CS 174C Week 6

Aidan Jan

February 14, 2024

Collision Detection and Response

Aside: Object Geometry

When considering particles colliding with objects, we need to know how to represent objects, and how to answer:

- Is a particle inside/outside an object?
- Does the particle trajectory cross into the object?
- What is the normal vector at some point on the object's surface?
- What is the distance/direction to the surface in space?

Standard geometric representations:

- Special geometries: plane, sphere, cylinder, prism, ...
- Height fields: $z = h(x, y)$
- Triangle meshes - closed or open
- Implicit functions: $f(x, y, z) = 0$

Ground Collisions

Collisions With a Plane

Represent the plane with a point P on the plane, and the (outward) normal n of the plane

- Often simply $P = [0, 0, 0]^T$, $n = [0, 1, 0]^T$ - the ground plane
- Particle at position x is "inside" plane if $(x - P) \cdot n < 0$
- Trajectory crosses if $(x - P) \cdot n$ changes sign
- Distance to surface:
  - if n is unit length, $(x - P) \cdot n$ is the "signed distance"
    - $vert(-P) \cdot n|$ is the regular distance
  - n or $-n$ is direction to the closest point on the surface
    - $-n$ if x is "outside"

Collisions With a Sphere

Represent the sphere with a center point C and a radius r

- Particle at position x is inside if $\|x - C\| - r < 0$
- Trajectory cross is complicated
  - Need to solve quadratic equation for intersection of straight line trajectory...
- Outward object normal is $(x - C)/r$
- Signed distance is $|x - C| - r$
- Direction to closest point on surface is $\pm(x - C)/|x - C|$
  - Sign depends on whether inside or outside
  - Beware of divide by zero at $x = C$
  - Note: matches up with object normal again!

Collisions With Height Fields

Especially good for terrain - a 2D array of heights

- Maybe stored as an image
  - i.e., a displacement map from a plane

Split up plane into triangles

- Particle inside:
  - Figure out which triangle (x, y) belongs to, check z against equation of triangle's plane
- Trajectory cross (for a stationary height field):
  - Check all triangles along path (use 2D line-drawing algorithm to figure out which cells to check)
- Object normal: get from the triangle
- Distance, etc.: not so easy, but vertical distance is easy for shallow height fields

Collisions With Triangle Meshes

For any decent sized mesh, will need to use an acceleration structure

- Could use background (hash-)grid, octree, kd-tree
- Can also use bounding volume (BV) hierarchy
  - Spheres, axis-aligned bounding boxes, oriented bounding boxes, polytopes, ...
- More exotic structures exist

Particle inside (for a closed mesh):

- Shoot a ray out to infinity and count the number of crossings

Trajectory cross (for a stationary mesh):

- For each candidate triangle (from acceleration structure), check a sequence of determinants

Collision Resolution

- We can now detect collisions. Now what?

Repulsion Velocity Fields

How do we create a repulsion velocity field?

- $v(x) = f(\text{distance}(x))n(x)$
  - $n(x)$ is the outward unit normal to surface at x
  - f is some function that monotonically decreases to zero
    * Exponential $f(d) = e^{-kd}$
    * Linear drop, truncated to zero: $f(d) = \max(0, m - kd)$
    * Or more complicated
  - Outward direction is plus/minus direction to closest point

