Notes

- Typo in test.rib --- fixed on the web now (PointsPolygon --> PointsPolygons)

Contact Friction

- Some normal force is keeping $v_N=0$
- Coulomb's law ("dry" friction)
  - If sliding, then kinetic friction:

$$F_{friction} = -\mu_k |F_{normal}| \frac{v_T}{|v_T|}$$

  - If static ($v_T=0$) then stay static as long as

$$|F_{friction}| \le \mu_s |F_{normal}|$$

- "Wet" friction = damping

$$F_{friction} = -D |F_{normal}| v_T$$

Collision Friction

- Impulse assumption:
  - Collision takes place over a very small time interval (with very large forces)
  - **Assume** forces don't vary significantly over that interval---then can replace forces in friction laws with impulses
  - This is a little controversial, and for articulated rigid bodies can be demonstrably false, but nevertheless…
  - Normal impulse is just $m\Delta v_N = m(1+\varepsilon)v_N$
  - Tangential impulse is $m\Delta v_T$

Wet Collision Friction

- So replacing force with impulse:

$$m\Delta v_T = -D |m\Delta v_N| v_T$$

- Divide through by m, use $v_T^{after} = v_T^{before} + \Delta v_T$

$$v_T^{after} = v_T^{before} - D|\Delta v_N| v_T^{before}$$
$$= \left(1 - D|\Delta v_N|\right) v_T^{before}$$

- Clearly could have monotonicity/stability issue
- Fix by capping at $v_T=0$, or better approximation for time interval
  e.g. $$v_T^{after} = e^{-D|\Delta v_N|} v_T^{before}$$

Dry Collision Friction

- Coulomb friction: assume $\mu_s = \mu_k$
  - (though in general, $\mu_s \geq \mu_k$)
- Sliding: $m\Delta v_T = -\mu |m\Delta v_N| \frac{v_T^{before}}{|v_T^{before}|}$
- Static: $|m\Delta v_T| \leq \mu |m\Delta v_N|$
- Divide through by m to find change in tangential velocity

Simplifying…

- Use $v_T^{after} = v_T^{before} + \Delta v_T$
- Static case is $v_T^{after} = 0 \Rightarrow \Delta v_T = -v_T^{before}$ when $|v_T^{before}| \leq \mu |\Delta v_N|$
- Sliding case is

$$v_T^{after} = v_T^{before} - \mu |\Delta v_N| \frac{v_T^{before}}{|v_T^{before}|}$$

- Common quantities!

Dry Collision Friction Formula

- Combine into a max
  - First case is static where v_T drops to zero if inequality is obeyed
  - Second case is sliding, where v_T reduced in magnitude (but doesn't change signed direction)

$$v_T^{after} = \max\left(0, 1 - \frac{\mu |\Delta v_N|}{|v_T^{before}|}\right) v_T^{before}$$

Where are we?

- So we now have a simplified physics model for
  - Frictionless, dry friction, and wet friction collision
  - Some idea of what contact is
- So now let's start on numerical methods to simulate this

"Exact" Collisions

- For very simple systems (linear or maybe parabolic trajectories, polygonal objects)
  - Find exact collision time (solve equations)
  - Advance particle to collision time
  - Apply formula to change velocity
    (usually dry friction, unless there is lubricant)
  - Keep advancing particle until end of frame or next collision
- Can extend to more general cases with conservative ETA's, or root-finding techniques
- **Expensive** for lots of coupled particles!

