

University of New South Wales
School of Computer Science and Engineering

Virtual Sandbox

Jacqueline Hawkins
jhawkins@cse.unsw.edu.au

Supervisor: Tim Lambert
lambert@cse.unsw.edu.au

2010

Table of Contents

1 Abstract	4
2 Introduction	5
3 Background	6
3.1 Physical Simulations.....	6
3.1.1 Introduction.....	6
3.1.2 Granular Materials.....	7
Granular Material Properties.....	7
Granular Simulation Techniques.....	7
3.1.3 Fluids.....	8
Fluid Properties.....	8
Fluid Simulation Techniques.....	8
Lagrangian view (particles).....	8
Eulerian view (grid cells).....	8
Combined.....	9
3.2 Previous work.....	9
3.2.1 Real-Time Simulation and Rendering of 3D fluids (Crane et al. 2007).....	9
How it Works.....	9
Algorithm for updating the fluid.....	9
3.2.2 Animating Sand as a Fluid (Zhu 2005; Zhu & Bridson 2005).....	10
Algorithm.....	11
3.2.3 Real-time Animation of Sand-Water Interaction (Rungjiratananon et al. 2008).....	11
How It Works.....	11
3.3 Maths.....	12
3.3.1 Symbols.....	12
3.3.2 Equations.....	12
Gradient.....	12
Divergence.....	12
Laplacian.....	12
Laplace's Equation.....	12
Poisson Equation	13
Material Derivative.....	13
Navier Stokes Equations.....	13
Incompressibility Condition.....	13
Boundary conditions.....	13
Solving for velocity.....	13
Semi-Langarian Advection.....	13
MacCormak Advection.....	13
Sand Frictional Forces.....	14
Frictional Stress.....	14
Yield Condition.....	14
3.4 Conclusion.....	14
4 Proposal/Plan	15
4.1 Overview.....	15
4.2 Choose a Fluid Simulation.....	15
Progress.....	15
4.3 Get Fluid Simulator Working.....	15
Progress.....	15

4.4 Modifying for Sand.....	15
4.5 Wetting Sand.....	15
4.6 Sand Interacting with Fluid.....	16
4.7 Sand Interacting with Solids (Extra).....	16
4.8 Evaluating Progress.....	16
4.9 Schedule.....	17
5 Conclusion.....	17
6 References.....	18

1 Abstract

People enjoy physics simulations, it is fun to set them up and watch them go. By the end of this thesis, it should be possible to have developed a virtual sandbox which can be appreciated in the same way as any other physics simulation. Due to recent developments with real time fluid simulators becoming available and the similar properties of sand to a fluid when it is flowing, it should be possible to modify an existing fluid simulator to do sand as well.

2 Introduction

This thesis is about trying to achieve an interactive sand simulation by modifying an existing fluid simulator. It should be efficient enough to create quantities of sand, yet have real enough properties to play with in real time. There is no software available to do this at the moment in 3D, however there are 2D sand games available on the internet at (thisissand.com 2010) and (<http://fallingsandgame.com/sand/sand1.html> 2010) which provide a similar interactive experience.

Real time fluid simulators are being added to current game physics engines, for example PhysX and SPE. It has also been shown that granular materials have similar properties to a fluid when they are flowing and it is possible to modify a fluid simulator to animate sand.

Fluid simulations are well established in computer graphics and a review of papers reveals that granular simulations are not so much. It is possible to download an open source fluid simulator to modify, for this thesis I will be using the one by (Crane et al. 2007).

Then this simulator needs to be modified to perform a sand simulation, and we can follow the paper by (Zhu 2005) on “Animating Sand as a Fluid” to do this.

Sand can be sculpted because the grains stick together when they are moist. To be able to create sand castles in the virtual sandbox should then have to be able to add water to the sand. Conveniently, we already have a fluid simulator working from the one modified to simulate sand. By following the details in the paper by (Rungjiratananon et al. 2008) it should be possible to implement this feature.

Finally if there is any time left, it would be the ultimate goal to be able to interact with the sand using solid objects as well. Sadly this is probably too much work to be within the scope of this thesis.

