CENG 732
Computer Animation

Spring 2006-2007

Week 6

Rigid Body Dynamics

Controlling Groups of Objects

This week

- Rigid Body Dynamics
  - Bodies in contact
- Controlling Groups of Objects
  - Particle Systems
  - Flocking Behavior
  - Autonomous Behavior

Bodies in Contact

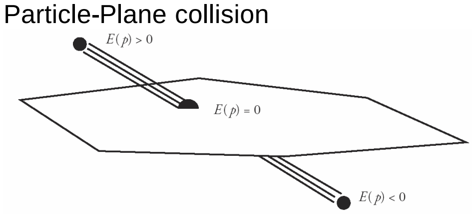
- Collision
  - Both kinematic and dynamic components
    - Kinematic: Do determine whether two objects collide or not. Only dependent on the position and orientations of the objects and how they change over time
    - Dynamic: What happens after collision, what forces are exchanged, how do they affect objects' motions

Collision handling

- Kinematic response
- Take actions after the collision occurs (the penalty method)
- Back up time to the first instant the collision occurs and determine the appropriate response

Kinematic Response

- Particle-Plane collision



$$E(p) = a \cdot x + b \cdot y + c \cdot z + d$$

Kinematic Response

- Particle's position is computed at every time step (particle is moving at a constant speed)

$$p(t_i) = p(t_{i-1}) + \partial t \cdot v_{ave}(t)$$

when $E(p(t_i)) \leq 0$ we understand that the particle has collided with the plane in the time interval t_{i-1} and t_i.

Kinematic Response

- When collision is detected, the component of the velocity vector in the normal direction is negated by subtracting it twice from the original velocity vector.
- To model the loss of energy during collision a damping factor 0<k<1 is multiplied with the normal component when it is subtracted the second time

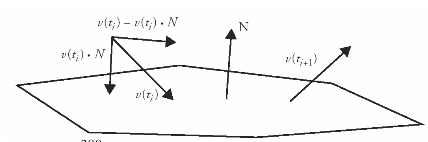
$$v(t_{i+1}) = v(t_i) - v(t_i)\cdot N - k\cdot v(t_i)\cdot N$$
$$= v(t_i) - (1+k)\cdot v(t_i)\cdot N$$

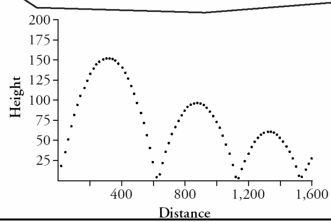
Projection Operation

- Projection of a vector **a** onto another vector **b** is given by:

$$\frac{\mathbf{a}\cdot\mathbf{b}}{|\mathbf{a}|^2}\mathbf{a}$$

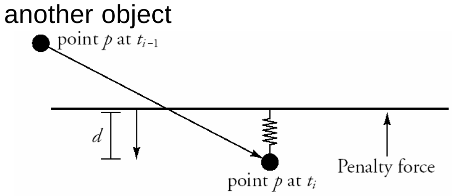
Kinematic Response





The Penalty Method

- A point is penalized for penetrating another object

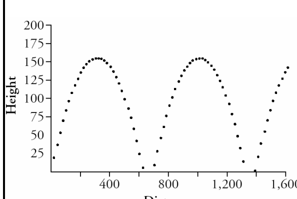


Penalty Spring

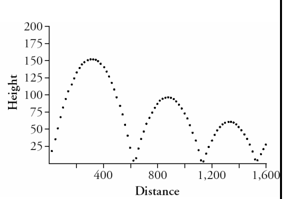
The spring produces an upward acceleration with a = - kd/m

Difficult to determine the mass and the spring constants that will produce a realistic effect

The Penalty Method



Penalty method with mass = 10 and spring constant = 250



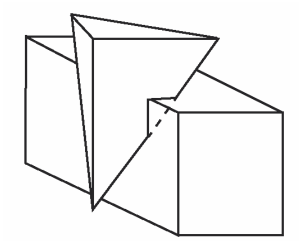
Kinematic Response

Testing Collisions Between Planar Polyhedra

- Bounding box tests
  - If they indicate a collision then more elaborate tests may be performed to make sure that the collision exists
- Convex polyhedra – point test
  - The point should be on the same side (inside) for all the polygons that make up the polyhedra
- Concave polyheadra – point test
  - Even-odd rule may be used to count the intersections of a ray emanating from the point to an arbitrary direction with the polygon faces
    - if its odd → point inside the polyhedra
    - if its even → point outside the polyhedra

