Ghost SPH for Animating Water

Hagit Schechter, Robert Bridson - University of British Columbia

ROMAIN CHAUSSONNIER, Institut Polytechnique de Paris, France

1 INTRODUCTION

The goal of the Ghost SPH algorithm is triple: to remove spurious tension artefacts at free surfaces, to prevent a loss of mass due to poor density estimation on boundaries, and to guarantee proper fluid cohesion with solids when separation forces occur. Those three problems are characteristic of the most simplistic SPH simulations and remained unsolved at the time of the publishing of Hagit Schechter and Robert Bridson's paper, in 2012 [1]. Our implementation of their improved algorithm shows that all issues were addressed properly, with little extra computational cost.

The implementation can be found at https://github.com/30L2AN0/Ghost-SPH.

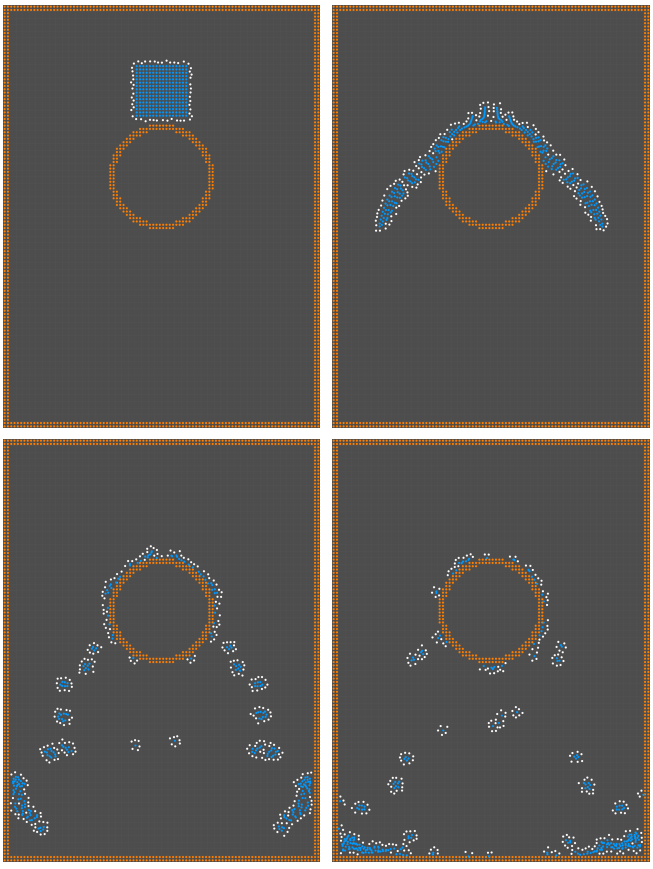

Fig. 1. A Ghost SPH simulation with $k = 2000$

Author's address: Romain Chaussonnier, Institut Polytechnique de Paris, France.

2 RELATED WORKS

The Ghost SPH method introduces an algorithm that handles properly solid and air boundaries for water. The obstacle researchers encountered can be summarised as a difficulty to satisfy two physical conditions during the simulation [3]. The first one regards velocity, and the second one regards pressure:

$$\vec{u} \cdot \hat{n} = \vec{u}_{solid} \cdot \hat{n} \qquad \text{at solid boundaries - no-stick} \tag{1}$$

$$p = 0 \qquad \text{at free surfaces (air)} \tag{2}$$

As a third condition, there can also be restrictions linked to viscosity if desired, such as the following one:

$$\vec{u} = \vec{u}^{solid} \qquad \text{at solid boundaries - no-slip} \tag{3}$$

Eq: (1) makes sure that the fluid only travels along the tangent of a solid at its contact. Eq: (2) makes sure that no pressure force will restrain the fluid from spreading at a free surface. This is a simplified model. In reality, since air is also a fluid, it can exert a non-zero pressure on our water. However, the pressure is quite negligible in most cases, and choosing it as zero is a commonly accepted approach for simulations that do not require the utmost realism. Eq: (3) enables particles to glue fully to solids they are in contact with. The smaller the scale of observation of the liquid, the more accurate this condition becomes. If we zoom in enough, we should indeed observe complete adhesion.

