# Smoothed Particle Hydrodynamic Fluid Simulation

Ryan L. Guy

## 1 Abstract

This paper presents a simple implementation of a liquid fluid simulation using the method of Smoothed Particle Hydrodynamics (SPH). The SPH model is a Lagrangian method that can be used to model fluid flow by treating each particle as a discrete element of fluid. The implementation covers a set of basic equations used to model a fluid, a spatial data structure to accelerate fluid calculations, and a method to handle interaction between fluid and solid obstacles.

## 2 Introduction

Each SPH fluid particle is treated as a discrete element of fluid. The motion of a fluid particle is influenced by other particles within some finite radius. The influence a particle has on another is dependent on the distance between the particles. How much influence a particle has on another depends on the smoothing kernel, which is a single variable function dependent on radius. Since the SPH method is pure Lagrangian, a particles motion is based on the current acceleration value at a point in time. The acceleration of a particle takes into account the density, pressure, and relative velocities of it's surrounding particles.

In order to calculate the density and pressure of a particle, we need to locate it's neighbour particles within some radius. To efficiently find neighbours for each particle, a spatial grid is used to refine the search to only particles that are near to each other.

In this implementation, solid objects are modelled in two different ways. The fluid boundaries are treated as planes that a particle cannot intersect. These planes also provide a repulsive force on the particles to keep them away from the boundaries. Moving objects are treated as a set of fluid particles whose motion are controlled by an external source. These obstacle particles do not move based on their acceleration, but do keep track of their density and pressure so other particles can react to them.

## 3 Summary of Cited Works

### 3.1 Physically-Based Fluid Modeling using Smoothed Particle Hydrodynamics (Trina M. Roy Masters Thesis)

This thesis gives implementation details such as fluid initialization, time step calculation, and an algorithm for modelling SPH fluid dynamics.

### 3.2 Fluid Flow for the Rest of Us: Tutorial of the Marker and Cell Method in Computer Graphics (D. Cline, D. Cardon, P. K. Egbert)

This paper gives implementation details for a grid based fluid. The dynamic spatial grid data structure described in the paper was modified for this implementation.

### 3.3 An Initiation to SPH (L. Braune, T. Lewiner)

This paper gives implementation details such as fluid initialization and provides constants for a stable fluid simulation.

### 3.4 Lei Wang, University of Ohio (Website)

This website gives an algorithm and equations for calculating density, pressure, viscosity, and acceleration.

### 3.5 SPH Survival Kit (UMU Fluid Lecture)

This lecture resource provides equations used for the kernel functions in the implementation, along with equation modifications for more numerically stable calculations.

## 4 Implementation

This section gives relevant implementation details for this SPH fluid simulation and will cover governing equations, time step calculation, dynamic spatial grid data structure, object-fluid interaction, and user interface.

### 4.1 Governing Equations

In order to find a particle's acceleration, it's density and pressure must be calculated. The following equation sums the masses of a particle's neighbours weighted by a kernel function.

$$p_i = \sum_j m_j W_{ij}$$

Where p is density, m is mass and W is the Poly6 smoothing kernel:

$$W_{ij} = \frac{315}{64\pi h^9}(h^2 - r^2)^3$$

Where h (chosen as 0.2) is the smoothing radius and r is the distance between particles. This equation has the advantage of not computing a square root for the distance between particles since r is already squared in the equation.

Once a particle's density is found, the pressure can be computed via the ideal gas law:

$$P = K(p - p_0)$$

Where P is pressure, p is the density of the particle, p0 is the reference density of the fluid and K is a pressure constant. For this implementation the reference density and pressure constant were both chosen to be 20. In this equation, density is set to the reference density if the value is less than the reference density to avoid negative pressures.

