**CMSC 498M: Chapter 7c**

**Modeling and Animation**

**Reading:** (Not covered in our readings).
- Some material from Lecture Notes for by Steve Rotenberg at UCSD.

**Overview:**
- Animation basics and skeletal animation
- Skin and Bones
- Binding
- Motion blending

---

**Kinematics & Dynamics**

**Forward Kinematics:**
- Given a hierarchical scene graph and joint angles for an articulated structure, find the locations of all end points.

**Inverse Kinematics:**
- Given the positions of the root, end points, and lengths, find a self-consistent set of joint angles.
- Much harder than forward kinematics, multiple solutions possible.

**Dynamics:**
- Given initial positions and forces, compute the final positions.

**Animation packages** (e.g., Maya, Lightwave, Kinetix):
- Support for kinematics and dynamics.
- Limited support for inverse kinematics.
Skeletal Animation

**Skeleton:**
- Objects represented as hierarchy of bones and joints.
- Joints and their types (e.g., hinge, universal, ball and socket).
- Joint degrees of freedom (rotation axes).
- Limits on movement may be given.

**Skin:**
- Provides smoothing, filling, and blending.

**Binding:**
- Correspondence between skeleton and skin geometries.

**Discussion:**
- Space efficient, since only skeleton joints need to be animated.
- Hierarchical structure of skeleton works well with scene graphs.
- Challenge is in making skin movement appear natural.

---

Character Animation

**Skeleton**

Skin

Binding

Motion Blending
Skeletal Animation

**Skeleton:**
- A pose-able framework of joints arranged in a tree structure.
- An invisible armature upon which to attach the skin and other geometric data of the character.

**Joint:**
- A joint allows relative movement within the skeleton.
- Each joint is represented by a transformation (4x4 matrix).
- Joints can be rotational, translational, or some non-realistic types as well.

**Bone:**
- Bones connect joints. Skin is placed relative to the bones.
- In many systems joints are the main entities, bones are implicit.
- For example, one might refer to the shoulder joint or upper arm bone (humerus) and mean the same thing.

---

Character Skeleton
Character Skeleton: Forward Kinematics

Position of left arm:
\[ M_{\text{elbow-arm}_L} \cdot M_{\text{arm}_L-\text{torso}} \cdot M_{\text{torso-world}} = M_{\text{elbow-world}} = M_{\text{elbow-arm}_L} \cdot M_{\text{arm}_L-\text{torso}} \cdot M_{\text{torso-world}} \]

Character Skeleton: Animation

Moving the Joints:

Mapping Angles to Positions: Each matrix is parameterized by its rotational degrees of freedom, each constrained to lie within some given angular limits:
- E.g., \( M_{\text{knee-Rleg}} = R_x(\phi) \), where \(-180^\circ \leq \phi \leq 0^\circ\).
- Each joint matrix also encodes a joint offset (length of the bone), thus in fact, \( M_{\text{knee-Rleg}} = T(\text{len}) \cdot R_x(\phi) \).

Animating Joints: To generate the animation each joint’s motion parameter (say the joint angle, \( \phi(t) \)) is changed from frame to frame.

Joint Angles: Changes (velocities, accelerations, etc) are specified by a higher-level animation system (e.g., keyframe, motion capture, or procedural).
Character Skeleton: Animation

**Pose:**
- A list of parameters \( \mathbf{\varphi} = (\varphi_1, \varphi_2, \ldots, \varphi_n) \) defining the various joint angles of the skeleton.

**Channel:**
- A sequence of parameters for a single joint angle \( \varphi_i(t) \).
- Often use parametric curves (Bezier, B-spline, NURBS, ...) to edit, interpolate, approximate, or compress.

**Animation:**
- An array of poses \( \varphi(t) \) or an array of channels \( (\varphi_1(t), \varphi_2(t), \ldots, \varphi_n(t)) \).
- Tradeoff memory access coherence vs. CPU computation.

Channel 2:  
\[ \varphi_1(0) \xrightarrow{\varphi_1(1)} \varphi_1(2) \]
\[ \varphi_2(0) \xrightarrow{\varphi_2(1)} \varphi_2(2) \]

Pose vectors: \( \varphi(0), \varphi(1), \varphi(2) \)

---

Computer Animation

- Key-frame animation
- Procedural Animation
- Motion capture
Keyframe Animation

Keyframe Animation:
- Main animator draws a few key frames.
- Others (artists or computers) draw intermediate frames.
- Drawing is simplified by drawing layers of the scene on translucent cels (cellular animation), superimposing, and photographing them.
- Traditional approach to animation in movies.

Pixar’s "Luxo Jr."

Interpolation Methods:

Linear Interpolation: \((\text{lerp})\) Take weighted averages.
\[
p(t) = (1-t)p_0 + tp_1, \quad \text{for } 0 \leq t \leq 1.
\]
Appropriate for points, but not for angles.

Spherical Interpolation: \((\text{slerp})\) Let \(\phi\) denote the angle between the unit vectors \(u_0\) and \(u_1\) \((\phi = \cos(u_0 \cdot u_1))\). The spherical interpolation is given by:
\[
u(t) = \frac{\sin((1-t)\phi)}{\sin \phi} u_0 + \frac{\sin(t\phi)}{\sin \phi} u_1, \quad \text{for } 0 \leq t \leq 1.
\]

Normalized Lerp: \((\text{nlerp})\) Do a lerp, then normalize to unit length. A quick-and-dirty cheat for slerp.
\[
u'(t) = (1-t)u_0 + tu_1; \quad u(t) = \frac{u'(t)}{\|u'(t)\|}, \quad \text{for } 0 \leq t \leq 1.
\]
Interpolating Key Frames

Smother Interpolation:
Why: A sequence of linear interpolations joined end-to-end can result in sudden changes in speed at joint points.
Better: Use B-Spline or Bezier curves to interpolate position.
Objectives: Local control, smooth motion, robustness.
Challenge: Maintain the right balance between interpolated position and timing (controlling velocity and acceleration). Almost an art.