The Ghost SPH method is a lagrangian approach to those three desired conditions. However, today, there are valid solutions brought by eulerian methods. The most common one used in production today is the PIC/FLIP simulation. It uses grids to store values and the goal of the algorithm is to retrieve the velocity $\vec{u}$ with a numerical iterative solving of the Navier Stokes equations:

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u} \tag{4}$$

$$\nabla \cdot \vec{u} = 0 \tag{5}$$

The algorithm is the following [3]:

- Start with an initial divergence-free velocity field $\vec{u}^0$
- For time step $n = 0, 1, 2, \ldots$
  - Determine a good time step Δt to go from time t_n to time t_{n+1}.
  - Set $\vec{u}^A = advect(\vec{u}^n, \Delta t, \vec{u}^n)$.
  - Add $\vec{u}^B = \vec{u}^A + \Delta t \vec{g}$.
  - Set $\vec{u}^{n+1} = project(\Delta t, \vec{u}^B)$.

Boundaries handling is done in the $project$ part. It develops as such :

$$\vec{u}^{n+1} = \vec{u} - \Delta t \frac{1}{\rho} \nabla p \tag{6}$$

Let's imagine ourselves in 1-dimensional grid. The cell i is liquid, and the cell $i + 1$ is air. ∇p can be rewritten $\nabla p = \frac{p_i - p_{i+1}}{\Delta x}$. Then, by choosing the ghost pressure p_{i+1} intelligently, Eq: (2) can be verified. One simply chooses $p_{i+1}^{ghost} = 0$.

For Eq: (1), unfortunately, it is more difficult. One needs to find a p^{ghost} which would respect the Neumann boundary condition on ∇p:

$$\frac{\Delta t}{\rho}\nabla p \cdot \hat{n} = (\vec{u} - \vec{u}^{solid}) \cdot \hat{n} \tag{7}$$

The problem is eventually solved by looking at surrounding cells one by one. Let's go back to 3D and consider our liquid cell i once again. If the neighbour cell j along one of the three dimensions is a solid one, then u's component along this dimension will be u^{solid}'s component. If the cell is liquid or air, Eq: (6) is used in 1D along x, y or z accordingly, with $p_{neighbor}$ or $p^{ghost} = 0$ respectively.

Similarly, the no-slip condition Eq: (3) can be implemented: if one of the neighboring cells is solid, then $u = u^{solid}$.

The goal of H. Schechter and R. Bridson was to keep this precise boundary handling while benefiting from the simplicity of the lagrangian SPH method.

3 TECHNICAL DETAILS

Our implementation is based on the Algorithm Fig: 2 showed in [1].

Algorithm 3 Ghost SPH Simulation Step

Sample new liquid particles, or air particles, if needed this step.
Compute density:
for all liquid particles i **do**
    Compute ρ_i with Equation 1
for all solid particles i **do**
    Find closest liquid particle j
    $\rho_i \leftarrow \rho_j$
Compute pressure:
for all particles i **do**
    Compute pressure p_i using the equation of state
Compute liquid accelerations and velocities:
for all liquid particles i **do**
    Compute the acceleration $\mathbf{a}_i$ from gravity and pressure
    $\mathbf{v}_i^* \leftarrow \mathbf{v}_i + \delta t \cdot \mathbf{a}_i$
Prepare solid boundary conditions:
for all solid particles i **do**
    Find closest liquid particle j
    $\mathbf{v}_i^* \leftarrow \mathbf{v}_i^N + \mathbf{v}_j^T$ as in Equation 3
Apply XSPH artificial viscosity:
for all liquid particles i **do**
    Update $\mathbf{v}_i^{\text{new}}$ using Equation 2
Extrapolate velocity into air:
for all air particles i **do**
    Find closest liquid particle j
    $\mathbf{v}_i^{\text{new}} \leftarrow \mathbf{v}_j$
Update positions:
for all liquid and air particles i **do**
    $\mathbf{x}_i^{\text{new}} \leftarrow \mathbf{x}_i + \delta t \cdot \mathbf{v}_i^{\text{new}}$