Testing Collisions Between Planar Polyhedra

- Point test may not be conclusive
  - May need to test the edges as well



Response in Polyhedra Collision

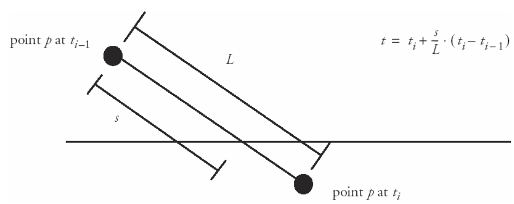
- The normal that defines the plane of intersection is used to calculate the response
  - point-face penetration
    - The normal of the face is used
  - edge-edge intersection
    - cross product of the edges is used as the normal

Backing Up Time

- Can be computationally expensive when too many collisions occur
- Time is backed up to the point of impact, an impulse force is computed, and the time is moved forward again
- The exact collision time can be found by binary search by setting L=t_{i-1} and U=t_i and searching between L and U, dividing the interval by half each time.

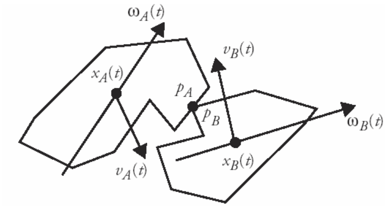
Backing Up Time

- The point of collision can also be found by assuming a linear path, constant-velocity motion. This provides an approximation.



Collision of two rigid objects

- Assume that two objects collide at points p_A and p_B.



Computing the impulse force

- The normal to the surface of contact, *n*, is determined. The relative positions of the contact points with respect to the center of masses are computed.

$$r_A = p_A - x_A(t)$$
$$r_B = p_B - x_B(t)$$

Computing the impulse force

- The relative velocity of the contact points in the direction of the normal, *n*, is computed

$$v_{rel} = (\dot{p}_A(t) - \dot{p}_B(t)) \cdot n$$

$$\dot{p}_A(t) = v_A(t) + \omega_A(t) \times r_A$$

$$\dot{p}_B(t) = v_B(t) + \omega_B(t) \times r_B$$

Computing the impulse force

$$v^+_{rel} = n \cdot (\dot{p}^+_A(t) - \dot{p}^+_B(t))$$

$$v^+_{rel} = n \cdot (v^+_A(t) + \omega^+_A(t) \times r_A - v^+_B(t) + \omega^+_B(t) \times r_B)$$

$$j = \frac{-((1+\varepsilon) \cdot v^-_{rel})}{\frac{1}{M_A} + \frac{1}{M_B} + n \cdot (I^{-1}_A(t)(r_A \times n)) \times r_A + (I^{-1}_B(t)(r_B \times n)) \times r_B}$$

Using the impulse to compute collision aftermath

$$v^+_A = v^-_A + \frac{j \cdot n}{M_A}$$

$$v^+_B = v^-_B + \frac{j \cdot n}{M_B}$$

$$\omega^+_A = \omega^-_A + I^{-1}_A(t) \cdot (r_A \times j \cdot n)$$

$$\omega^+_B = \omega^-_B + I^{-1}_B(t) \cdot (r_B \times j \cdot n)$$

Enforcing constraints

- Enforcing constraints to physically based animations can be done by introducing additional forces in the system, such as:
  - Springs
  - Internal energy constraints (such as distance of vertices from the center of mass)
- Constraints can be
  - Hard constraints
  - Soft constraints

Controlling Groups of Objects

Type of Group	Number of Elements	Incorporated Physics	Intelligence
Particles	many	much—with environment	none
Flocks	some	some—with environment and other elements	limited
Autonomous behavior	few	little	much

Controlling Groups of Objects

- Particle systems
- Flocking behavior
- Autonomous behavior

Type of Group	Number of Elements	Incorporated Physics	Intelligence
Particles	many	much—with environment	none
Flocks	some	some—with environment and other elements	limited
Autonomous behavior	few	little	much