Aside: useful for more than just collisions

- e.g., firing particles streaming out of an object

Repulsion Force Fields

Can do exactly the same trick for force-based motion

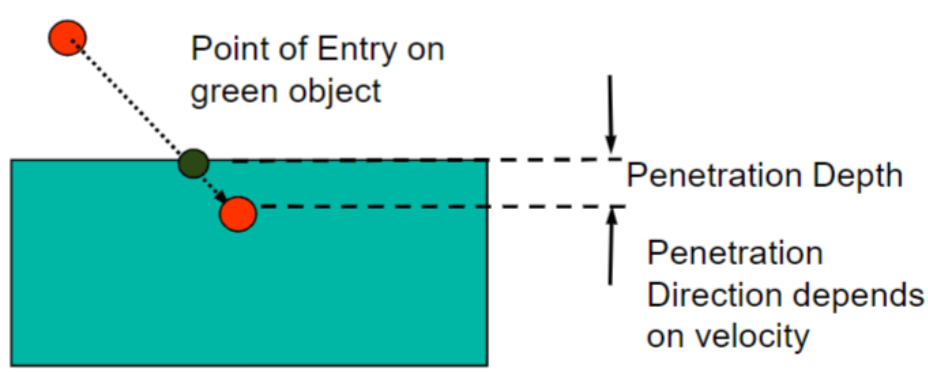
- Add repulsion field to $f(x)$

Simple, often works, but there are sometimes problems

- What are you trying to model?
- Robustness - high velocity impacts can penetrate arbitrarily far
  - High velocity impacts may go straight through thin objects!
- How much of a rebound do you want?

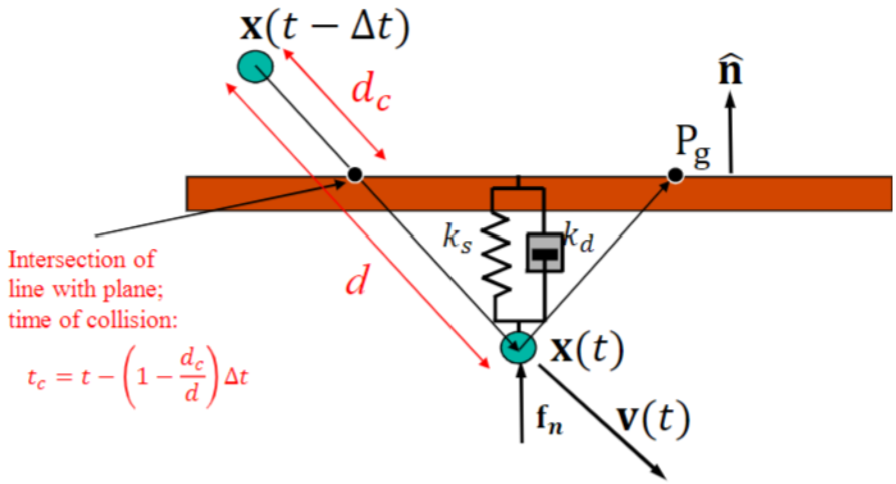
Penalty Methods

Penalize Penetration:



- Springs and damers to the rescue!
  - Attach a zero-length "virtual" viscoelastic element at the point of entry

For a Plane:



- Normal force:

$$f_n = k_s\left((P_g - x(t)) \cdot \hat{n}\right)\hat{n} - k_d(v(t) \cdot \hat{n})\hat{n}$$

$$f_n \cdot \hat{n} > 0$$

Spring and Damping Constants

How do you come up with reasonable values for spring constants and damping constants?

- And how do you pick good step sizes for differential equation solver (Symplectic Euler, etc.)

Look at 1-dimensional, simplified model

- $ma = F = -Kx - Dv$
- where...
  - m is the mass, a is the acceleration, F is the force
  - K is the spring constant, D is the damping constant
- Can solve it analytically.

Aside: Underdamped

$$D^2 - 4MK < 0$$

- Oscillation with frequency:

$$\omega \sim \frac{1}{2\pi}\sqrt{\frac{K}{M}}$$

- Characteristic time:

$$t \sim 2\pi\sqrt{\frac{M}{K}}$$

- Exponentially decays at rate:

$$r = -\frac{D}{2M}$$

- Characteristic time:

$$t \sim \frac{2M}{D}$$

Aside: Overdamped

$$D^2 - 4MK > 0 \qquad D = 2\sqrt{MK}$$

- No continued oscillation
- Fastest decay possible at rate:

$$r = -\frac{D}{2M}$$

- Characteristic time:

$$t \sim \frac{2M}{D}$$

Aside: Critically Damped

$$D^2 - 4MK = 0$$

- No continued oscillation
- Exponentially decays at rates:

$$r \sim -\frac{K}{D}, -\frac{D}{M}$$

- Characteristic times:

