All the Simulations

Taking as inspiration one of four suggested simulations, you shall produce a tech demo style implementation of that simulation suitable for inclusion into a wider game engine. This should be implemented in DirectX possibly using the supplied code base. Whilst the simulation might not take place in 3D, any visualization of it should do. You will also be required to produce a development report which outlines your research into developing this implementation and how you went about producing it.

In all cases:

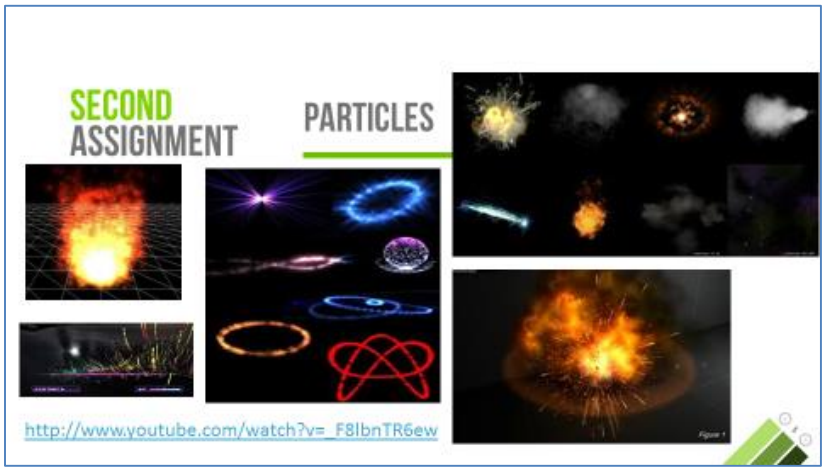
- Think about how the implementation would sit within a wider game engine AND asset pipeline.
- How would a designer be able to customise this to produce a particular effect in a game?
- Simulation should be customisable to produce multiple variants of the underlying simulation.

System 1: Particles

You retweeted @mikeBithell

Mike Bithell
@mikeBithell

The next generation seems to so far be predominantly about particle effects and detailed embroidery on characters' backs. I am ok with this.



Particles are commonly a simulation of a large numbers of non-interacting point particles. Particles are used for many different effects. A particle system thus needs to be easily extendable and easily customisable to produce multiple effects. Remember that the components that build up a particle effect are more “atomic” than fire, cloud, fog etc.

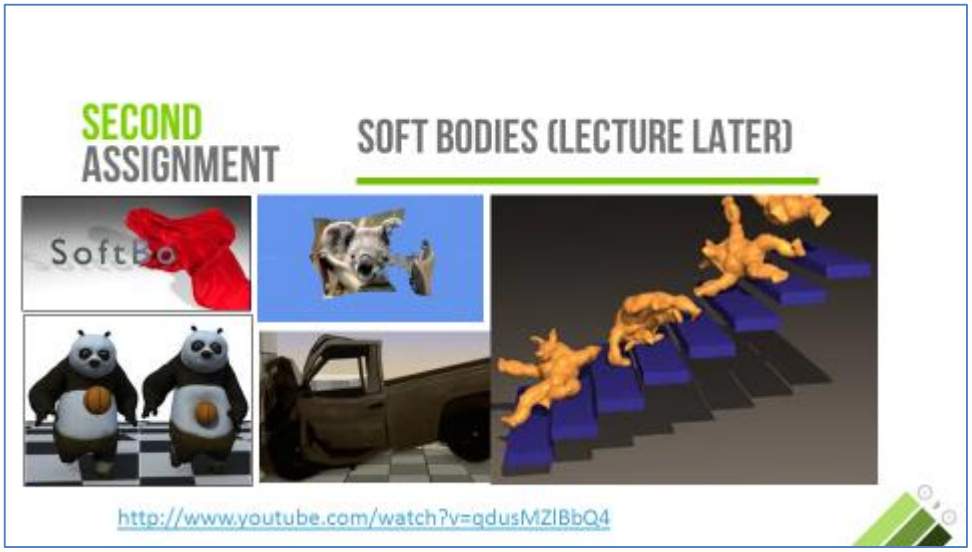
Potential Parameters/Features:

- Various force fields
- emitter types
    - (Omni-directional, directional, volume emitters of various shapes, ring emitter),
- colour/alpha variation over life-time of the particle
- rotation
- spawn rate of particles
- initial velocities of particles over a random distribution.

