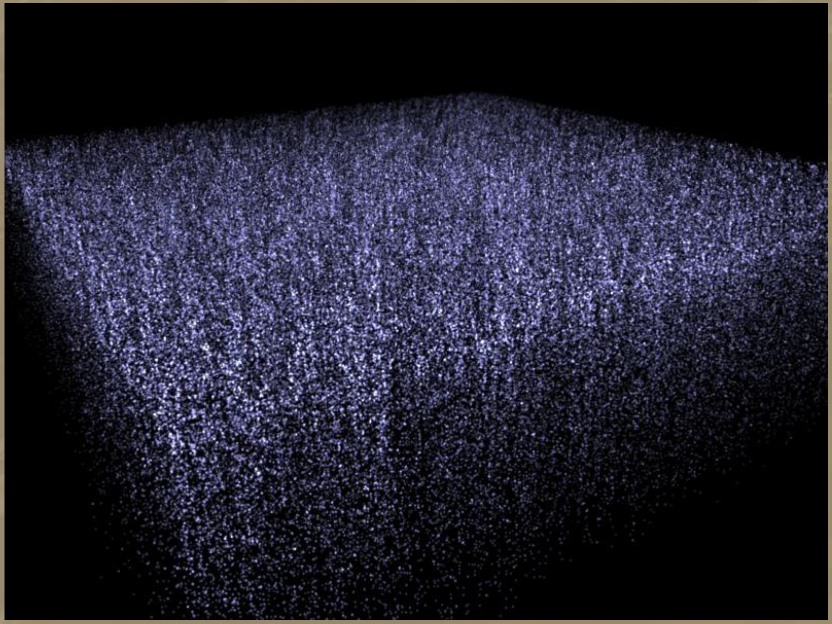
Particle Systems

CS 418 – Interactive Computer Graphics
TA: Gong Chen
Fall 2012

Particle System

- **Particle Dynamic System:**
  Simulate a massive number of interacting elements



Particle System

- **Basic Examples:**
  - **F=ma rule**
  - **Gravity force**
  - **Bounce back from floor.**
- **Particle examples:**
  - **simple points, or billboard *sprites***
    - http://www.lighthouse3d.com/opengl/billboarding/index.php
  - **You cannot use a particle system library**