$$t \sim \frac{D}{K}, \frac{M}{D}$$

Numerical Time Steps

Should be proportional to minimum characteristic time

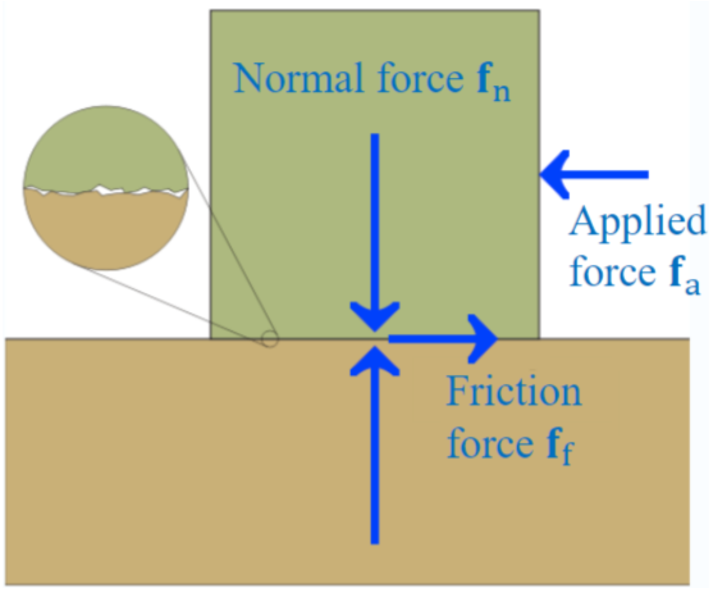
- Implicit time integration methods like Backward Euler actually let you take larger steps with stability, but kill all hope of accuracy for systems with small charateristic time

For nonlinear multi-dimensional forces, what are K and D?

- Estimate them by figuring out what is the fastest $|F|$ can change if you modify x or v respectively
- This is all very approximate, so don't get hung up on getting the "right" answer
- Anyhow, will ultimately need a "fudge factor" (from trial-and-error experiments)

Friction

Coulomb friction: $f_{\mathrm{f}} \leq \mu f_{\mathrm{n}}$



- Coefficient of friction μ
- f_f opposes f_a:

$$f_f \leq -\mu \|f_n\| \frac{f_a}{\|f_a\|}$$

- Traction: $f_f = -f_a$ until $f_f > f_T \ldots$, then motion occurs

Friction Models

Static (Coulomb) friction: $\|f_s\| = \mu_s \|f_n\|$

- $f_n = -(f \cdot n)n$ is the normal component of the force f
- Must exceed static friction for object to start moving

For object in motion,

- Kinetic (constant) friction model: $f_k = -\mu_k \|f_n\| \frac{v_t}{\|v_t\|}$
  - Tangential component of velocity: $v_t = v - v_n$
  - Normal component of velocity: $v_n = (v \cdot n)n$
- Viscous (linear) friction model: $f_v = -\mu_k \|f_n\| v_t$
  - i.e., a drag force that acts tangentially with magnitude proportional to the tangential velocity and normal force

Collisions

Impulsive Collisions

Model collision as a discrete event - a bounce

- Input: incoming velocity, object normal
- Output: outgoing velocity

Need some idea of how "elastic" the collision

- Fully elastic - reflection
- Fully inelastic - sticks (or slides)

Newtonian Collisions

For normal vector n, compute the components of the velocity v

- Normal component $v_N = (v \cdot n)n$
- Tangential component $v_T = v - v_N$
- Reflect normal component to obtain rebound velocity with coefficient of restitution r

$$v_{new} = v_t - rv_n$$

Relative Velocity in Collisions

What if the particle hits a moving object?

- Now process collisions in terms of the relative velocity
  - $v_{\text{ref}} = v_{\text{particle}} - v_{\text{object}}$
  - Resolve normal and tangential components of the relative velocity
  - Reflect normal part appropriately to get new v_{rel}
  - Then new $v_{\text{particle}} = v_{\text{object}} + v_{\text{rel}}^{\text{new}}$