Why animate humans?
- Movies
- Television
- Videogames
- Training
- Simulation
- Analysis

Why is this hard?
- People are good at watching people!
- Human appearance is very complex
- People do many things
  - In many ways
  - Subtlety matters
  - Hard to describe movement
- "Normal" movements aren't interesting

Aspects of the Problem
- "Gross" Body movement
  NOT:
  - Appearance Models
  - Facial animation
  - Cloth, clothing, secondary movement
  - Hands

These lectures
1. Representation of humans
2. Motion capture processing and editing
3. Concatenative synthesis
4. Parametric synthesis
5. Skinning

Animation Apreciation 101
- Luxo Jr. Pixar, 1986
- Brilliance (Sexy Robot)
  - Robert Abel and Associates, 1985
  - Early motion capture
  - Early computer graphics look (chrome)
- Final Fantasy
  - Square Studios, 2001
  - Realistic, animated, human characters
- Hollowman
  - Sony Imageworks (effects), 2000
  - Complex human models, terrible dialog
Why did I show those?
- Motion is rich, expressive, complex
- Hard to describe mathematically
- Amount of detail in characters varies
  - Different representations needed

Where’s the math problem?
- How do we describe movement mathematically?
  - So we can use it on a computer
- How do we describe the thing that is moving?
  - The “character”
What is the character?
- Way to interpret a configuration
- A vector of parameters
- Some interpretation of these parameters such that a value can be drawn
- Representation

What is a motion?
- A motion maps times to configurations
  \[ m(t) \in \mathbb{R} \Rightarrow \mathbb{R}^n \]
- Vector-valued, time-varying signal
- Representation comes from creation
- All we have to do is define the functions!

Why is this so hard?
- We are good at looking at motion!
- Motion is very expressive
  - Mood, activity, personality, ...
- But those attributes are subtle
  - What makes a motion sad? Realistic?
- We lack vocabulary
  - Talk about motion with metaphor

Three main ways to make motion
- Create it by hand
- Compute it
- Capture it from a performer
- Animate by example
  - Re-use existing motions
  - Editing
  - Synthesis by Example

Creating Motion by Hand: Keyframing
- Skilled animators place “key” poses
  - Computer “in-betweens”
- Requires incredible amounts of talent
  - But can be done extremely well

Verdict: Produces the highest quality results, at a very high cost
Computing Motion: Procedural and Simulation

- Define algorithms to create motions
- Ad-hoc rules, or simulate physics
- Physics provides realism
- But how do you control it?

Verdict: Good for secondary effects, not for characters (yet)

Motion Capture and Performance Animation

- Use sensors to record a real person
- Get high-degree of realism
- Which may not be what you want...
- Possibility for real-time performance

Verdict: Good for realistic human motions. Scary to animators.
Motion Capture Technology: Optical Tracking
- User markers and special cameras
- Tracking + Math

Motion Capture Technology: Video
- An interesting and open problem...
- Limited information
  - But seemingly enough
  - Problem can be arbitrarily hard
  - Or easy – if you make assumptions
- Video is surprisingly bad

The subject of these lectures...
Animation by Example
- Good motion is hard to get
- Can’t get everything you need
- Need to create motion on the fly

- Re-use existing motions
  - Editing (change an existing motion)
  - Synthesis by example
    • (make a new motion from old ones)

Where to begin...
Some preliminaries
- Human Representation
  - Rigid bodies
  - Kinematics
- Motion Capture and Processing
  - Motion Signal Processing
Representation of Humans
- Need concise description of pose

Goal:
- Summarize pose as a vector
- Motion is vector valued function
- Compact, yet flexible
- Make constraints implicit

Modeling Humans
- Humans are complex!
  
