the variations of the rheological parameters to a linear function of the temperature, leading to a simple heat equation for governing the phase transition.

### (a) Lagrangian formulation

The governing equations of a viscoplastic fluid flow can be formulated by the following two equations which describes the conservation of the mass (equation (1)) and of the momentum (equation (2))

$$\frac{d\rho}{dt} = -\rho \nabla \cdot \mathbf{v} \tag{1}$$

$$\frac{d\mathbf{v}}{dt} = -\frac{1}{\rho}\nabla p + \frac{1}{\rho}\nabla \cdot \mathbf{S} + \mathbf{g} \tag{2}$$

where $t$ denotes the time, $\mathbf{v}$ the velocity vector, $\rho$ the density, $p$ the pressure, $\mathbf{g}$ the gravity acceleration vector and $\mathbf{S}$ the viscoplastic stress tensor field. This last term plays a fundamental role in viscoplastic simulation.

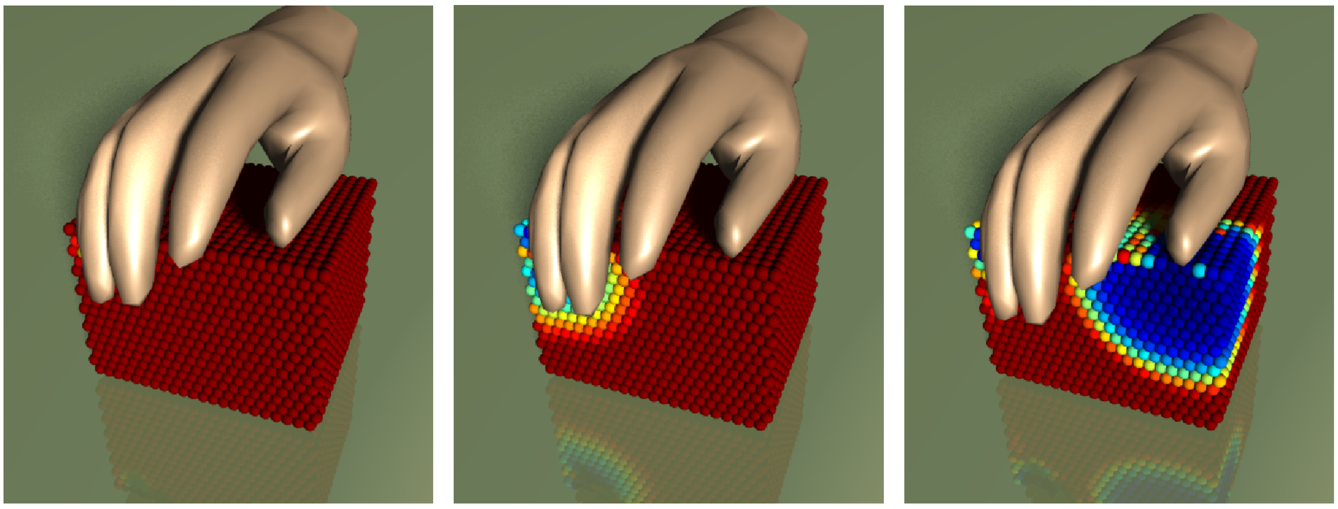
### (b) Generalized Newtonian fluid model

Non-Newtonian fluids are characterized by the non-linear dependence of the stress tensor with respect to the rate-of-deformation tensor: $\mathbf{D} = \nabla\mathbf{v} + (\nabla\mathbf{v})^T$. In this work, we use the Generalized Newtonian Fluid model proposed by Mendes *et al.* [17], described by:

$$\mathbf{S} = \eta(D)\,\mathbf{D}, \quad \text{with} \quad D = \sqrt{\tfrac{1}{2}\cdot \text{trace}\,(\mathbf{D})^2} \quad \text{and} \tag{3}$$

$$\eta(D) = (1 - \exp[-(J+1)D])\left(D^{n-1} + \frac{1}{D}\right). \tag{4}$$

This formulation models the viscosity $\eta$ as an exponential and as a power-law of the intensity of the rate-of-deformation tensor $D$. The exponential depends on a single, new rheological parameter $J$ called the *jump number*, which simplifies our viscosity model. For viscoplastic fluid, Mendes *et al.* concisely represent through $J$ several previous rheological parameters such as the yield stress, low shear rate viscosity and the consistency index. The jump number can reproduces dramatic drop of the viscosity in the yield stress (see Figure 2). It controls directly the fluid viscosity: increasing the value of $J$ intuitively increases the fluid viscosity (see Figure 3). The power-law index $n$ is responsible for the flow behavior: if $n < 1$, it predicts the effective viscosity decreases with increasing shear, characterizing viscoplastic behavior. In our simulation, we fixed the power-law index $n = 0.5$.



(a) 25 iterations (b) 890 iterations (c) 1225 iterations

**Figure 2:** *Generalized Newtonian Fluid model: the color map represents the viscosity of each particle, from low (blue) to high (red) viscosity. Note that the viscosity drop created by the force applied by the hand.*

### (c) Viscosity control and phase transition

Physical behavior become delicate to simulate when the properties of the material change, e.g. during phase transitions. In this work, we focus on viscosity transitions, which may be induced by temperature or high forces, e.g., in large deformations, melt and lava flow. Such transitions are hard to control with classical models such as used in previous works since the viscosity model involves many parameters. Using the model of the previous section, we model these transitions directly through the jump number.

Since no exact physical model has been stated yet for viscosity, we propose here a simple linear control of the jump number. For example for melting, the temperature of some part of the object increases until it reaches the melting point, where it becomes liquid (see Figure 4). The jump number $J$ must thus decrease with the temperature. We approximate this dependency through a linear variation with respect to the temperature:

$$J\,(T) = (1 - u(T))J_{max} - u(T)J_{min}$$

with $u(T) = (T - T_{min})/(T_{max} - T_{min})$. In the case of lava flows, the same coupling is used, but the temperature may often go above and below the melting point. In these specific cases, the variation of the temperature $T$ is described by the heat equation ($\frac{\partial T}{\partial t} = k\nabla^2 T$). The jump number can also be fixed to a constant even for large deformations, which provides a very concise and efficient interface for the user (see Figure 3).