Fig. 2. Algorithm used in H. Schechter's publication

The skeleton of the algorithm is a common SPH simulation which will be called "Basic SPH" in the rest of the paper. The following subsections expose which elements of Basic SPH simulation were untouched and which elements were improved in the Ghost SPH implementation as well as how they were modified.

3.1 Density

The density evaluation of Basic SPH and Ghost SPH is made using the popular summation:

$$\rho_i = \sum_j m_j W_{ij} \tag{8}$$

It is used as such for liquid, solid, and air particles

3.2 Pressure

The Tait equation is used to evaluate pressure.

$$p_i = k\left(\left(\frac{\rho_i}{\rho_0}\right)^7 - 1\right) \tag{9}$$

This equation is also quite popular and was not introduced in H. Schechter and R. Bridson's paper. However, the expression is sometimes clamped to avoid negative pressures: $p_i' = max(p_i, 0)$. The Basic SPH implementation uses the clamped expression, when the Ghost implementation uses the regular one. The choice to keep negative pressures was made in order to simulate proper physical properties of fluids. Indeed if the pressure only pushes particles away from each other, it results in air creation between them which should not be possible. It becomes particularly relevant in the hydrostatic test Fig: 3 where a dense equilibrium is desired.

The formula to then update particles' acceleration is Monaghan's usual pressure gradient:

$$f_i^{pressure} = -m_i \sum_j m_j \left(\frac{p_j}{\rho_j^2} + \frac{p_i}{\rho_i^2}\right)\nabla W_{ij} \tag{10}$$

3.3 Viscosity

A commonly-used formula for viscosity in SPH simulations is Morris et al.'s:

$$f_i^{viscosity} = 2\nu m_i \sum_j \frac{m_j}{\rho_j}\left(v_j - v_i\right)\frac{(x_j - x_i)\cdot\nabla W_{ij}}{||x_j - x_i||_2 + 0.01h^2} \tag{11}$$

where h is the spacing between particles and ν is a chosen coefficient. It is the one used in Basic SPH.

In Ghost SPH, however, Monhagan's XSPH artificial viscosity is prefered, updating velocities directly:

$$v_i^{new} = v_i^* + \epsilon \sum_j \frac{m_j}{\rho_j}\left(v_j^* - v_i^*\right) W_{ij} \tag{12}$$

where $\epsilon = 0.05$, the velocities v_i^* are either step-advanced velocities $v_i^* = v_i + \delta t \cdot a_i$ when dealing with liquid particles, or ghost velocities when dealing with solid particles. See Fig: 2.

Let's take a moment to clarify what those ghost velocities are. If the no-stick condition Eq: (1) is needed, then $v_i^{ghost} = v_N^{solid} + v_T^{liquid}$. However, if the fluid is viscous, one would prefer the no-slip condition. In that case, $v_i^{ghost} = v^{solid}$, gives the proper adherence.

3.4 Acceleration structure

In both Basic SPH and Ghost SPH, one needs to access the neighboring particles of a considered one. For this reason, a grid is used. At each step, the grid is updated to store all the particles that are contained in each cell. Then, for each particle i, we look at the cells

within reach of the kernel radius, and evaluate all the particles j they contain. If j is not too far, it is stored in a container for i.

In Ghost SPH, at different steps of the algorithm Fig: 2, the closest liquid neighbour of every solid or air particle is also necessary. We decided to find and store it as well using the acceleration structures described above.

