Fluidsimulation in Computergraphics:
Weakly Compressible Smoothed Particle Hydrodynamics
and Position Based Fluids

Abstract

1   Introduction

> Reading maketh a full man; conference a
> ready man; and writing an exact man.
>
> *Sir Francis Bacon*

2   Methods

In the following section we will give an overview
of our implementation of a fluidsolver based on
the SPH and PBF frameworks. We will take a
look at the general structure of the implementa-
tion and discuss specific points of interest that
were done differently than in the given assign-
ment. For a thorough discussion of the SPH and
PBF framework we refer the reader to the re-
view of SPH fluids by Ihmsen et al. [1] and to
the original paper about position based fluids by
Macklin et al. [2].

2.1   Implementation

The implementation uses a C-style imperative
approach with the exception of a few object
oriented paradigms. Data encapsulation in the
form of C++ classes is applied to ensure data
consistency during computation. Basic inheri-
tance is used to be able to reuse code for different
parts of the program. This style of programming
allows us to be flexible enough to incrementially
implement the practical assignment tasks while
sacrificing little to no performance due to design
overload.

The implementation is divided into two broad
categories, systems and solvers. What we define
as a 'system', includes fluid- and boundaryparti-
cles, their generalization as particlesystems and
auxilary constructs concerning particles. These
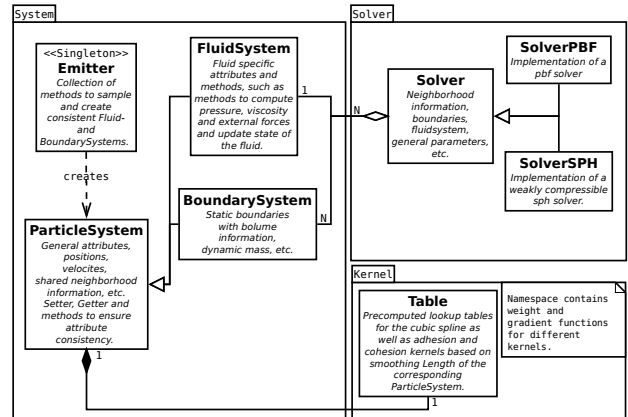represent the creation, state and behaviour of

Figure 1: Schematic overview of the implementation.

fluid- and boundary particles. On the other
hand, we have two seperate specific solvers for
WCSPH fluids and PBF. They aggregate fluid-
and particlesystems to model the progression of
the given system. Figure 1 gives a schematic
UML diagram of the relation between the differ-
ent parts of the implementation.

A particlesystem is a collection of particles
which are represented by basic attributes, such
as mass, density, position, velocity and the accu-
mulated external forces on the particles. Since
it is vital to keep all attributes of a particlesys-
tem consistent during computation a particlesys-
tem object is only allowed to be created by the
emitter object. The emitter object is a single-
ton, which means that there is always one and
only one single instance of the object during the
runtime of the program. During runtime a parti-
clesystem is present in two forms, either as fluid
particles or as boundary particles. These sub-
types posess attributes and methods responsible
for the behaviour of fluid or boundary particles
in a system. For example, boundary particles
possess a seperate variable volume attribute and
a method to compute the volume for each parti-
cle depending on the density, e.g. the number of

boundary particles in close neighbourhood.

Boundary particles are used to realise the interaction between the fluid and rigid boundaries. As boundaries in a scene are usually represented by triangular mesh models, we create a boundary particlesystem by sampling each triangle of a given input mesh. To this end, it is reasonable to dissolve a higher resolution mesh into a low resolution mesh which still contains the shape information and use this low resoltion mesh for the simulation and keep the original high resolution mesh for rendering the final simulation. This will reduce the overall number of triangles and therefore the number of sampled particles.

Figure 2: High resoltuion mesh for rendering (left) and low resolution mesh for simulation (right). The mesh on the left maintains the qualitative shape of the mesh.

