CGI2012 manuscript No.
(will be inserted by the editor)

A Prediction-Correction Approach for Stable SPH Fluid Simulation from Liquid to Rigid

François Dagenais · Jonathan Gagnon · Eric Paquette

Abstract The simulation of highly viscous fluids using an SPH (Smoothed Particle Hydrodynamics) approach is a tedious task. Since the equations are typically posed as stiff problems, simulating highly viscous fluids involves strong forces applied to the particles. With these strong forces, a very small time step is needed to keep the simulation stable and produce good results. The approach detailed in this paper uses an iterative prediction-correction scheme to optimize forces that act on the fluid, in order to produce a behaviour that varies from liquid to solid. This approach significantly reduces the computation times when the fluid is very viscous and almost rigid. At every time step, each particle position is predicted. The deformation is then compared with a target deformation and forces are adjusted to counteract the deformation. In addition to requiring lengthy computation times and tedious adjustment of time step to maintain a stable simulation, typical SPH simulators make it difficult to achieve the desired behavior. This difficulty is caused by the highly non-linear effect that the viscosity has on the behavior of the fluid. Compared to the typical viscosity parameter which varies from zero to infinity, the proposed rigidity parameter is easier to control, providing an intuitive variation from 0 (liquid) to 1 (solid). Since simulating high viscosity fluids is subject to large computation times and instabilities, we complement the proposed model with some important improvements. Firstly, an improved time step adjustment is proposed that results in both reduced computation times and increased stability. Secondly, an implicit temperature diffusion provides stable melting and solidification, regardless of the size of the time step. Thirdly, a constraint propagation provides faster convergence of the rigid forces to visually realistic behaviours. Together, these improvements and the proposed model allow the simulation of fluids with viscous behaviours that were very difficult, if not impossible, to simulate with current SPH approaches.

Keywords Fluid simulation · SPH · Melting · Extreme viscosity · Heat diffusion · Prediction-correction · Stability

1 Introduction

Current techniques can simulate relatively viscous fluids using Lagrangian simulations. However, it is difficult to simulate fluids that are so viscous that they almost look rigid. This problem arises because simulating an almost rigid fluid involves generating strong viscous forces. Simulating these strong forces requires an extremely small time step to ensure stability and correctness of the simulation. This implies high computational times and the result is still far from looking like a rigid object. Also, since the fluid is discretized into particles which can only influence their neighbors, several time steps are needed for the effect of a collision with an obstacle to propagate throughout the whole fluid. Again, the use of a smaller time step helps in solving this problem, but at a large computational cost.

Consequently, our goal is to lower the computational cost of simulating extremely viscous fluids, while in-

F. Dagenais
Multimedia Lab, ÉTS, Montréal, Canada
E-mail: francois.dagenais.2@ens.etsmtl.ca

J. Gagnon
Mokko Studio, R&D, Montréal, Canada
Multimedia Lab, ÉTS, Montréal, Canada
E-mail: jgagnon@mokkostudio.com

E. Paquette
Multimedia Lab, ÉTS, Montréal, Canada
E-mail: eric.paquette@etsmtl.ca

creasing stability and the range of viscous behaviors that can be simulated using a Lagrangian simulation. Instead of using a smaller time step, the proposed approach executes additional work at each time step to properly handle viscous behaviors of the fluid. While these extra calculations are iterative, each iteration takes less time to compute than a simulation time step. Furthermore, the proposed approach introduces a new model to handle viscosity. Although this new model is not physically based, it produces visually plausible results while providing a more intuitive control over the particles rigidity. With this new model, the viscosity simulation is not posed as a stiff problem, making it possible to use a larger time step, thus reducing the total computation time. The contributions of this paper are summarized as follows:

– A model able to simulate extremely viscous fluids (almost rigid) using a Lagrangian simulation.
– Reduced computation times when simulating extremely viscous fluids.
– An unconditionally stable simulation of heat diffusion between particles using an implicit scheme.
– A constraint propagation method that reduces computation times while improving the results when dealing with constraints such as collision with rigid objects.
– An accurate calculation of the time step to maintain the stability while providing large time steps.
– A simulation controlled by a single easy to adjust parameter ranging from 0 (liquid) to 1 (solid).

2 Related Work

2.1 Lagrangian Gas-Liquid Phases

SPH simulation was introduced in computer graphics by Desbrun and Cani [3]. Later, Müller and al. [6] extended this method focusing on fluid simulation. They introduced a new implementation of the Navier-Stokes equation of viscosity and pressure. The SPH method has been used to simulate phase changes by Müller and al. [7] who introduced multiple fluid simulation using SPH. Phase transition uses the temperature of the particles to model boiling water by changing the types and densities of particles dynamically. These approaches are related to ours as they use SPH to simulate different phases or states of matter, in this case, liquid and gas. Nevertheless, since they do not handle liquid to solid transitions and interactions, they cannot be used to solve the problem that the proposed approach solves.

2.2 Eulerian Gas-Liquid-Solid Phases

Carlson and al. [1] proposed a multiphase method handling liquid and solid. The Eulerian fluid simulation enables complex water behaviour. The approach used Eulerian (Marker-and-Cell) simulation with an implicit scheme allowing large time steps. To simulate the transition between liquid and solid, it used a very high viscosity fluid. With this technique, a realistic melting object is possible. However, it needs high computation times when simulating very viscous fluids. Rasmussen and al. [10] enhanced the viscosity solver of Carlson and al. [1] to better handle fluids with varying viscosity. The approach provides control over the viscosity using control particles. However, regions of the fluid with many control particles can be time consuming and less realistic. While Eulerian approaches have been successfully used to simulate liquids with high viscosity, our contribution focuses on improving SPH simulations. It should also be noted that both Eulerian and Lagrangian fluid simulations require lengthy computation times when considering liquids with high viscosities and one of our contributions is in reducing the computation times of Lagrangian SPH highly viscous simulations.