## 3 Particle-based approximation scheme

This work uses the Smoothed Particle Hydrodynamics (SPH) framework to simulate the viscoplastic fluid behavior. In our animation framework, we use the complete formulation of SPH for weakly compressible fluid [20], extended by XSPH technique [19]. A wide description of SPH method can be found in Monaghan's survey [21].

### (a) SPH approximation

The main idea of SPH is to replace the fluid for a set of particles representing elements of the fluid (see Figure 2), which carry individual material properties. The dynamics of a Lagrangian viscoplastic fluid is due to the governing equations introduced in the last section. The properties in point $\mathbf{x}$ are determined by discrete convolutions of a kernel function with compact support as follows:

$$A(\mathbf{x}) = \sum_{j \in N(\mathbf{x})} A(\mathbf{x}_j)\frac{m_j}{\rho_j} W\left(\mathbf{x} - \mathbf{x}_j, h\right) \qquad (5)$$

$$\nabla A(\mathbf{x}) = \sum_{j \in N(\mathbf{x})} A(\mathbf{x}_j)\frac{m_j}{\rho_j} \nabla W\left(\mathbf{x} - \mathbf{x}_j, h\right)$$

where the set $N(\mathbf{x})$ contains all the particles at distance below $h$ from $\mathbf{x}$, $j$ is the particle index, $\mathbf{x}_j$ the particle position, $m_j$ the particle mass and $\rho_j$ the particle density. In

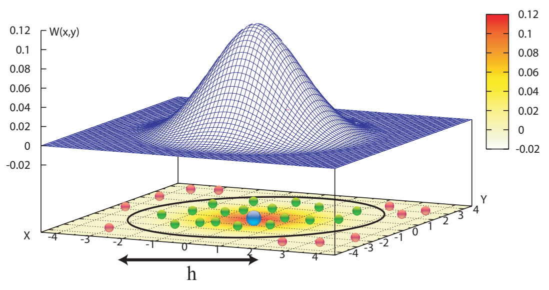


**Figure 5:** *Quartic smoothing kernel: the particles farther than the smoothing length h are discarded.*

this work, we choose a piecewise quartic smoothing kernel

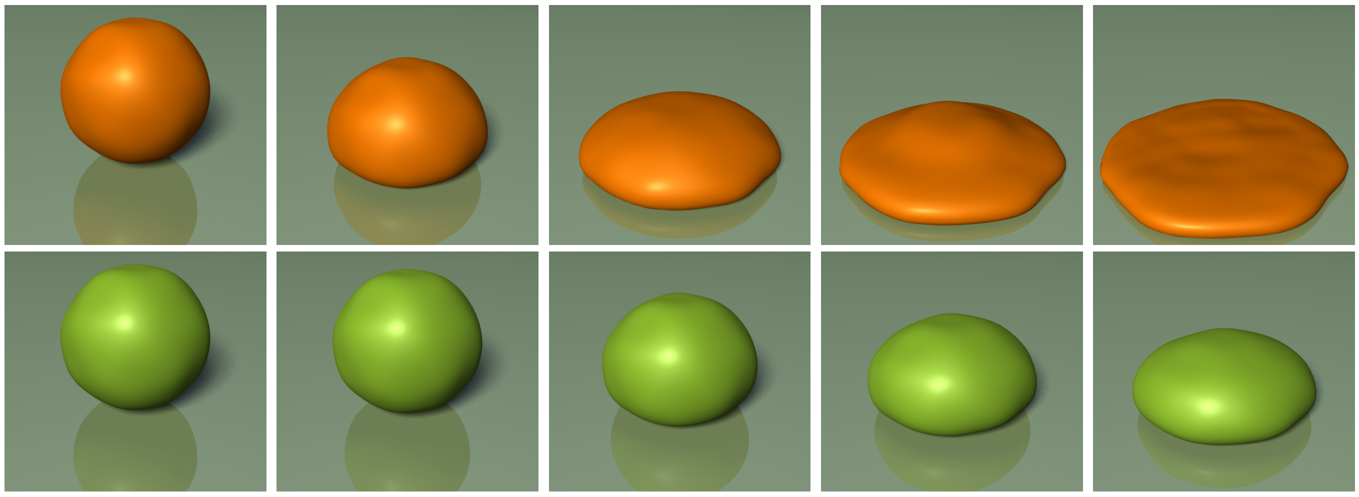


**Figure 3:** *Explicit control of the viscosity using jump number J: viscoplastic sphere model with 1200 particles, from low viscosity with* $J = 15$ *(top) to high viscosity with* $J = 150$ *(bottom).*

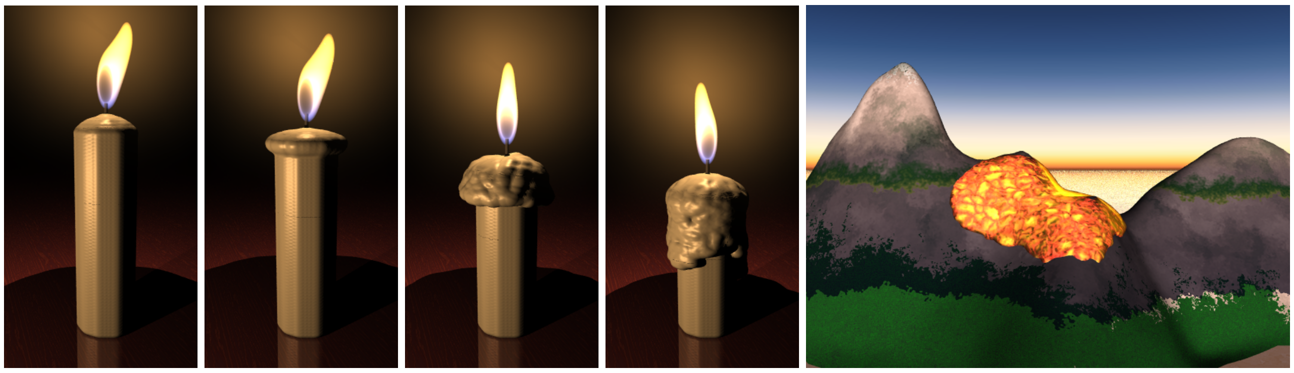


**Figure 4:** *Examples of viscoplastic materials: wax melting and lava flow at Rio's Sugar Loaf.*

function (see Figure 5) with a fixed smoothing length of twice the initial particle spacing.