To sample the particles on a triangle surface we perform a variation of the Pineda algorithm [3] in the local coordinate system of the triangle surface. Given a triangle specified by the three vertex points $a, b, c \in \mathbb{R}^3$, we construct the orthonormal base vectors $u, v, n \in \mathbb{R}^3$ defining the triangle surface. The basis transform that will yield the local triangle coordiantes $p \in \mathbb{R}^{\not\models}$ for a point in global space $x \in \mathbb{R}^3$ is given in equation 1.

$$p = \begin{pmatrix} u_x & u_v & u_z & u^T a \\ v_x & v_v & v_z & v^T a \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot x \qquad (1)$$

Note that the projected point is in homogeneous two dimensional space, as we are only interested in sampling the surface of the triangle. Applying the pineda algorithm is now straight forward. We first enclose the triangle

by a bounding box by determining the minimal and maximal coordinates in the projected space. We then go along the bounding box and check for each sample point if its inside the triangle by checking against its edges, as described by Pineda [3]. We use a hexagonal sampling pattern to ensure maximum coverage of the surface. Figure 3 demonstrates the hexagonal sampling in the bounds of the rectangle enclosing the triangle in its local coordinate system. Sample points inside the triangle are shaded oqaue blue and sample points outside are shaded transparently. Note that sample points that slightly touch the border are also taken to be in the surface of the triangle. This is achieved by allowing an offset of the particle radius and is used to ensure that the triangle mesh of the boundary has no holes where the edges of two triangles meet. The oversampling is later compensated by recomputing the volume of each sample depending on the number of samples in its close neighbourhood.
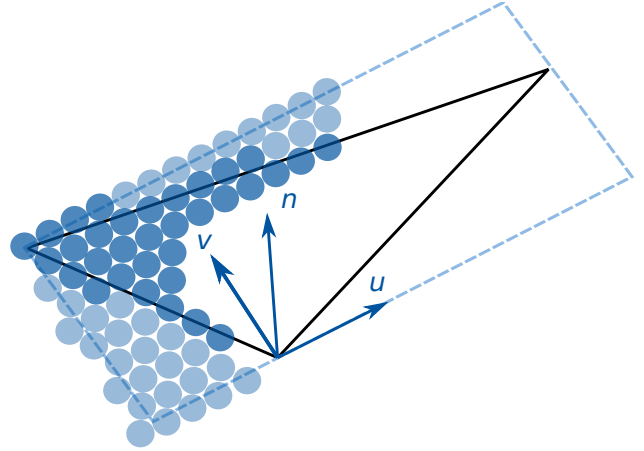

Figure 3: Hexagonal sampling using pineda algorithm in local triangle space.

Given the sample points computed in the two dimensional surface plane, we transform the points back into three dimensional global space by multiplying them by the inverse of the matrix given in equation 1. Since the matrix is orthonormal its inverse is given by simply transposing the matrix.

The boundary particle radius should optimally be the same as the fluid particle radius. This is due to the timestep being determined

by the Courant-Friedrich-Levy (CFL) condition,
e.g. $\Delta t \leq \lambda \frac{h}{v^{max}}$ which dictates, that a parti-
cle moves a maximum distance of its own radius
h per time-stamp Δt. If the boundary particle
radius is much smaller. The fluid particles may
tunnel through the boundary if the timestep is
too high. If the boundaryparticle radius is too
high, the boundary shape information is lost and
the interaction therefore incorrect. The constant
factor λ in the CFL is set around 0.4 in all fur-
ther simulations.

Results

Conclusion

Outlook

The WCSPH and PBF implementation as well
as the surface reconstruction offers a high level
of concurency in the form of data parallelism.
This means, that there are many parts in the
programm where sequentially processed data can
be split into independant groups which can be
processed in parallel.

References

[1] M. Ihmsen, J. Orthmann, B. Solenthaler, A. Kolb,
and M. Teschner, "SPH Fluids in Computer Graph-
ics," in *Eurographics 2014 - State of the Art Reports*
(S. Lefebvre and M. Spagnuolo, eds.), The Eurograph-
ics Association, 2014.

[2] M. Macklin and M. Müller, "Position based fluids,"
ACM Transactions on Graphics (TOG), vol. 32, no. 4,
pp. 1–12, 2013.

[3] J. Pineda, "A parallel algorithm for polygon rasteriza-
tion," in *Proceedings of the 15th annual conference on
Computer graphics and interactive techniques*, pp. 17–
20, 1988.