2.3 Lagrangian Fluid-Solid Phases

The approach of Paiva and al. [8] used a transition model based on a stress tensor. It allows the possibility to simulate melting objects by controlling a single parameter called the "jump number" which affects the "apparent viscosity" of the fluid. The numerical stability is improved by using artificial viscosity and an adaptive time step based on the CFL (Courant-Friedrichs-Lewy) condition. Later, they adapted their approach to handle collisions with complex object surfaces [9]. Chang and al. [2] later proposed a viscoelastic model which they used to simulate melting objects by modifying the elasticity and viscosity of the particles according to their temperature. Both approaches have the disadvantage of requiring lengthy computation times when simulating extremely viscous fluids. Heat diffusion was also simulated in both approaches in order to produce more visually realistic results. Both authors uses a similar approach, but Paiva and al. [9] approximated the laplacian of the kernel using only its first derivative, which is always positive. While this represents an improvement, this heat diffusion approach can become unstable for large values of the diffusion coefficient or the time step.

Solenthaler and al. [12] developed a Lagrangian simulation to create a unified model which allows the computation of fluid-solid interaction. They used an elastic model to handle the phase transition between solid and liquid. Melting is done by modifying the elasticity and updating the rest model using temperature and "stress" between particles. Melting and viscoelastic deformation is also possible with the work of Chang and al. [2]. This approach adds an elastic stress tensor to the SPH simulation in order to achieve the viscoelastic behavior. It uses a coefficient, according to the temperature, that affects the melting and flowing phenomena. Unfortunately, for both approach the time step needed to simulate "close to" solid material is really small, thus affecting computation time.

A similar problem arises when simulating incompressible fluids using SPH, which also involves strong forces. Solenthaler and Pajarola [11] introduced an approach based on a "prediction-correction" solver to optimize pressure forces for the current time step. Using this approach, they could ensure stability of the simulation with a larger time step while generating good visual results. Although the optimization process takes much more time than a normal simulation iteration, the large time step allowed with this technique enables faster simulation of incompressible fluids. There is no corresponding approach for simulating extremely viscous fluids. This paper fills that gap by introducing an approach to simulate high viscosity and almost rigid fluids based on a prediction-correction approach and a new deformation error metric. The proposed approach smoothly handles the transition between a liquid and an almost completely rigid fluid.

3 Foundations of the SPH Simulator

The proposed approach is based on a fluid simulation to which forces controlling its rigidity are added (see Sec. 4). Although it can be applied to any particle-based simulation, our implementation is based on an SPH (Smoothed Particle Hydrodynamics) simulation. Our simulator is based on the work of Müller and al. [7]. At every time step, particles velocities v_i are updated using standard SPH forces:

$$\frac{dv_i}{dt} = \frac{1}{\rho_i}\left(f_i^{pressure} + f_i^{external}\right) \tag{1}$$

The force $f_i^{external}$ includes any external forces such as gravity. The force $f_i^{pressure}$ is the internal pressure force of the fluid. It is computed using this equation:

$$f_i^{pressure} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W\left(x_{ij}, h\right)$$

Note that Table 1 summarizes all the symbols used in this paper. Density ρ_i and pressure p_i of particle i, can be computed using these equations:

Table 1 Symbols

Symbols	Definition
v_i,x_i,m_i	Velocity, position and mass of particle i
μ_i,p_i,ρ_i	Viscosity, pressure and density of particle i
ρ_0	Rest density
h	Smoothing radius
x_{ij}	$x_i - x_j$
W_{ij}	$W(x_{ij}, h)$, Smoothing kernel
c	Speed of sound
α_v	Bulk viscosity
x_{ij}^0	Initial relative position of i from j
$x_i^{solid_j}$	Position of i when solid according to j
x_i^{liquid}	Position of i when liquid
α_{ij}	Rigid bond coefficient of particles i and j
s_i	Rigidity of particle i
T_i	Temperature of particle i
T_{solid}	Temperature when fluid is completely solid
T_{liquid}	Temperature when fluid is completely liquid
$dist^0$	Initial distance between particles

$$\rho_i = \sum_j m_j W\left(x_{ij}, h\right)$$

$$p_i = c^2\left(\rho_i - \rho_0\right)$$

The choice of kernels influences the visual result as well as the stability of the simulation. We used the same kernels as Müller and al. [6] who used strictly positive gradient and laplacian kernels.

A damping force [4,9] is used in order to prevent particles from oscillating due to the pressure forces:

$$f_i^{damping} = -\rho_i \sum_j m_j \Pi_{ij} \nabla_i W\left(x_{ij}, h\right)$$

$$\Pi_{ij} = \begin{cases} -\frac{2\alpha_v \mu_{ij} c}{\rho_i + \rho_j},\ (v_i - v_j)\cdot(x_i - x_j) < 0 \\ 0,\ (v_i - v_j)\cdot(x_i - x_j) \geq 0 \end{cases}$$

$$\mu_{ij} = \frac{h\left(v_i - v_j\right)\cdot\left(x_i - x_j\right)}{\left|x_i - x_j\right|^2 + 0.01h^2}$$

This force is added to equation 1:

$$\frac{dv_i}{dt} = \frac{1}{\rho_i}\left(f_i^{pressure} + f_i^{external} + f_i^{damping}\right) \tag{2}$$

Finally, to improve the quality of the results, a leap-frog integration scheme is used to update particles in order to provide second order accuracy without additional computational costs.

