

1

CS-184: Computer Graphics

Lecture #16: Forward and Inverse Kinematics

Prof. James O'Brien
University of California, Berkeley

©2016 UC, B.O.

2

Today

- Forward kinematics
- Inverse kinematics
 - Pin joints
 - Ball joints
 - Prismatic joints

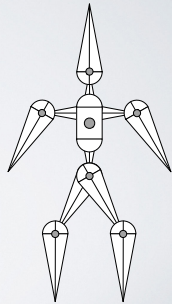
2

Forward Kinematics

3

• Articulated skeleton

- Topology (what's connected to what)
- Geometric relations from joints
- Independent of display geometry
- Tree structure
 - Loop joints break "tree-ness"

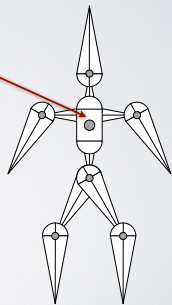


Forward Kinematics

4

• Root body

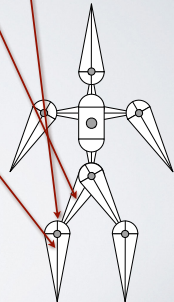
- Position set by "global" transformation
- Root joint
 - Position
 - Rotation
- Other bodies relative to root
- *Inboard* toward the root
- *Outboard* away from root



Forward Kinematics

5

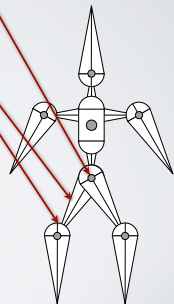
- A joint
- Joint's inboard body
- Joint's outboard body



Forward Kinematics

6

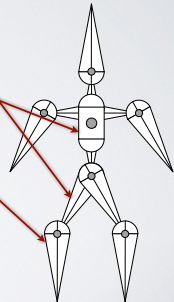
- A body
- Body's inboard joint
- Body's outboard joint
- May have several outboard joints



Forward Kinematics

7

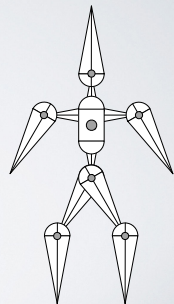
- A body
 - Body's inboard joint
 - Body's outboard joint
 - May have several outboard joints
- Body's parent
- Body's child
 - May have several children



Forward Kinematics

8

- Interior joints
 - Typically not 6 DOF joints
 - Pin - rotate about one axis
 - Ball - arbitrary rotation
 - Prism - translation along one axis

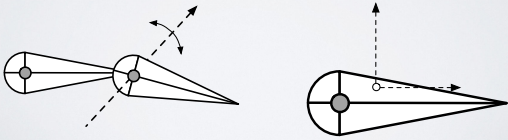


Forward Kinematics

9

• Pin Joints

- Translate inboard joint to local origin
- Apply rotation about axis
- Translate origin to location of joint on outboard body

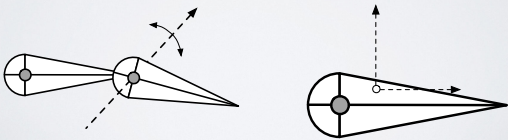


Forward Kinematics

10

• Ball Joints

- Translate inboard joint to local origin
- Apply rotation about arbitrary axis
- Translate origin to location of joint on outboard body

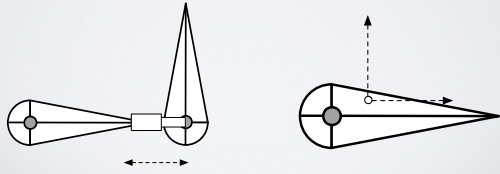


Forward Kinematics

11

• Prismatic Joints

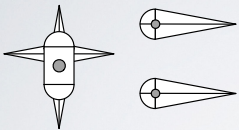
- Translate inboard joint to local origin
- Translate along axis
- Translate origin to location of joint on outboard body



Forward Kinematics

12

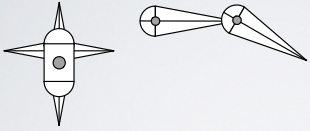
• Composite transformations up the hierarchy



