Smoothed Particle Hydrodynamics Solver for 3D Fluid Simulations

Guilherme G. Schüler
Computer Animation 2022/1 – PUCRS
Professora Soraia Musse

ABSTRACT

Fluid simulation refers to a set of techniques and algorithms in computer graphics to procedurally render fluids in a realistic manner. Distinct techniques may aim for specific fluid behavior – such as vast volumes or turbulent flows. This paper describes an implementation of Smoothed Particles Hydrodynamics (SPH) in 3D space, a particle-based technique known for its real-time rendering capability. The resulting algorithm allows real-time rendering while simulating under ±700 particles.

1 INTRODUCTION

Fluid phenomena are everywhere: wind gusts, ocean waves, air flow through turbines and petroleum flow through pipelines. Simulating these phenomena has been an interest and a challenge for the computer graphics community. There are many techniques available, sometimes diverging on its result. Some simulations aim for accuracy regarding world phenomena, while others aim for easy and fast rendering. Typically, fluid phenomena is simulated offline and visualized in a later stage [4].

A common trade-off regarding fluid simulation techniques is between render speed and physics-accuracy. As a general rule, real-time rendering comes at the expense of accuracy[2, 7]. Smoothed Particles Hydrodynamics (SPH) is a technique most known for its real-time rendering capability, while maintaining accuracy, both physically – by its analogy between particle and molecule – and visually – by the absence of mesh rendering problems present in other techniques.

This paper describes an implementation of the SPH technique for simulating fluids in 3D space. Related work is discussed in Section 2. Section 3 contains the description of the relevant equations and the implementation of SPH, as well as the structure of the simulation scene and performance testing. Results and limitations of the implemented technique are discussed in Section 4. Finally, Section 5 offers considerations and prospects for future work.

guilherme.schuler@edu.pucrs.br, ,

2 RELATED WORK

Smoothed Particle Hydrodynamics was originally developed in 1977 for solving astrophysical problems [1]. It was first implemented for fluid simulation purposes by Müller, Charypar, and Gross [4] and is commonly called weakly compressible SPH (WCSPH). One of the main drawbacks of Müller's proposed technique is its enforcing of incompressibility, that is, that the overall pressure of the fluid remains the same throughout the simulation. WCSPH uses a equation of state (EOS) to relate the particle's pressure and density. Using such equations limits the permissible time-steps of the simulation, affecting its computational cost. Solenthaler and Pajarola [6] proposed a method by which particle's position and velocity values are projected forward in time, density is computed, and pressure is updated accordingly. This specific method is named predictive-corrective PCSPH (PCSPH) and is part of a set of techniques denominated incompressible SPH (ISPH). This discards EOS and allows for bigger time-steps while maintaining accuracy.

Another common problem with SPH methods, including PCSPH, is its instability – most notable when particles do not have enough neighbors for computing accurate density and pressure values. Using sufficiently small time-steps or a sufficiently large set of particles generally solve this, but at the cost of performance. Recently, SPH has been implemented using the Position Based Dynamics (PBD) framework [3, 5], which completely omits velocity treatment. Using this framework let SPH solve position values directly, which in turn deals with instability. In PCSPH, for example, an isolated particle will have increasingly negative pressure values; when nearing a neighbor – in a subsequent iteration – considerably inaccurate pressure force is applied. By considering only position, pressure is completely bypassed and such problems are absent.

Given that, the following implementation is based on the original SPH formulation for fluids simulation, and does not have either PCSPH optimizations or PBD framework functionality. Next section describes the algorithm developed and presents some experiments.

3 METHODS

This section describes the SPH algorithm implementation. The algorithm is based on Müller *et al.*'s paper [4] and was done using C# in Unity (2021.3.3f1). The following is divided into subsections. First, the proccess of modelling fluids as particles is briefly explained. Then the algorithm is discussed, mainly the computation of particles' properties and forces. Integration scheme and simulation parametrization is also discussed.

3.1 Particles

Isothermal fluids are usually described by two equations, describing mass and momentum conservation. Eqn. (1) and (2) are, respectively, such equations. Using particles let us completely discard Eqn. (1), since the number of particles is constant and so is its mass. Furthermore, since SPH is a Lagrangian approach – that is, particles are said to be moving *with* the fluid –, the partial derivative can be replaced by a simple derivative of type $\frac{D\mathbf{v}}{Dt}$ and the term $\mathbf{v}\nabla\mathbf{v}$ can be ommited.

$$\frac{\partial\rho}{\partial t} + \nabla\rho\mathbf{v} = 0 \tag{1}$$

$$\rho(\frac{\partial\mathbf{v}}{\partial t} + \mathbf{v}\nabla\mathbf{v}) = -\nabla p + \rho\mathbf{g} + \eta\nabla^2\mathbf{v} \tag{2}$$

