Character Animation

COS 426, Spring 2021

Felix Heide
Princeton University

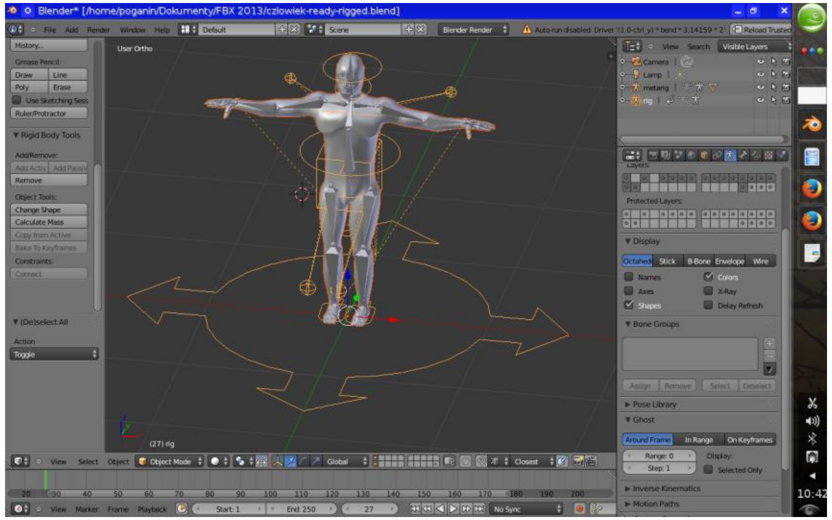
Computer Animation

- Challenge is balancing between …
  - Animator control
  - Physical realism

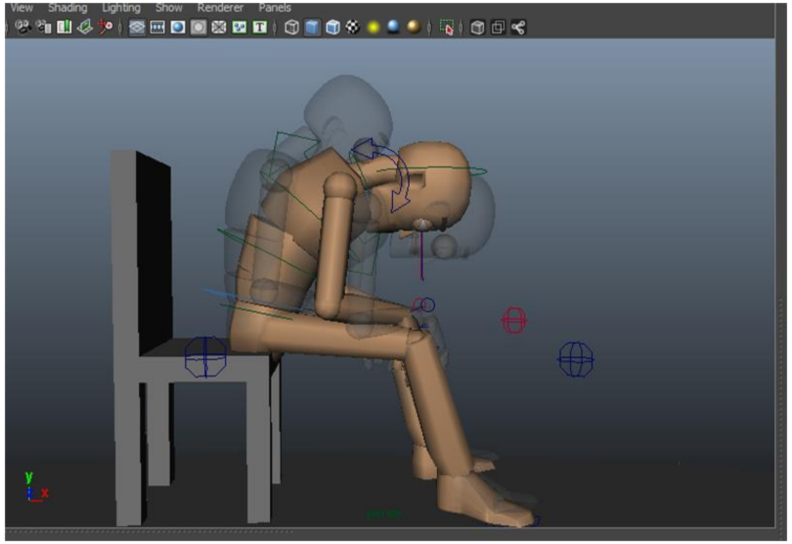
Computer Animation

- Manipulation
  - Posing
  - Configuration control

- Interpolation
  - Keyframes
  - In-betweens



https://blenderartists.org/

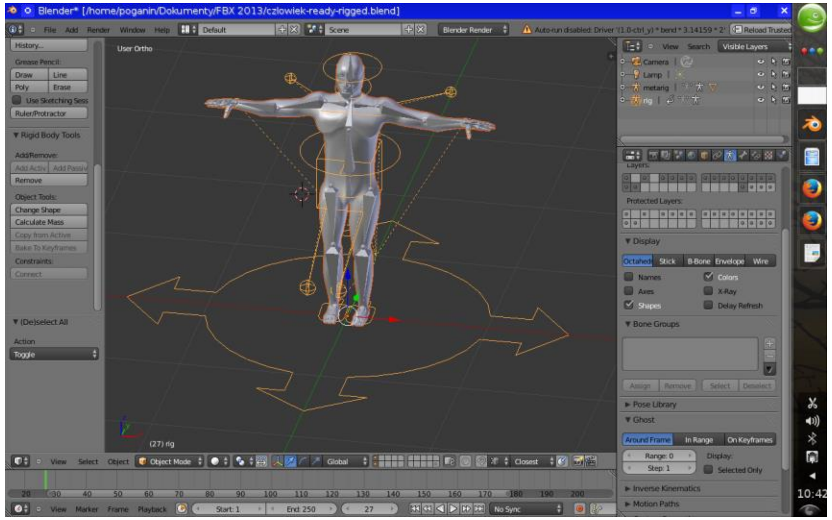


focus.gscept.com

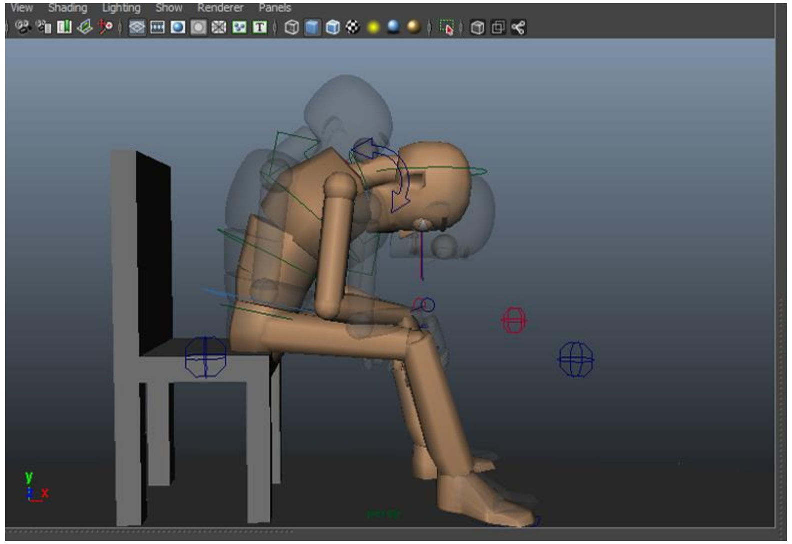