Forward Kinematics

13

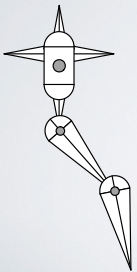
- Composite transformations up the hierarchy



Forward Kinematics

14

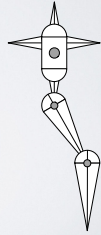
- Composite transformations up the hierarchy



Forward Kinematics

15

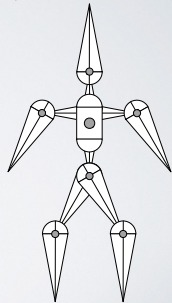
- Composite transformations up the hierarchy



Forward Kinematics

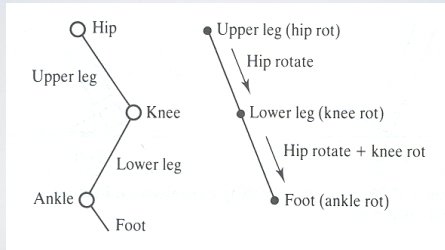
16

- Composite transformations up the hierarchy



Example: Walk Cycle

17

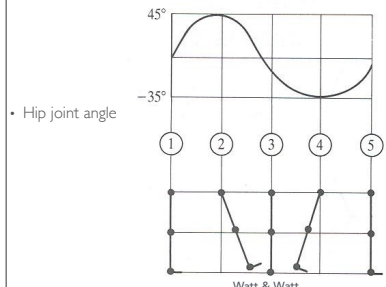


Watt & Watt

Slide credit: Tom Funkhouser

Example: Walk Cycle

18

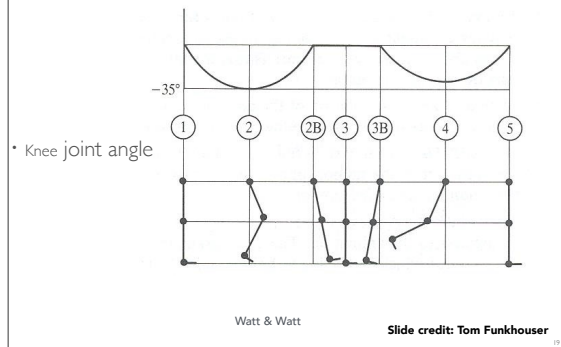


Watt & Watt

Slide credit: Tom Funkhouser

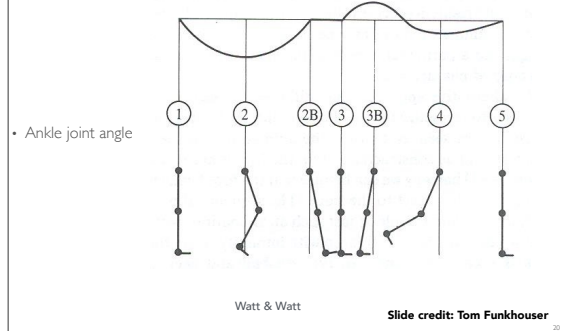
Example: Walk Cycle

19



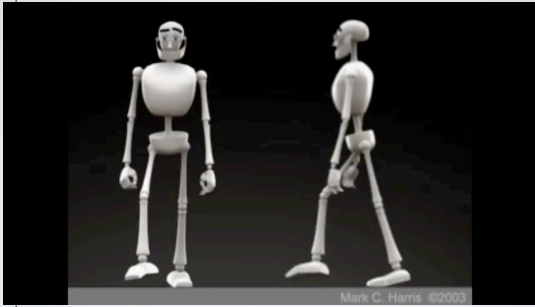
Example: Walk Cycle

20



Example Walk Cycle

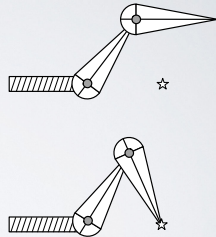
21



Inverse Kinematics

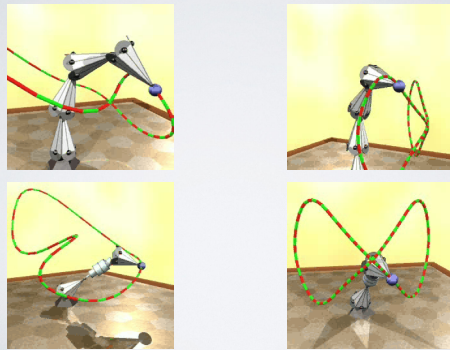
22

- Given
 - Root transformation
 - Initial configuration
 - Desired end point location
- Find
 - Interior parameter settings



