CSD1130 Game Implementation Techniques

Lecture 13

Outline

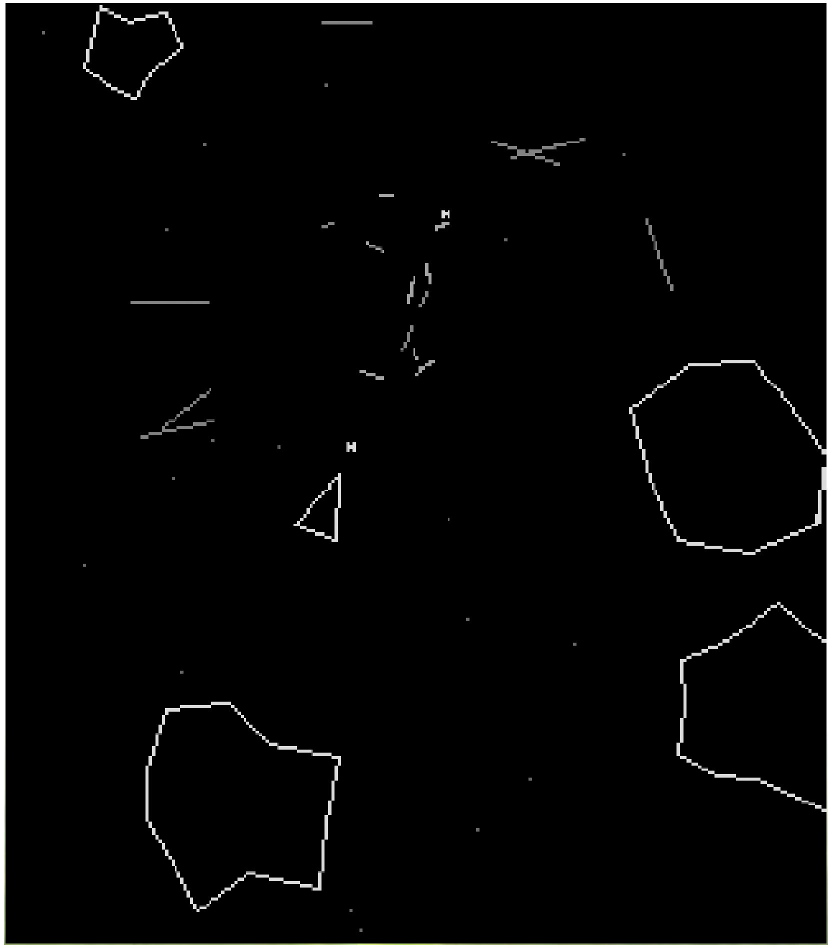
- History of Particle Systems
- What is a Particle System?
- Basic Model of Particle Systems
  - Particle Attributes
  - Particle Life Cycle
- Random Numbers

History of Particle Systems (1/3)

- Spacewar
  - 1962
  - Second video game ever
  - Uses pixel clouds as explosions (random motion)

History of Particle Systems (2/3)

- Asteroids
  - 1978
  - Uses short moving lines for explosions (physical particle simulation)



History of Particle Systems (3/3)

- Star Trek II: The Wrath of Kahn
  - 1983
  - Movie Visual FX
  - First CG paper about particle systems by William T. Reeves
  - This concept is still used today
  - Watch the trailer: http://www.youtube.com/watch?v=UJTi7KJPx_E

Outline

- History of Particle Systems
- What is a Particle System?
- Basic Model of Particle Systems
  - Particle Attributes
  - Particle Life Cycle
- Random Numbers

What is Particle System? (1/2)

"A particle system is a collection of many many minute particles that together represent a fuzzy object. Over a period of time, particles are generated into a system, move and change from within the system, and die from the system."

– Reeves Particle Systems—a Technique for Modeling a Class of Fuzzy Objects.

What is a Particle System? (2/2)

- Movement of particles is defined from forces and constraints (e.g. gravity)
- Stochastically defined attributes, and that is to use random numbers to control particle attributes such as position, color, …
- Often rendered as individual primitive geometry (e.g. point)

Uses of Particle Systems

- The use of Particle systems is a way of modeling fuzzy objects, such as:
  - Fire (explosions, ...)
  - Clouds
  - Smoke
  - Water
  - Fog
  - etc...