4 Melting and Solidification

Alg. 1 provides an overview of the simulation loop of the proposed approach. It can be seen that the approach provides improvements that integrate well within the standard SPH framework. Phase 1 from Alg. 1 is related

```
while Simulation time ≤ End time do
    // Phase 1: Compute liquid SPH forces (Sec. 3)
    Compute particles density $\rho_i$ and pressure $p_i$
    Compute forces $f_i^{pressure}$, $f_i^{external}$, $f_i^{damping}$
    Compute $\Delta t$ (Sec. 5.1)
    // Phase 2: Optimize rigid forces
    while Stopping criterion not met do
        Predict particle positions (Sec. 4.2)
        Compute particle deformations (Sec. 4.1)
        Adjust $f_i^{rigid}$ (Sec. 4.2)
    // Phase 3: Integrate
    Update particle positions and velocities
    Diffuse the temperature (Sec. 5.2)
    Simulation time += $\Delta t$
```

Algorithm 1: Simulation step

to the typical SPH simulation of forces with an improvement on the computation of the time step. Phase 2 is the core part of our approach where rigid forces are iteratively optimized in order to maintain the fluid in a more or less viscous state based on the user specified rigidity factor. Phase 3 is also quite typical of SPH simulations that handle melting and solidification, with our proposed improvement for a more stable temperature diffusion. These improvements are detailed in this section, which addresses the iterative adjustments of the rigid forces, and Sect. 5, which covers contributions regarding the stability of the simulation.

In our model, the rigidity s_i varies from 0 (completely liquid) to 1 (completely solid) as a function of the temperature:

$$s_i = \begin{cases} 1,\ T_i \le T_{solid} \\ \frac{T_{liquid}-T_i}{T_{liquid}-T_{solid}},\ T_{solid} < T_i < T_{liquid} \\ 0,\ T_{liquid} \le T_i \end{cases}$$

The initial temperature is provided by the user. Then, the diffusion of the temperature is used (see Sec. 5.2)

To produce the desired deformation based on s_i, an additional force, f^{rigid} is added to the SPH equation:

$$\frac{dv_i}{dt} = \frac{1}{\rho_i}\left(f_i^{pressure} + f_i^{external} + f_i^{damping} + f_i^{rigid}\right)$$

To correctly control the deformation of the material, the computation of f^{rigid} is based on a *target deformation.* This target deformation corresponds to a deformation that can range from completely liquid to completely solid, based on s_i. It is used to compute the deformation error φ which represents the difference between the current deformation and the target deformation (see Sec. 4.1). The proposed method seeks to find rigid forces that minimize this error:

$$min_{f^{rigid}}\left(\varphi(particules, f^{rigid})\right)$$

Finding the optimal f^{rigid} forces that minimize the global deformation error would require too much computation. As can be seen in Alg. 1, the optimization of the rigid forces is computed iteratively. From the particles deformation errors, the best rigid forces are computed on a per particle basis. The next iteration then computes the resulting position of the particles, the new deformation errors and the new rigid forces. With the mutual influence of the neighbour particles, the rigid forces converge to a visually realistic solution. Sec. 4.1 explains how to compute the deformation error while Sec. 4.2 explains how to obtain the rigid forces.

4.1 Deformation Error

In order to control the deformation of the material, a deformation error metric is used. The deformation error is the difference between the current deformation and a target deformation. Considering a solid material, the deformation error for particle i is computed based on the difference in position with the neighbor particles $j \in 1..n$:

$$\varphi_i = \sum_j \alpha_{ij}\left(x_i^{solid_j} - x_i\right) W_{ij}\left(x_{ij}, h\right)$$

The difference in position is computed using relative positions. The initial relative position x_{ij}^0 is computed before every time step using the current value of x_i and x_j. The solid position $x_i^{solid_j}$ is computed using the position x_j of the neighbor j and x_{ij}^0:

$$x_i^{solid_j} = x_j + x_{ij}^0$$

In order to have a continuous transition between solid and liquid, the target position $x_i^{target_j}$ is used instead of $x_i^{solid_j}$:

$$x_i^{target_j} = s_{ij} x_i^{solid_j} + (1 - s_{ij})\, x_i^{liquid}$$

$$\varphi_i = \sum_j \alpha_{ij}\left(x_i^{target_j} - x_i\right) W_{ij}\left(x_{ij}, h\right)$$

The position x_i^{liquid} is the position according to the SPH forces excluding the rigid force. The solid position $x_i^{solid_j}$ and the liquid position x_i^{liquid} of the particle i are blended together in order to find the target position of particle i according to j. This approach to compute the deformation allows for the melting and solidification of the material.

If two particles have different s_i factors, the minimum of both rigidity coefficients is used ($s_{ij} = min(s_i, s_j)$). This way, liquid particles and semi-rigid particles will not act like rigid particles when surrounded by rigid particles. Similarly, more importance is given to neighbors when both particles are rigid or highly viscous using the rigid bond coefficient $\alpha_{ij} = {s_{ij}}^b$. In the tests presented in this paper, the exponent b was empirically fixed to $b = 20$.

This new metric allows to measure how much the actual position of the particles is diverging from the desired position. Using a prediction-correction scheme, it is possible to determine forces that minimizes this metric in order to produce the desired visual result.

4.2 Deformation Correction

This section describes how the rigid forces are computed. The terminology used in this section follows the one used in the PCISPH approach [11]. While inspired by the prediction-correction scheme of PCISPH, the proposed approach provides a new correction formulation applied to a different phenomenon. A prediction-correction solver is used to minimize the deformation error metric (see sect. 4.1). The proposed iterative solver applies a correction to the rigid forces at every sub step to reduce the deformation error. While iteratively applying sub steps, the rigid forces converge to smaller and smaller deformation errors.

The main goal here is to adjust the forces so that the position of the particles at the next time step will result in smaller deformation errors. In order to adjust the force for each particle, the relation between the rigid force and the deformation error of a particle i is first detailed. From the beginning (time t) to the end (time $t + \Delta t$) of a simulation time step, the difference in deformation error is as follows:

$$\begin{aligned}
\varphi_i(t+\Delta t) &= \sum_j \alpha_{ij} \left[x_i^{target_j} - x_i(t+\Delta t)\right] W_{ij} \\
&= \sum_j \alpha_{ij} \left[x_i^{target_j} - (x_i(t) + \Delta x_i(t))\right] W_{ij} \\
&= \sum_j \alpha_{ij} \left(x_i^{target_j} - x_i(t)\right) W_{ij} \\
&\quad - \sum_j \alpha_{ij} \Delta x_i(t) W_{ij} \\
&= \varphi_i(t) - \sum_j \alpha_{ij} \Delta x_i(t) W_{ij} \\
&= \varphi_i(t) + \Delta\varphi_i(t)
\end{aligned}$$

Since all of this happens at a single time step, W_{ij} and $x_i^{target_j}$ remain constant.