Inverse Kinematics

23



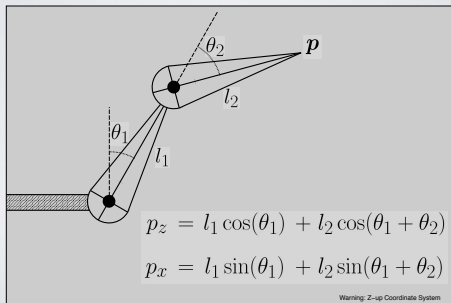
Egon Pasztor

23

Inverse Kinematics

24

• A simple two segment arm in 2D



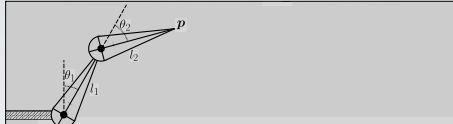
Warning: Z-up Coordinate System

24

Inverse Kinematics

25

- Direct IK: solve for the parameters



$$\theta_2 = \cos^{-1} \left(\frac{p_z^2 + p_x^2 - l_1^2 - l_2^2}{2l_1 l_2} \right)$$

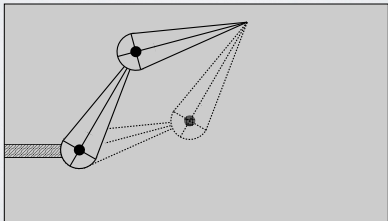
$$\theta_1 = \frac{-p_z l_2 \sin(\theta_2) + p_x (l_1 + l_2 \cos(\theta_2))}{p_x l_2 \sin(\theta_2) + p_z (l_1 + l_2 \cos(\theta_2))}$$

25

Inverse Kinematics

26

- Why is the problem hard?
 - Multiple solutions separated in configuration space

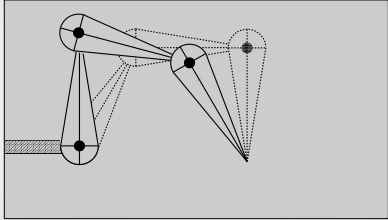


26

Inverse Kinematics

27

- Why is the problem hard?
 - Multiple solutions connected in configuration space

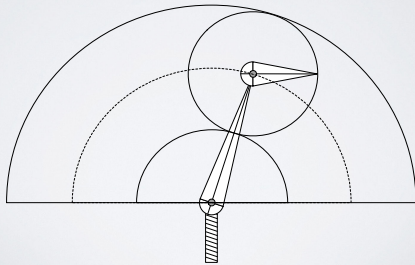


27

Inverse Kinematics

28

- Why is the problem hard?
 - Solutions may not always exist



28

Inverse Kinematics

29

• Numerical Solution

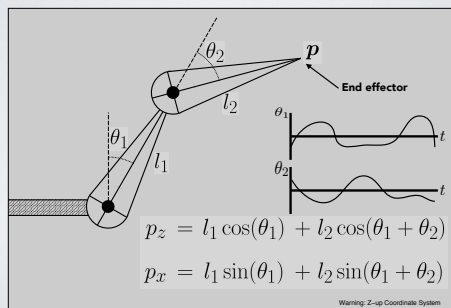
- Start in some initial configuration
- Define an error metric (e.g. goal pos - current pos)
- Compute Jacobian of error w.r.t. inputs
- Apply Newton's method (or other procedure)
- Iterate...