The right-hand side is left with a density term and the derivative, while the left-hand side contains three forces: pressure, gravity, and viscosity. These forces are calculated for each particle and a resulting acceleration $\mathbf{a}$ is computed. By means of $\mathbf{a}$, then, the particles' positions are updated in each iteration. The equation for acceleration exerted on particle i is given by Eqn. (3):

$$\mathbf{a}_i = \frac{\mathbf{f}_i}{\rho_i} \tag{3}$$

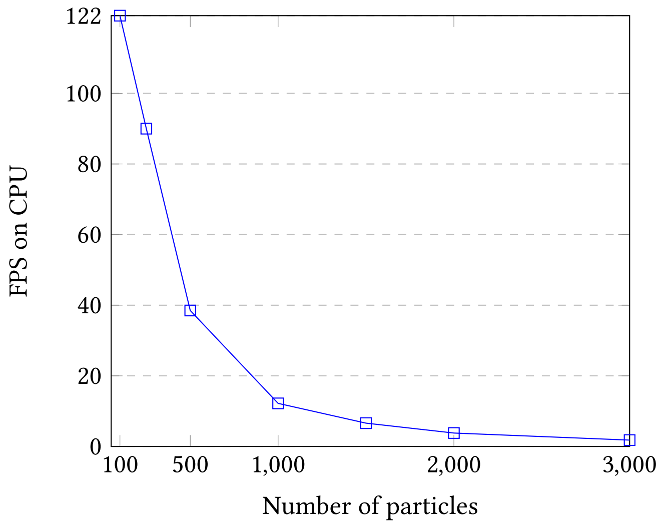
3.2 Density, Pressure, and Forces

Particles have vector properties – in 3D space – and scalar properties. Vector properties are position, velocity and acting forces; scalar properties are density and pressure. While the mass of each particle is constant, density varies and is evaluated at every time-step. Density ρ of particle i at position $\mathbf{r}$ is given by Eqn. (4): the sum of all j-particles times a weight function acting on a radially symmetric radius of h.

$$\rho_i = \rho(r_i) = \sum_j m_j W_p(|r_i - r_j|, h) \tag{4}$$

Then, the pressure property of particle i is computed by an equation of state $p = k(\rho - \rho 0)$, where k is a gas constant and $\rho 0$ is the rest density of the simulated fluid. What remains is to calculate the acting forces on each particle. This is done by a loop over all other particles and weighting of their contributions. Eqn. (5) and (6) describe the force computation:

Figure 1: Rendering speed on CPU



$$F_i^p = -\sum_j m_i m_j (\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2})\nabla W_s(|r_i - r_j|, h) \tag{5}$$

$$F_i^v = -\eta \sum_j m_j \frac{u_j - u_i}{\rho_j}\nabla^2 W_v(|r_i - r_j|, h) \tag{6}$$

Each of the three kernel functions (or weight functions), is described below. These are proposed by Müller *et al.* and exactly reproduced in the current implementation.

$$W_p(r, h) = \frac{315}{64\pi h^9} \tag{7}$$

$$W_s(r, h) = \frac{15}{\pi h^6} \tag{8}$$

$$W_v(r, h) = \frac{15}{2\pi h^3} \tag{9}$$

3.3 Integration and Simulation

For integration of Eqn. (3), forward Euler method is used, by its simplicity. In, Muller2003, Leap-Frog scheme is used. The time-set that best fitted this implementation was 8ms. Values over 10ms generated unexpected behavior, such as abrupt gains of large velocities. All of the kernel functions were pre-computed for performance sake. The smoothing radius was set to 1, best suited for the corresponding particle radius chosen. Although water's rest density is $1g/cm^3$ and viscosity is 0.01 poise, these were set respectively to 15 and 1. Gas constant k was set to 2000. Parametrization was done in accordance with expected behavior of the fluid.

4 RESULTS

Three experiments were developed. Each one of them had different particle instantiation formations, shown in Fig. 2. Case 1 is a parallelepiped of 1000 particles; case 2 is a front collision of two sets of 500 particles; and case 3 is a side collision of two sets of 500 particles. Each case was executed with 1000 particles with a radius of 1 and mass of 0.25. It was found that different cases, despite having the same parameters and equal number of particles, had distinct rendering speeds. Namely, case 1 had a minimum value of frames per second (FPS) of 14.2, case 2 had 23.9 and case 3 had 23.8. This was the case because of a larger number of particles being computed by the kernel functions, as a result of smaller distance in-between particles. A set of 5 frames from Case 1 is illustrated in Fig. 3.