Particle System: Demos

- Demo 1
  - Particle Dreams by Karl Sims (1988)
- Demo 2
  - Particle System by Lutz Latta

Outline

- History of Particle Systems
- What is a Particle System?
- Basic Model of Particle Systems
  - Particle Attributes
  - Particle Life Cycle
- Random Numbers

Basic Model of Particle Systems (1/2)

- Particle Attributes
  - Position
  - Velocity (Speed and Direction)
  - Color
  - Lifetime
  - Shape
  - Size
  - Transparency

Basic Model of Particle Systems (2/2)

- Particle Life Cycle:
  - Generation
  - Dynamics
  - Extinction
  - Rendering

Particle Generation

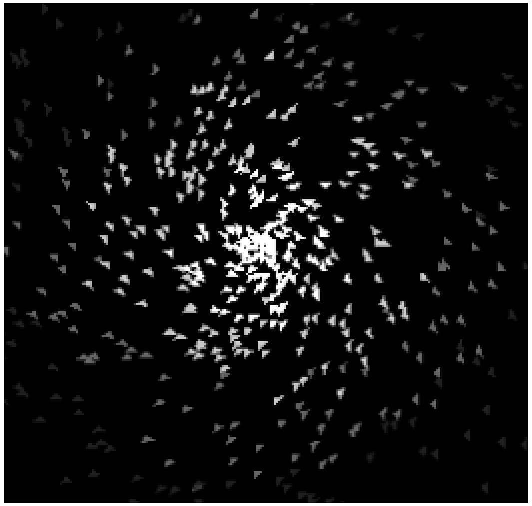
- Each of the attributes are given an initial value
- These values can be fixed or determined by a stochastic process

Particle Dynamics

- Applying forces (e.g. gravity, wind, …)
- Particle attributes can be functions of both time and other particle attributes
  - Ex:
    - Color of a particle in an explosion gets darker as it gets further from the center of the explosion

Particle Extinction

- The particle is destroyed when:
  - The lifetime reaches zero
  - The color is below a threshold (becomes invisible or fades out)
  - Running out of bounds



Particle Rendering (1/2)

- Particles can obscure other particles behind them, can be transparent, and can cast shadows on other particles.

Particle Rendering (2/2)

- Particles can act as light sources
  - Particles that map to the same pixels in a frame, the color of the pixel is the sum of the color of all the particles that map to it.

References

- William T. Reeves, "Particle Systems - A Technique for Modeling a Class of Fuzzy Objects"

Outline

- History of Particle Systems
- What is a Particle System?
- Basic Model of Particle Systems
  - Particle Attributes
  - Particle Life Cycle
- Random Numbers

Random Numbers

- In computer applications we use what is called **pseudo-random numbers**

- **Pseudo** because:
  - It's based upon specific mathematical algorithms which are repeatable and sequential or pre-calculated tables to produce sequence of numbers that appear random

Pseudo-Random Numbers Generator (1/2)

- Goal:
  - To produce a sequence of numbers in [0,1] that simulates, or imitates, the ideal properties of random numbers

Pseudo-Random Numbers Generator (2/2)

- Characteristics:
  - Fast
  - Portable to different computers
  - Have a long cycle
  - Uniform and independent

Linear Congruential Generator

- Oldest and best known PRNG
- The generator is defined as:

$$X_{n+1} = (aX_n + c) \mod m$$

$m > 0$ (the modulus)

$0 < \mathrm{a} < \mathrm{m}$ (the multiplier)

$0 < \mathrm{c} < \mathrm{m}$ (the increment)

$0 \leq X_0 < \mathrm{m}$ (the seed or start value)

Example

- LCG (a, c, m, X0)

  LCG (5, 1, 16, 1)

  Output:

  - 1,6,15,12,13,2,11,8,9,14,7,4,5,10,3,0, …

Characteristics

- Periodic
  - The period is at most m
- Deterministic
  - Next "random" number depends heavily on the previous X

Further Reading

- Numerical Recipes in C – Second Edition

Basic Particle System

- **Particle system structure:**
  - Array of particles
  - Number of particles
  - Blending factor
  - Initialization per particle
    - Color range
    - Lifetime range
    - Scale range
    - Etc..

Basic Particle System

- **Particle system structure:**
  - Particle emitter (default point emitter)
  - Enable cycles
  - Relative transformation
  - Rotation, scale and position
  - Emission rate
  - Warm up time
  - Etc..

Basic Particle System

- **Particle system initialization:**
  - When creating the particle system, you need to pass the number of particles and the texture to be used by all particles.

  - A particle system uses a mesh for all the particles.
  - This mesh is instantiated, in the graphics system, every time we render a new particle

Basic Particle System

- **Particle system initialization:**
  - The final output is a series of quads (in 2D).
  - Each quad is made of 2 triangles to represent one particle.
  - Using a “triangle list” type of rendering, you will need 6 vertices per particle.

Basic Particle System

- **Particle system initialization:**
  - Therefore, each particle system will allocate memory as follow:
    - Vertices buffer
    - Color buffer
    - Indices buffer
    - Texture coordinates buffer

Basic Particle System

- **Particle system initialization:**
  - By default, the size of each particle will be the same as of the texture size.
  - The rendering of all particles in one particle system should be done in one pass.