29

Inverse Kinematics

30

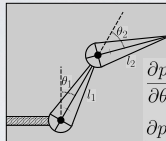
• Recall simple two segment arm:



Inverse Kinematics

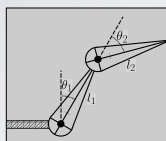
31

- We can write of the derivatives


$$\frac{\partial p_z}{\partial \theta_1} = -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2)$$
$$\frac{\partial p_x}{\partial \theta_1} = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2)$$
$$\frac{\partial p_z}{\partial \theta_2} = -l_2 \sin(\theta_1 + \theta_2)$$
$$\frac{\partial p_x}{\partial \theta_2} = l_2 \cos(\theta_1 + \theta_2)$$

Inverse Kinematics

32



Direction in Config. Space

$$\theta_1 = c_1 \theta_*$$
$$\theta_2 = c_2 \theta_*$$
$$\frac{\partial p_z}{\partial \theta_*} = c_1 \frac{\partial p_z}{\partial \theta_1} + c_2 \frac{\partial p_z}{\partial \theta_2}$$

Inverse Kinematics

33

The Jacobian (of p w.r.t. θ)

$$J_{ij} = \frac{\partial p_i}{\partial \theta_j}$$

Example for two segment arm

$$J = \begin{bmatrix} \frac{\partial p_z}{\partial \theta_1} & \frac{\partial p_z}{\partial \theta_2} \\ \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} \end{bmatrix}$$

33

Inverse Kinematics

34

The Jacobian (of p w.r.t. θ)

$$J = \begin{bmatrix} \frac{\partial p_z}{\partial \theta_1} & \frac{\partial p_z}{\partial \theta_2} \\ \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} \end{bmatrix}$$

$$\frac{\partial p}{\partial \theta_*} = J \cdot \begin{bmatrix} \frac{\partial \theta_1}{\partial \theta_*} \\ \frac{\partial \theta_2}{\partial \theta_*} \end{bmatrix} = J \cdot \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

34

Inverse Kinematics

35

Solving for c_1 and c_2

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \quad d\mathbf{p} = \begin{bmatrix} dp_z \\ dp_x \end{bmatrix}$$

$$d\mathbf{p} = \mathbf{J} \cdot \mathbf{c}$$

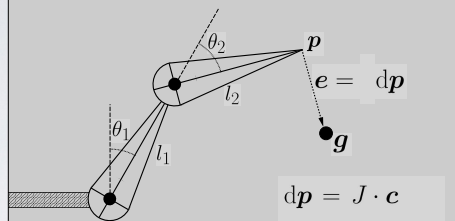
$$\mathbf{c} = \mathbf{J}^{-1} \cdot d\mathbf{p}$$

35

Inverse Kinematics

36

Solving for c_1 and c_2



Is the Jacobian invertible?

36

Inverse Kinematics

37

• Problems

- Jacobian may (will!) not always be invertible
 - Use pseudo inverse (SVD)
 - Robust iterative method
- Jacobian is not constant

- Nonlinear optimization, but problem is (mostly) well behaved

$$J = \begin{bmatrix} \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} \\ \frac{\partial p_z}{\partial \theta_1} & \frac{\partial p_z}{\partial \theta_2} \end{bmatrix} = J(\theta)$$

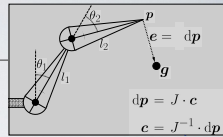
37

Inverse Kinematics

38

Jacobian is not always invertible

- Use pseudo inverse (SVD)



Computing a linear approximation

$$J = \begin{bmatrix} \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} \\ \frac{\partial p_z}{\partial \theta_1} & \frac{\partial p_z}{\partial \theta_2} \end{bmatrix} = J(\theta)$$

- End effector only locally moves linearly
- So iterate (choosing proper step size) and update Jacobian after each step
- Choosing step size requires line search at each step
 - Choose some step size (say 5 degrees) and compute how to update joint parameters
 - Calculate distance of end effector from goal
 - If distance decreased take step
 - If distance did not decrease set parameters to be half the current change and try again

38

Inverse Kinematics

39

- More complex systems
 - More complex joints (prism and ball)
 - More links
 - Other criteria (COM or height)
 - Hard constraints (joint limits)
 - Multiple criteria and multiple chains

