Notes

- Assignment 3 due today
  - Or by the time I get into my office tomorrow morning

Spacetime problems

- For simple models, simple objectives, simple constraints: works great!
- For more complex stuff, unless initial guess at motion is really good, optimization gets stuck
  - Local minimum problem
- We're relying too heavily on optimization
  - We have lots more knowledge, data, etc. that we could bring in
- Let's next look at a completely non-physical alternative to generating new motion:
  motion warping

Motion Warping

- First reference:
  - Witkin & Popovic, "Motion warping", SIGGRAPH'95
- Idea: start with detailed motion clip
  - E.g. motion capture
- Smoothly deform (warp) it to satisfy new keyframes
- Hope that result still looks realistic

Keyframe constraints

- Animator specifies new motion by selecting a few frames from original, and modifying them
  - Change the pose any way you want
  - Change the time (when the frame should happen)
- Break this down separately for each motion curve (e.g. joint angles, root position, etc.)

Warping a motion curve

- First do the retiming:
  - We basically did this before: correspondences provide control points for a spline that controls time map
- Next alter the curve values:
  - Model is $\theta_{new}(t_{new}) = a(t)\theta(t) + b(t)$
  - Scaling by a(t), offset by b(t)
  - User specifies if a specific keyframe change should be accomplished by scaling or by offset: this sets $a(t_i)$ and $b(t_i)$
  - Then pass splines through the a's and b's to get final result

Why should it work?

- For sparse keyframes, spline warp varies very slowly
- The motion detail varies quickly - so isn't effected much by warp
- As long as warp isn't too extreme, motion detail should still be realistic

Motion Parameterization

- Warping extrapolates from 1 motion: scaled or offset details could be noticeably wrong
- Better to interpolate between existing motion
  - For example, Kovar & Gleicher, "Flexible Automatic motion blending with registration curves", SCA03; Kovar & Gleicher, "Automated extraction and parameterization…", SIGGRAPH04
  - Automatically find similar clips of motion
  - Automatically construct timewarps to synchronize
  - Take weighted average to blend between them
    - Or solve for blend that achieves some user constraint
  - Can make it a transition: blend over time
  - Constraints are tricky

Background: Fourier Series

- Alluded to notions of
  - "smooth" vs. "rough"
  - "slowly-varying" vs. "quickly varying"
  - "low frequency" vs. "high frequency"
- There is a formal way to discuss this: Fourier analysis
- The cornerstone of signal processing
  - Motion curves look like signals…
  - Also comes up in physics-based animation

The Fourier Basis

- A few different ways to express it
  - Start with simplest to visualize
- Say we have a vector of n values:
  - $(F_0, F_1, F_2, \ldots, F_{n-1})$
  - In standard basis: $F(i) = \sum_{j=0}^{n-1} F_j e_j(i)$
- Introduce Fourier basis consisting of

$$\left\{\cos\left(2\pi j \tfrac{i}{n}\right)\right\}_{j=0}^{n/2}, \left\{\sin\left(2\pi j \tfrac{i}{n}\right)\right\}_{j=1}^{n/2-1}$$

  - Change to the new basis:

$$F(i) = \sum_{j=0}^{n/2} C_j \cos(2\pi j \tfrac{i}{n}) + \sum_{j=1}^{n/2-1} S_j \sin(2\pi j \tfrac{i}{n})$$

  - C's and S's are the Fourier coefficients

Fourier Properties

- Basis vectors are orthogonal
  - Easy to normalize - some people do
  - If you put basis vectors into columns of Q, then coefficients are $Q^{-1}F$ which is just Q^TF up to rescaling
- Fast Fourier Transform (FFT):
  O(n log n) time to compute QC or Q^TF
- Coefficients tell us about frequency content of signal
  - Low frequency coefficients correspond to smooth, slowly varying part
  - High frequency coefficients correspons to rough, quickly varying part