3 Background

3.1 Physical Simulations

3.1.1 Introduction

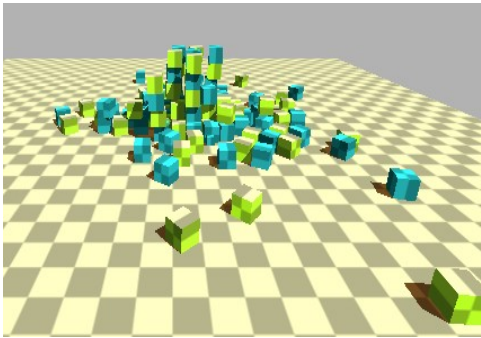
As outlined by (Gourlay 2010, p.1), the physical phenomena simulated in modern real-time rendering and physics engines include:



Particles

Screenshot of a particle simulation, in the Ogre 1.7.0 rendering engine.

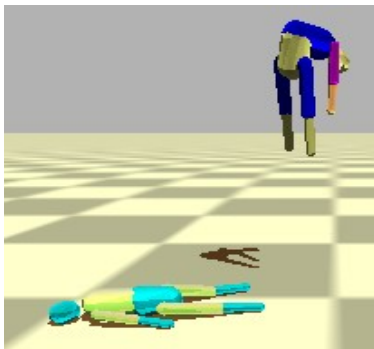
Particles are simulated as points and have no shape or volume.



Rigid Bodies

Screenshot of a stack of rigid bodies, simulated in the Bullet 2.76 physics engine.

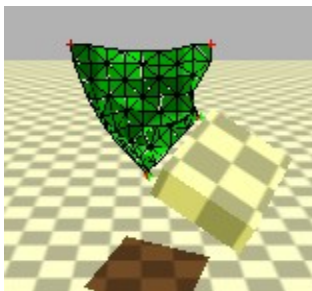
Have shape and volume, the volume never changes.



Articulated Bodies

Screenshot of two rag dolls, simulated in the Bullet 2.76 physics engine.

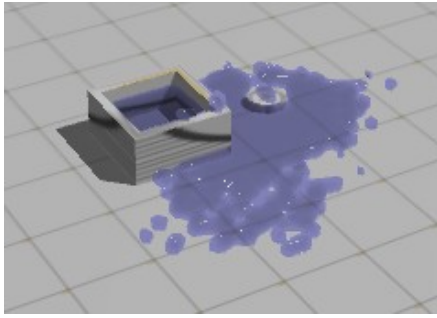
Rigid bodies that are held together with constraints. There are different types of constraints, for example points where the two bodies must touch, but can rotate in any direction or hinges where the bodies' rotation is limited to around one axis.



Soft Bodies

Screenshot of a cloth, which has a block hanging from it, simulated in the Bullet 2.76 physics engine.

Instead of being rigid, their shape is deformable, however the points along the body must retain their connections to each other.



Fluids

Screenshot of an SPE (Langarian) fluid simulation, simulated in the SPE 3.3 physics engine.

Fluid simulations can include water, smoke and fire. Fluids can change completely their shape to fill a container.

All of the simulations pictured are interactive and can be downloaded and run on a home computer. It is the aim of this thesis to create a similar interactive simulation for sand, but first an understanding of the properties of sand is required.

3.1.2 Granular Materials

Granular Material Properties

Granular materials are substances that consist of small solid particles grouped together (Wikipedia 2010). They exhibit behaviours similar to solids, liquids and gases under different situations (Jaeger et al. 1996). They behave like solids when the friction forces between their grains are stronger than any applied forces, and begin to flow like liquids when they receive stress over a certain threshold, powder can also be agitated to form a gas (Zhu 2005; Jaeger et al. 1996). These materials also have many other unique properties which separate them from fluids, but for the purposes of animation under most conditions, they go unnoticed. For example, the grains can exhibit pattern forming behaviours, on which there is an entire paper by (Aranson & Tsimring 2006). Also, as detailed by (Zhu 2005, p.15 (22 in pdf)), the pressure in sand eventually reaches a maximum, due to friction between the grains and the sides of the container. Which is why sand in an hourglass always falls at a constant rate.

Granular Simulation Techniques

Granular materials such as sand and powders, have previously been modelled as particle systems, height-fields, collections of rigid bodies and as fluids (Zhu 2005, p.4 (11 in pdf)).

