Character Animation and Skinning
Animation

Motion over time
Traditional Character Animation

Lead animator draws sparse key frames

Secondary artists fill in (by hand) the intermediate frames: inbetweening
Computer Character Animation

How to inbetween automatically on a 2D sprite?
Cage-Based Animation

Surround object with animation cage

Moving the cage moves interior points
Simplest Cage: Triangle

Use barycentric interpolation

Matches points’ pixels between triangles
Polygonal Cages

Must generalize barycentric coordinates to arbitrary polygons

Many ways to do this: generalized barycentric coordinates not unique
Generalized Barycentric Coordinates

Partition of unity:

\[ 1 = \sum \alpha_i \]

Reproduces the verts:

\[ \alpha_i(p_j) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \]

\[ q = \sum \alpha_i p_i \]
Polygonal Cages

Other properties:
1. Weights must be positive inside the polygon (or get leaks)
Polygonal Cages

Other properties:

1. Weights must be positive inside the polygon (or get **leaks**)
2. Weights must unique (or get **flips**)

---

**Polygonal Cages**

Other properties:

1. Weights must be positive inside the polygon (or get **leaks**)
2. Weights must unique (or get **flips**)

---
Polygonal Cages

Other properties:

1. Weights must be positive inside the polygon (or get **leaks**)
2. Weights must unique (or get **flips**)
3. Smooth
4. Easy to compute
Some Possible Schemes

- Wachspress Coordinates
- Mean-value Coordinates
- Green Coordinates
- Bounded Biharmonic Weights
- etc…
Cage-Based Animation

Extends to 3D from 2D naturally

Full control, but not intuitive
Handle-Based Animation

Pick special points (handles) on object

Moving handles moves nearby points
Character Rigs

Skeletons inside the geometry

moving bones moves surrounding geometry

the industry standard for character animation

how to build rig?
Building a Rig

Usually done by hand using Maya etc.

Expressiveness/complexity tradeoff
Rigging in Practice

https://www.youtube.com/watch?v=WxZz-yH-mKU
Building a Rig

Some automatic tools exist…

[Pinocchio]

[Mixamo]
Mixamo Demo

https://www.mixamo.com/

Automatic rigging can work well for humans/humanlike objects

• Assumess bipedal with standard placement and orientation of joints
Not so impressive for arbitrary characters…

https://www.youtube.com/watch?v=-5jWtz3rzco
Representing a Rig

Tree of bones connected by joints

bones have two endpoints
- first attached to parent
Bone Local Coordinates

Origin O
One natural direction: \textbf{tangent} axis \( \hat{t} \)
Bone Local Coordinates

Origin O

One natural direction: **tangent** axis $\hat{t}$

Two perpendicular directions: $\hat{n}, \hat{b}$

$$(x, y, z) = x\hat{t} + y\hat{n} + z\hat{b}$$
Bone Local Coordinates

Origin O

One natural direction: \textbf{tangent} axis $\hat{t}$

Two perpendicular directions: $\hat{n}, \hat{b}$

$$(x, y, z) = xt\hat{t} + yn\hat{n} + zb\hat{b}$$

second endpoint: $(L, 0, 0)$
Bone Local Coordinates

Child bone can be expressed in terms of parent coordinate system

\[ O_2 = (L, 0, 0) = T_2 O \]
\[ \hat{t}_2 = R_2 (1, 0, 0) = R_2 \hat{t} \]
Bone to World Coordinates

In local coordinates:

\[ q = (x, y, z) = O_3 + x\hat{t}_3 + y\hat{n}_3 + z\hat{b}_3 \]
Bone to World Coordinates

In local coordinates:

\[ q = (x, y, z) = O_3 + x\hat{t}_3 + y\hat{n}_3 + z\hat{b}_3 \]

In world coordinates:

\[ q = T_1 R_1 T_2 R_2 T_3 R_3 \begin{bmatrix} x \\ y \\ z \end{bmatrix} = M_3 \begin{bmatrix} x \\ y \\ z \end{bmatrix} \]
What About the Base?

(0, 0, 0)
What About the Base?

$$(0, 0, 0)$$

write origin & axes in **world coordinates**, then

$$T_1 = T_{O_1}$$

$$R_1 = \begin{bmatrix} \hat{t}_1 & \hat{n}_1 & \hat{b}_1 \end{bmatrix}$$
Forward Kinematics

\[ q = T_1 R_1 T_2 R_2 T_3 R_3 \begin{bmatrix} x \\ y \\ z \end{bmatrix} \]

changing \( R_1 \) also changes child coordinate systems
Additional Reading

Skinning

Moving bones moves the character

Closer bones have more influence
Nearest-Bone Skinning

Given: **undeformed** (rest) skeleton and **deformed** skeleton
Coordinate Systems

world

undeformed bone

dehformed bone
Coordinate Systems

\[ M_3 = T_1 R_1 T_2 R_2 T_3 R_3 \]

world

undeformed bone

deformed bone
Coordinate Systems

\[ M_3 = T_1 R_1 T_2 R_2 T_3 R_3 \]

\[ \hat{M}_3 = \hat{T}_1 \hat{R}_1 \hat{T}_2 \hat{R}_2 \hat{T}_3 \hat{R}_3 \]

undeformed bone

deformed bone
Coordinate Systems

\[ M_3 = T_1 R_1 T_2 R_2 T_3 R_3 \]

\[ \dot{M}_3 = \dot{T}_1 \dot{R}_1 \dot{T}_2 \dot{R}_2 \dot{T}_3 \dot{R}_3 \]