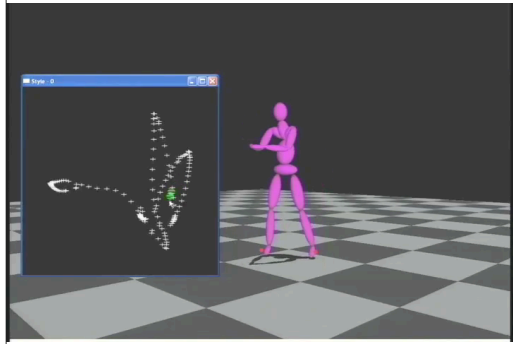
Inverse Kinematics

40

- Some issues
 - How to pick from multiple solutions?
 - Robustness when no solutions
 - Contradictory solutions
 - Smooth interpolation
 - Interpolation aware of constraints
- Numerical evaluation of Jacobian

Style-Based IK

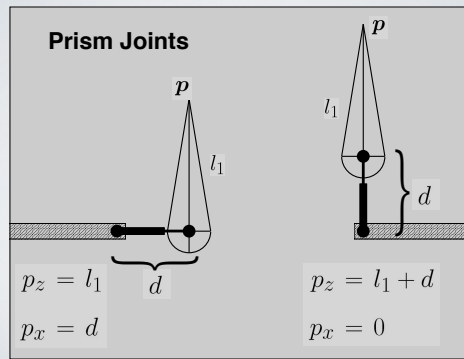
41



Grochow et al., Style Based Inverse Kinematics

Inverse Kinematics

42

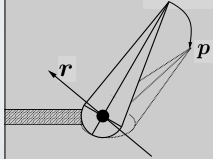


Inverse Kinematics

43

Ball Joints

$$\begin{aligned} \mathbf{p} &= \hat{\mathbf{r}}(\hat{\mathbf{r}} \cdot \mathbf{x}) \\ &+ \sin(\|\mathbf{r}\|)(\hat{\mathbf{r}} \times \mathbf{x}) \\ &- \cos(\|\mathbf{r}\|)(\hat{\mathbf{r}} \times (\hat{\mathbf{r}} \times \mathbf{x})) \end{aligned}$$



Inverse Kinematics

44

Ball Joints (moving axis)

$$d\mathbf{p} = [d\mathbf{r}] \cdot e^{[\mathbf{r}]} \cdot \mathbf{x} = [d\mathbf{r}] \cdot \mathbf{p} = -[\mathbf{p}] \cdot d\mathbf{r}$$

That is the Jacobian for this joint

$$[\mathbf{r}] = \begin{bmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix}$$

$$[\mathbf{r}] \cdot \mathbf{x} = \mathbf{r} \times \mathbf{x}$$

Inverse Kinematics

45

Ball Joints (fixed axis)

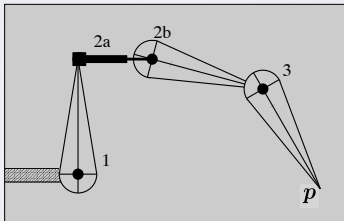
$$d\mathbf{p} = (d\theta)\hat{\mathbf{r}} \cdot \mathbf{x} = -[\mathbf{x}] \cdot \hat{\mathbf{r}} d\theta$$

That is the Jacobian for this joint

Inverse Kinematics

46

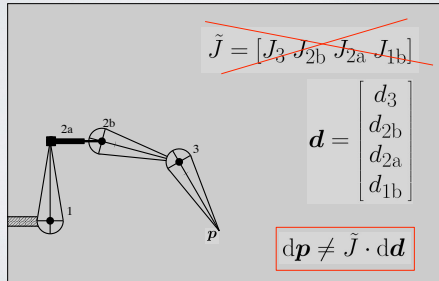
- Many links / joints
- Need a generic method for building Jacobian



Inverse Kinematics

47

- Can't just concatenate individual matrices



Inverse Kinematics

48

Transformation from body to world

$$X_{0 \leftarrow i} = \prod_{j=1}^i X_{(j-1) \leftarrow j} = X_{0 \leftarrow 1} \cdot X_{1 \leftarrow 2} \cdots$$