These models can be separated into two categories, ones which attempt to simulate individual grains or ones that neglect the grains and treat sand as a continuum or “granular flow” (Zhu 2005, p.2 (9 in pdf)).

Particle models which simulate each grain individually have the advantage that you can see individual grains, for example a powder coating a surface. The disadvantage is that only small quantities can be simulated because of the quantity of particles. (Zhu 2005, p.2 (9 in pdf)) An example of this technique is the Discrete Element Method (DEM) algorithm used by (Rungjiratananon et al. 2008) to simulate a pile of coarse sand interacting with a fluid.

Continuum models are similar to fluid simulation, with the advantage that a larger quantity can be simulated and the disadvantage is that the individual particles are not visible. Fluid simulations have the same problem with splashing (O'Brien & Hodgins 2005). Models include using height-fields (Li & Moshell 1993; Sumner et al. 1999; Onoue & Nishita 2003; Zhu 2005, p.4 (9 in pdf)) or using fluid simulation algorithms (Zhu 2005; Carlson et al. 2002).

To simulate as many grains as possible in a real-time, interactive fashion we will be using a continuum model in this thesis. Because fluid simulators are being added to new physics engines, it should be a feasible to acquire one and then modify it to model the properties of sand.



Image: Left to right: grains, pile and lots of (real) sand. The continuum model used in this thesis will not be able to simulate the grains that get stuck to surfaces as shown on the left.

3.1.3 Fluids

Fluid Properties

As described previously, fluids are substances that flow to take the shape of their container. A fluid is not only in collision with surroundings, but also with the particles within itself, so the whole body of a fluid has to respond to a collision (Gourlay 2010, p.1). The motion of a fluids is governed by the Navier-Stokes equations and are usually modelled as being incompressible because the amount they do compress is negligible and it makes solving the equations more efficient (Zhu 2005; Gourlay 2010; Crane et al. 2007; Harris 2004; Stam 1999; Bridson & Muller-Fischer 2007). A description of these equations can be found in the maths chapter.

Fluid Simulation Techniques

There is a lot of research on fluid simulations, however, much of it is for non-real-time applications and for this thesis we are more interested in interactive simulations, so we will just consider a simple model as used by real time simulators.

Fluids are continuous, but need to be discretised into small pieces to calculate in a simulation, they can be split up into particles (Lagrangian) or grids (Eulerian) discretisation (Gourlay 2010; Zhu 2005; Bridson & Muller-Fischer 2007; Crane et al. 2007).

Lagrangian view (particles)

Each particle represents a small amount of fluid with a position and a velocity (Gourlay 2010; Bridson & Muller-Fischer 2007). This works well on CPU, but is more difficult to implement on GPU, however this has been done by (Harada et al. 2007). It also makes it harder to find adjacent particles, compared to a grid where you can just look up the coordinates. The surface can also appear blobby, see the image from the SPH engine in the introduction to this chapter.

Eulerian view (grid cells)

Space is split into a grid (which may or may not be uniformly spaced) (Gourlay 2010; Bridson &

Muller-Fischer 2007). Using a uniform grid works well on the GPU because grid can be represented by the voxels in a 3D texture (Crane et al. 2007). It is also easy to propagate values between grid cells, because they do not move relative to each other you just look up the coordinates.

Combined

There also exist combined particle and grid methods, which combine the advantages of both methods when solving equations (Gourlay 2010; Zhu 2005), however they are beyond the scope of this paper.

3.2 Previous work

3.2.1 Real-Time Simulation and Rendering of 3D fluids (Crane et al. 2007)



Image: gargoyle and smoke (Crane et al. 2007)

The paper describes the fluid simulator comes as part of NVIDIA's DirectX SDK and is free to use. It performs the fluid calculations on the GPU using fragment shaders. Because it runs on the GPU is more efficient to use an Eulerian (grid-based) simulation with the values for the fluid properties stored in 3D textures. It also does volume rendering of smoke and collisions with an animated model in real time.

How it Works

Grid cells are mapped to voxels in a 3D texture, with different textures for each of the properties of the fluid, i.e. velocity and pressure etc. At each time step pixel shaders go through the cells in the grid and update the simulation. The 3D grid is sliced into 2D slices, each slice is rendered as a quad which is rasterised and sent to the shader

Volume rendering of smoke uses a ray-marching pixel shader.