Fixed collision time stepping

- Even "exact" collisions are not so accurate in general
  - [hit or miss example]
- So instead fix $\Delta t_{collision}$ and don't worry about exact collision times
  - Could be one frame, or 1/8th of a frame, or …
- Instead just need to know did a collision happen during $\Delta t_{collision}$
  - If so, process it with formulas

Relationship with regular time integration

- Forgetting collisions, advance from x(t) to $x(t+\Delta t_{collision})$
  - Could use just one time step, or subdivide into lots of small time steps
- We approximate velocity (for collision processing) as constant over time step:

$$v = \frac{x(t+\Delta t) - x(t)}{\Delta t}$$

- If no collisions, just keep going with underlying integration

Numerical Implementation 1

- Get candidate $x(t+\Delta t)$
- Check to see if $x(t+\Delta t)$ is inside object (interference)
- If so
  - Get normal n at $t+\Delta t$
  - Get new velocity v from collision response formulas applied to average $v=(x(t+\Delta t)-x(t))/\Delta t$
  - Integrate $x(t+\Delta t)=x(t+\Delta t)_{old} + \Delta t \Delta v$

Robustness?

- If a particle penetrates an object at end of candidate time step, we fix that
- But new position (after collision processing) could penetrate another object!
- Maybe this is fine-let it go until next time step
- But then collision formulas are on shaky ground…
- Switch to repulsion impulse if x(t) and x(t+Δt) both penetrate
  - Find Δv_N proportional to final penetration depth, apply friction as usual

Making it more robust

- Other alternative:
  - After collision, check if new x(t+Δt) also penetrates
  - If so, assume a 2nd collision happened during the time step: process that one
  - Check again, repeat until no penetration
  - To avoid infinite loop make sure you lose kinetic energy (don't take perfectly elastic bounces, at least not after first time through)
  - Let's write that down:

Numerical Implementation 2

- Get candidate x(t+Δt)
- While x(t+Δt) is inside object (interference)
  - Get normal n at t+Δt
  - Get new velocity v from collision response formulas and average v
  - Integrate collision: $x(t+\Delta t)=x(t+\Delta t)_{old}+\Delta t\Delta v$
- Now can guarantee that if we start outside objects, we end up outside objects

Micro-Collisions

- These are "micro-collision" algorithms
- Contact is modeled as a sequence of small collisions
  - We're replacing a continuous contact force with a sequence of collision impulses
- Is this a good idea?
  - [block on incline example]
- More philosophical question: how can contact possibly begin without fully inelastic collision?

Improving Micro-Collisions

- Really need to treat contact and collision differently, even if we use the same friction formulas
- Idea:
  - Collision occurs at start of time step
  - Contact occurs during whole duration of time step

Numerical Implementation 3

- Start at x(t) with velocity v(t), get candidate position x(t+Δt)
- Check if x(t+Δt) penetrates object
  - If so, process **elastic collision** using v(t) from start of step, **not** average velocity
  - Replay from x(t) with modified v(t) or simply add ΔtΔv to x(t+Δt) instead of re-integrating
  - Repeat check a few (e.g. 3) times if you want
- While x(t+Δt) penetrates object
  - Process **inelastic contact** (ε=0) using **average** v
  - Integrate +Δt Δv

Why does this work?

- If object resting on plane y=0, v(t)=0 though gravity will pull it down by the end of the timestep, t+Δt
- In the new algorithm, elastic bounce works with pre-gravity velocity v(t)=0
  - So no bounce
- Then contact, which is inelastic, simply adds just enough Δv to get back to v(t+Δt)=0
  - Then x(t+Δt)=0 too
- NOTE: if ε=0 anyways, no point in doing special first step - this algorithm is equivalent to the previous one

Moving objects

- Same algorithms, and almost same formulas:
  - Need to look at relative velocity $v_{particle}-v_{object}$ instead of just particle velocity
  - As before, decompose into normal and tangential parts, process the collision, and reassemble a relative velocity
  - Add object velocity to relative velocity to get final particle velocity
- Be careful when particles collide:
  - Same relative Δv but account for equal and opposite forces/impulses with different masses…

Moving Objects…

- Also, be careful with interference/collision detection
  - Want to check for interference at end of time step, so use object positions there
  - Objects moving during time step mean more complicated trajectory intersection for collisions

Collision Detection

- We have basic time integration for particles in place now
- Assumed we could just do interference detection, but…
- Detecting collisions over particle trajectories can be dropped in for more robustness - algorithms don't change
  - But use the normal at the collision time