Furthermore, Fig. 1 shows the average rendering speed of the three cases, related to the number of particles simulated. It is observed that, for values less than ±700 particles, the simulation has real-time rendering speed.

5 CONCLUSIONS AND FUTURE WORK

This paper described an implementation of SPH, as formulated in Müller *et al.*'s 2003 paper. This technique aims to be an approachable way of simulating fluids in real-time, while keeping accurate enough physical attributes. It was observed that the simulation was able to run in real-time rendering speed while under ±700 particles.

The implemented algorithm can be greatly optimized. Regarding integration, switching from forward Euler method to Leap-Frog would increase performance. Most of the computational cost of the algorithm comes from the $O(n^2)$ complexity of the solver. A simple way of decreasing cost is to implement a spatial grid and pre-sorting particles into it. Future work should start by the adding the grid functionality.

REFERENCES

[1] R. A. Gingold and J. J. Monaghan. 1977. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society* 181, 3 (12 1977), 375–389. https://doi.org/10.1093/mnras/181.3.375 arXiv:https://academic.oup.com/mnras/article-pdf/181/3/375/3104055/mnras181-0375.pdf

[2] Steven J. Lind, Benedict D. Rogers, and Peter K. Stansby. 2020. Review of smoothed particle hydrodynamics: towards converged Lagrangian flow modelling. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 476, 2241 (2020), 20190801. https://doi.org/10.1098/rspa.2019.0801 arXiv:https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.2019.0801

[3] Miles Macklin and Matthias Müller. 2013. Position Based Fluids. *ACM Trans. Graph.* 32, 4, Article 104 (jul 2013), 12 pages. https://doi.org/10.1145/2461912.2461984

[4] Matthias Müller, David Charypar, and Markus Gross. 2003. Particle-Based Fluid Simulation for Interactive Applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, California) *(SCA '03)*. Eurographics Association, Goslar, DEU, 154–159.

[5] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. 2006. Position Based Dynamics. In *Vriphys: 3rd Workshop in Virtual Realitiy, Interactions, and Physical Simulation*, Cesar Mendoza and Isabel Navazo (Eds.). The Eurographics Association. https://doi.org/10.2312/PE/vriphys/vriphys06/071-080

[6] B. Solenthaler and R. Pajarola. 2009. Predictive-Corrective Incompressible SPH. *ACM Trans. Graph.* 28, 3, Article 40 (jul 2009), 6 pages. https://doi.org/10.1145/1531326.1531346

[7] Zhi-Bin Wang, Rong Chen, Hong Wang, Qiang Liao, Xun Zhu, and Shu-Zhe Li. 2016. An overview of smoothed particle hydrodynamics for simulating multiphase flow. *Applied Mathematical Modelling* 40, 23 (2016), 9625–9655. https://doi.org/10.1016/j.apm.2016.06.030

Figure 2: From top to bottom, Cases 1, 2, and 3

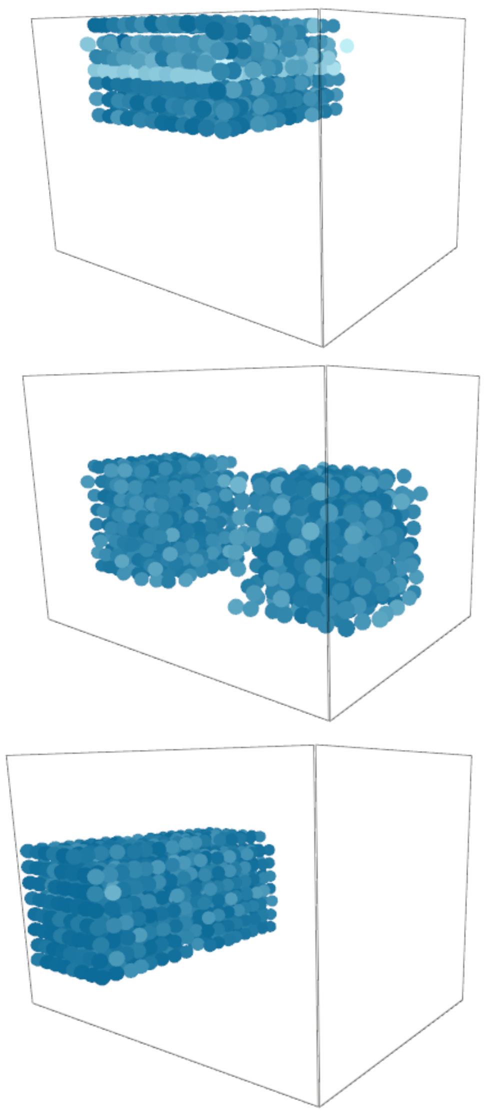


Figure 3: Five frames of water block case