For collisions of the smoke with gargoyle it calculates the fluid-solid boundary conditions by voxelising the solid object into an “inside-outside” texture and a velocity texture and then does calculations.

Algorithm for updating the fluid

- For each time step
 - Advection (semi-Lagrangian or MacCormak)
 - Advection of Density or Level set (shader)
 - Advection of Velocity (shader)
 - Diffusion

- Diffusion of Density (only for smoke) (shader)
- Diffusion of Velocity field (shader)
- Vorticity confinement (shader)
- Add External Forces (shader)
- Add new matter (shader)
- Pressure projection
 - Compute Velocity Divergence (shader)
 - Compute Pressure (shader)
 - Project Velocity (shader)
- End

3.2.2 Animating Sand as a Fluid (Zhu 2005; Zhu & Bridson 2005)

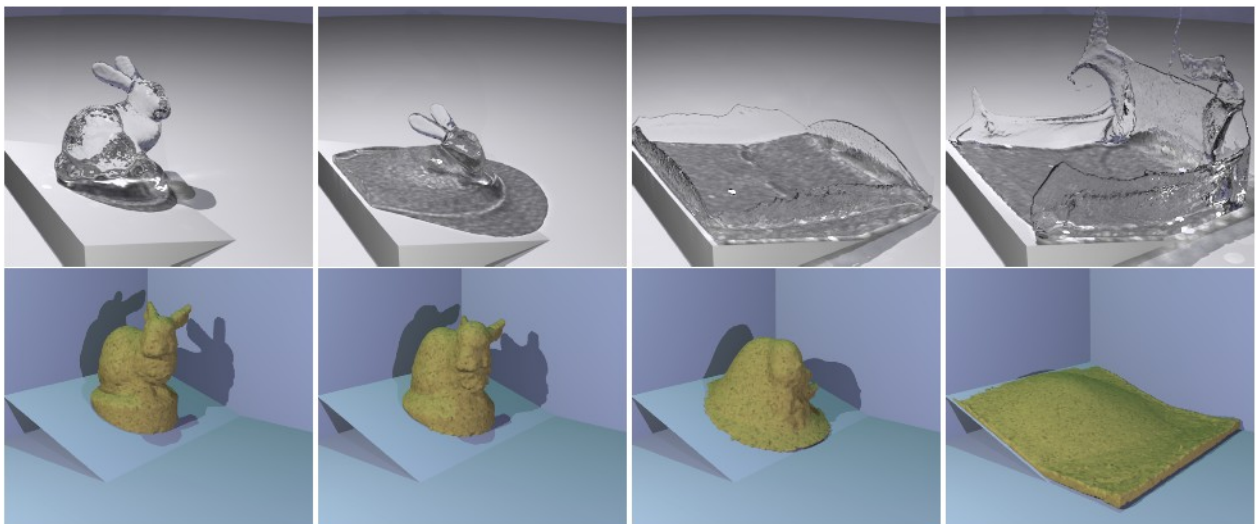


Image: water bunny and sand bunny (Zhu 2005)

In this paper a modified a fluid simulator is used to simulate sand in non-real-time.

It assumes that sand is a continuum rather than as individual particles and also that the grains are of the same size. Zhu only considers granular flow when it is in its “frictional regime”, which is when the grains are moving slowly and friction forces between the grains are dominant. This is opposed to a flow in a “rapid or collisional regime”, where there grains are moving quickly and the frictional forces are small compared to the collisional forces between the grains. It also ignores sand dilation, which is where the grains move apart a little when they start to flow (and the volume changes), because this effect is too small to see in animation. The pressure within the sand is also assumed to behave like that of water, even though it does not, due to friction between the grains and the sides of the container.

In this model the grains only begin to flow after a minimum force is applied, this is called the “yield condition”. When the forces on the sand in a grid cell are lower than the frictional forces between

the sand grains then the cell is “rigidified” and the sand does not flow.

