Notes

- Assignment 1:
  - Check on the web site for a software particle renderer (w/o OpenGL, with antialiasing and alpha-blending)
- Please read Stam & Fiume, "Turbulent wind fields…", SIGGRAPH'93

Aside: noise

- You know there should be some detail in there, but you don't know *what* detail
  - So use random numbers
- The physics comes in to guide the random distribution
- Things get interesting when you look at "noise" fields
  - n(x,y,z,t)
  - White noise - no correlation (and no smoothness) - isn't so useful
  - Need to introduce correlation/smoothness

Perlin Noise

- (sort of) standard in graphics
- See web-page for references
- In a nutshell:
  - Randomly decide on (unit) gradient of n at grid points: $\nabla n_{i,j,k} = G_{H(i,j,k)} \quad (with\ |G_h| = 1)$
  - Enforce n=0 at grid points
  - Interpolate with (sort of) spline
- Result is smooth with irregular variations on the scale of the grid spacing but no larger

Combining noise

- While Perlin noise is very useful (fast to evaluate!), how do you control it?
- Can use f(n(x,y,z)), but that still doesn't introduce different length scales
- Graphics notion of "turbulence":
  - Add dilated noise: $\sum_{i=0}^{N} \alpha_i f\left(n(\beta^i x)\right)$
  - Use $\beta \approx 2, \quad \alpha_i \propto \frac{1}{\left(\beta^i\right)^s}$

Spectral perspective

- Smooth noise function is (approximately) band-limited:
  - Close to no frequencies below grid scale,
  - Fast decay of higher frequencies (smoother=faster)
- Adding dilated noise makes sense if we want to shape spectrum better
- Why not just directly go for spectrum?
  - Many physical models give you a spectrum anyways

Spectral noise

$$n(x,y,z) = \sum_{i,j,k} A_{i,j,k} \cos\left(ix + jy + kz + \theta_{i,j,k}\right)$$

- Specify amplitude as function of frequency (wave number)
  - Randomize phase shifts
  - Can randomize amplitude a little too
- Use Fast Fourier Transform to get a (periodic) grid of noise
- Then interpolate from the grid
- Avoid periodicity by adding 2+ different size grids (not integer multiples!)

Collision and Contact

- We can integrate particles forward in time, have some ideas for velocity or force fields
- But what do we do when a particle hits an object?
- No simple answer - depends on problem as always
- General breakdown:
  - Interference vs. collision detection
  - What sort of collision response: (in)elastic, friction
  - Robustness: do we allow particles to actually be inside an object?

Interference vs. Collision

- Interference (=penetration)
  - Simply detect if particle has ended up inside object, push it out if so
  - Works fine if $v\Delta t < \frac{1}{2} w$ [w=object width]
  - Otherwise could miss interaction, or push dramatically the wrong way
  - The ground, thick objects and slow particles
- Collision
  - Check if particle **trajectory** intersects object
  - Can be more complicated, especially if object is moving too…
- For now, let’s stick with the ground (y=0)

Repulsion Forces

- Simplest idea (conceptually)
  - Add a force repelling particles from objects when they get close (or when they penetrate)
  - Then just integrate: business as usual
  - Related to penalty method: instead of directly enforcing constraint (particles stay outside of objects), add forces to encourage constraint
- For the ground:
  - $F_{repulsion}=-Ky$ when $y<0$ [think about gravity!]
  - …or $-K(y-y_0)-Dv$ when $y<y_0$ [still not robust]
  - …or $K(1/y-1/y_0)-Dv$ when $y<y_0$

Repulsion forces

- Difficult to tune:
  - Too large extent: visible artifact
  - Too small extent: particles jump straight through, not robust (or time step restriction)
  - Too strong: stiff time step restriction, or have to go with implicit method - but Newton will not converge if we guess past a singular repulsion force
  - Too weak: won't stop particles
- Rule-of-thumb: don't use them unless they really are part of physics
  - Magnetic field, aerodynamic effects, …

Collision and Contact

- Collision is when a particle hits an object
  - Instantaneous change of velocity (discontinuous)
- Contact is when particle stays on object surface for positive time
  - Velocity is continuous
  - Force is only discontinuous at start

Frictionless Collision Response

- At point of contact, find normal n
  - For ground, n=(0,1,0)
- Decompose velocity into
  - normal component $v_N=(v\cdot n)n$ and
  - tangential component $v_T=v-v_N$
- Normal response: $v_N^{after}=-\varepsilon v_N^{before}, \quad \varepsilon\in[0,1]$
  - $\varepsilon=0$ is fully inelastic
  - $\varepsilon=1$ is elastic
- Tangential response
  - Frictionless: $v_T^{after}=v_T^{before}$
- Then reassemble velocity $v=v_N+v_T$

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 $v_T^{after} = v_T^{before} + \Delta v_T$

$$\begin{aligned} v_T^{after} &= v_T^{before} - D|\Delta v_N| v_T^{before} \\ &= (1 - D|\Delta v_N|) v_T^{before} \end{aligned}$$

- 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, $\mu_s \ge \mu_k$)
- Sliding: $m\Delta v_T = -\mu |m\Delta v_N| \frac{v_T^{before}}{|v_T^{before}|}$
- Static: $|m\Delta v_T| \le \mu |m\Delta v_N|$
- 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 $\left|v_T^{before}\right| \le \mu\left|\Delta v_N\right|$
- Sliding case is

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

- 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)

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

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**! [think springs]

Fixed collision time stepping

- Even "exact" collisions are just first order 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, forget this average v, and 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 and average v
  - Replay $x(t+\Delta t)=x(t)+\Delta t v$

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… [show example in concavity]
  - Switch to repulsion impulse if x(t) and $x(t+\Delta t)$ 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 $x(t+\Delta t)$ 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 $x(t+\Delta t)$
- While $x(t+\Delta t)$ is inside object (interference)
  - Get normal n at $t+\Delta t$
  - Get new velocity v from collision response formulas and average v
  - Replay $x(t+\Delta t)=x(t) + \Delta t\, v$
- 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)
  - Could 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
  - Replay x(t+Δt)=x(t)+Δt v

Why does this work?

- If object resting on plane y=0, v(t)=0, though gravity will pull it down by 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