FPGA Based Particle Engine for Textile 3D Surface Modeling and Simulation

Dorian Gorgan, Stefan Mathe
Computer Science Department
Technical University of Cluj-Napoca
dorian.gorgan@cs.utcluj.ro, mstefan@cs.toronto.edu
http://users.utcluj.ro/~gorgan
Fashion design

- Objectives
  - Development and experimentation of a virtual 3D studio
  - Research in fashion textile design
  - Allows a designer to achieve by a natural manner the modelling and the simulation of the textile materials and objects
Modelling and simulation of 3D textile surfaces

- Objectives
  - Simulation of the physical characteristics of the textile materials (elasticity, rigidity, transparency) but also their association with the material’s chromatic composition
  - Modelling and editing of the 2D garments considering the current trends in fashion
  - Design of clothing items and their presentation on models
  - Configuring the surrounding environment (colours, textures, lights and shadows) for viewing the static or dynamic models
  - Experimenting interaction techniques and the implementation of software tools for virtual 3D operations: cutting, fitting, sewing and laying out material pieces on the models.
Software Platform Architecture

Diagram:

1. Particle Graphics Engine
2. Particle Simulator
3. Textile Materials Editor
4. 2D Garments Editor
5. Mannequin Editor
6. 3D Cloth Editor
7. Dresser
8. Scene Editor
9. Studio 3D Animator

Nodes:
- Textile Materials
- 2D Garments
- 3D Cloth Objects
- Scenes
- Mannequins
- Dressed Mannequins
- Accessories
- Mannequin Editor
- 2D Garments Editor
- 3D Cloth Editor
- Scene Editor
- Studio 3D Animator
- Particle Graphics Engine
- Particle Simulator
Particle based modeling of the textile 3D surface
Garments editor
Main issues

- Expensive software computation
  - high memory data model
  - particle movement
  - force computation
  - constraint satisfaction
- Graphics presentation
- Animation
- User interaction
- Collision
Particle model execution algorithm

1. Force accumulation
2. Verlet integration
3. Global constraint satisfaction
4. Local constraint satisfaction

Considerations:
- Global constraint
  1. Sphere
  2. Box (axis aligned)
- Local constraint
  1. Rope (dist < L)
  2. Stick (dist = L)
Verlet integration

\[ x' = x + v \cdot \Delta t \]

\[ v' = v + a \cdot \Delta t, \]

Verlet integration

\[ x' = 2x - x^* + a \cdot \Delta t^2 \]

\[ x^* = x. \]

Verlet computation:

next_pos = 2 crt_pos – prev_pos + a Δt
prev_pos = crt_pos
crt_pos = next_pos
Solution

- Particle engine
- Computation through hardware engine
- FPGA (Field Programmable Gate Arrays)
- VHDL (VHSIC Hardware Description Language)
- VHSIC (Very High Speed Integrated Circuits), 1980
- Xilinx (1984)


Research:

- design alternatives and the performance of an FPGA based particle engine
Particle model parameters

- Number of particles (N)
- Current positions of the particles (cpos)
- Previous positions of the particles (prev)
- Gravitational acceleration (g)
- Time step size (deltat)
- Number of iterative relaxations (R).
- The total number of time iterations (M)
- The number of global constraints (G)
- The set of global constraints (global).
- Number of local constraints (L)
- Set of local constraints (local)
Commands to the particle engine

1. SET_GLOBAL: load the global constrains
2. SET_LOCAL: load the local constrains
3. SET_PARTICLES: load the initial positions of the particles
4. RUN_TIMESTEP: execute a simulation step
5. GET_PARTICLE: read the current positions of the particles
Implementation issues

Parallelization, but with limitations:

- Spatial limitation of the design
  - cannot instantiate as many integration and constraint modules as the number of particles.

- Limited number of simultaneous accesses (read and write) to the particle memory.
Pipeline architecture
Particle engine - pipeline architecture

Extended version of the pipeline architecture for Verlet integration and global constraints.
Local constraints

Order of computation for the local constraints: A, B, D

A pipeline of length greater than 2 would fail, despite of the number of relaxation iterations.
Local constraints

Solutions:

- Avoid the dependencies - detect the dependencies at runtime, followed by:
  - Stalling – delay the processing
  - Bypassing – pass the execution of another constraint

- Remove the dependencies before the execution (e.g. while the particles are loaded into the system):
  - Software environment - before the execution of the SET_LOCAL command.
  - Hardware system – during the execution of the SET_LOCAL command
Local constraint execution

Pipeline architecture for local constraints execution.
Chip space usage

- Free, 1098, 16%
- Verlet Integrator, 20, 0%
- Sphere Constraint, 1510, 22%
- FIFOs and other logic, 2173, 32%
- Local Constraint, 1664, 24%
- Controller, 447, 6%
Software/hardware speed comparison

![Comparison chart showing processing speed in particles/second between software and hardware implementations. The hardware implementation (VirtexE600, 50MHz) processes 5,380,836 particles/second, while the software implementation (P4, 2.4GHz) processes 334,952 particles/second.]
Thanks

Prof. Dorian Gorgan
Computer Science Department
Technical University of Cluj-Napoca
dorian.gorgan@cs.utcluj.ro
http://users.utcluj.ro/~gorgan