Let us say that the movement of the particle was the result of a force f_i^{deform}. This force influences the position of the particle i in the following manner:

$$\Delta x_i(t) = \frac{dv_i}{dt}\Delta t^2 = \frac{f_i^{deform}\Delta t^2}{\rho_i}$$

Introducing this into $\Delta\varphi_i(t)$ yields:

$$\begin{aligned}
\Delta\varphi_i(t) &= -\sum_j \alpha_{ij}\Delta x_i(t) W_{ij} \\
&= -\sum_j \alpha_{ij}\left(\frac{f_i^{deform}\Delta t^2}{\rho_i}\right) W_{ij} \\
&= -\frac{f_i^{deform}\Delta t^2}{\rho_i}\sum_j \alpha_{ij} W_{ij}
\end{aligned}$$

From these equations, the force needed to produce a specific change in a particle deformation error function can be computed as follows:

$$f_i^{deform} = -\Delta\varphi_i(t)\frac{\rho_i}{\Delta t^2 \sum_j \alpha_{ij} W_{ij}}$$

Since the goal is to have $\varphi_i(t+\Delta t) \approx 0$, we set $\Delta\varphi_i$ to $-\varphi_i$, resulting in the following frigid force:

$$f_i^{rigid} = \varphi_i(t)\frac{\rho_i}{\Delta t^2 \sum_j \alpha_{ij} W_{ij}}$$

As a result of the interrelation between each particle, the computed rigid forces are not likely to yield the minimal $\varphi(t)$. This is why an iterative process is introduced in Alg.1, where the rigid forces are repeatedly adjusted. At the first iteration, the rigid forces are set to null forces. Then, at each iteration, the position of the particles are predicted using the previously adjusted rigid forces, the deformation error φ_i of each particle is computed and finally the forces are adjusted based on this predicted deformation error.

4.3 Stopping Criterion

In order to determine whether or not the iterative adjustment of the rigid forces should stop, a stopping criterion is evaluated after every iteration. Firstly, the sum φ of the deformation errors φ_i is computed. Then, the difference between that sum and the one computed during the previous iteration is evaluated. The algorithm stops when this difference is below the threshold $0.01\varphi^{previous}$.

However, that approach might be sensitive to irregular variations of φ over successive iterations. To avoid an erroneous early termination of the iterative process, a window of 5 iterations (fixed empirically) is used when evaluating the variation of φ at the current iteration.

4.4 Constraints Propagation

While the proposed iterative optimization of the rigid forces produces good overall results inside the material, it is not always effective in adapting the rigid forces to hard constraints at the boundaries of the material. Examples of such constraints include particles in collision with the surface of rigid objects outside of the SPH simulator, as well as particles that would be key frame animated. When dealing with such constraints, the iterative process requires too many iterations to converge, and sometimes it converges to a local minimum that is too far from a realistic result. To overcome this issue, the proposed approach uses a priority based constraint propagation method, which correctly and efficiently propagates the effects of the constraints to the rigid forces of all particles. Right before the iterative rigid forces adjustment of phase 2 in Alg. 1 begins, particles affected by a constraint are flagged as high priority, while the remaining particles are flagged as low priority. Whenever a particle has one or more neighbors flagged as high priority, only these high priority neighbors are considered during the calculation of the deformation error and the correction of rigid forces. This way, neighbors of particles affected by constraints are adjusted correctly according to these particles. At each iteration of phase 2, particles that become high priority are considered as correctly affected by the constraints and allow their neighbours to become adjusted accordingly. This process is continued until the constraint has been propagated through the whole fluid. At each rigid forces iterative adjustment of phase 2, new particles can become high priority if these two conditions are met:

– The sum of the lengths $\|x_i^{target_j} - x_i\|$ is below a threshold:

$$\sum_j \|(x_i^{target_j} - x_i)\| \leq 0.01 \sum_j dist^0$$

– Out of the particles that have at least one high priority neighbor, the particle is among the ones with the highest s_i.

If a particle meets the second condition but fails to fulfill the first one after a predetermined number of iterations (5 in our examples), it becomes high priority in order to prevent the algorithm from being blocked. The second condition prevents artifacts caused by constraints propagating through particles with different values of s_i. The most rigid particles are affected by the constraints first, preventing less rigid parts from transferring unrealistic deformation into the rigid regions.

While the constraints are propagated, the iterations of phase 2 are forced to continue, even if the stopping criterion might have been fulfilled. After the constraints have reached all the particles, the iterations continue until the stopping criterion of Sec. 4.3 is met. This ensures that the constraints can propagate their effects to all of the particles. As a result, the number of iterations required to achieve a satisfactory result is much smaller and the result preserves the effect of the constraints much better.

5 Simulation Stability

A variety of factors affect the stability of the simulation. For example, when simulating extremely viscous fluids, strong forces are generated which require a small time step to maintain stability. The approach introduced in this paper eliminates these strong forces as well as the instability, making it possible to use a larger time step while simulating fluids with high viscosity. However, because larger time steps are used, other sources of instability, such as strong pressure forces and fast heat diffusion can affect the simulation. Approaches able to handle these instabilities are presented in this section.