Abstractions
- 206 bones, muscles, fat, organs, clothing, ...
- 206 bones, complex joints
- 53 bones, Kinematic joints

Abstractions vs. Reality (skeletons vs. humans)
- Representation of complex human structure with varying degrees of simplification
- Simple Pin Joint
- Complex tendon and bone system

How Realistic do you need?
- It depends!
  - Generally, small numbers of degrees of freedom (50-60)
  - Easier to animate/specify
  - Don’t really see the details from far away

Standard simplified models of humans
- Small numbers of degrees of freedom for gross motion
- Articulated figures
  - Rigid pieces
  - Sometimes stretching allowed
- Kinematic joints
  - Rotations between pieces
- Why this?
How to best match

- Can't be exact
- Something gets lost
  - Don't want to lose what is important
- What is important?
  - Essence! (not details)
- Data provides details, essence is hidden inside

Articulated figure representation

 Sets of rigid pieces

 What are the rigid pieces?

Rigid Body

- A set of points that undergoes a rigid transformation

- Describe configuration by the rigid transformation

  \[ P' = f(q, P) \]

  Transformed points  
  Transform parameters  
  "Rest" state points

Rigid Transforms

- Mapping \( f : \mathbb{R}^n \rightarrow \mathbb{R}^n \)

- Defined by properties:
  - Has a zero
  - Preserves distances
  - Preserves handedness

- Is a linear mapping

Parameterizing Rotations

- Goal: encode rotations in a vector
  - \( \mathbb{R}^n \rightarrow \) "set of rotations"
- Give "names" to members of the set of possible rotations

- Many ways to do this, all flawed
  - No perfect method
  - Use the best one for the job

Goals for Parameterization

- Compact
  - (as few variables as possible)
- Complete
  - Every rotation can be represented
- 1-to-1
  - Every rotation has one value
  - Every value has one rotation
- Singularity free
  - "close" rotations are "close" in value
Parameterization 1: The Rotation Matrix

- We know the rotation is a linear function (e.g., Matrix)
- Use the matrix as the parameterization!
- Any rotation is represented by 1 matrix
- Must preserve distance
- Must preserve handedness
- Must preserve angles
- Positive, Orthonormal matrices

Problems with Matrix as Parameterization

- Not compact
  - 9 numbers (but 3 d.o.f.)
- Not all matrices are orthonormal
  - Change 1 number, it’s not orthonormal
    - Sensitive to numerical issues
  - Can’t tell quickly
    - Given a matrix, determine if orthonormal
  - Can’t project quickly
    - Given a matrix, find the “closest” orthonormal one

More problems...

- Given two rotation matrices, $M_1$ and $M_2$
  - Can you measure how different they are?
  - Can you interpolate them?
    - (e.g., find halfway)
- Fortunately, they are closed under multiplication
  - Modulo numerical issues

Problems are worse in 3D

- 3x3 matrices – 9 parameters
- No intuitive meaning to parameters
- Only supports a few operations
  - Apply to point
  - Multiply (compose) – beware drift
- Use rotation matrices to apply rotations
- Use other methods to parameterize and manipulate them

Parameterizations of Rotations

- Rotation Matrices
- Euler Angles
- Axis Angle formulation
- Unit Quaternions
- Exponential Co-Ordinates
  - Local linearizations

Two theorems of Euler

- Any rotation can be represented by a single rotation about an arbitrary axis
  - Axis / Angle Representation
- Any rotation can be represented by a sequence of 3 rotations around fixed axes
  - Euler Angles
**Axis / Angle**
- Not compact (4 numbers, not 3)
- Each rotation represented by many groups of 4 numbers
- Can’t compute with
  - Hard to compose
  - Hard to compare
  - Hard to interpolate
- Inefficient

**Euler Angles**
- Pick 3 axes (XYZ, ZXZ, ZXY, …)
- Compact
- Any 3 numbers is a rotation
- Every rotation has many values
- Singularities
- Not metric (close rotations->different numbers)
- Interpolations can be weird
- OK when 1 axis at a time
- False sense of security that can do math

**What else?**
- Other parameterizations more recent in Computer Graphics
  - Quaternions (introduced 1985, popular recently)
  - Exponential co-ordinates (introduced 1995, popular recently)
- Both method are old
  - Graphics just took a while to discover them

**Easy case: 2D**
- Rotations in 2D aren’t too hard
  - Examine them to see what happens in 3D (where it is much harder)
  - Basic problems still occur

**2D Rotations**
- Consider 1 point in 2D, center is the origin
- A rotation maps the point somewhere on the circle

**Each rotation is a point on the circle**
- Not exactly...
  - There’s the handedness thing
So how to name points on a circle?