$$W(\mathbf{x}-\mathbf{x}_j,h) = \frac{315}{208\ \pi\ h^3}\cdot w\left(\frac{\|\mathbf{x}-\mathbf{x}_j\|}{h}\right) \qquad \text{with}$$

$$w(q) = \begin{cases} \frac{2}{3} - \frac{9}{8}\,q^2 + \frac{19}{24}\,q^3 - \frac{5}{32}\,q^4 & ; \quad 0 \leq q \leq 2 \\ 0 & ; \quad q > 2 \end{cases}$$

Due to the similarity of the SPH kernel with the radial basis functions (RBF), the particle approximation can be improved and the smoothing length can be dispensed by the use of a RBF interpolant [2]. The reader can find a complete discussion about kernel functions in [13].

### (b) SPH approximation of momentum

Since SPH suits better for compressible fluid, we approximate the incompressible fluid by a weakly compressible fluid through an equation of state [20] for the pressure. However, modeling of pressure remains a delicate point for SPH simulations of incompressible fluids, due to the lack of explicit control of the local density. In this work, we use an equation of state proposed by Morris *et al.* [22]:

$$p_i = c^2\left(\rho_i - \rho_0\right) \tag{6}$$

where $p_i$ is the pressure at particle $i$, $c$ the speed of sound, which represents the fastest velocity of a wave propagation in that medium, and $\rho_0$ is a reference density. This equation of state is very similar to the ideal gas equation of state used by Desbrun and Cani [8]. In particular, when the constant of the gas is equal a $c^2$ we have the same equation.

After updating the pressure at all particles using equation (6), we can evaluate the pressure term in equation (2) at each particle:

$$\frac{1}{\rho_i}\nabla p_i = \sum_{j\in N(\mathbf{x}_i)} m_j\left(\frac{p_i}{\rho_i^2}+\frac{p_j}{\rho_j^2}\right)\nabla_i W\left(\mathbf{x}_{ij},h\right),$$

where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$.

In order to compute the stress tensor in equation (3), we first need to compute the SPH approximation for the velocity tensor field $\nabla\mathbf{v}_i$ at each particle:

$$\nabla\mathbf{v}_i = \sum_{j\in N(\mathbf{x}_i)} \frac{m_j}{\rho_j}\left(\mathbf{v}_j-\mathbf{v}_i\right)\otimes\nabla_i W\left(\mathbf{x}_{ij},h\right). \tag{7}$$

After that, we compute the particle rate-of-deformation tensor:

$$\mathbf{D}_i = \nabla \mathbf{v}_i + (\nabla \mathbf{v}_i)^T . \tag{8}$$

Finally, the stress tensor $\mathbf{S}_i$ is updated for each particle $i$ according to equation (4). The stress term in equation (2) is then approximated by:

$$\frac{1}{\rho_i} \nabla . \mathbf{S_i} = \sum_{j \in N(\mathbf{x}_i)} \frac{m_j}{\rho_i \rho_j} (\mathbf{S}_i + \mathbf{S}_j) \cdot \nabla_i W(\mathbf{x}_{ij}, h) .$$

### (c) SPH approximation of density

The SPH frameworks usually approximate density using the *density summation*, which follows directly from the SPH approximation of equation (5):

$$\rho_i = \sum_{j \in N(\mathbf{x}_i)} m_j W(\mathbf{x}_{ij}, h) .$$

Therefore, due to the particles deficiency in the fluid free surface the approximation above leads us to spurious results. We consequently chose another approximation for the density by using the following SPH version of *continuity equation* (1) which derives from Gauss theorem [20, 28]:

$$\frac{d\rho_i}{dt} = \rho_i \sum_{j \in N(\mathbf{x}_i)} \frac{m_j}{\rho_j} (\mathbf{v}_i - \mathbf{v}_j) \cdot \nabla_i W(\mathbf{x}_{ij}, h) , \tag{9}$$

where $\mathbf{v}_i$ and $\mathbf{v}_j$ are velocities at particles $i$ and $j$ respectively. However, we can avoid an extra loop to compute equation (9) by using equation (8) as follows:

$$\frac{d\rho_i}{dt} = \text{trace}\left(\tfrac{1}{2} \cdot \mathbf{D}_i\right) . \tag{10}$$

### (d) Particle velocity correction

In order to avoid particle inter-penetration problem without the use of several kernel functions, we computed a velocity correction for each particle using the XSPH method [19]. This method consists in computing an average velocity from the velocities of the neighboring particles. This maintains a more ordered move of particles in a weakly compressible flow (Figure 6) by only using the standard spline kernel.

In XSPH, each particle further moves according to a constant global parameter $\varepsilon \in [0, 1]$ in the following way:

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \varepsilon \sum_{j \in N(\mathbf{x}_i)} \frac{2 m_j}{\rho_i + \rho_j} (\mathbf{v}_j - \mathbf{v}_i) W(\mathbf{x}_{ij}, h) . \tag{11}$$

## 4 Numerical improvements

We can improve the numerical stability and accuracy of our framework by using some usual numerical techniques such as artificial viscosity and CFL condition in the SPH particle context.