5.1 Dynamic Time Step

As stated previously, the size of the time step affects the stability of the simulation. While a small time step improves the stability of the simulation, it results in lengthy computation times. Hence, it is important to find a time step that will ensure stability with the minimum computational costs. Paiva and al. [9] proposed an approach to adapt the time step dynamically at every iteration of the fluid simulation. They adapt the time step based on the pressure term of the SPH equation and the “apparent viscosity” of the fluid using the CFL condition. Since our approach eliminates the instability caused by viscosity forces, we will focus only on the pressure forces. Their approach relies on the speed of sound c and the maximum velocity at the previous time step $|v_{max}^{(t-\Delta t)}|$, to compute Δt:

$$\Delta t = 0.1 \frac{h}{|v_{max}^{(t-\Delta t)}| + c}$$

However, when the pressure is very strong their approach might not ensure stability since the strongest acceleration might be larger than c. Also, when the fluid is at rest, the impact of pressures forces on the velocity is smaller than c. Therefore, their approach will not give the optimal time step while still being prone to instability when there are high pressure forces. By using the maximum acceleration of the current time step instead of c it is possible to compute a larger time step while keeping the simulation stable.

The CFL condition tells us that a particle should not travel more than a fraction β of its radius in one time step. Therefore the ideal time step should be computed using the following equation:

$$\Delta t = \beta \frac{h}{|\tilde{v}_{max}|} \tag{3}$$

It is impossible to know the maximum velocity without knowing the time step, since the velocity is affected by the acceleration. However, it is possible to approximate the maximum possible velocity using $|\tilde{v}_{max}| = |v_{max}^{(t-\Delta t)}| + |a_{max}|\Delta t$, where a_{max} is the maximum acceleration of a particle during the current time step. Applying this to equation 3 yields:

$$\Delta t = \beta \frac{h}{|v_{max}^{(t-\Delta t)}| + |a_{max}|\Delta t}$$

Solving for Δt yields:

$$\Delta t = -\frac{|v_{max}^{(t-\Delta t)}| - \sqrt{|v_{max}^{(t-\Delta t)}|^2 + 4|a_{max}|\beta h}}{2|a_{max}|} \tag{4}$$

At every simulation step, equation 4 is used to compute a time step value that will ensure stability while providing good performances.

5.2 Implicit Heat Diffusion

When handling melting and solidification, computing heat diffusion is an important simulation step in order to provide realistic results. As seen in Sec. 4, the approach presented in this paper allows larger time steps while simulating a highly viscous fluid. Typical heat diffusion approaches are prone to instabilities when the time step or the heat diffusion coefficient are large. By using larger time steps, typical heat diffusion techniques [2,8] are more subject to instability. Let's take the heat diffusion equation of Chang and al. [2], which is a commonly used formulation as an example:

$$\frac{dT_i}{dt} = k \sum_j \frac{m_j}{\rho_j} (T_{ji}) \nabla^2 W_{ij} \tag{5}$$
$$T_{ji} = T_j - T_i$$

It is obvious that it can become unstable when the time step or k is too large since dT_i might become larger than $T_j - T_i$. Computing the limits of dT_i according to k and Δt gives $dT_i = \pm\infty$ in both cases, confirming the unstable nature of the equation.

By adapting the approach of Monaghan [5], who used an implicit formulation of the viscosity equation for each pair of particles, it is possible to compute heat diffusion using an implicit formulation. First, let's rewrite equation 5 in its implicit form using the temperature at time $t + \Delta t$:

$$\frac{dT_i(t)}{dt} = k \sum_j \frac{m_j}{\rho_j} (T_{ji}(t + \Delta t)) \nabla^2 W_{ij}$$

With a few simplifications and assumptions, it is possible to derive from this equation an unconditionally stable expression for the temperature T_i at time $t + \Delta t$. If each pair of particles is treated individually, it is easy to determine how particle i and j affect each other:

$$T_i(t + \Delta t) = T_i(t) + k\frac{m_j}{\rho_j} (T_{ji}(t + \Delta t)) \nabla^2 W_{ij} \Delta t \tag{6}$$

$$\begin{aligned} T_j(t + \Delta t) &= T_j(t) + k\frac{m_i}{\rho_i} (T_{ij}(t + \Delta t)) \nabla^2 W_{ij} \Delta t \\ &= T_j(t) - k\frac{m_i}{\rho_i} (T_{ji}(t + \Delta t)) \nabla^2 W_{ij} \Delta t \end{aligned} \tag{7}$$

By making the assumption that the density (ρ_i and ρ_j) and the laplacian of the kernel $\nabla^2 W_{ij}$ are constant over Δt it is possible to find an expression to compute the temperature T_i at time $t + \Delta t$ involving only variables at time t. Furthermore, this new expression produces good results. Using equations 6 and 7, we can express $T_{ji}(t + \Delta t)$ as a function of $T_{ji}(t)$, thus removing any dependencies on variables at time $t + \Delta t$:

$$\begin{aligned} T_{ji}(t + \Delta t) &= T_j(t + \Delta t) - T_i(t + \Delta t) \\ &= \left[T_j(t) - k\frac{m_i}{\rho_i} (T_{ji}(t + \Delta t)) \nabla^2 W_{ij} \Delta t\right] \\ &\quad - \left[T_i(t) + k\frac{m_j}{\rho_j} (T_{ji}(t + \Delta t)) \nabla^2 W_{ij} \Delta t\right] \\ &= (T_{ji}(t)) - k\left(\frac{m_i}{\rho_i} + \frac{m_j}{\rho_j}\right) (T_{ji}(t + \Delta t)) \nabla^2 W_{ij} \Delta t \end{aligned}$$

Moving terms containing $T_{ji}(t + \Delta t)$ to the left hand side we get:

$$(T_{ji}(t + \Delta t)) \left(1 + k\left(\frac{m_i}{\rho_i} + \frac{m_j}{\rho_j}\right) \nabla^2 W_{ij} \Delta t\right) = T_{ji}(t)$$

Keeping only $T_{ji}(t + \Delta t)$ on the left hand side we get:

$$T_{ji}(t + \Delta t) = \frac{T_{ji}(t)}{1 + k\left(\frac{m_i}{\rho_i} + \frac{m_j}{\rho_j}\right) \nabla^2 W_{ij} \Delta t} \tag{8}$$

Replacing $T_{ji}(t + \Delta t)$ in Eq. 6 with the formulation of $T_{ji}(t + \Delta t)$ of Eq. 8, it is now possible to compute $T_i(t + \Delta t)$ using only variables at time t.