3.5 Sampling

If we have a look at the global algorithm Fig: 2, every step has been explained, except for the very first one. At each new pass through the loop, new liquid and air particles can be sampled. In the proposed implementation, there were no sources of liquid. So no fresh particles appear at all. There is no sampling needed. For air particles, it is different. Their desired behaviour is to surround liquid appropriately. Because air moves at each step, a new sampling is regularly necessary to keep fitting the shape of the liquid. It is done every 20 steps in our implementation.

Resampling is done using Fast Poisson Disk Sampling algorithm [2]. This algorithm suggest to create within a disk around a central particle. In the case of SPH, neighbours must not be created further than the kernel radius. This restriction was ensured as well while adding the sampling algorithm.

4 MODIFICATIONS & IMPROVEMENTS

In contrast to H. Schechter and R. Bridson's paper, the implementation proposed here do not use level sets. They are introduced mostly to sample liquids and solids from their contours. Indeed, it is to find the boundary of an environment that level sets are usually used. But here, to simplify the model, particles were disposed by hand.

Because there were no level sets, and because no alternatives were implemented, it is not possible to determine whether we stand inside a volume in our algorithm, whether it is solid or liquid. The consequence is that liquid can enter the solid in the simulation. The no-stick condition only intervenes as a repulsive force, but has no way to ensure the non-penetration and water particles can not be projected back to the boundary when trespassing it. To address the problem, the pressure was modified for solid particles. In Eq (9), the factor k is chosen as 20000 instead of 2000, and it is multiplied by 10 again for solid particles' pressures. The repulsion from solid to the liquid was then drastic and no trespassing occurred. As a consequence, the negative pressures and so the attraction of the solid also became higher. But this secondary effect was appreciated as it simulates water drops sticking to solid after the passage of a flow. It was decided that it should be kept.

5 OBSERVATIONS & FUTURE WORKS

Placing liquid particles by hand had another benefit apart from simplicity, but also came with a withdraw. The benefit is that the simulation displays symmetric patterns when the original disposition is supposed to be symmetrical. It is ensured by the symmetrical disposal of the initialised particles. Nevertheless, those symmetrical dispositions at the start of the simulation are also responsible for strange, non-physical behaviours. Aligned in vertical columns for example, particles had the tendency to stack on top of each other Fig: 4.

One of the main goals of the original paper was attained in our implementation. Density for liquid particles is correctly computed at the boundary, thanks to the presence of air particles. Therefore, pressure is also well computed. The hydrostatic test shows that in Basic SPH, the lower density on the sides and particularly on the angles results in a shrinking of the global shape. In Ghost SPH, either no shrinking can be seen, in the case of a manual sampling of air particles, or the shrinking is extremely reduced and takes time to appear, in the case of a stochastic sampling. But even this problem can be tackled by increasing the number of air particles and by generating more chaotic distribution at the start.

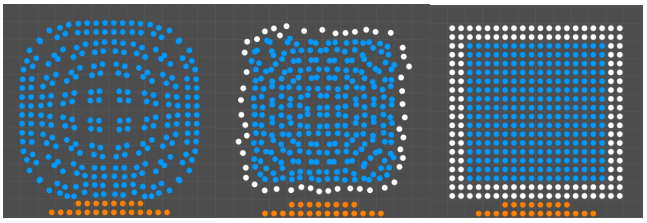

Fig. 3. The hydrostatic test: with Basic SPH after 250 iterations (left), with Ghost SPH after 500 iterations with stochastic sampling (middle), and even sampling (right)

The stickiness of the water is also very well addressed in H. Schechter and R. Bridson's paper. When in contact with the solid, it should be hard for a liquid particle to break away from the surface, as this would result in the creation of air out of nowhere. A fluid surface can not separate from a solid surface except if another fluid takes its place. This phenomenon is not treated in Basic SPH. Water is simply pushed away like a rigid body. In ghost SPH, once water is in contact with the solid, it gets difficult for it to separate. The consequence is that the liquid slides along the solid surface, as one would except. It is easier to observe the phenomenon by reducing the pressure. But as doing so, we increase the risk of solid penetration. Ideally, in future works, we would reduce the pressure and introduce level sets to guarantee the respect of the no-slip condition.