Rotation from body to world

$$R_{0 \leftarrow i} = \prod_{j=1}^i R_{(j-1) \leftarrow j} = R_{0 \leftarrow 1} \cdot R_{1 \leftarrow 2} \cdots$$

Inverse Kinematics

49

Need to transform Jacobians to common coordinate system (WORLD)

$$J_{i,WORLD} = R_{0 \leftarrow (i-1)} \cdot J_i$$

Inverse Kinematics

50

$$J = \begin{bmatrix} R_{0 \leftarrow 2b} \cdot J_3(\theta_3, \mathbf{p}_3) \\ R_{0 \leftarrow 2a} \cdot J_{2b}(\theta_{2b}, X_{2b \leftarrow 3} \cdot \mathbf{p}_3) \\ R_{0 \leftarrow 1} \cdot J_{2a}(\theta_{2a}, X_{2a \leftarrow 3} \cdot \mathbf{p}_3) \\ J_1(\theta_1, X_{1 \leftarrow 3} \cdot \mathbf{p}_3) \end{bmatrix}^T$$

Note: Each row in the above should be transposed....

$$\mathbf{d} = \begin{bmatrix} d_3 \\ d_{2b} \\ d_{2a} \\ d_{1b} \end{bmatrix}$$
$$d\mathbf{p} = J \cdot d\mathbf{d}$$

Rigging

51

- Rigging is a set of higher level controls on a character that allow more rapid & intuitive modification of pose, deformations, expression, etc.

- Important

- Like strings on a puppet
- Captures all meaningful character changes
- Varies from character to character



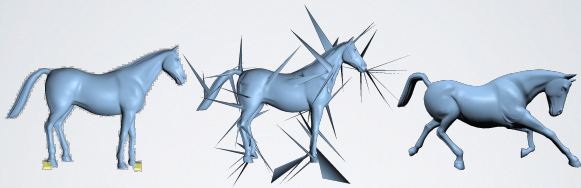
- Expensive to create

- Manual effort
- Requires both artistic and technical training

From Ren Ng

Rigging

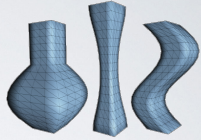
52



From Ren Ng

Types of Rigging

- Procedural Rigging
- Skeletal Rigging
- Anatomical Rigging



Al Barr. Global and Local Deformations of Solid Primitives. SIGGRAPH 1984.



Skeleton

Skinning on top



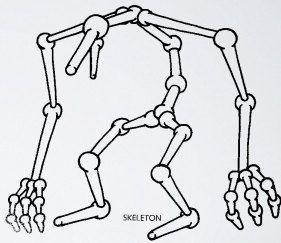
Anatomy-Based Modeling of the Human Musculature. Scheepers et al. SIGGRAPH 1997.

From Ren Ng

53

Skeletal Rigging

- Parameterize character deformation with a skeleton.
- Approximate actual skeleton of the character.



SKELETON

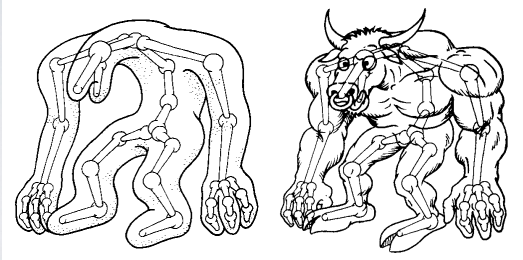
From Ren Ng

54

Skeletal Rigging

55

- Then add skin on top.

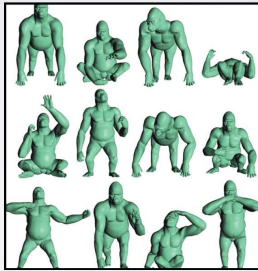


From Ren Ng

Posing

56

- Use the rigging controls to put the character into a given pose.

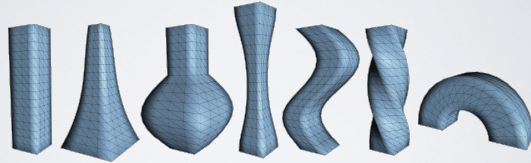


From Ren Ng

Non-linear Deformation

57

- Barr's "global and local deformations."
- Non-linear deformations for bends, twists, tapering, bulges, etc.