**Algorithm 1** Particle dynamics

```
 1: repeat
 2:   for each particle $i$ do
 3:     Update $\nabla \mathbf{v}_i$ (equation 7)
 4:     Update $\mathbf{D}_i$ (equation 8)
 5:     Update derivative density (equation 10)
 6:     Update pressure $p_i$ (equation 6)
 7:     Update viscosity $\eta_i$ (equation 4)
 8:   end for
 9:   for each particle $i$ do
10:     Update acceleration (equations 2 and 12)
11:   end for
12:   for each particle $i$ do
13:     Update $\mathbf{v}_i$ and $\rho_i$ with Leap-Frog scheme
14:     Correct $\mathbf{v}_i$ with XSPH (equation 11)
15:     Eventual collision: $(\mathbf{x}_i, \mathbf{v}_i) \leftarrow \text{collision}(\mathbf{x}_i + \delta t \mathbf{v}_i)$
16:   end for
17:   Update $\delta t$ using CFL condition (equation 13)
18:   $time = time + \delta t$
19: until $time < time_{total}$
```

### (a) Artificial viscosity

In order to avoid numerical instabilities due to oscillations in the velocity vector field, a common technique adds an artificial viscous stress term in equation (2):

$$\frac{d\mathbf{v}_i}{dt} \leftarrow \frac{d\mathbf{v}_i}{dt} - \sum_{j \in N(\mathbf{x}_i)} m_j \Pi_{ij} \nabla_i W(\mathbf{x}_{ij}, h) . \tag{12}$$

The effect of artificial viscosity in SPH systems is given by:

$$\Pi_{ij} = \begin{cases} -\frac{2\alpha\mu_{ij} c}{\rho_i + \rho_j}, & (\mathbf{v}_i - \mathbf{v}_j) \cdot (\mathbf{x}_i - \mathbf{x}_j) < 0 \\ 0, & (\mathbf{v}_i - \mathbf{v}_j) \cdot (\mathbf{x}_i - \mathbf{x}_j) \geq 0 \end{cases}$$

$$\text{and} \qquad \mu_{ij} = \frac{h (\mathbf{v}_i - \mathbf{v}_j) \cdot (\mathbf{x}_i - \mathbf{x}_j)}{|\mathbf{x}_i - \mathbf{x}_j|^2 + 0.01 h^2},$$

where $\alpha$ corresponds to bulk viscosity [21].

### (b) Numerical integration

We integrate the SPH fluid equations with a Leap-Frog scheme [12], which is, among the second order accurate solvers, the most computationally efficient (only one evaluation per step) and it only requires low memory storage for the evaluation. The numerical stability in this explicit time integration scheme is due to the CFL condition, where adaptive time step is given by

$$\delta t = 0.1 \min \left\{ \frac{h}{|\mathbf{v}_{max}| + c}, \frac{h^2}{6\, \eta_{max}} \right\} . \tag{13}$$

Note that, for simulations of materials with high viscosity the CFL condition tells us that there is a tradeoff between stability and the computational time.

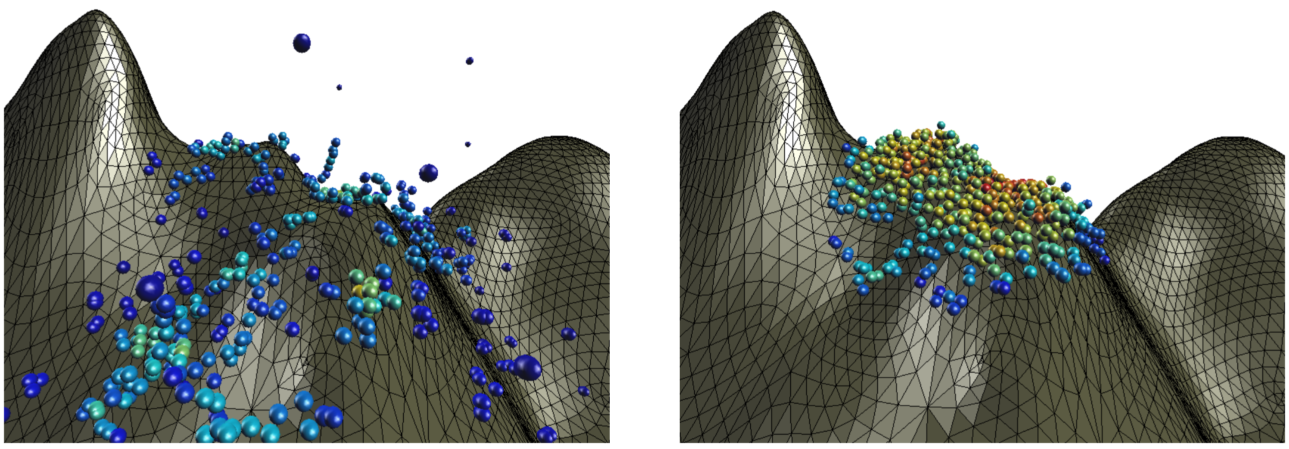


**Figure 6:** *Lava flow simulation after 1409 iterations, without* XSPH *(left) and with the* XSPH *correction (right), with the same parameters: 545 particles, 9566 triangles for the scene. The simulation explodes without* XSPH*, due to arbitrarily small distance between particles.*

## 5 Results and Implementation

In our implementation, there are system attributes and particle attributes. The system attributes such as mass, speed of sound and the smoothing length $h$ are global and they do not change in relation to time. The particle attributes vary according to time, and must therefore be stored at each particle. These attributes are density, viscosity, position, velocity and rate-of-deformation tensor. We update the particle attributes as follow the sequence of Algorithm 1.

### (a) Neighbors' retrieval

Unlike grid-based methods, where the positions of neighboring grid-cells are well defined, the neighbors of a given particle in the SPH method can vary with time. The approaches for finding these neighbors are traditionally: all-pair search ($\mathcal{O}(n^2)$), tree search ($\mathcal{O}(n \log n)$). In this implementation, the SPH approximations are computed from the neighbors at a *fixed* maximal distance $h$. Since $h$ is fixed, the neighbors can be retrieved using a grid of fixed resolution $2h$, where each grid cell contains the list of nodes. This structure, sometimes referred as linked-list, has a memory consumption $\mathcal{O}(n)$ and computational complexity of $\mathcal{O}(1)$: the neighbors of a particle are among the 27 cells surrounding it in the grid, leading to reasonable computation time (see Table 1).

### (b) Collision response

The collision handling between particles and scene model is done in two steps: detecting the collision and computing the new state of the particle. A collision response involves computing a new particle position and modifying the normal and tangential components of its velocity (Figure 7). A perfectly elastic collision inverts the normal component and maintains the tangential component. For softer collisions as the one needed for lava flow simulations, these normal and tangential velocity components can be further be scaled by constant factors between 0 and 1, until factor 0 for no-slip collision (Figure 8).

