|  |  |
| --- | --- |
| A picture of a winding road and trees  simulated worlds  Soft-Body Water Simulation | Abstract  [Draw your reader in with an engaging abstract. It is typically a short summary of the document. When you’re ready to add your content, just click here and start typing.]  Edward McDowell  UFCFG5-30-2 Simulated Worlds |

Table of Contents

[**Introduction** 1](#_Toc416858817)

[**Research** 2](#_Toc416858818)

[Spring Models 2](#_Toc416858819)

[Smoothed Particle Hydrodynamics 2](#_Toc416858820)

[Verlet Integration 3](#_Toc416858821)

[**Implementation** 3](#_Toc416858822)

[**Evaluation** 3](#_Toc416858823)

[**References** 3](#_Toc416858824)

## **Introduction**

Water is everywhere. It is the very foundation of life on this planet; without it life as we know it would not exist. Given its importance to our planet and its abundance, it is understandable that we would want to create simulations for numerous reasons. Yet so many things need to be considered when modeling fluids, and this changes drastically depending on what effect you wish to achieve.

Some simulations may want to simply mimic reality as closely as possible to help further out understanding of fluid mechanics, or to model real life situations to map out ocean currents.

Figure 1: Cities Skylines dynamic water flow [5]

Figure 2: Water effects in Half Life 2

In the context of games it can add an element to a world that makes it feel more real. In the recent title “Cities: Skylines”, a water simulation is used to great effect. The simulation models flow of water down rivers, carrying pollution and overflowing banks. The player can build a dam across rivers to provide power for the city, but if the water flow is too fast it can back up and overflow the banks of the river. This reinforces the game’s theme of city planning as bad planning can have serious consequences and undo your work. This effect suits Cities: Skylines, but other games may want need something different.

In the title “Half Life 2”, a level exist where the player uses a hovercraft to move around over water and land. The water in this situation does not need to flow, but it reflects the world around, refracts objects underneath the surface and a small splash/ripple occurs where any object interacts with it. This simulation is very different but fits well in the situation.

## **Research**

There are various different approaches for actually coding a simulation, and almost all have their own individual advantages and disadvantages depending on what may be required e.g. fast real-time simulation vs. accurate but with a slow calculation time, or only a surface vs. a fully-fledged fluid which can flow.

### Spring Models

Spring models rely on creating a set of point masses to form the shape of an object. These are then connected using various springs obeying a form of Hooke’s Law. Hooke’s Law states that:

F= k X\,

where  F is the magnitude of the force pulling on the free end of the spring,  X is the displacement of the spring from its resting position and k is the spring constant, a positive real number associated with that spring.

Usually it is written as:

F= -k X\,

to give the restoring force of spring(the force the spring is exerting to restore itself to the resting position).

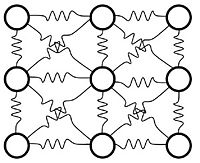


Figure 3: A set of nodes connected by springs

(Image taken from source [1])

In practice the spring model creates something akin to Figure 2. The nodes may be the vertices on the surface of a shape or `virtual` points forming the internal structure of an object. By tweaking the parameters of the springs, a shape can be allowed to compress and bounce back to its original shape, or with a low spring constant it could simple crumple up.

While the spring model is a good starting point for soft body objects with a particular shape, in terms of a water simulation it is rather poor at imitating flow. However the concept of springs is one to take note of.

### Smoothed Particle Hydrodynamics

This is a computational method for simulating fluid flow. It was not initially designed for fluids, but rather for simulating theoretical astrophysics (stellar collisions and the formation of galaxies and stars being a few examples) [2]. The model works by using a large set of elements (referred to as particles) to model the fluid which are able to freely flow around each other. Each of these particles can be given a variety of different properties. The basic formula gives quantity A at any point \mathbf{r} : 
A(\mathbf{r}) = \sum_j m_j \frac{A_j}{\rho_j} W(| \mathbf{r}-\mathbf{r}_{j} |,h),


where   m_j  is the mass of the particle   j ,  A_j  is the value of the quantity  A  for particle   j ,   \rho_j  is the density associated with particle  j , \mathbf{r} denotes position and  W  is the kernel function [4] which determines the range of particles to take into account in the formula (various different kernel formulas exist). An example of this kind of simulation can be seen here [3].

One advantages of using this method are that different fluids may interact as the particles can be given different densities, which will let them disperse within each other and still be able to separate out and be distinct. It can also be used for creating a full-fledged fluid with flow. However there are some drawbacks to this method; to create a simulation that resembles fluid some extra rendering is needed to create a surface geometry over all the particles. It also tends to require a huge number of particles to cover the same size area as other approaches which would be less intensive. In the wider context of most games this is a significant drawback, and the extra fidelity of the simulation is not worth the processing cost.

### Verlet Integration

This approach is best applied to simple surfaces/planes, which is ideal in the consideration of a game engine. The

## **Implementation**

## **Evaluation**

## **References**

[1] Wiki.blender.org, (2015). Doc:2.6/Manual/Physics/Soft Body/Interior Forces - BlenderWiki. [online] Available at: http://wiki.blender.org/index.php/Doc:2.6/Manual/Physics/Soft\_Body/Interior\_Forces [Accessed 14 Feb. 2015].

[2] Owen, J., Villumsen, J., Shapiro, P. and Martel, H. (1998). Adaptive Smoothed Particle Hydrodynamics: Methodology. II. ASTROPHYS J SUPPL S, 116(2), pp.155-209.

[3]4GamerMovie, (2014). *Smoothed Particle Hydrodynamicsベースの波表現*. [video] Available at: https://www.youtube.com/watch?v=3Y\_QTY4C08I [Accessed 13 Mar. 2015].

[4] Wikipedia, (2015). Smoothed-particle hydrodynamics. [online] Available at: http://en.wikipedia.org/wiki/Smoothed-particle\_hydrodynamics#Method [Accessed 14 Apr. 2015].

[5] Cities Skylines. (2015). Paradox Interactive & Colossal Order Ltd.