Now that the pressure and density of all particles are calculated, the acceleration of a particle can be computed as:

$$a_i = -\sum_j \frac{m_j}{m_i}\frac{P_i + P_j}{2p_i p_j} \bigtriangledown W_{ij}\hat{r_{ij}}$$

Where a is the acceleration, P is the pressure of the respective particle, r is the normalized vector of the position of particle j to particle i and W is the gradient of the following Spiky kernel smoothing function:

$$W_{ij} = -\frac{45}{\pi h^6}(h - r)^2$$

The above acceleration is caused by the mass, density and pressure of the particles. To dampen the motion of the system, the velocities of surrounding particles can be taken into account to add a viscosity term to the acceleration:

$$av_i = \epsilon \sum_j \frac{m_j}{m_i}\frac{1}{p_j}(v_j - v_i) \bigtriangledown^2 W_{ij}\hat{r}_{ij}$$

Where av is the acceleration due to viscosity, epsilon is the viscosity constant (chosen as 0.018 in this implementation), v is the velocities of the respective particle, and W is the laplacian of the following viscosity smoothing kernel:

$$W_{ij} = -\frac{r^3}{2h^3} + \frac{r^2}{h^2} + \frac{h}{2r} - 1$$

In addition to adding the viscosity term to the particle's acceleration, the force of gravity is also added.

### 4.2 Calculating Time Step

The goal of choosing a time step is to choose the largest value such that a particle will move less than one smoothing radius. The following equation was used in this implementation:

$$\triangle t = max\left(\frac{Ch}{max(1, v_{max})}, \sqrt{\frac{h}{a_{max}}}, \frac{Ch}{c_{max}}\right)$$

Where v and a are the maximum velocities and accelerations in the particle system, c is the speed of sound, and C is the Courant safety constant, chosen as 1.0 in this simulation. The safety constant can be set to a higher value for larger timesteps. The speed of sound is calculated with the following equation:

$$c = \sqrt{\frac{\gamma P}{p}}$$

Where c is the speed of sound and gamma is the ratio of specific heats, which is set to 1.0 in this simulation.

### 4.3 Dynamic Spatial Grid

In order to accelerate the nearest neighbour search of all particles, a fixed width spatial grid data structure was implemented. The size of the grid is dynamic, and grid cells only exist where there is fluid. This means that the memory usage of the grid is proportional to the amount of fluid.

Grid cells are contained in a chained hash table according to the following hash function:

$$hash = 541i + 79j + 31k$$

Where i, j, k are the respective x, y, z grid indices. The grid cell width is set to one smoothing radius.

Since the particles are only looking up the nearest neighbours within one smoothing radius, we know that at most the particle's cell and 26 cell neighbours need to be looked up. To speed up this process, and to avoid repetitive hash lookups, each grid cell keeps track of it's direct neighbours in an array.

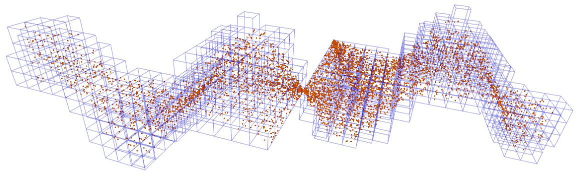

*Figure 1. Dynamic Spatial Grid*

### 4.4 Object-Fluid Interaction

Object-fluid interaction is handled in two separate ways depending on the type of object. Static Fluid boundaries are handled with planes that the particles cannot intersect. The boundary planes are given an increasing repulsive force to push away particles that are too close to the boundary. If a particle hits a boundary, the particle's velocity is adjusted so that it bounces off the surface.

Moving objects are handled by constructing the shape with fluid particles. These obstacle particles interact with regular fluid particles as if they were a fluid, except that their motion is controlled by an external force instead of acceleration. Since obstacle particles are not affected by acceleration, only their pressure and density values are computed. Each obstacle particle is updated for velocity depending on the motion of the object.

