

Particles

CS-116B: Computer Graphics Algorithms
Spring 2018

Particles

Particles are useful in simulations of:

- “smoke”
- “fire”
- “explosions”
- “water”
- “dust clouds”
- “swarm of insects”

Particles

Two essential elements of particle simulations:

- “particle model”
- “integrator”

Particle model: defines the physical traits of the particle and their behavior during a simulation.

Integrator: iteratively updates each particles' velocity and displacement over time based on external forces and collisions.

C++ particle class

```
class Particle
{
public:
    float fMass;           // Total mass
    Vector vPosition;     // Position
    Vector vVelocity;     // Velocity
    float fspeed;        // Speed (magnitude of the velocity)
    Vector Vforces;      // Total force acting on the particle
    float fRadius;      // Particle radius used for collision
detection
    Vector vGravity;     // Gravity force vector

    Particle (void);     // Constructor
    void CalcLoads (void); // Aggregates forces acting on the particle
    void UpdateBodyEuler (double dt); // Integrates one time step
    void Draw (void)     // Draws the particle
};
```

Source: *Physics for Game Developers*, p. 167

C++ constructor for initializing the particle class

```
#define _GRAVITYACCELERATION -9.8f

Particle::Particle (void)
{
    fMass = 1.0;
    vPosition.x = 0.0;
    vPosition.y = 0.0;
    vPosition.z = 0.0;
    vVelocity.x = 0.0;
    vVelocity.y = 0.0;
    vVelocity.z = 0.0;
    fspeed = 0.0;
    Vforces.x = 0.0;
    Vforces.y = 0.0;
    Vforces.z = 0.0;
    fRadius = 0.1;
    vGravity.x = 0;
    vGravity.y = fMass * _GRAVITYACCELERATION;
}
```

Source: *Physics for Game Developers*, p. 168

C++ method CalcLoads from the particle class

```
void Particle::CalcLoads (void)
{
// Reset forces:
    Vforces.x = 0.0f;
    Vforces.y = 0.0f;

// Aggregate forces:
    Vforces += vGravity;
}
```

C++ method UpdateBodyEuler from the particle class

```
void Particle::UpdateBodyEuler (double dt)
{
    Vector a;
    Vector dv;
    Vector ds;

    // Integrate equation of motion:
    a = vForces / fMass;

    dv = a * dt;
    vVelocity += dv;

    ds = vVelocity * dt;
    vPosition += ds;

    // Misc. calculations
    fSpeed = vVelocity.Magnitude();
}
```