Algorithm

Quoted directly from (Zhu 2005, p.16 (24 in pdf))

“

- For every time step
 - Apply advection
 - Apply gravity and boundary conditions, solve for pressure to enforce incompressibility, and subtracting $\Delta t/\rho \nabla p$ from the intermediate velocity field to make it incompressible.
 - Evaluate the strain rate tensor D_{strain} in each grid cell with standard central differences, and decide yielding condition for each grid cell:
 - If the resulting σ_{rigid} from equation 2.5 satisfies the yield condition with the pressure we computed for incompressibility, we mark the grid cell as rigid and store σ_{rigid} at that cell.
 - Otherwise, we store the sliding frictional stress σ_f from equation 2.4.
 - Treat the two regions differently:
 - Find all connected groups of rigid cells, and project the velocity field in each separate group to the space of rigid body motions.
 - For all the yielding cells, we update with the frictional stress, using standard central differences for $u+ = \Delta t/\rho \nabla \cdot \sigma_f$
- EndFor

“

3.2.3 Real-time Animation of Sand-Water Interaction (Rungjiratananon et al. 2008)

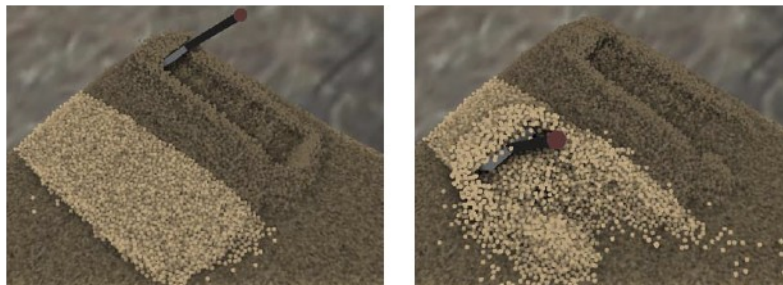


Image: wet and dry sand with a shovel (Rungjiratananon et al. 2008)

Simulates the properties of wet and dry sand and water propagating through sand along with changes in sand stickiness and colour in real time.

How It Works

Used a particle-based granular simulation for sand (DEM) and a particle-based fluid simulation for water (SPH). Water particles absorbed by grain particles and it adds a “stickiness” value to each grain.

Sand can be:

- dry and able to absorb more fluid
- wet and sticky, but still able to absorb more
- over-wet and runny, sand is saturated

3.3 Maths

Description of the equations used in this paper.

3.3.1 Symbols

\vec{u}	velocity (m/s^2)
ρ	density (kg/m^3)
p	pressure (N/m^2)
\vec{g}	gravity $= (0, -9.8, 0)(m/s^2)$
μ, ν	kinematic viscosity (m^2/s)
σ	stress tensor (N/m^2)
f_{bd}	forces body density (N/m^3)
D_{strain}	rate of strain (1/s)
Δx^2	cross sectional area (of a grid cell) m^2

3.3.2 Equations

To perform a fluid simulation involves solving the Navier Stokes equations, which requires vector calculus. A brief outline is given below, for a review read (Bridson & Muller-Fischer 2007) or (Wikipedia 2010).

Gradient

$$\nabla f(x, y, z) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right)$$

Divergence

$$\nabla \cdot \vec{f} = \nabla \cdot (u, v, w) = \left(\frac{\partial u}{\partial x}, \frac{\partial v}{\partial y}, \frac{\partial w}{\partial z} \right)$$

Laplacian

$$\Delta f = \nabla \cdot \nabla f = \nabla^2 f = \left(\frac{\partial^2 f}{\partial x^2}, \frac{\partial^2 f}{\partial y^2}, \frac{\partial^2 f}{\partial z^2} \right)$$

Laplace's Equation

A laplacian where the RHS must be 0.

$$\nabla \cdot \nabla f = 0$$

Poisson Equation

A laplacian where the RHS is non-zero.

$$\nabla \cdot \nabla f = q$$

Material Derivative

$$\begin{aligned} \frac{Dq}{Dt} &= \frac{\partial q}{\partial t} + \vec{u} \cdot \nabla q \\ &= \frac{\partial q}{\partial t} + u \frac{\partial q}{\partial x} + v \frac{\partial q}{\partial y} + w \frac{\partial q}{\partial z} \end{aligned}$$

