Animation by Example
Lecture 2: Motion Signal Processing

Michael Gleicher
University of Wisconsin - Madison
www.cs.wisc.edu/~gleicher
www.cs.wisc.edu/graphics

Recap: Representation
- Represent human as hierarchical skeleton
- Vector with 1 position, 1 absolute orientation, many relative orientations
- Vector really isn’t in \( \mathbb{R}^n \)
- Many different ways to do this
- Many things to be careful of

How do skeletons differ?
- Obvious ways?
  - Topology
    - number of bones
    - Connectivity of bones
  - Joint Types
  - Bone lengths
  - Anatomical / skin relations
    - Is spine in middle of body, or up the back?

Subtle Skeletal Differences
- What to measure angles with respect to
  - Doesn’t matter, as long as we agree
- Poses (design of a skeleton)
  - Zero Pose / Base Pose
  - Dress or Binding pose
  - Frankenstein Pose
  - Da Vinci Pose
  - Rest Pose (real pose of actor)
- Need to figure out how to get between these

Target Poses
- AKA Zero Pose. What happens when all joints are set to zero
- AKA Dress Pose. What is required to fit in the skin
- Poses specified by technical needs
- May look different for each setup

Reference Poses
- All limbs vertical (AKA Zombie)
- Arms Horizontal, Legs Spread
- Defined by how pose looks
- Not the same values
- How the actor stands at rest
Why do we care?

- Motion data is relative to base pose
  - Tells us how to interpret data
- Need binding pose to skin character
- Need reference poses for calibration
- Try to unify poses
  - Base pose = Frankenstein?
  - Base pose = Bind pose?
  - Base pose = Rest pose?
  - Animator’s T-Pose vs. Anatomical T-Pose?

Recap: Representation

- Represent human as hierarchical skeleton
- Vector with 1 position, 1 absolute orientation, many relative orientations
- Vector really isn’t in \( \mathbb{R}^n \)
  - \( \mathbb{R}^3 \times \text{SO}(3) \times \text{SO}(3)^{n-2} \)
- Many different ways to do this
- Many things to be careful of

Now... on to motion!

- Motion is a function of time
  - Given time, provide a pose
- Often represented as samples
  - Sparse samples + interpolation
  - Dense samples (at frames)
- How to manipulate sets of samples?

Motions

- A motion is a map \( t \rightarrow \text{pose} \)
  \( m(t) : \mathbb{R} \rightarrow \mathbb{R}^n \)
- Not really \( \mathbb{R}^n \), but close enough
  - Only matter if we manipulate vector
  - Euler Angles – arithmetic + pray
  - Quaternions
    - Arithmetic + renormalize
    - Log map + arithmetic + Exp map

The General Challenge

- Get a specific motion
  - From capture, keyframe, ...
  - Specific character, action, mood, ...
- Want something else
  - But need to preserve original
  - But we don’t know what to preserve
  - Can’t characterize motion well enough*

*This is a working assumption of my research. I’d love to be proven wrong.

Three Problems

- Where does \( X \) live in the data?
  - Where \( X \in \{ \text{style, personality, emotion, } \ldots \} \)
  - The things to keep or add
- Small artifacts can destroy realism
  - Eye is sensitive to certain details
  - Amazing what you can’t get away with
    - Kovar, Schreiner and Gleicher, SCA ’02
  - How to specify what you want
Why Edit Motion?

- What you get is not what you want!
- You get observations of the performance
  - A specific performer
  - A real human
  - Doing whatever they did
  - With the noise and “realism” of real sensors
- You want animation
  - A character
  - Doing something
  - And maybe doing something else...