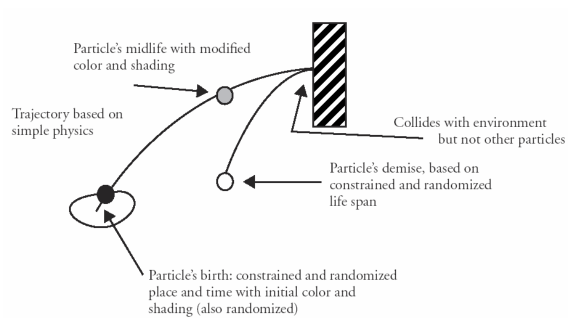
Particle Systems

- Some common assumptions due to large number of particles
  - Particles do not collide with other particles
  - Particles do not cast shadows, except in an aggregate sense
  - Particles only cast shadows on the rest of the environment, not on each other
  - Particles do not reflect light, they are each modeled as light emitting objects

A frame of a particle system

- Any new particles that are born during this frame are generated
- Each new particle is assigned attributes
- Any particles that have exceeded their allocated life span are terminated
- The reaming particles are animated and their shading parameters changed according to the controlling processes
- The particles are rendered

Life of a Particle



Particle Generation

- A random number of particles around an average can be generated
- Similar number of particles should be terminated to ensure constant number of particles at each frame

$$\#\text{ of particles} = average + Rand(\_) \cdot range$$

$$\#\text{ of particles} = average + Rand(\_) \cdot range \cdot screenArea$$

Attributes of Particles

- Position
- Velocity
- Shape Parameters
- Color
- Transparency
- Lifetime (in number of frames)

Particle Animation

- Effect of forces modeled on the environment is computed as acceleration on the particle
  - Gravity, wind, force fields, collisions with environment objects
- Acceleration is used to update the particle's velocity
- Average velocity is used to update the position
- Shape can be a function of velocity
  - An ellipse that elongates with respect to velocity

Flocking Behavior

- Example:
  - A flock of birds in flight
- Local perception and behavior of boids (i.e. flock members)
  - Limited intelligence and simple physics
- Emergent behavior
  - Flying in a diamond shape or V shape
  - Splitting, merging

Flocking Behavior

- Two main forces to keep a collection of objects behaving like a flock
  - Collision avoidance
    - With the environment
    - With other members
  - Flock centering
    - Can be achieved using localized control

Local Control

- Computationally desirable
- More realistic
- Three processes modeled in local control
  - Physics
  - Perception
  - Reasoning and reaction
    - Negotiates among the various demands due to perception
    - Collision avoidance
    - Flock centering
    - Velocity matching

Perception

- Sight of a boid is restricted to a neighborhood. To maintain a localized filed of view a boid should
  - Be aware of itself and two or three of its neighbors
  - Be aware of what is in front of it and have a limited field of view
  - Be influenced by objects within the line of sight based on size and distance
  - Have a general migratory urge but no global objective
  - Not follow a designated leader
  - Not have knowledge about a global flock center

Interacting with other flock members

- Attractive force
  - To move with the flock
- Repulsive force
  - A shorter range force for not to collide with neighbors

Global Control

- A global goal that can effect all the members of the flock or just the leader
- Following a specific path

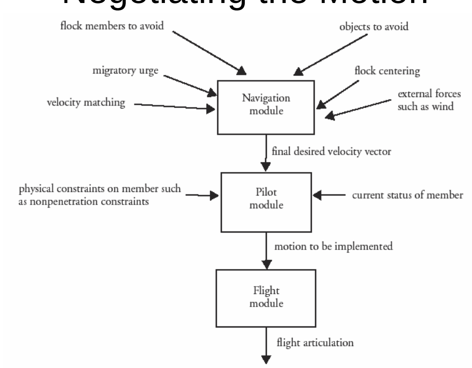
Flock Leader

- In real life leaders change periodically
  - The wind resistance is strongest for the leader of a flying flock of birds
- But in animation, it may be easier to have one designated leader whose motion is scripted along a path