In order to find the temperature $T_i(t + \Delta t)$, T_i is updated for each neighbor using the following operation:

$$T_i \leftarrow T_i + k\frac{m_j}{\rho_j} T_{ji} \nabla^2 W_{ij} \Delta t \tag{9}$$

Since the result is dependent on the neighbors traversal order, this operation is applied a second time using the

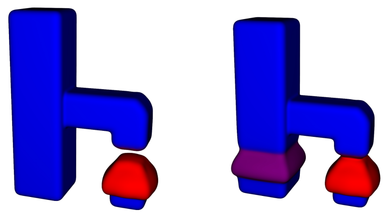

Fig. 3 The "h" example.

reverse order and then the average of the two results is used as the new temperature T_i. This approach is unconditionally stable, which allows us to use a large time step or diffusion coefficient. Calculating the limits according to Δt and k confirms that Eq. 9 stays stable even with a high time step or diffusion coefficient:

$$\lim_{k \to +\infty} dT_i(t + \Delta t) = \frac{m_j \rho_i \left(T_{ji}(t)\right)}{m_i \rho_j + m_j \rho_i}$$

$$\lim_{\Delta t \to +\infty} dT_i(t + \Delta t) = \frac{m_j \rho_i \left(T_{ji}(t)\right)}{m_i \rho_j + m_j \rho_i}$$

6 Results

The proposed approach was tested on many scenes, with several values of s_i and in the context of melting and solidification with varying s_i. See the accompanying video for animations of all examples. The Stanford bunny in Fig. 1 shows how a whole object can melt in a plausible way. Notice how the top part of the bunny stays rigid while following the movement of the bottom part that is melting. In Fig. 2, the "two blocks" test was repeated with different rigidity factors. It shows that the rigidity factor s_i provides a continuous, visually realistic, and easy to control transition between solid and liquid. The "h" test of Fig 3 illustrates the robustness of this approach. In Fig 3(a), note how the left part remains rigid while the right part melts. Fig. 3(b) illustrates the propagation of two constraints through particles with different rigidities. Note that the top part remains solid. In Fig. 4, the armadillo arms are melted, and they stretch and deform in a reasonable way. In that exemple only, a subset of particles was not simulated and kept still (head, torso and legs). While these particles are fixed, they still influence the other particles in the simulation. The result shows the arms correctly stay attached, therefore the approach has the advantage that it can be coupled with particles that are not simulated by the solver.

Table 2 summarizes the statistics for our simulations. The optimization of the rigid forces takes most of the simulation time. The more rigid the particles are, the more time it takes to optimize the forces. However, simulations where s_i is larger often need fewer sub steps. This can be seen in the examples where $0.9 < s_i \leq 1.0$ for all particles. Although it takes more time to compute the rigid forces, fewer times steps are needed since the fluid remains still, hence the smaller simulation time. Simulations with varying viscosity suffer from larger simulation times per time step. This is caused by the constraint propagation mechanism which must iterate while the most rigid particles stabilize, before continuing the propagation.

Table 2 Statistics for the figures presented in this paper: number of particles, rigidity factor s_i, average time per frame, average time per time step, average number of time steps per frame, ratio of the time spent adjusting rigid forces with respect to the total time of a simulation time step. Note that all times are in seconds.

Fig. #	# part.	s_i	Avg. time / frame	Avg. time / $t.s.$	Avg. t. s. / frame	ratio t rigid / t total
1	52.4k	$[0.8, 1.00]$	480.1	50.3	9.5	0.97
2	52.7k	0.00	17.0	1.0	16.2	0.33
2	52.7k	0.25	88.1	9.0	9.7	0.88
2	52.7k	0.50	90.2	9.9	9.0	0.89
2	52.7k	0.75	56.8	7.4	7.6	0.91
-	52.7k	0.90	94.5	14.5	6.4	0.92
-	52.7k	0.95	92.0	15.9	5.7	0.93
-	52.7k	0.99	65.5	17.1	3.8	0.94
2	52.7k	1.00	23.5	21.4	1.1	0.97
3a	55.7k	$[0.8, 1.00]$	619.7	49.3	12.5	0.97
3b	55.7k	$[0.8, 1.00]$	848.7	53.1	15.9	0.98
4	17.1k	$[0.9, 1]$	165.2	14.1	11.6	0.92

Compared to the standard SPH approach [6], our approach is slower when the rigidity of the fluid is low. However, the computation times of theses approaches vary linearly with respect to the viscosity μ, while the variation of the perceived rigidity of the fluid is considerably sub-linear. This makes our approach more practical with particles that are almost rigid.

The proposed adaptive time step was compared with that of previous works [4,8]. For a completely liquid simulation (no rigid forces), our adaptive time step was more aggressive, by a factor of 1.5, with an average time step Δt of 0.00530 for our approach compared to 0.00347 [4] and 0.003221 [8]. During the impact of the water drop, a similar timestep value was observed compared to the previous approaches, while a higher value was observed once the fluid became more calm. This results in a faster simulation while preserving stability and visual realism as can be seen in the video.

Figure 5 compares our implicit heat diffusion with a standard explicit approach [2]. Our approach gives similar results, but stays stable even with a large time step (also see the accompanying video).