- No good mapping to the real line
  - Real line goes on forever
  - Circle wraps around
- Same problems as rotation!
- Note: circle (in 2D) is a 1D set

Method 1: use a 2D coord

- Name point by x, y on circle
  - Could be a complex number

Extra coordinates

- Good points
  - Every point can be named
  - Every point has a unique name
  - Close points have similar names
    - (no singularity)
- Bad points
  - Not all points are on the circle
  - Can’t manipulate vectors
    - How to add? Takes you off the circle

Quaternions

- Extension of this idea to 3D rotation
  - 4 dimensional complex number
  - Real part, 3 imaginary axes (vector)
- Represent 3D rotation as a point on the unit 4-sphere
- Need to stay on sphere
  - E.g. UNIT Quaternions

Good points about Quaternions

- Multiplication is defined
  - Easy composition
- Interpolation is defined
  - Special methods worked out
- Relatively compact
- Singularity free
- “Nearly” 1-to-1

Bad point about Quaternions

- Can’t add
- Can’t take linear combinations
- Can’t average
- Can’t linear filter
- Distance metric is unclear
A “hack”
- It's easy to get “back on the circle” via reprojection
- Pretend points are in 2D, then project back
- Example: averaging

Warning on the hack...
- Gets the right answer for averaging
- Not for other linear combinations
- Works well when difference is small
- Small angle approximation
- Fails when opposite
- Useful since we can renormalize if computations have problems

Method 2: distance
- How far around circle?
  - (unit radius makes things easier)
  - Basically an angle

Method 3: velocity
- Suppose the particle starts “at zero” and has a constant velocity $\omega$
- Where does it end up at the end of a unit of time?

Method 4: velocity
- Velocity is tangent to circle – therefore it is initially upwards
- If circle is in the complex plane, the velocity is purely imaginary

Velocity (cntd)
- Velocity as “up” only works if we start at origin
- So always measure from origin and shift the start around
Initial velocity is good...
- It’s linear!
  - Linearizes the circle around the origin
- Can operate on it
  - Add
  - Scalar multiply
- Not perfect...
  - Many different ways to get to any place

Local linearization
- Logarithmic map / Exponential map
- Good for describing the differences between orientations
- Good basis for performing linear operations on orientations
- Filtering
- Averaging

In general...
- Use quaternions to represent orientation
- Use tangent space (log map) to perform linearized computations
  - Hack often works, almost as well
  - Don’t tell anyone I said that!

Back to our real question...
Abstraction of Human Motion
- Humans too complex
  - Need tractable models
  - Some number of connected, rigid pieces
  - (usually)

Representations of Motion
- Angle vs. positional data
- Global vs. relative
- Hierarchical vs. non-hierarchical
- Skeletal vs. Non-Skeletal

Representations of 2 bodies
- Independent: Absolute Orientation
  - Absolute Position
- Hierarchical: Relative Orientation
  - Fixed Relative Position
  - Position Only
Good Points of Hierarchical Skeletons
- Enforce key constraints
  - Connected segments
  - Rigid limbs
- Fewer DoF’s
  - Only store angles between segments
  - Easy for skinning
  - Local coordinate systems defined

Bad Points of Hierarchical Skeletons
- Need 3D rotations
- Coupled parameters
- End effector controls require IK
- Forces rigidity
- Problems with reference
  - Different ways of defining things

Complexities of Skeletal Representation
- Can’t just measure
  (even x-rays wouldn’t help, no real "joints")
- Abstraction
- Don’t know parameters
- Need to know skeleton and relation of skeleton to markers