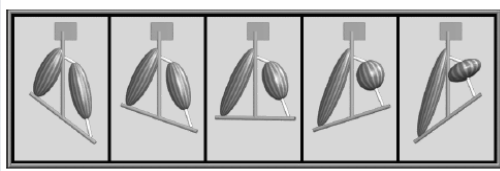
Al Barr: Global and Local Deformations of Solid Primitives SIGGRAPH 1984.

From Ren Ng

Anatomical Models

58

- Muscles are attached to bones, sometimes with tendons as well
- The muscles contract in a volume preserving way, thus getting wider as they get shorter

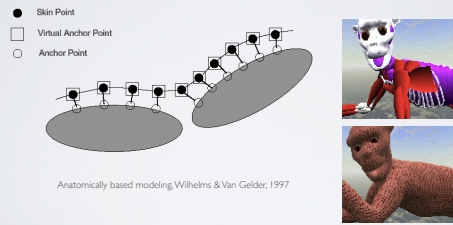


From Ren Ng

Anatomical Models

59

• Skin can be attached to the muscles with springs/dampers and physically simulated with collisions against bone & muscle

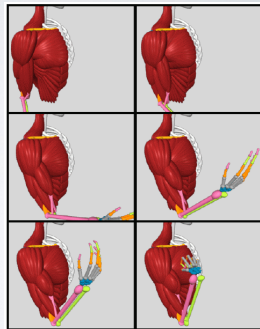
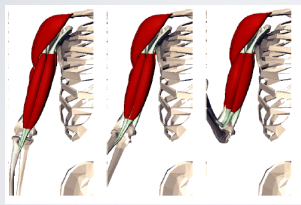


From Ren Ng

Anatomical Models

60

Complex musculature built up from lots of simple primitives.



From Ren Ng

Rigging Example

61



Provided by Matthew Lallier via Keenan Crane

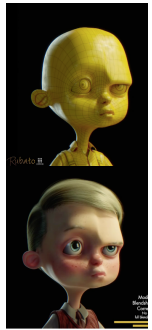
Ariane Rig

From Ren Ng

Blend Shapes

62

- Instead of skeleton, interpolate directly between surfaces
- E.g., model a collection of facial expressions:
- Simplest scheme: take linear combination of vertex positions
- Spline used to control choice of weights over time

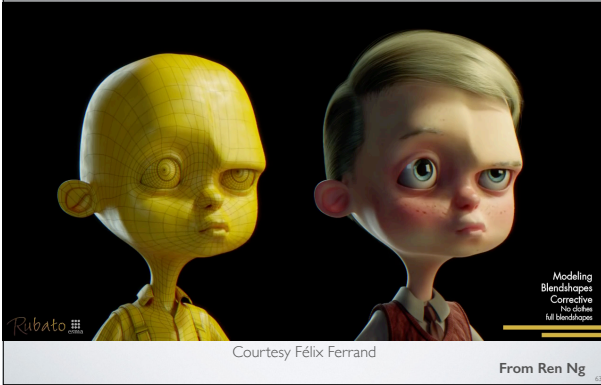


Courtesy Felix Ferard

From Ren Ng

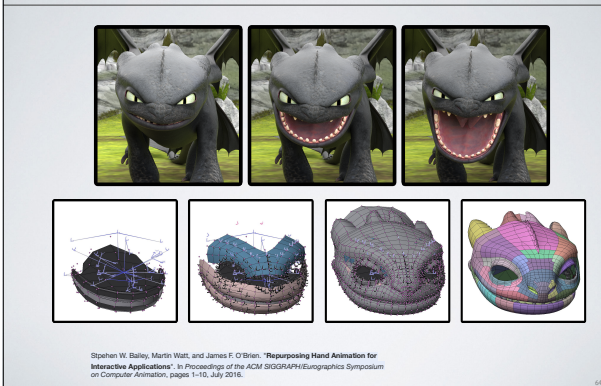
Blend Shapes

63



Rib-Based Facial Animation

64



Rib-Based Facial Animation

65