**undeformed bone**

**deformed bone**
Coordinate Systems

Key (and confusing) point:

- $M_3$ maps from undeformed local to world coords (doesn’t move point)
- **Identity** maps undeformed to deformed bone coords (and does move point)
Nearest-Bone Skinning

Undeformed to deformed skin position (world coordinates):

$$\tilde{q} = \tilde{M}_3 M_3^{-1} q$$
Nearest-Bone Skinning

Undeformed to deformed skin position (world coordinates):

\[ \tilde{q} = \tilde{M}_3 M_3^{-1} q \]

changes during animation
Problems with Nearest-Bone

Which bone does point belong to?

\[ \tilde{M}_1 M_1^{-1} q \]

\[ \tilde{M}_2 M_2^{-1} q \]
Problems with Nearest-Bone

Which bone does point belong to?
One solution: average \[
\left[ \frac{1}{2} \tilde{M}_1 M_1^{-1} + \frac{1}{2} \tilde{M}_2 M_2^{-1} \right] q
\]
Linear-Blend Skinning

Each vertex feels **weighted average** of each bone’s transformations

\[ \tilde{q}_i = \sum_{\text{bones } j} w_{ij} \tilde{M}_j M_j^{-1} q_i \]

Nearby bones have higher weight
Linear-Blend Skinning

How to determine skinning weights \( w \)?
Linear-Blend Skinning

How to determine skinning weights $w$?

- Use only nearest bone
- Spatially blend the weights
- In practice: paint weights by hand
Painting Weights

https://www.youtube.com/watch?v=dD5adx9SdH0
The “Arm Twist” Problem

(Why does this happen?)
Blending Transformations

Each individual bone undergoes a **rigid transformation**

- Combination rotation and translation
- Linear blend of rigid motions not rigid
Blending Transformations

Translations alone: trivial to blend

$$\{t_1, \ldots, t_n\} \mapsto \sum_i \alpha_i t_i$$

Rotations alone: blend using SLERP

- use quaternions
- do not use Euler angles!
Separate Transforms

Idea: separately blend translation and rotation components of rigid motion

\[ \text{blend}(T_1, T_2, t) \]
Separate Transforms: Problem

where is the child bone half way in between the motion?

(where it the origin?)
Separate Transforms: Problem

where is the child bone half way in between the motion?

(Where is the origin?)

blend($T_1, T_2, 1/2$)
Separate Transforms: Problem

where is the child bone half way in between the motion?

(where it the origin?)

blend\( (T_1, T_2, 1/2) \)
Separate Transforms: Problem

Blended transformation not coordinate-independent

- different origin positions in bone hierarchy result in different blends
- unify translation and rotation into single state
- blend centers of rotation
Dual Quaternion Skinning

• No more arm twisting issues
• Less deformation
• The industry standard (used in Maya, etc)
Dual Quaternion Skinning

- Expresses a rotation (quaternion) and translation
- Blended dual quaternions represent a rotation and translation as quaternions

\[
\dot{q} = q_r + q_d \epsilon
\]

where

\[
q_r = r
\]

\[
q_d = \frac{1}{2} tr
\]

\[
\dot{q} = \frac{\sum_{i=1}^{n} w_i \dot{q}_i}{\| \sum_{i=1}^{n} w_i \dot{q}_i \|}
\]
Dual Quaternion Skinning

Prevents loss of volume during rigid motion

Other Skinning Challenges

Volume Conservation
Other 3D Animation Techniques

Anatomy-based animation
• simulate the tendons and muscles

Most correct motion
Restricted to “real” animals
Slow
Other 3D Animation Techniques

Faces are hard to rig: many muscles

instead, precompute a small set of basis deformations

blend between them (blendshapes)

knobs not always intuitive
Other 3D Animation Techniques

SCAPE: data-driven statistical model of human body shapes

- rig a “mean” template human once by hand for an arbitrary human,
- transfer the rig by fitting to a SCAPE model
- works well only for (nearly) naked humans
Simplifying Keyframing

Simulation (“inverse kinematics”)

https://www.youtube.com/watch?v=dEvE0oCJpUA
Motion Capture

https://www.youtube.com/watch?v=CmrXK4fNOEo
Machine Learning and Motion Synthesis

- Machine learning can allow for good-looking animations
  - https://www.youtube.com/watch?v=urf-AAIwNYk
- Can lower cost of artistic pipeline
- Downside is that learned function is mostly unmodifiable/incomprehensible
  - Difficult to correct in pathological cases
- Also realism is not necessarily as appealing as artistic interpretation
Animation Recap

Most common pipeline:

• build a 3D model of the character
• **rig** the 3D model (build a skeleton inside)
• **skin** the model (determine bone-skin weights)
• animate the bones by specifying **keyframes**; skin moves with them
Animation Recap

Most common pipeline:
• model, rig, skin, animate

Automatic approaches exist for each step
• not great, but getting better

Still a grand challenge: use Kinect to build a fully rigged and skinned digital digital avatar