Navier Stokes Equations

For incompressible flow, because fluids do not compress by much and it makes the equations more efficient to solve (Zhu 2005; Gourlay 2010; Bridson & Muller-Fischer 2007).

$$\begin{cases} \frac{\partial \vec{u}}{\partial t} = \mu \nabla \cdot \nabla \vec{u} - \frac{\nabla p}{\rho} + \vec{f}_{external} - \vec{u} \cdot \nabla u \\ \quad = \mu \nabla \cdot \nabla \vec{u} - \frac{\nabla p}{\rho} + \vec{g} - \vec{u} \cdot \nabla u \\ \nabla \cdot \vec{u} = 0 \end{cases}$$

Incompressibility Condition

Change in density = 0, so the divergence of velocity = 0

$$\begin{aligned} \frac{D\rho}{Dt} &= 0 \\ \nabla \cdot \vec{u} &= 0 \end{aligned}$$

Boundary conditions

Occur at the edges of the fluid, where it interacts with other objects and must be dealt with in a simulation. Since it is unknown if there will be time to simulate the collisions between the sand and solid objects in this thesis they are ignored.

Solving for velocity

Semi-Langarian Advection

The simple advection scheme used by (Crane et al. 2007), is based on the “stable fluids” method by (Stam 1999). It is “unconditionally stable” which means that the simulation does not get errors and “blow up”.

MacCormak Advection

A more detailed advection scheme is also used by (Crane et al. 2007) which is not stable, but provides improved detail and looks less blobby and is better than using a larger grid resolution. It uses 2 semi-Langarian steps and a limiter is applied to keep it stable.

Sand Frictional Forces

These are the forces used in the paper by (Zhu 2005) to model the friction forces between the sand grains.

Frictional Stress

The frictional stress is the point that resists sliding the most. It represents the resistance of movement between grid cells.

$$\sigma_f = -\sin\left(\phi p \frac{D_{strain}}{\sqrt{\frac{1}{3}|D_{strain}|_F}}\right)$$

where

$$D_{stress} = \frac{\nabla u + \nabla u^T}{2}$$

Yield Condition

The minimum stress at which to rigidify sand in a cell.

$$\sigma_{rigid} = -\frac{\rho D_{strain} \Delta x^v}{\Delta t}$$

3.4 Conclusion

After searching, there aren't any 3D physics simulators currently and openly available that have sand to play with. It is also apparent that simply using a particle simulator will not work when simulating larger amounts of sand than a single pile. However, the latest real time physics engines are starting to add fluid simulations and it is also possible to modify a fluid simulator to simulate the properties of sand.

4 Proposal/Plan

4.1 Overview

For this thesis I propose to combine the techniques from these papers to make an interactive sandbox. By the end of the project the user should at least be able to drop sand into a box, watch it and be satisfied it behaves like sand. Apart from plain sand properties to be simulated include colour, stickiness, wetting from a fluid and, if time permits, interactions with solids.

4.2 Choose a Fluid Simulation

This requires choosing a fluid simulator based on how easily it can be modified to do sand. Selection is based on real time performance vs. quality of simulation, documentation, ease of use and if it's cross platform.

Progress

Already chosen NVIDIA DirectX SDK Smoke example which is also documented by (Crane et al. 2007). It was chosen because it's real-time, works well and has excellent documentation. It runs on the GPU which is suited to an Eulerian simulation which also suits propagating water between the cells. The only disadvantage is that it is written with DirectX. Most of the other open source simulators that were sampled were also only for Windows/DirectX, and were poorly documented or just failed to compile and run easily.

4.3 Get Fluid Simulator Working

For this I will modify NVIDIA's sample code to use OpenGL instead of DirectX, since I'm learning OpenGL and the goal is to be able to run it on my laptop. This also requires studying the code in detail, which I will have to do to be able to modify it for sand anyway.

Goal: have the fluid simulation in a box, the box being bounded by the size of the 3D texture and the user can use the mouse to add more fluid.

Progress

Currently stage in project, it doesn't really do much until all of the shaders are converted and working.

4.4 Modifying for Sand

At this stage it is time to add the sand calculations from paper by (Zhu 2005). And research a different/faster rendering technique to what is used by the fluid simulator, because sand is opaque (compared to smoke, water and fire) it should be possible to render faster. This will be done by investigating a few different ones and choose one based on performance and how good it looks.

