Realtime Particle System Simulation and Rendering in Embedded Systems

Linköpings University
Information Coding Group
Jens Ogniewski
Motivation

- Smartphones / tablets (so called embedded systems) will replace PCs in consumer households
  - Already happening
  - PCs in the future will only be used for work, and probably hardcore gaming
Motivation

- Smartphones / tablets (so called embedded systems) will replace PCs in consumer households
  - Already happening
  - PCs in the future will only be used for work, and probably hardcore gaming

- Rapid growing market for games on embedded systems
  - Considered to be one of (if not: the) most important market
  - Mainly small, casual games, but even many with advanced graphics
Motivation

- Embedded systems: performance constrained
  - Efficiency optimized, not performance optimized like PCs
  - Also: very high resolution screens, no upscale
Motivation

- **Embedded systems: performance constrained**
  - Efficiency optimized, not performance optimized like PCs
  - Also: very high resolution screens, no upscale

- **Different architecture as PCs / gaming consoles**
  - Shared memory, shared bus between ALL the components
  - In comparison to PC only limited distributed memory (e.g. caches, in case of the GPU in the Nexus 10: <=256 kbytes)
  - => need to make the most of it
Motivation

- Ice Storm: cross platform benchmark
Motivation

- Ice Storm: cross platform benchmark
- Nexus 10 as used during this project: 8006
Motivation

- Ice Storm: cross platform benchmark
- Nexus 10 as used during this project: 8006
- NVIDIA GeForce GTX 660: 137246
  
  => more than 17 times faster!
Background

- Particle effects in computer graphics
  - Water, smoke, fire etc.
  - Navier-Stokes based solutions, e.g. “Simple and Fast Fluids”

- Current games for smartphones / tablets
  - Particle systems
  - Basically animated billboards moving in predetermined or pseudorandom way
Our contribution

- First work on effects based on simulated particle movement on embedded systems
- Based on a novel, forced-based motion model
  - No need for additional, space-consuming pressure field
  - Simulation completely done in 2D
Particle Fields
Particle Fields
Particle Fields

4 2 3 4 4 2 3 3
Motion Model
Motion Model
Motion Model
Motion Model
Force based motion

- 4 different forces
  - Diffusion
  - External forces (e.g. gravity)
  - Inertia
  - Random
Force based motion

- 4 different forces
  - Diffusion
  - External forces (e.g. gravity)
  - Inertia
  - Random

- Combined using different weights for each
Force based motion
Force based motion
## Results

<table>
<thead>
<tr>
<th></th>
<th>Size of particlefield</th>
<th>Reference (time/ms)</th>
<th>Simulation (time/ms)</th>
<th>Rendering (time/ms)</th>
<th>Est. FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nexus 10, resolution: 2560x1600</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fire</td>
<td>32x32</td>
<td>4.89</td>
<td>3.17</td>
<td>26.7</td>
<td>36.3</td>
</tr>
<tr>
<td>Water</td>
<td>64x64</td>
<td>3.59</td>
<td></td>
<td>43.2</td>
<td>22.3</td>
</tr>
<tr>
<td>Smoke</td>
<td>64x64</td>
<td>3.93</td>
<td></td>
<td>11.4</td>
<td>84.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iPhone 5, resolution: 1136x640 (preliminary)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fire</td>
<td>32x32</td>
<td>3.4</td>
<td></td>
<td>34.2</td>
<td>28.2</td>
</tr>
<tr>
<td>Water</td>
<td>64x64</td>
<td>6.3</td>
<td></td>
<td>47.6</td>
<td>19.7</td>
</tr>
<tr>
<td>Smoke</td>
<td>64x64</td>
<td>7.6</td>
<td></td>
<td>7.2</td>
<td>138</td>
</tr>
</tbody>
</table>
Conclusion

- Effects based on simulated particle movement for games in embedded systems

- Based on novel, force-based motion model
  - Faster than fastest Navier-Stokes (by 35%)
  - Much less data (up to 80% less)
  - Easy to configure for the designer
  - Allows fast particle spreading
Conclusion

- Future work
  - Optimize the code
  - More unified approach
  - Improve visual quality