Procedural Animation

Procedural Animation:
- Specify the motion in terms in purely mathematical terms (e.g. equations, procedures, solution of differential equations).
- Usually best for natural objects/scenes:
  - Smoke, fire, explosions via particle systems.
  - Physical simulations like collisions.
  - Periodical motion like waves in water, blinking eyes, walking.
- Add random noise to periodic motion.
  - Players notice identical gait/blink cycles.
- Tricky to fine tune parameters to produce natural-looking motion.

Source: Kaufman, Edmunds, and Pai (SIGGRAPH 2005)
Motion Capture

Motion Capture:
- Attach virtual reality trackers on a person or use cameras and attach markers on the person.
- Record tracker/marker motion and play back the motion.

Pros: Real-time, easy to capture complex human motions.

Cons: May not be feasible. May be hard to edit if changes needed.

Character Animation

Skeleton

Skin

Binding

Motion Blending
**Skin and Bones**

**Skin:**
- A mesh attached to each bone.
- Each vertex of this mesh is placed relative to (one or more) bones.

Source: [http://www.flipcode.com/articles/article_dx8shaders.shtml](http://www.flipcode.com/articles/article_dx8shaders.shtml)

**Problems:**
- When joints flex, vertices attached to them move rigidly.
- While this definition is simple, it is not very realistic.
- Flexing of joints cause mesh to stretch and/or self-intersect.

**Smooth Skin:**
- Want a system that flexes naturally.

Source: [http://www.flipcode.com/articles/article_dx8shaders.shtml](http://www.flipcode.com/articles/article_dx8shaders.shtml)
Rigid vs. Smooth Skin

Rigid Skin:
- Every vertex is associated with exactly one joint. It is expressed relative to the joint's local coordinate system:
  \[ v'(t) = M_{\text{joint}}(t) \cdot v \]

Smooth Skin:
- Each vertex is associated with multiple joints (usually two).
- It is expressed as a weighted average of joint coordinates:
  \[ v'(t) = w_1 M_{\text{joint1}}(t) \cdot v + w_2 M_{\text{joint2}}(t) \cdot v + \ldots + w_n M_{\text{jointn}}(t) \cdot v, \]
  where \( w_1 + w_2 + \ldots + w_n = 1 \).
- This is called geometry blending.
- Direct3D supports geometry blending. (I don’t know about OpenGL).

Smooth Skin

Problems with Smooth Skin:
- Although smooth skin produces smooth results, it is still only a heuristic.
- Flexing and twisting can result in unnatural collapsing at joints.
- More sophisticated approaches are needed to achieve better realism.

Source: Lewis, Cordner, Fong, SIGGRAPH 00
Character Animation

Skeleton

Skin

Binding

Motion Blending

Binding Skin with Bones

Layering:
- Animation is usually easiest to do with a skeleton.
- Final appearance involves additional layers: skin, muscles, clothes, ...
- Character layers are modeled in a natural pose (e.g., standing).
- This pose may not correspond to the skeleton’s zero pose ($\phi_i = 0$).

Binding: A three-step process:
1. Pose the skeleton in a natural pose, called the binding pose.
2. Establish correspondence (and weights) between skin vertices and the skeleton (several heuristics).
3. Transform skin vertices to local coordinate system of joint(s).

Establishing Correspondences:
By Proximity: Assign each skin vertex to the closest bone(s).
Manually: Create bounding volumes (spheres, ellipsoids, cubes) for each bone to enclose skin vertices that belong to it.
Character Animation

Skeleton
Skin
Binding

Motion Blending

Blending: Unique to Games

Need for Blending:
- Characters in games have a library of actions that need to be sequenced (generally seamlessly) on demand, at interactive rates.
- Examples:
  - General: Sitting, walking, running, climbing.
  - Battle: Sword fighting, punching.
  - Sports: Throwing a ball, swinging a bat, kicking.

Motion blending:
- Interpolate between:
  - ending pose of previous action and
  - starting pose of next action.

Assassin's Creed
Cross Dissolve

Cross Dissolve:
- Lerp or Slerp from one pose to the next.
- The simplest type of blend.
- The term "cross" comes from increasing one weight while decreasing the other.
- Unfortunately, does not always produce realistic results.

Challenges: How can we produce more realistic transitions?
- Ensuring phase synchronization: E.g., running to kicking.
- Adapting to changes in velocity: E.g., walking to running.
- Mixing rotations and translations: E.g., sitting to walking.

Animation State Machine

Animation State Machine:
- A finite state machine where:
  States: represent animation clips.
  Transitions: represent motion blends.

Features:
- Enables complex motion sequences:
  - Sitting to walking should have an intermediate stage of standing.
- Object-oriented way to build complex motions:
  - Encapsulate simpler motions (state machine) as a single state of a more complex motions.
- Allows greater control of transitions between selected states.
- Simplifies design of animated games:
  - Transitions triggered by user events, game AI, randomness, ...

Chapter 7, Slide 25
Copyright © David Mount and Amitabh Varshney

Chapter 7, Slide 26
Copyright © David Mount and Amitabh Varshney
Summary

Summary:
- Animation basics and skeletal animation
- Skin and Bones
- Binding
- Motion blending

What's Next?
- Procedural modeling