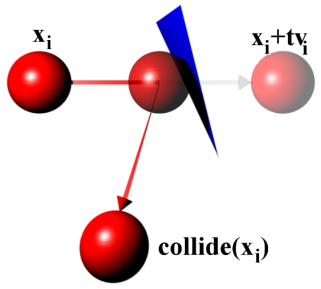


**Figure 7:** *Response to a particle/triangle collision.*

### (c) Collision detection

We represent the scene model explicitly through a triangle mesh and computing the collision between particles (spheres) and triangles. In order to accelerate the detection of eventual collisions between the particles and the scene elements (triangles), we store the triangles in the same search grid as for the neighbors' retrieval. Although a triangle may appear in several cells, it guarantees a constant time search for each position. The complete collision detection is performed as a Bresenham line drawing on the 3D grid, where the line connects $\mathbf{x}_i$ to $\mathbf{x}_i + \delta t \mathbf{v}_i$ and the collision test is computed for each triangle $T$ contained in the current cell. The intersection between particles and triangles is made through a simplified version of the algorithm proposed by Karabassi *et al.* [10]. The performance of the collision test can be improved by using a back-face culling process to remove triangles from the collision test. This is done by comparing the sign of the dot product between triangle's normal $\mathbf{n}_T$ and $\mathbf{v}_i$.

### (d) Rendering

The tracking of the fluid free surface is done by rendering an isosurface from the SPH approximation of its character-

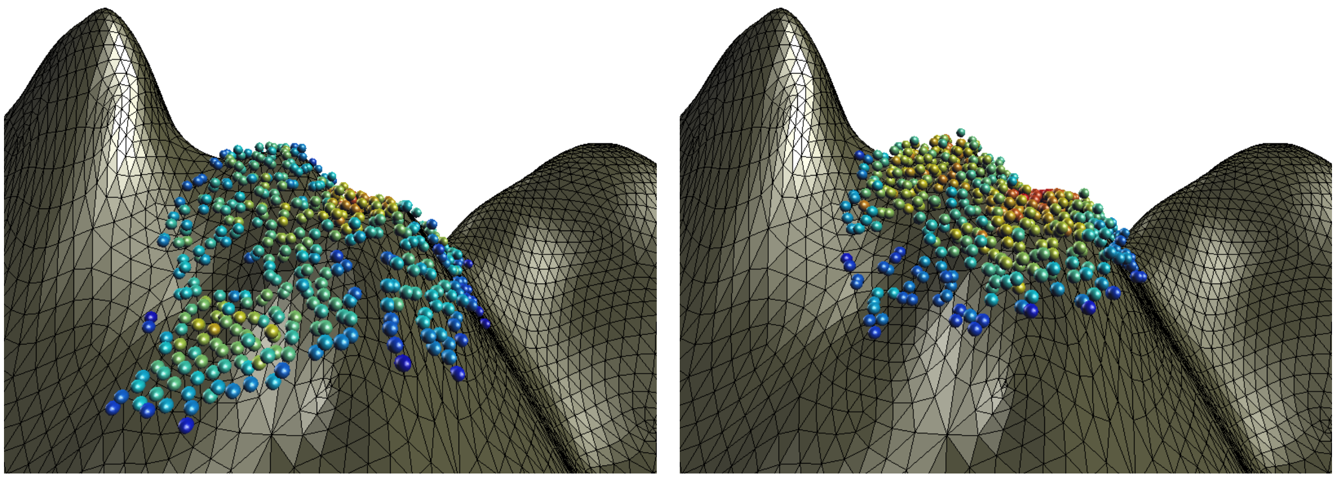

(a) Soft collision. (b) No-slip collision.

**Figure 8:** *Lava flow simulation after 1909 iterations with soft collision (left) and with no-slip collision (right), with the same parameters as Figure 6. The colors map the density of particles: soft collision makes the lava slide too fast.*

istic function [24]:

$$\chi(\mathbf{x}) = \sum_{j \in N(\mathbf{x})} \frac{m_j}{\rho_j} W(\mathbf{x} - \mathbf{x}_j, h)$$

where the isovalue is in the range $[0, 1]$.

We use an efficient and robust implementation of the marching cubes algorithm [30] to generate the triangle mesh for the isosurface. In order to improve the evaluation of the characteristic function, we use the same data structure as for search neighboring particles.

The animations of this paper and the supplementary materials were rendered by using the open-source ray tracer POV–Ray (`http://www.povray.org`).

### (e) Results

We experiment the framework described in this paper in diverse contexts, looking for realistic behaviors in each of them. The main intuition behind viscoplastic deformation is illustrated in Figure 1, where strong forces applied upon the material are able to deform it ("toothpaste effect"). In particular, the yield stress increases with the jump number.

This dependency can be linked to a physical parameter, such as temperature, generating a melting effect. For example in Figure 4, a melting candle with 7000 particles is simulated, initializing with a linear gradient of temperature such that the region near the flame melts while the region far from it remains solid. In this case, the small computation time (see Table 1) is due to the few fluid particles interaction in the melt process.

The use of scene collision in the SPH framework with our viscoplastic formulation results in clear work-softening effects, as in Figure 10 where a viscoplastic Stanford bunny model with 5000 particles collides against a rigid David's face with 10000 triangles. In order to demonstrate the scalability of our framework, we test this example with different particle resolutions and the results show a linear time complexity (see Figure 9).

Our simulations have also been successful in reproducing morphology seen in real lava flows such as spreading of lava front in the absence of solidification and developing complex, non-axi-symmetrical finger shapes called *lobes* (see Figure 11). Our viscoplastic model suits well here due to its conciseness while preserving its physical meaning.

The SPH method already offers simple handling of topological changes. For example in Figure 12, we simulate the interaction between a complex viscoplastic object with 7000 particles and the detailed scene represented by skeleton hand with 2351 triangles. Note that even when the material has flown across the fingers, part of it remains glued to the skeleton.

The quality of the viscoplastic formulation allows direct simulation of advanced viscous effects. In the example of Figure 13, a metallic sphere impacts a plastic wall. Observe that before the sphere drills the wall, the energy dissipated by the shock induces a large deformation of the wall.