Goal: is to have the sand simulation in the same box, where the mouse drops more sand.

4.5 Wetting Sand

Adding sand wetness and stickiness calculations from the paper by (Rungjiratananon et al. 2008). This will also involve a change of colour for wet or dry sand so I might as well implement a choice of other colours of sand to drop. Dropping sand in coloured layers should be similar to the 2D

(thisissand.com 2010) toy.

Goal: to be able to choose to drop different coloured sand, or wet or dry sand from mouse.

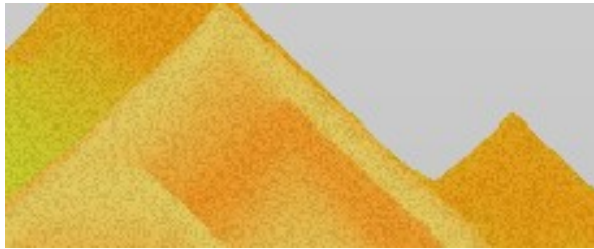


Image: Screenshot of piles of 2D sand from (thisissand.com 2010)

4.6 Sand Interacting with Fluid

Now that the fluid and sand simulators are working they can be combined, so that the fluid from the fluid simulation is absorbed by the sand, using calculations from the paper by (Rungjiratananon et al. 2008).

Goal: the user should be able to drop sand or water into the box and watch the water wet the sand.

4.7 Sand Interacting with Solids (Extra)

This stage is only if time permits. I can start with NVIDIA's collision system for smoke, and then research more about sand-solid interaction forces so as to modify it to do sand. I would also research about integrating the project into the Bullet physics engine, choosing Bullet because:

- it is open source
- has soft body simulations, which is the closest to fluid simulation
- popular and well documented

Goal: would be to have something more like the “hell of sand” game (<http://fallingsandgame.com/sand/sand1.html> 2010).

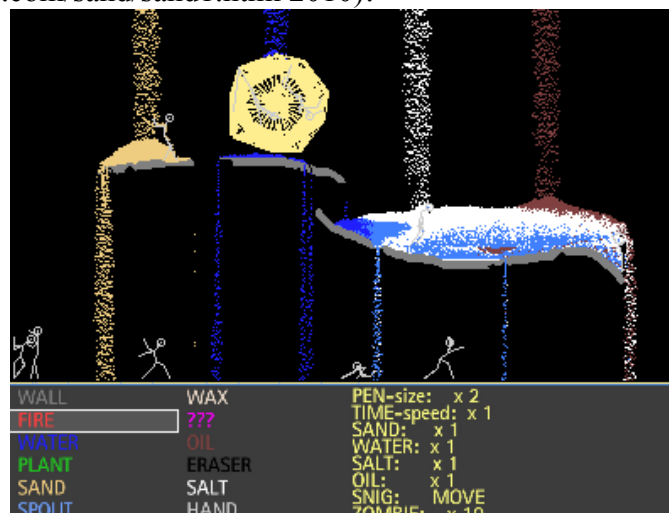


Image: Screenshot of the “Hell of sand” game (<http://fallingsandgame.com/sand/sand1.html> 2010)

4.8 Evaluating Progress

Evaluation is mainly based on the subjective experience of sand and how well the goals detailed above have been achieved. Efficiency of the simulation will mainly be due to the initial choice of

simulation engine. Ideally there should be a decent enough quantity of sand to play with and it should run on my laptop.

4.9 Schedule

Below is a rough schedule for completing the development of the simulator. It is difficult to estimate how long each of these steps will take because it also involves prototyping and researching to get a result that works well and provides a good experience.

