CS425: Game Programming 1
Lecture 2: Particle System
8/28/2013

Lecturer: Jyh-Ming Lien

1 Introduction

1.1 Data Structure

State of a particle $s = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} = \begin{bmatrix} x \\ v \end{bmatrix}$, where x and position and v is the velocity.

State of a particle system Given n particles in a particle system, we can represent the system as

$$S = \begin{bmatrix} s_1 \\ s_2 \\ \dots \\ s_n \end{bmatrix}$$

1.2 Physics Engine

Physics engine is usually used in game to simulate the motion of objects, such as particles. The following simple procedule is the core of a physics engine (using a particle system as an example).

1. Get states of particles (positions and velocities)
2. Get forces F applied to each particles
3. Compute derivatives from the forces F
4. Using ODE solvers to compute the new positions and velocities
5. Set the states back to the particles

2 Forces

There are many different type of forces, such as gravity and friction. In general, when multiple forces are applied to a single particle, these forces are combined linearly (with weights) into a single force F (as the F in step 3 above). There are typically three types of forces:

1. Unary Forces:
   - Gravity
   - Viscous drag

2. N-ary Forces:
   - mass-spring system (usually used for modeling deformable objects)
   - flocking system (coherence, alignment, separation)
   - fluid dynamics (Navier-stroke equations)
3. Spatial related forces:
   - collision
   - wind
   - user interaction

2.1 Viscous drag

This is similar to sampling in spring system. This is highly recommended for stability in simulating particles.

$$F = -k_d v,$$

where k_d is a user defined constant and v is the linear velocity.

2.2 Hook's law spring

This is similar to sampling in spring system. This is highly recommended for stability in simulating particles.

$$F = -\left(k_s(|\ell| - r) + k_d \frac{v\ell}{|\ell|}\right) \frac{\ell}{|\ell|},$$

where k_s and k_d are spring and damping constants, ℓ is the vector pointing at the particle and parallel to the spring (i.e., a vector between two connected particles), r is the resting length of the spring, and, finally, v is the linear velocity of the particle .

2.3 Flocking

Flocking system is used to simulate the motion of a group of coherent entities, like a school of fishes and a flock of birds. The motion is governed by three simple local rules, whose forces are linearly combined to create the flocking force. For a given particle, its flocking force is defined as:

$$F = k_{co} F_{coherent} + k_{se} F_{separation} + k_{al} F_{alignment},$$

where k_{co}, k_{se}, k_{al} are user defined constants that can be used to influence the behavior of the flock. Each of these threes forces are defined as the following.

Coherence $F_{coherent} = x - O_{nei}$

Separation $F_{separation} = O_{nei} - x$

Alignment $F_{alignment} = V_{nei}$

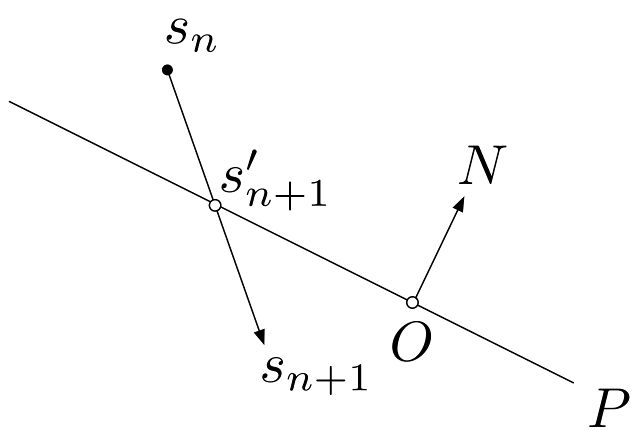
Here O_{nei} is the centroid of the positions of the neighboring particles and V_{nei} is the averaged velocity of of the neighboring particles. The neighboring particles of a given particle can be defined in many ways, such as particles within δ distance from the given particle or k closest particles.

2.4 User interaction

Allows user to exert force to particles. Idea is to attached a spring between the mouse pointer and particles.

2.5 Collision

Given that we found the collision of a particle between state s_n and s_{n+1} with a plane $P = \{N, O\}$, we can find the exact collision state s'_{n+1} where the particle is on the colliding plane P. Here, N is the normal direction of P and O is a point on P. You can find s'_{n+1} using binary search since s_n is above (inside) and s_{n+1} is below (outside) the plane (or any 2D/3D objects).



Collision response To handle collision, we decompose the velocity of the particle at s'_{n+1} into two vectors: v_N and v_T, where v_N is parallel to N and v_T is perpendicular to N (i.e. parallel to the plane). The new velocity of the particle at s'_{n+1} now becomes

$$v'_{n+1} = v_T + k_r(-v_N) ,$$

where k_r is called *restitution coefficient*, a property of the plane. We then continue to simulate the particle from s'_{n+1} with this new velocity.

Constact We say that the particle is in contact with the plane P if $v_N = \emptyset$. If an external force F is applied to push the particle into P a contact force is applied back as $-F$.

Friction Friction force is applied in the direction opposite to v_T and is proportional to the external force pushing the particle into P. Therefore, $F = -k_f(-F_{ext}N)v_T$, where k_f is friction constant and F_{ext} is an external force applied on the particle.