## 6 Conclusions and future works

This paper proposes a simplified viscosity formulation for viscoplastic materials using a non-Newtonian fluid model instead of the traditional continuum mechanics model. Our setup relies on the SPH framework, improved regarding the scene interaction and numerical stabilized by adapted discretization of the governing equations of a viscoplastic fluid flow. The effectiveness of the method is illustrated on simple examples, which match the physical laws, leading to an efficient scheme for both animation and physical simulation purposes.

This work can be improved mainly in three directions. On the physical side, we aim at simulating more non-Newtonian fluid behaviors, such as viscoelastic materials. Concern-

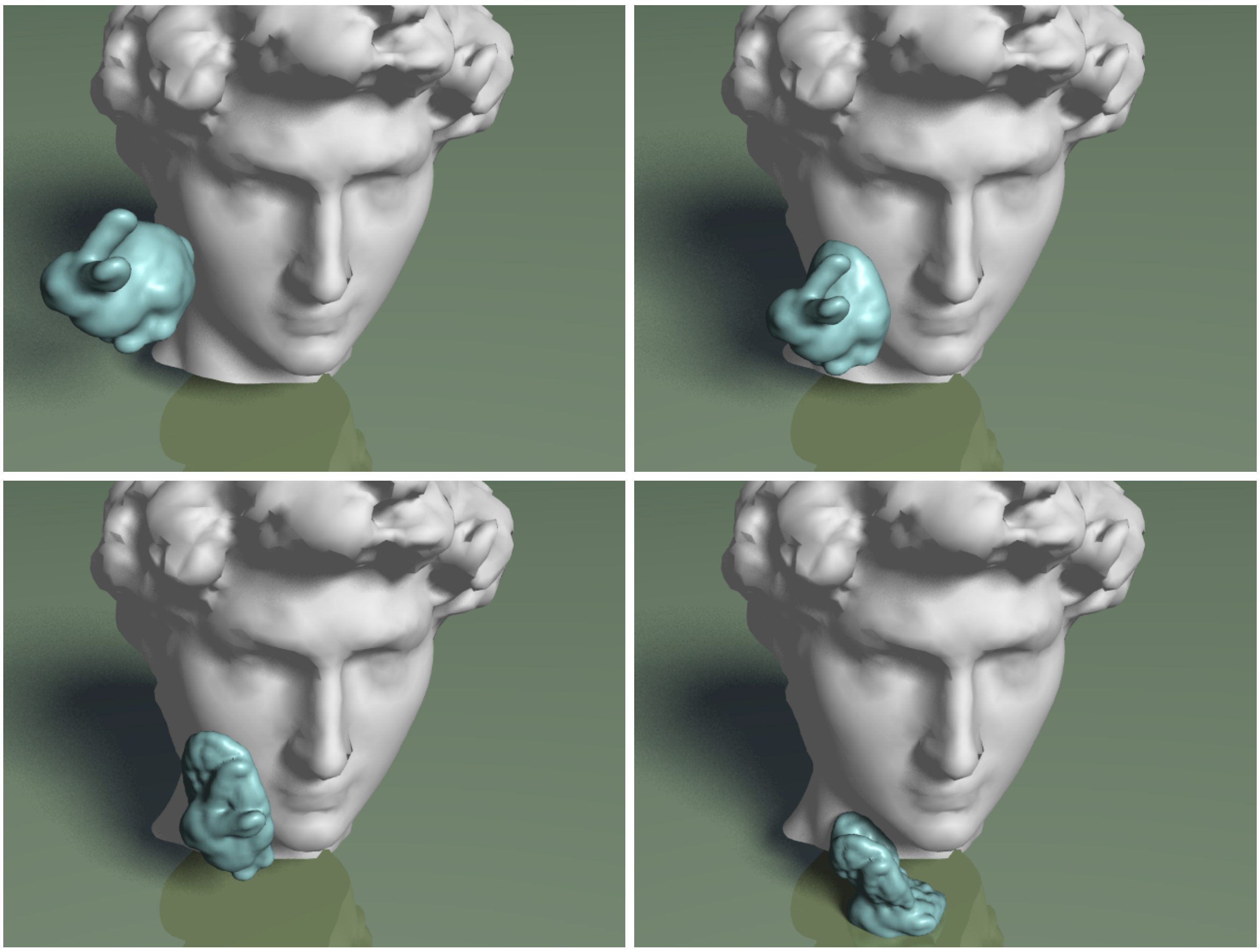


**Figure 9:** *The Stanford Bunny model with 5000 particles is deforming after colliding David's face with 10000 triangles.*

| **Animation** | **# particles** | $\delta t$–**CFL**($\times 10^{-4}$) | | | **CPU time** | | |
|---|---|---|---|---|---|---|---|
| | | *min* | *max* | *avg* | *min* | *max* | *avg* |
| Hand (Figure 1) | 6000 | 6.06 | 6.70 | 6.45 | 0.53 | 0.59 | 0.55 |
| Candle (Figure 4) | 7000 | 6.77 | 7.85 | 7.48 | 0.09 | 0.19 | 0.14 |
| Sugar Loaf (Figure 4) | 10000 | 5.03 | 5.31 | 5.22 | 0.42 | 0.77 | 0.73 |
| David (Figure 10) | 5000 | 4.35 | 7.26 | 6.08 | 0.32 | 0.51 | 0.46 |
| Lava (Figure 11) | 4900 | 6.53 | 6.70 | 6.58 | 0.26 | 0.49 | 0.42 |
| Chair (Figure 12) | 7100 | 5.20 | 6.76 | 6.25 | 0.37 | 0.74 | 0.65 |
| Wall (Figure 13) | 5000 | 20.33 | 40.47 | 30.54 | 0.18 | 0.30 | 0.24 |

**Table 1:** *The adaptive times in seconds by using the CFL condition and the computation time (in seconds) per iteration of the examples running on Centrino – 1.86 GHz.*

ing the efficiency of our simulations, the inner nature of SPH systems allows a straightforward parallelization of the algorithm, which would increase the possible number of particles used during the simulation. Finally, the rendering may be completed by inferring the texture of the objects from the fluid simulation.