Week	Tasks
Recess	Get fluid simulator working - calculation shaders (advection etc), UI (add fluid with mouse), using a debugging rendering algorithm
Recess	Get fluid simulator working - calculation shaders (advection etc), UI (add fluid with mouse), using a debugging rendering algorithm
Recess	Get fluid simulator working - rendering algorithms for smoke and water
1	Modify for sand - calculation shaders (friction & rigidifying cells)
2	Modify for sand - calculation shaders (friction & rigidifying cells)
3	Modify for sand - rendering algorithm for sand surface and sand texture
4	Wetting sand - colour
5	Wetting sand - stickiness
6	Sand interacting with fluid - colliding water with sand
7	Sand interacting with fluid - propagation of water through sand
Break	Misc time - fixing problems or sand interacting with solids
8	Misc time - fixing problems or sand interacting with solids
9	Misc time - fixing problems or sand interacting with solids
10	Demo week (Mon 27 Sept - Fri 1 Oct)
11	Report writing - results etc
12	Report writing - results etc
13	Report due (Tues, 19 Oct)

5 Conclusion

I propose to create a sand simulation based on an existing fluid simulation, using the methods outlined in previous work it should be possible. The methods are summarised above, and do not cover everything in too much detail, in particular the maths, and the rendering algorithm. This is because these are implementation details, which will be covered more in the Thesis B report, because I will be prototyping the simulator and the details might change anyway.

I've planned out a set of goals for how to complete the project and what it should be expected to look like after, because the main evaluation of the simulator will be on its animation of sand.

The projects is currently at the “get the fluid simulator working” stage and should be completed before the start of session 2.

6 References

- Aranson, I. & Tsimring, L., 2006. Patterns and collective behaviour in granular media: Theoretical concepts. *Reviews of modern physics*, 78(2), 641–692.
- Bridson, R. & Muller-Fischer, M., 2007. Fluid Simulation. In SIGGRAPH. Available at: <http://people.cs.ubc.ca/~rbridson/fluidsimulation/>.
- Carlson, M. et al., 2002. Melting and Flowing. *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*.
- Crane, K., Llamas, I. & Tariq, S., 2007. GPU Gems 3 - Chapter 30. Real-Time Simulation and Rendering of 3D Fluids. In *GPU Gems 3*. GPU Gems. Addison-Wesley Professional. Available at: http://http.developer.nvidia.com/GPUGems3/gpugems3_ch30.html [Accessed March 16, 2010].
- Gourlay, M., 2010. Fluid Simulation for Video Games. *Intel® Software Network*. Available at: <http://software.intel.com/en-us/articles/fluid-simulation-for-video-games-part-1/> [Accessed March 17, 2010].
- Harada, T., Koshizuka, S. & Kawaguchi, Y., 2007. Smoothed Particle Hydrodynamics on GPUs. *Computer Graphics International*.
- Harris, M., 2004. GPU Gems - Chapter 38. Fast Fluid Dynamics Simulation on the GPU. In *GPU Gems*. Available at: http://http.developer.nvidia.com/GPUGems/gpugems_ch38.html [Accessed March 30, 2010].
- <http://fallingsandgame.com/sand/sand1.html>, 2010. Hell of Sand Game. Available at: <http://fallingsandgame.com/sand/sand1.html>.
- Jaeger, H., Nagel, S. & Behringer, R., 1996. Granular solids, liquids, and gases. *Reviews of Modern Physics*, 68(4), 1259-1273.
- Li, X. & Moshell, J.M., 1993. Modeling Soil: Realtime Dynamic Models for Soil Slippage and Manipulation. *International Conference on Computer Graphics and Interactive Techniques*.
- O'Brien, J. & Hodgins, J., 2005. Dynamic Simulation of Splashing Fluids. *Proceedings of the Computer Animation*.
- Onoue, K. & Nishita, T., 2003. Virtual Sandbox. *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*.
- Rungjiratananon, W. et al., 2008. Real-time Animation of Sand-Water Interaction. *Pacific Graphics*, 27(7).
- Stam, J., 1999. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. p. 128.
- Sumner, R., O'Brien, J. & Hodgins, J., 1999. Animating Sand, Mud, and Snow. *Computer Graphics Forum*.

thisissand.com, 2010. thisissand.com. Available at: <http://thisissand.com/> [Accessed May 25, 2010].

Wikipedia, 2010. Granular material - Wikipedia, the free encyclopedia. *Wikipedia*. Available at: http://en.wikipedia.org/wiki/Granular_material [Accessed May 25, 2010].

Zhu, Y., 2005. *Animating Sand as a Fluid*. Masters. The University of British Columbia.

Zhu, Y. & Bridson, R., 2005. Animating Sand as a Fluid. *ACM SIGGRAPH 2005*.