Manipulating motion

- Manipulate time
  \[ m(t) = m_0(f(t)) \]
  \[ F : \mathbb{R} \rightarrow \mathbb{R} \text{ “time warp”} \]
- Manipulate value
  \[ m(t) = f(m_0(t)) \]
  \[ F : \mathbb{R}^n \rightarrow \mathbb{R}^n \]
  \[ F : \mathbb{R}, \mathbb{R}^n \rightarrow \mathbb{R}^n \]

Time Manipulations

- \[ m(t) = m_0(f(t)) \]
- Time scaling
  \[ f(t) = kt \]
- Time shifting
  \[ f(t) = t + k \]
- Time warping
  - Interpolate a table
  - Align events

Value Manipulations

- Scale?
- Shift?
- Convole (linear filter)
  \[ “Add” \text{ to another motion } \]
  \[ m(t) = m_0(t) + a(t) \]

Signal Processing Example: Noise Removal

- Noise comes from errors in process
- Sensor errors
- Fitting errors
- Bad movements
- Noise is “data” that we don’t want

Where’s the Noise?

- Sometimes identification is easy:
  - Clearly wrong (foot through floor)
  - Marked wrong (missing data - gaps)
  - More often, need to guess
- Might be a subtle twitch...
- Might be person shaking...
- Might be sensor errors...
Noise Detection

- Use heuristics and rules of thumb to identify noise
- Use info about which body part as a discriminator
  - Extremities are more likely to have sharp movement
  - "Speed" of the movement affects how prevalent noise is
    - Visual signal/noise ratio decreases as movement gets slower

Mocap Noise Misconception

- Things in the world don't change that fast (have high freq)
- If there are high freqs, must be noise
- Get rid of high freqs (quick changes)
- Low-Pass Filter (LPF) easy (weighted average, FIR, ...)

Low-Pass Filters vs. Noise

- We want to remove the noise, to get back a signal that looks like

High Frequencies

- PROBLEM: High frequencies can be important!
  - Getting rid of them makes motion look soggy
- ANSWER: Do not over apply LPF
  - How much is enough?
  - Use a little LPF

Treating Mocap Noise

- Small amounts of Low Pass Filtering
- Noise modeling
- Adaptive filters
- Non linear filters
- Hybrid solutions
Important Intuition

- High Frequencies are Important!
  - Don’t occur often
  - Always significant
    - Impact
    - Rapid, sudden movement
    - Emphasis
    - Sensitivity of perception

Additive Editing

- “add” two motions together
  - Addition really means combine
  - Interpolate between
  - Compose
- Two common uses
  - Motion displacement maps
  - Motion blending

Changing Motions

- High Frequencies are important
- Can’t remove
  - Don’t want to take away
- Can’t add
  - Don’t want to put something in

One way to use this: Motion Displacement Maps

- A.K.A. Motion Warps
- Add in another motion
  \[ m(t) = m_0(t) + d(t) \]
  - Pick other motion so that it doesn’t stick out (no high frequencies)

Band-limited adaptation

- High frequencies are important
  - Eye is sensitive to them
  - Always signifies important events
- Avoid high frequency changes
  - Preserve existing high frequencies
  - Avoid adding new ones
- Band limit the changes
  - Not the resulting motions

Band-limited adaptation?

- Can’t look at individual frames
- Need to look across space and time
- Popping can be worse than skating
Motion Blending
- “Add” two motions together
  - Really interpolate
    \[ m(t) = a m_0(t) + (1-a) m_1(t) \]
- Note: this is a per-frame operation
  - We’re really interpolating between poses!

Does interpolation make sense?
- No!
- Yes – but only if poses are similar

How to use blending
- Interpolate similar motions
- Be sure to make time correspondence
- Transition between motions
  - Time varying blend \((a=0-1)\)
  - Over a short period of time
    - A bad pose isn’t such a big deal
  - Avoids discontinuities
    \[ m(t) = a(t) m_0(t) + (1-a(t)) m_1(t) \]

Transistion
- Very useful!
  - Often get small pieces of motion
  - Need to connect
  - Easy if motions are similar
  - Hard if motions are not similar

Transitions

What’s next...
- Use Transitions and Blending to synthesize new motions based on a library of examples!
Thanks!

- To the UW graphics gang.
- Animation research at UW is sponsored by the National Science Foundation, Microsoft, and the Wisconsin University and Industrial Relations program.
- House of Moves, IBM, Alias/Wavefront, Discreet, Pixar and Intel have given us stuff.
- House of Moves, Ohio State ACCAD, and Demian Gordon for data.
- And to all our friends in the business who have given us data and inspiration.