Negotiating the Motion

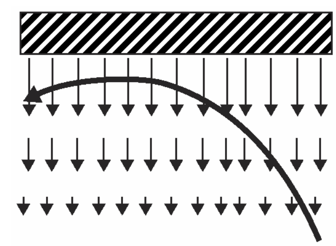
- Three low-level controllers
  - Collision avoidance
  - Velocity matching
  - Flock centering
- A priority based weighting scheme can be used to combine the individual requests from the low level controllers

Negotiating the Motion

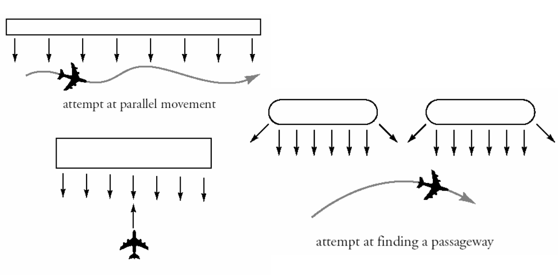


Avoiding Collision

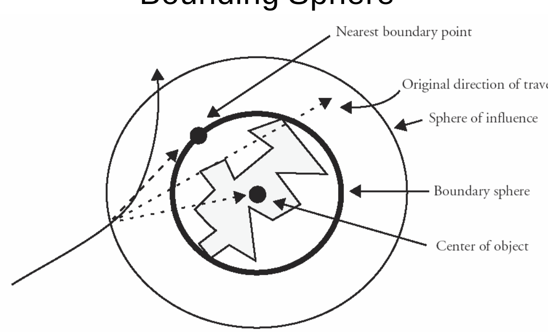
- Using a force field to direct flock members away from an obstacle



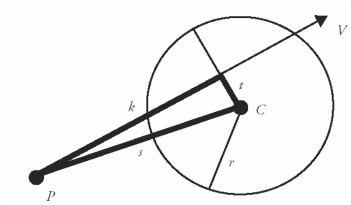
Problems with Force Field Technique



Collision Avoidance Using A Bounding Sphere



Testing for Collision with the Bounding Sphere



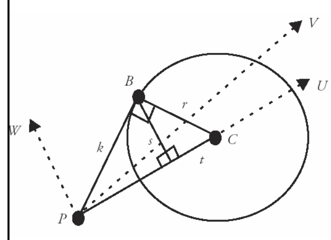
$$s = |C - P|$$

$$k = (C - P) \bullet \frac{V}{|V|}$$

$$t = \sqrt{s^2 - k^2}$$

$$t < r$$

Calculation of the Nearest Boundary Point



$$k = \sqrt{|C - P|^2 - r^2}$$

$$r^2 = s^2 + t^2$$

$$k^2 = s^2 + (|C - P| - t)^2$$

$$k^2 = r^2 - t^2 + (|C - P| - t)^2$$

$$k^2 = r^2 - t^2 + |C - P|^2 - 2 \cdot |C - P| \cdot t + t^2$$

$$t = \frac{k^2 - r^2 - |C - P|^2}{-2 \cdot |C - P|}$$

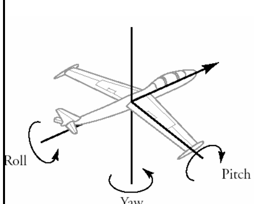
$$s = \sqrt{r^2 - t^2}$$

$$U = \frac{C - P}{|C - P|}$$

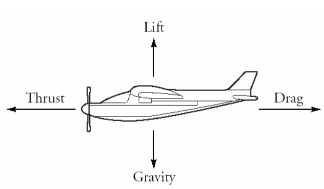
$$W = \frac{(U \times V) \times U}{|(U \times V) \times U|}$$

$$B = P + (|C - P| - t) \cdot U + s \cdot W$$

Modeling Flight

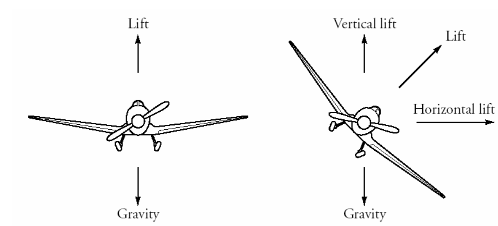


Rotation Types of Flight



Forces of Flight

Lifting Forces



Important Points on Modeling Flight

- Turning is effected by horizontal lift
- Increasing pitch increases drag
- Increasing speed increases lift

Autonomous Behavior