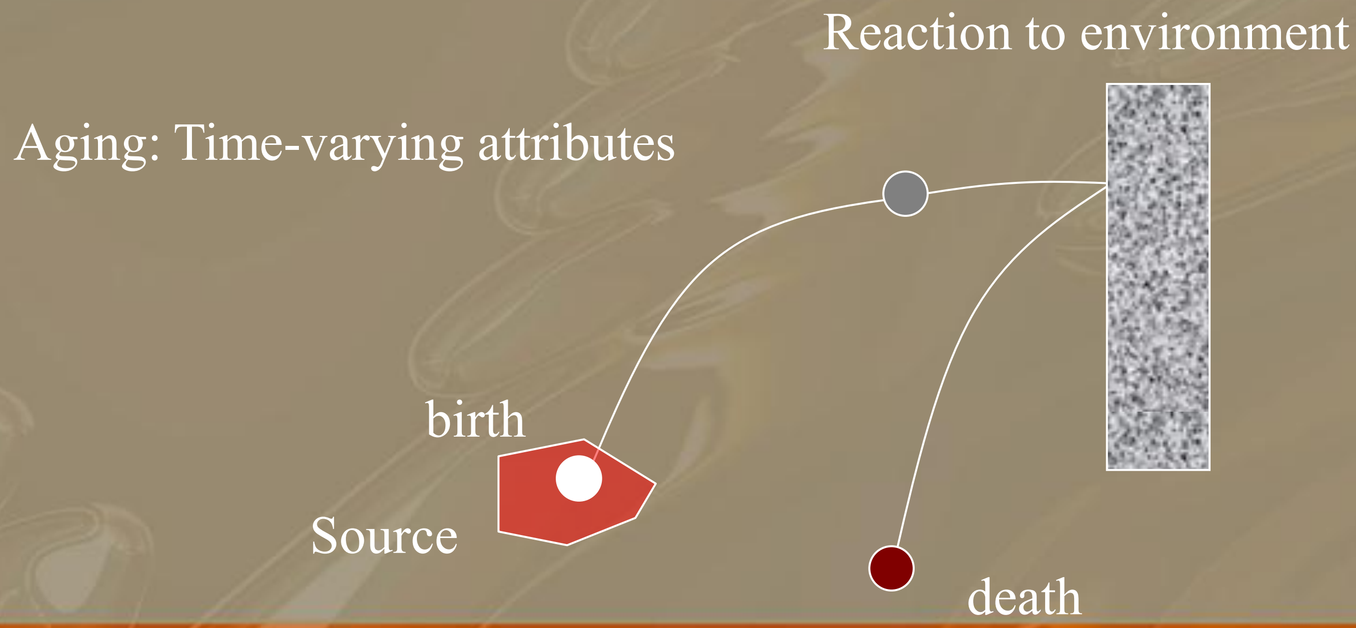
Particle Object

For each particle you need to store:

- **Mass**
- **Position**
- **Velocity**
- **Acceleration**
- **Life Span (Optional)**

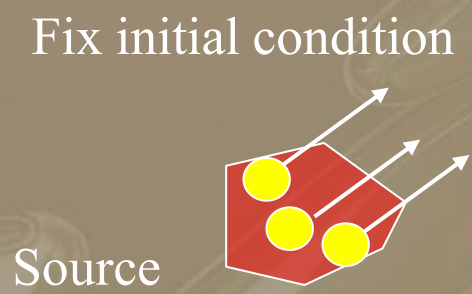
Basic Flow

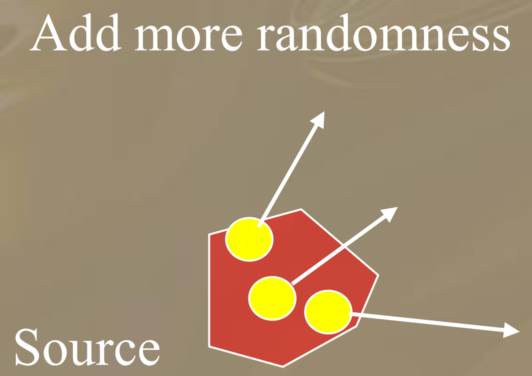
- **For each frame you should :**
  - **Create some new particles**
  - **Delete "dead" particles**
  - **Update particle "Position" based on physics**
  - **Render particles in new positions.**



Particle Generation

- Specify a source location to generate particles
- Each particle has initial position & velocity
- Add some randomness in initial condition.





Update particles

- **Given forces on this particle. How do you determine its next position ?**
- **Euler Method** $x(t_0 + h) = x_0 + h\dot{x}(t_0)$
  - Simplest to implement.
  - Not very stable, so don't jump too much at time.
  - Beware of accumulated numerical error
- **Midpoint Method**

$$y_{n+1} = y_n + hf\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}f(t_n, y_n)\right)$$

Types of Forces

- **Unary forces:**
  - Gravity
    - Make object moving down.
    - Constant acceleration on all particles.

- **N-ary forces:**
  - Spring force :
    - Add a spring to connect two particles.
    - Force depends on deviation from rest length.
    - Damping : Force that depends on Rate of change in length.

k_s : spring constant

k_d : damping factor

s : rest length

$$\mathbf{d} = \mathbf{x}_i - \mathbf{y}$$

$$\dot{\mathbf{d}} = \dot{\mathbf{x}}_i - \dot{\mathbf{y}}$$

$$\mathbf{f}_i = -\left(k_s \left(\|\mathbf{d}\| - s \right) + k_d \frac{\dot{\mathbf{d}} \cdot \mathbf{d}}{\|\mathbf{d}\|} \right) \frac{\mathbf{d}}{\|\mathbf{d}\|}$$

Update Rules

- Apply all forces on this particle (gravity, etc).
- Acc = F/m
- V = V + Acc* Δt
- P = P + V* Δt
- Life = Life - Δt

- KEEP IN MIND:

  Δt should not be too large !