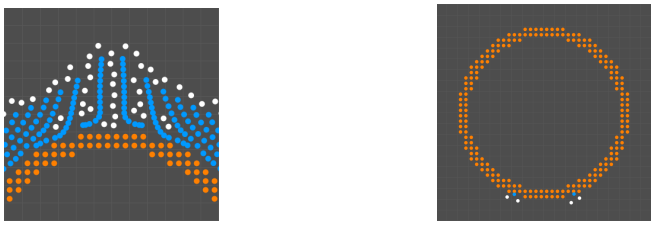

Fig. 4. Left: columns forming as a consequence of a perfectly even distribution with high viscosity ($\epsilon = 0.1$). Air is created between them. Right: droplets of water sticking to the solid at the end of a simulation.

6 CONCLUSION

Ghost SPH is a solid improvement of the popular SPH. With this method, density is properly calculated at free surface, without introducing any unnecessary pressure. As a result, we always get a pressure $p^{ghost} = 0$ for surrounding air particles. On this part, it

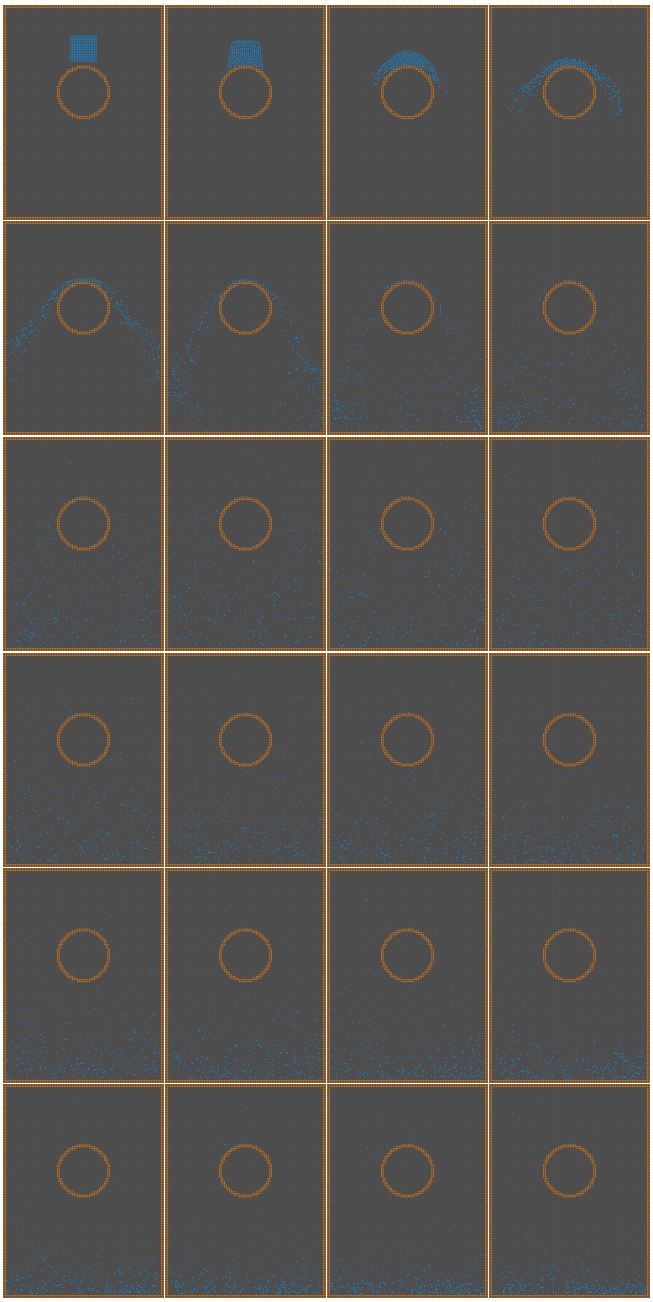

Fig. 5. A Ghost SPH simulation with $k = 20000$

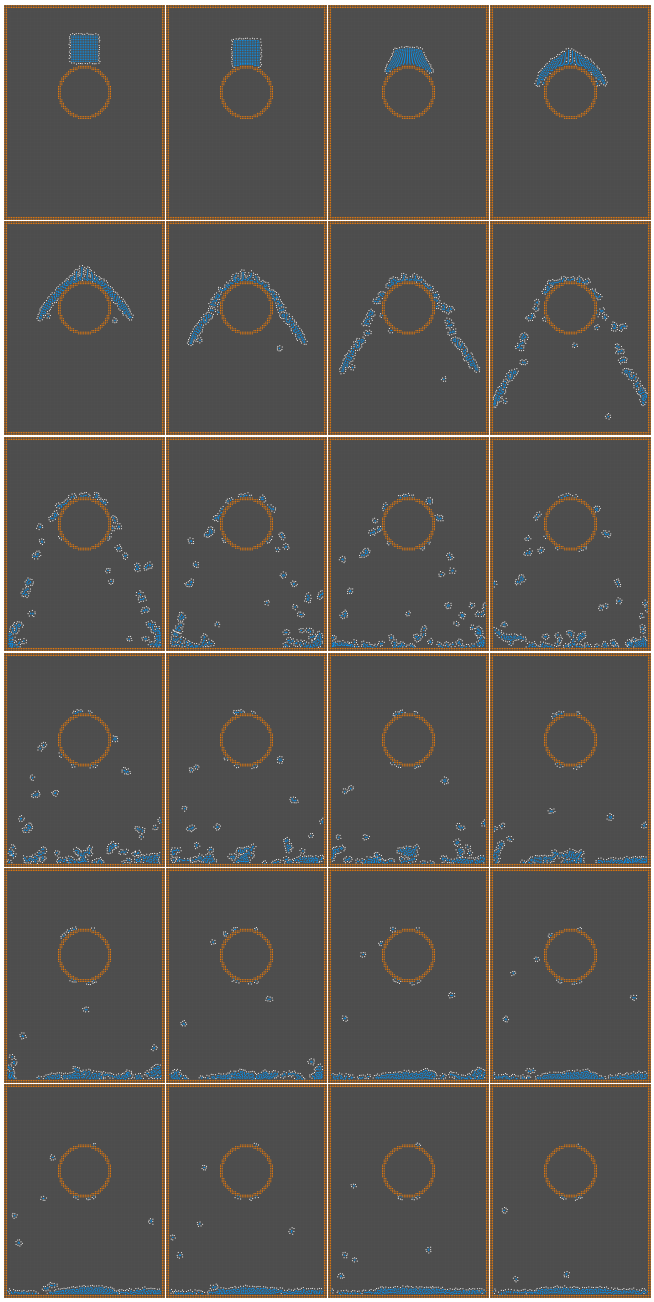

Fig. 6. A Ghost SPH simulation with $k = 20000$

is as efficient as common eulerian approaches. The no-stick and no-slip conditions are well taken into account with XSPH viscosity, and even if it does not prevent liquid penetration in solids, it is easily fixed by the use of level sets. Ghost SPH does not create incredibly precise simulations due to its stochastic nature - an inheritage of Basic SPH in way - but does fix important issues of common simplistic models and produces pleasing results for the eye at a very low cost.

REFERENCES

[1] Hagit Schechter, Robert Bridson, "Ghost SPH for Animating Water" In ACM Transactions on Graphics 2012, Volume 31, Issue 4, Article No.: 61, pp 1–8
[2] Robert Bridson, "Fast Disk Sampling in Arbitrary Dimensions" In SIGGRAPH sketches, 2007
[3] Robert Bridson, Matthias Müller-Fischer, Eran Guendelman, "Fluid Simulation" In SIGGRAPH 2006 Course Notes