## Acknowledgments

We would like to thank Prof. Paulo Roberto Mendes (Department of Mechanical Engineering, PUC–Rio) for suggesting us to use his viscoplastic fluid model, and Marcos Citelli for his help. The authors are members of Matmidia laboratory at PUC–Rio which is sponsored by CNPq, FAPERJ and Petrobras.

## References

[1] J. D. Anderson. *Computational Fluid Dynamics*. McGraw-Hill, 1995.

[2] R. Brownlee, J. Levesley, P. Houston and S. Rosswog. Enhancing SPH using moving least-squares and radial basis functions. In *Algorithms for Approximation*, pages 103–112. Springer, 2007.

[3] T. Harada, S. Koshizuka and Y. Kawaguchi. Smoothed particle hydrodynamics in complex shapes. In *Proc. of Spring Conference on Computer Graphics*, pages 26–28, 2007.

[4] B. Solenthaler, J. Schläfli and R. Pajarola. A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds*, 18(1):69–82, 2007.

[5] A. W. Bargteil, C. Wojtn, J. K. Hodgins and G. Turk. A finite element method for animating large viscoplastic flow. *ACM Trans. on Graph.*, 26(3):16:1–16:8, 2007.

[6] M. Carlson, P. J. Mucha, R. B. Van Horn III and G. Turk. Melting and flowing. In *ACM SIGGRAPH Symposium on Computer Animation*, pages 167–174, 2002.

[7] S. Clavet, P. Beaudoin and P. Poulin. Particle-based viscoelastic fluid simulation. In *2005 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 219–228, 2005.

[8] M. Desbrun and M. P. Cani. Smoothed particles: A new paradigm for animating highly deformable bodies. In *EGCAS '96: Seventh International Workshop on Computer Animation and Simulation*, pages 61–76, 1996.

[9] T. G. Goktekin, A. W. Bargteil and J. F. O'Brien. A method for animating viscoelastic fluids. *ACM Trans. on Graph.*, 23(3):463–468, 2004.

[10] E.-A. Karabassi, G. Papaioannou, T. Theoharis and A. Boehm. Intersection test for collision detection in particle systems. *J. Graph. Tools*, 4(1):25–37, 1999.

[11] R. Keiser, B. Adams, D. Gasser, P. Bazzi, P. Dutré and M. Gross. A unified lagrangian approach to solid-fluid animation. In *Symposium on Point-Based Graphics 2005*, pages 125–134, 2005.

[12] S. Li and W. K. Liu. *Meshfree Particle Methods*. Springer, 2004.

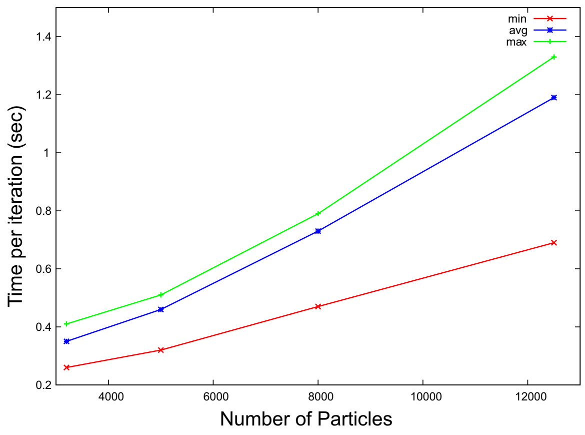


**Figure 10:** *Computational cost of the example given by Figure 10.*

[13] G. R. Liu and M. B. Liu. *Smoothed Particle Hydrodynamics*. World Science, 2005.

[14] F. Losasso, G. Irving, E. Guendelman and R. Fedkiw. Melting and burning solids into liquids and gases. *IEEE Trans. on Vis. and Comput. Graph.*, 12(3):343–352, 2006.

[15] L. B. Lucy. Numerical approach to testing the fission hyphotesis. *Astronomical Journal*, 82:1013–1024, 1977.

[16] H. Mao and Y. Yang. A particle-based model for non-Newtonian fluid. Technical Report TR05-21, Department of Computing Science, University of Alberta, 2005.

[17] P. R. S. Mendes, E. S. S. Dutra, J. R. R. Siffert and M. F. Naccache. Gas displacement of viscoplastic liquids in cappilary tubes. *J. of Non-Newton. Fluid Mech.*, 142:1–11, 2007.

[18] R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Month. Not. Roy. Astr. Soc.*, 181:375–389, 1977.

[19] J. J. Monaghan. On the problem of penetration in particle methods. *J. Comput. Phys.*, 82:1–15, 1989.

[20] J. J. Monaghan. Simulating free surface flow with SPH. *J. Comput. Phys.*, 110:399–406, 1994.

[21] J. J. Monaghan. Smoothed particle hydrodynamics. *Rep. Prog. Phys.*, 68:1703–1759, 2005.

[22] J. P. Morris, P. J. Fox and Y. Zhu. Modeling low reynolds number for incompressible flows using SPH. *J. Comput. Phys.*, 136:214–226, 1997.

[23] M. Müller, S. Schirm, M. Teschner, B. Heidelberger and M. Gross. Interaction of fluids with deformable solids. *Computer Animation and Virtual Worlds*, 15(3-4):159–171, 2004.

[24] M. Müller, D. Charypar and M. Gross. Particle-based fluid simulation for interactive applications. In *2003 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 154–159, 2003.

[25] M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross and M. Alexa. Point based animation of elastic, plastic and melting objects. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 141–151, 2004.

[26] J. F. O'Brien, A. W. Bargteil and J. K. Hodgins. Graphical modeling and animation of ductile fracture. *ACM Trans. on Graph.*, 21(3):291–294, 2002.

[27] A. Paiva, F. Petronetto, T. Lewiner and G. Tavares. Particle-based non-Newtonian fluid animation for melting objects. In *Proc. of SIBGRAPI 2006 – XIX Brazilian Symposium on Computer Graphics and Image Processing*, pages 78–85, 2006.

[28] D. Stora, P.-O. Agliati, M.-P. Cani, F. Neyret and J.-D. Gascuel. Animating lava flows. In *Graphics Interface '99*, pages 203–210, 1999.

[29] D. Terzopoulos, J. Platt and K. Fleischer. Heating and melting deformable models (from goop to glop). In *Graphics Interface '89*, pages 219–226, 1989.