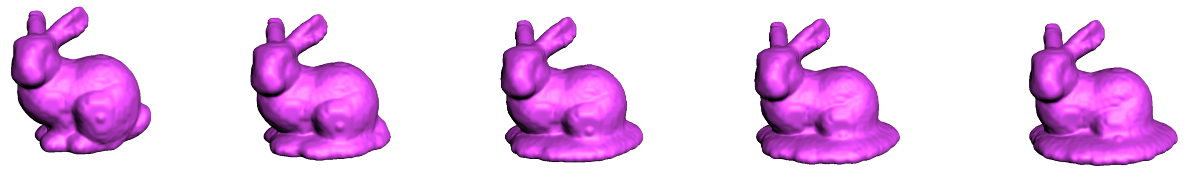

Fig. 1 Stanford bunny.

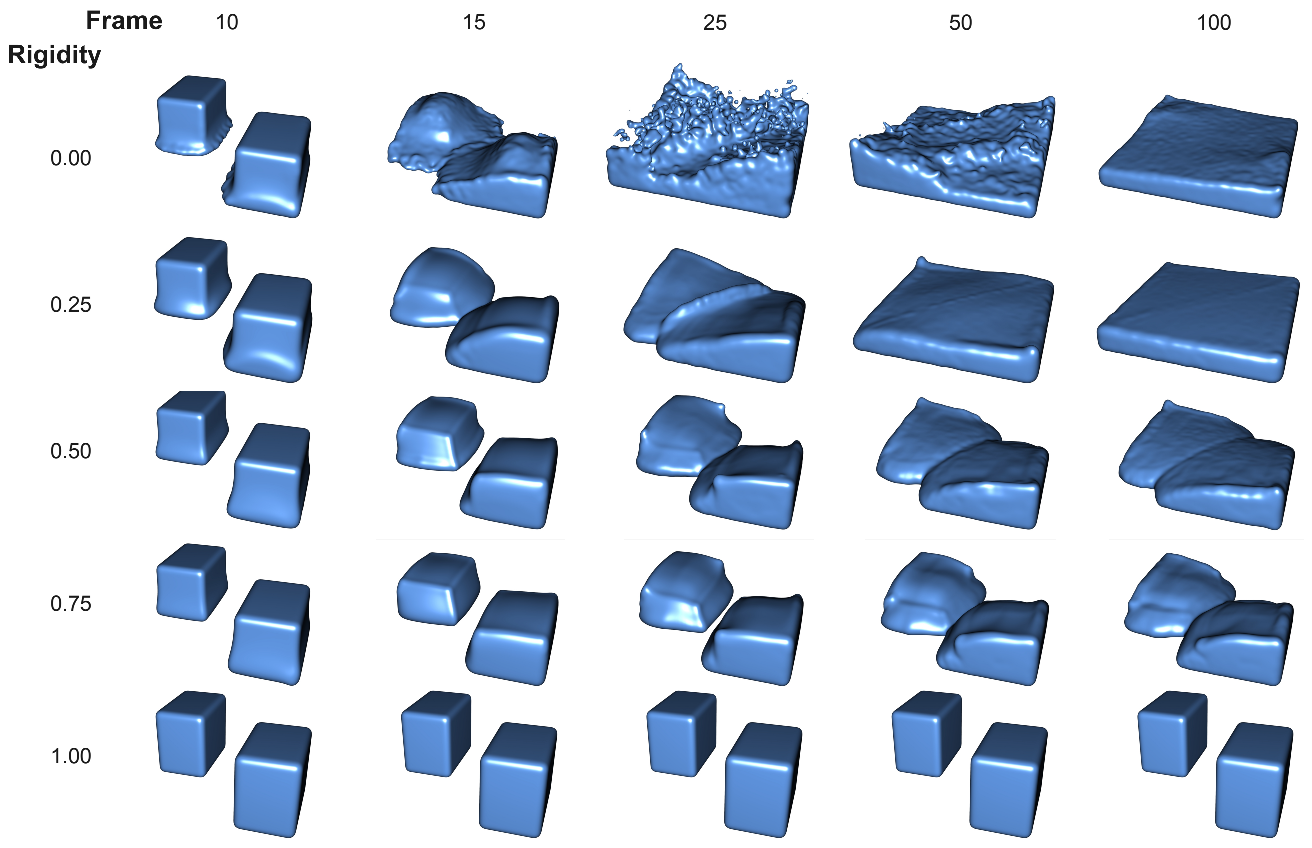

Fig. 2 Two blocks.

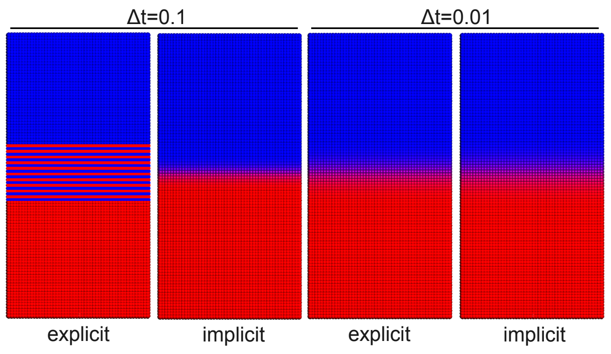

Fig. 5 Heat diffusion comparison. The color represents the temperature.

7 Discussion

In this paper we presented an approach for simulating extremely viscous fluids using a very stable simulation. Even though it is not physically based, it produces very convincing results. The approach enables the simulation of extremely viscous fluids that were very difficult to reproduce with standard SPH simulation, while providing a smooth transition between the liquid and solid states through parameter s_i which is intuitive to control.

The proposed approach requires long computation times. Fortunately, the computation of the rigid forces optimization could be parallelized since it is composed of several independent computations. Currently, our implementation is not optimized and only uses one core on a single processor.