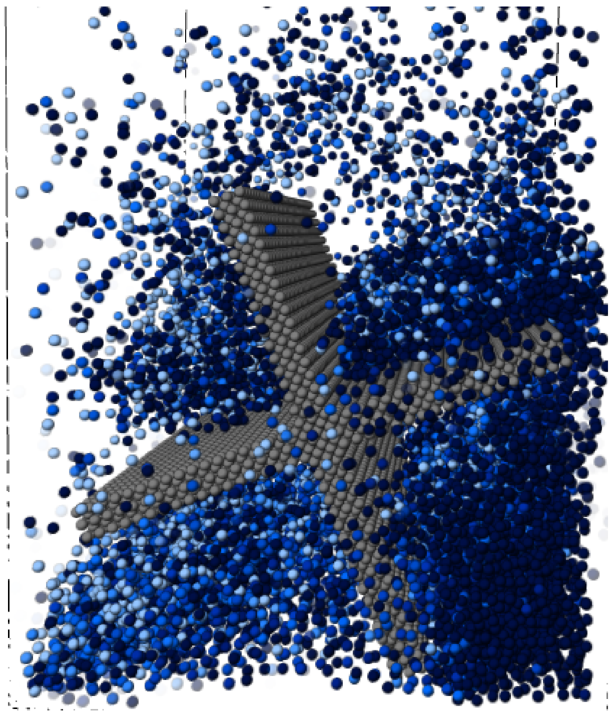
*Figure 2. Object-Fluid Interaction*

### 4.5 User Interface

The simulation of a SPH fluid requires the fine-tuning of many constants. Instead of providing a graphical user interface, the simulation is controlled by a Lua script file. This allows the user to change constants and control simulation settings and rendering options while the program is running.

## 5 Results

To gauge the speed of the simulation, a dam break scenario was created and tested on three datasets of 48958, 98958, and 210504 particles respectively. The dam break consists of three object panels holding back a column of fluid. The middle panel is then lifted to release the fluid.

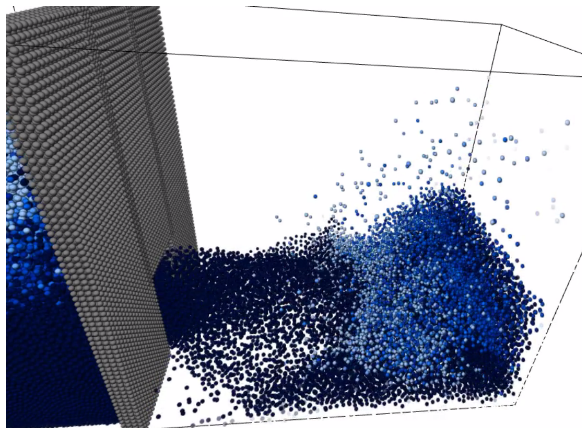
*Figure 3. Dam Break Scenario*

During testing, five timing metrics were logged for each frame of simulation: total time, nearest neighbour search time, simulation update time, graphics update time, and graphics draw time.

The following table displays the total time results for each test in average minutes per simulation seconds (how long it takes to simulate one second). Each test was ran for 1000 frames at 120 frames per second.

| **Number of Particles** | **minutes/simulation second** |
|---|---|
| **48958** | 6.4 |
| **98958** | 17.8 |
| **210504** | 60.0 |

*Table 1. Simulation Times*

The following table displays the proportion of time spent running the major processes of the simulation.

| **Dataset** | **Neighbour Search** | **Simulation** | **Graphics update** | **draw** |
|---|---|---|---|---|
| **48958** | 86.5% | 9.2% | 1.3% | 3% |
| **98958** | 80.5% | 15.3% | 1.2% | 3% |
| **210504** | 77% | 20% | 1% | 2% |

*Table 2. Proportion of Simulation Times*

We can see that the major bottleneck in this simulation is the nearest neighbour search.

## 6 Conclusions

Overall, SPH fluid simulation is a simple to implement method that gives graphically believable results for a compressible fluid. Simulation constants are difficult to tweak in order to achieve an incompressible fluid such as water. To model an incompressible fluid, a grid based Navier-Stokes solver would give for realistic results.

The major bottleneck in simulating a SPH fluid is the nearest neighbour search. To speed up simulation times, effort should be focused on optimizing the spatial grid.

## 7 References

Roy, T. M. 1995. *Physically-Based Fluid Modeling using Smoothed Particle Hydrodynamics*

Cline, D., Cardon, D., Egbert, P. K. 2013. *Fluid Flow for the Rest of Us*

Braune, L., Lewiner, T. *An initiation to SPH*

Wang, L. "*Fluid with free surfaces by Smoothed Particle Hydrodynamics*" Lei Wang http://web.cse.ohio-state.edu/~wangle/courses/788au11/788au11.html

Jakobsen, T. *2003. SPH Survival Kit*