[30] T. Lewiner, H. Lopes, A. W. Vieira and G. Tavares. Efficient implementation of marching cubes with topological guarantees. *J. Graph. Tools*, 8(2):1–15, 2003.

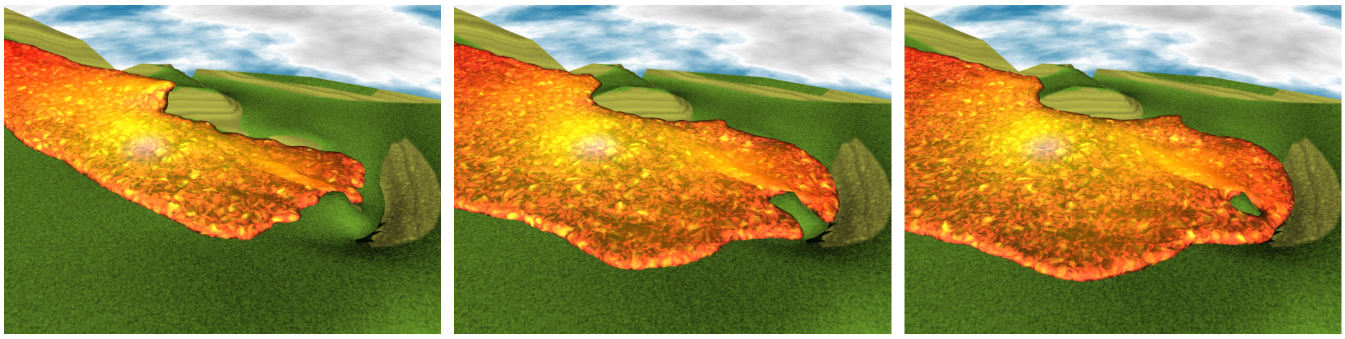


**Figure 11:** *Lava flow on a small virtual terrain of 1547 triangles, with 4900 particles.*

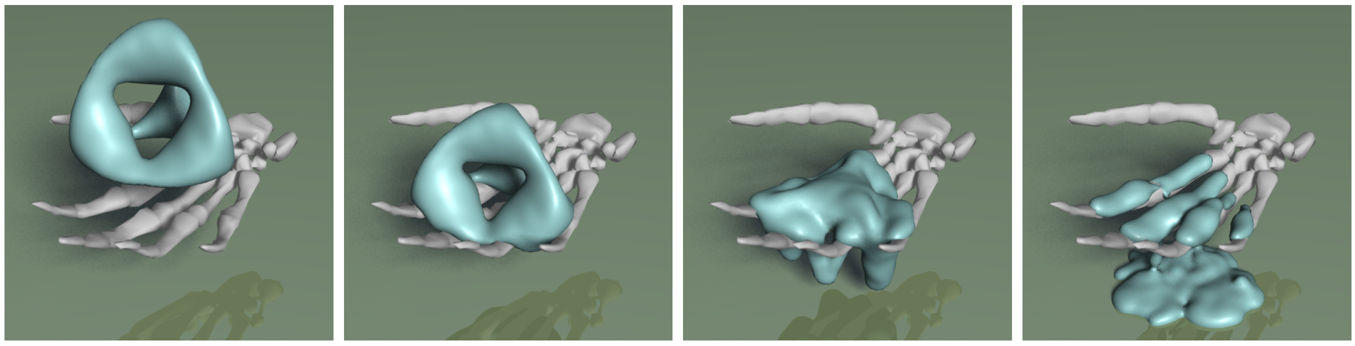


**Figure 12:** *The viscoplastic chair model with 7000 particles interacting with a complex solid hand skeleton with 2351 triangles.*

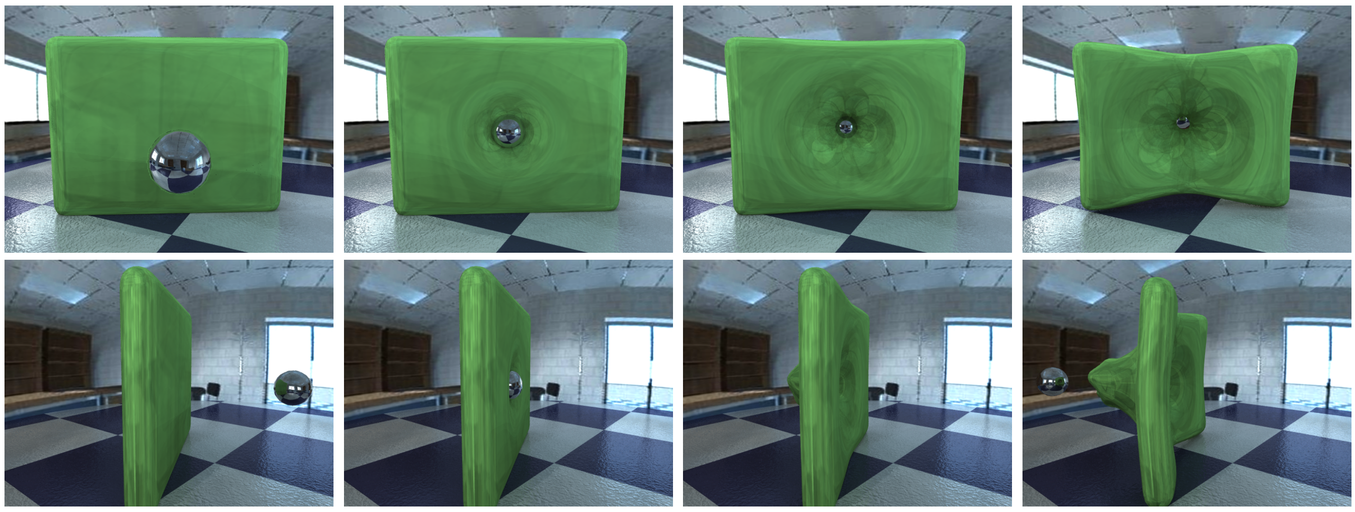


**Figure 13:** *Collision between a metal sphere and a plastic wall (green) made of 5000 particles. The simulation is showed in the front view (top) and side view (bottom).*