Stretch Goals

- Particles specific render effects
    - E.g. Soft particles
- Collisions
- GPU particles

System 2: Soft Body-Water Surface



A soft body systems is usually simulated using point masses located at the vertices of a mesh. These are then linked together with hook springs. In some game engines all one has to do to make an object into the cloth is to set a flag in its "GameObject" and then the triangulated model otherwise used to render the object becomes the spring system. Although note that to properly simulate a soft body would usually require a different organisation of vertices than rendering it.

You should produce a customisable water-surface simulation for which the parameters of the simulation can be altered. This should at least be the "height diffusion" model, but additional marks for attempting a more involved model.

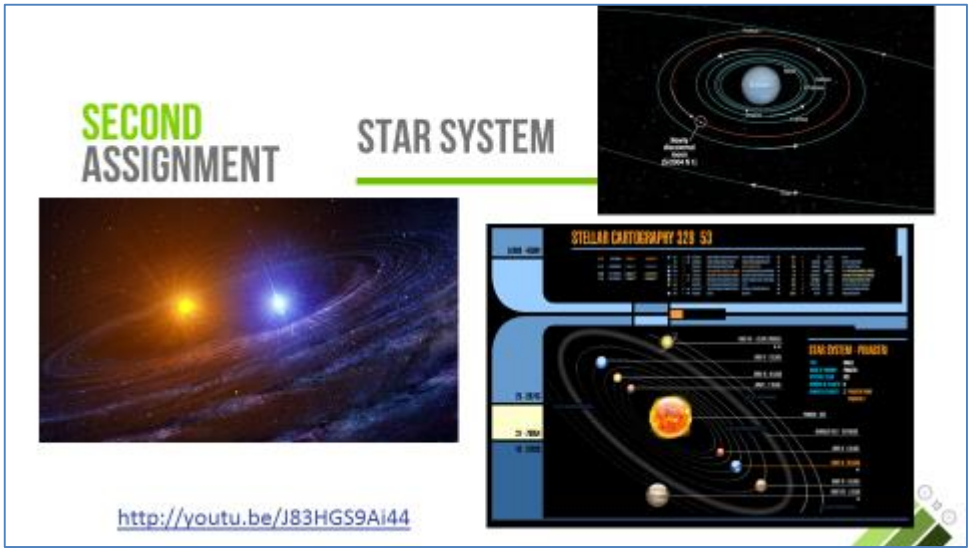
Potential Parameters/Features:

- "Poke" water surface to create ripples / waves
- Drive a boat about the surface
- Interface with a terrain object
- Transparent surface / depth fog

Stretch Goals

- Reflections
- GPU based simulation

System 3: Solar System



Whilst technical almost all aspects of the motion of our entire star system can be modelled with a single equation: Newton's law of Gravitation. Perhaps alarmingly this hardly ever produces a stable system. Remembering that we are really only producing our simulations to be visual interesting not scientifically accurate, we can implement either limitations on the forces in a system or additional "hidden" forces to achieve an interesting but stable system.

Potential Parameters/Features:

- Newton's law of Gravitation
- Moons, asteroids, comets
- Jupiter-like mini-star systems
- Faking / forcing stability
- Procedural generation of a star system

Stretch Goals

- Stability
- Procedural generation of planet textures.

System 4: Lion & Wildebeests

A simulation very similar to the Boids model can carried out a simulation of predators chasing after prey. This should include a range of obstacles for the creatures to avoid, spawn points where they enter the environment and destinations for them to leave. All of these could potentially be placed by a level designer.

Potential Parameters/Features:

- Different prey types.
- Multiple predators.

Stretch Goals

- Hunting FSM.
- No fixed parameters.