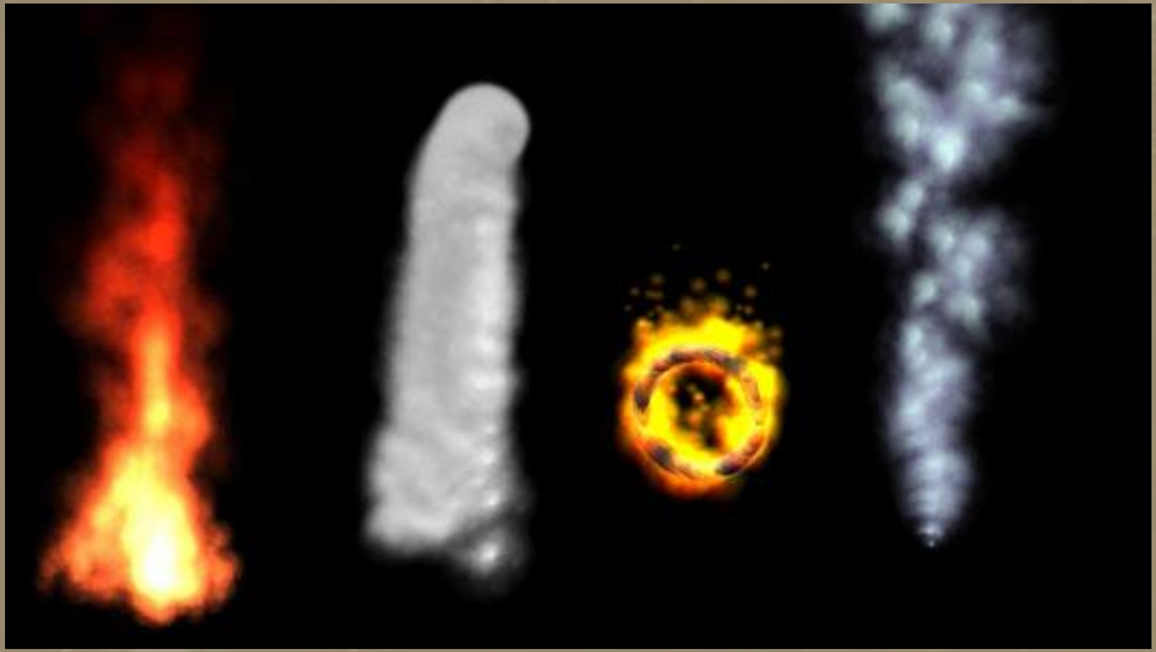
Bounce from floor

- Particle can not fall through floor.
- Detect if P.y <= floor height.

- If collide with floor
  - Bounce back (Ex: V.y = abs(V.y))
  - Add some friction ? → Reduce velocity for each bounce
  - Add some randomness in how particles bounce back.

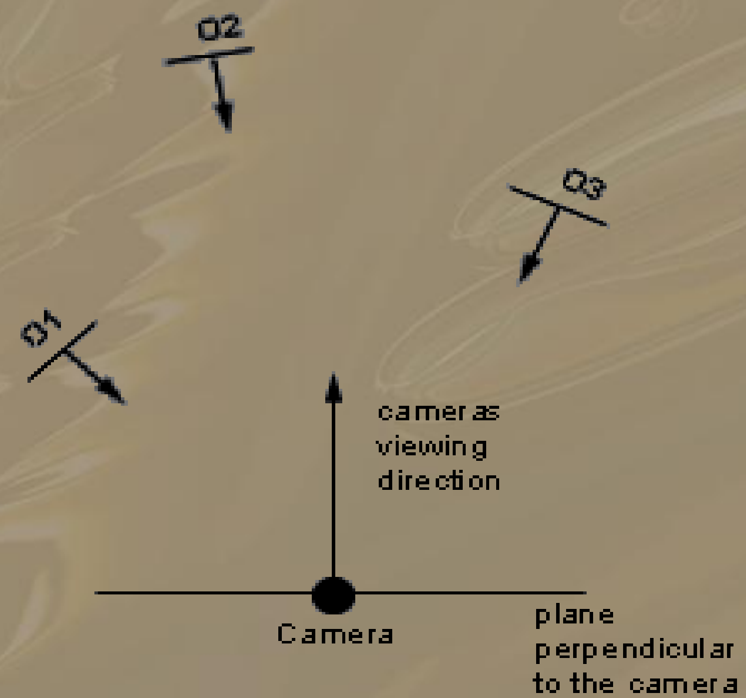
Rendering

- Simple point will do.
- Alpha blend points for better visual quality.
- Or use bilboards to enhance visual result.



Billboard Stripes

- **Use images mapped to quads, rotated to face the camera to represent particles (remember texture mapping)**



Q&A