Another limitation of the approach is that it does not handle rotations inside the material very well. In the Armadillo of Fig. 4, the arms fall straight instead of curving toward the body. This is a limitation of the

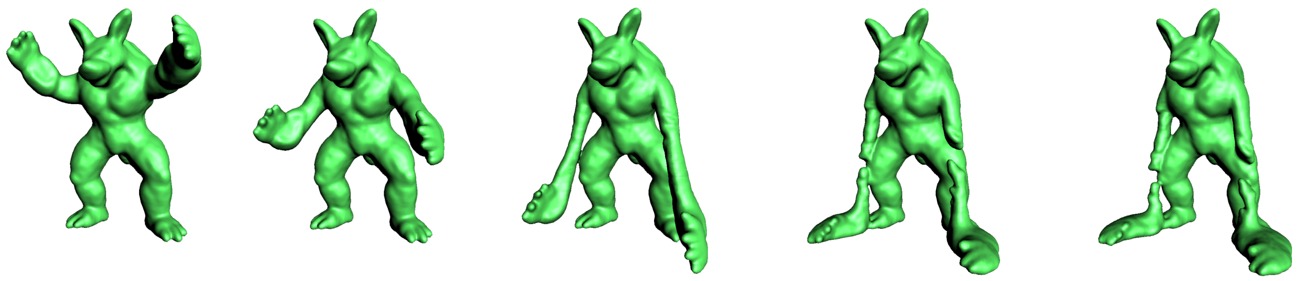

Fig. 4 Melting Armadillo's arms.

proposed model which does not consider the orientation variation of a particle according to its neighbors.

In this paper an SPH simulation is used, but this approach could be applied to other Lagrangian simulations. The only step needed is to add the optimization of the rigid forces before integrating the particles and adding the rigid forces to the advection equation.

8 Conclusion

In this paper, we presented an approach able to simulate highly viscous fluids using a Lagrangian simulation. The approach is based on a deformation error used in an iterative optimization of rigid forces. It allows a smooth transition between the liquid and solid states using a single parameter that can be easily controlled. While it may take more time per iteration to compute the rigid forces needed to keep the fluid rigid, the approach allows a larger time step to be used, hence making it more suitable for extremely viscous fluids, while maintaining stability. A constraint propagation mechanism also helps converging toward a more visually realistic solution. Furthermore, the proposed approach also introduces some numerical stability improvements to the SPH simulation. Firstly, it introduces a new way to compute a more efficient time step, to ensure simulation stability with less sub steps, thus less computational time. Since the technique presented in this paper allows larger time steps to be used, typical heat diffusion equations pose a problem as they are prone to instability when considering larger time steps. To overcome this problem, an implicit and unconditionally stable formulation of the heat diffusion equations has been derived. Together, these contributions allow the simulation of extremely rigid fluids using a Lagrangian simulation, while ensuring the stability of the simulation.

With respect to future work, the method is unable to realistically simulate rotations inside the material. This limitation can be overcome for completly rigid particles by using a a rigid body simulation as in the works of Solenthaler and Pajarola [12]. However, melting particles attached to theses particles won't follow correctly the rotational behavior. Therefore, it would be interesting to modify the deformation error model to allow these behaviors. Another interesting avenue for future work would be to conduct user studies to compare the realism of the proposed approach with respect to other current approaches.

Acknowledgements We want to thank FRQNT, CINQ and CRSNG for their financial support; Danny Bergeron, president of Mokko Studio, for making this project realizable; The Mokko Studio R&D team for its involvement in this project.

References

1. Carlson, M., Mucha, P.J., Van Horn III, R.B., Turk, G.: Melting and flowing. In: Proc. of 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation, SCA '02, pp. 167–174. ACM, NY, USA (2002)
2. Chang, Y., Bao, K., Liu, Y., Zhu, J., Wu, E.: A particle-based method for viscoelastic fluids animation. In: Proc. of ACM Symp. on Virtual Reality Software and Technology, VRST '09, pp. 111–117. ACM (2009)
3. Desbrun, M., Cani, M.P.: Smoothed particles: A new paradigm for animating highly deformable bodies. In: Eurographics Workshop on Computer Animation and Simulation (EGCAS), pp. 61–76. Springer-Verlag (1996)
4. Monaghan, J.J.: Smoothed Particle Hydrodynamics. Ann. Rev. of Astronomy and Astrophysics **30**, pages 543–574 (1992)
5. Monaghan, J.J.: Implicit sph drag and dusty gas dynamics. J. Comput. Phys. **138**, 801–820 (1997)
6. Müller, M., Charypar, D., Gross, M.: Particle-based fluid simulation for interactive applications. In: Proc. of ACM SIGGRAPH/Eurographics symposium on Computer animation, SCA '03, pp. 154–159 (2003)
7. Müller, M., Solenthaler, B., Keiser, R., Gross, M.: Particle-based fluid-fluid interaction. In: Proc. of ACM SIGGRAPH/Eurographics symposium on Computer animation, SCA '05, pp. 237–244. ACM, NY, USA (2005)
8. Paiva, A., Petronetto, F., Lewiner, T., Tavares, G.: Particle-based non-newtonian fluid animation for melting objects. In: Processing, 2006. SIBGRAPI '06. 19th Brazilian, pp. 78–85. SIBGRAPI (2006)
9. Paiva, A., Petronetto, F., Lewiner, T., Tavares, G.: Particle-based viscoplastic fluid/solid simulation. Comput. Aided Des. **41**, 306–314 (2009)

10. Rasmussen, N., Enright, D., Nguyen, D., Marino, S., Sumner, N., Geiger, W., Hoon, S., Fedkiw, R.: Directable photorealistic liquids. In: Proc. of ACM SIGGRAPH/Eurographics symposium on Computer animation, SCA '04, pp. 193–202 (2004)
11. Solenthaler, B., Pajarola, R.: Predictive-corrective incompressible SPH. ACM Trans. Graph. **28**, 40:1–40:6 (2009)
12. Solenthaler, B., Schläfli, J., Pajarola, R.: A unified particle model for fluid–solid interactions. Computer Animation and Virtual Worlds **18**(1), 69–82 (2007)