Character Animation Methods

- Modeling (manipulation)
  - Deformation
  - Blendshapes
  - Skeletons

- Interpolation
  - Key-framing
  - Kinematics
  - Motion Capture



https://blenderartists.org/

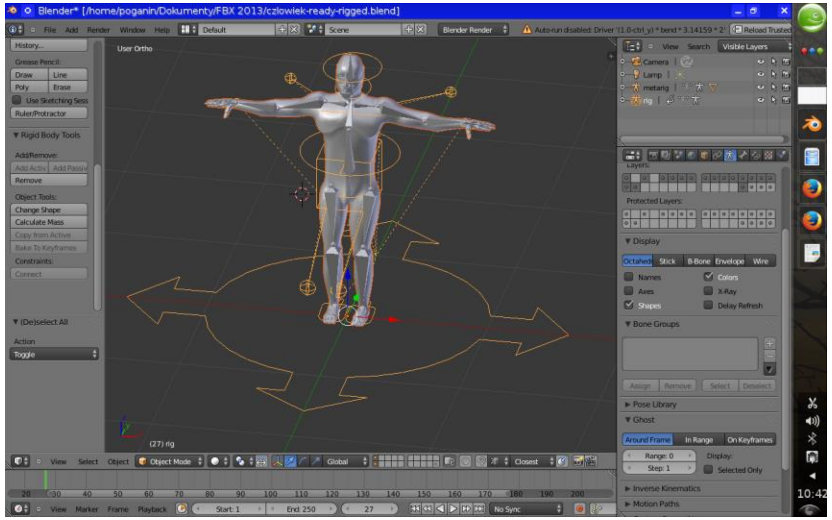


focus.gscept.com

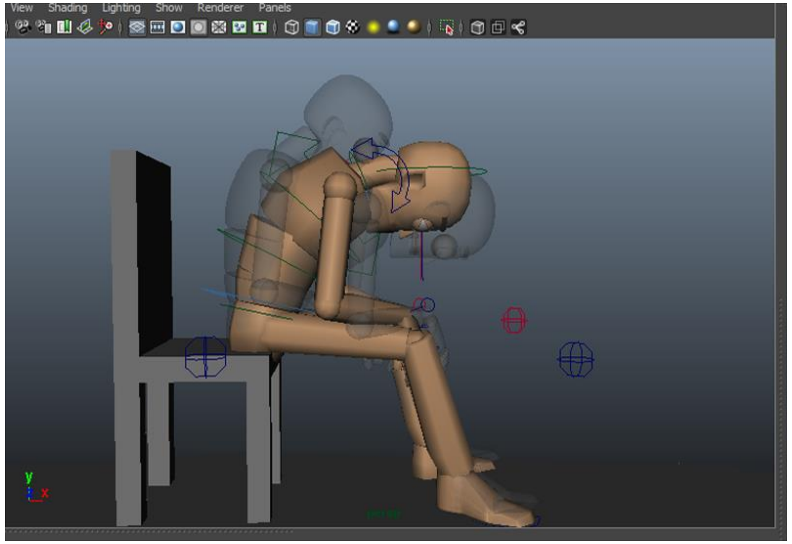
Character Animation Methods

- Modeling (manipulation)
  - Deformation
  - Blendshapes
  - Skeletons

- Interpolation
  - Key-framing
  - Kinematics
  - Motion Capture



https://blenderartists.org/

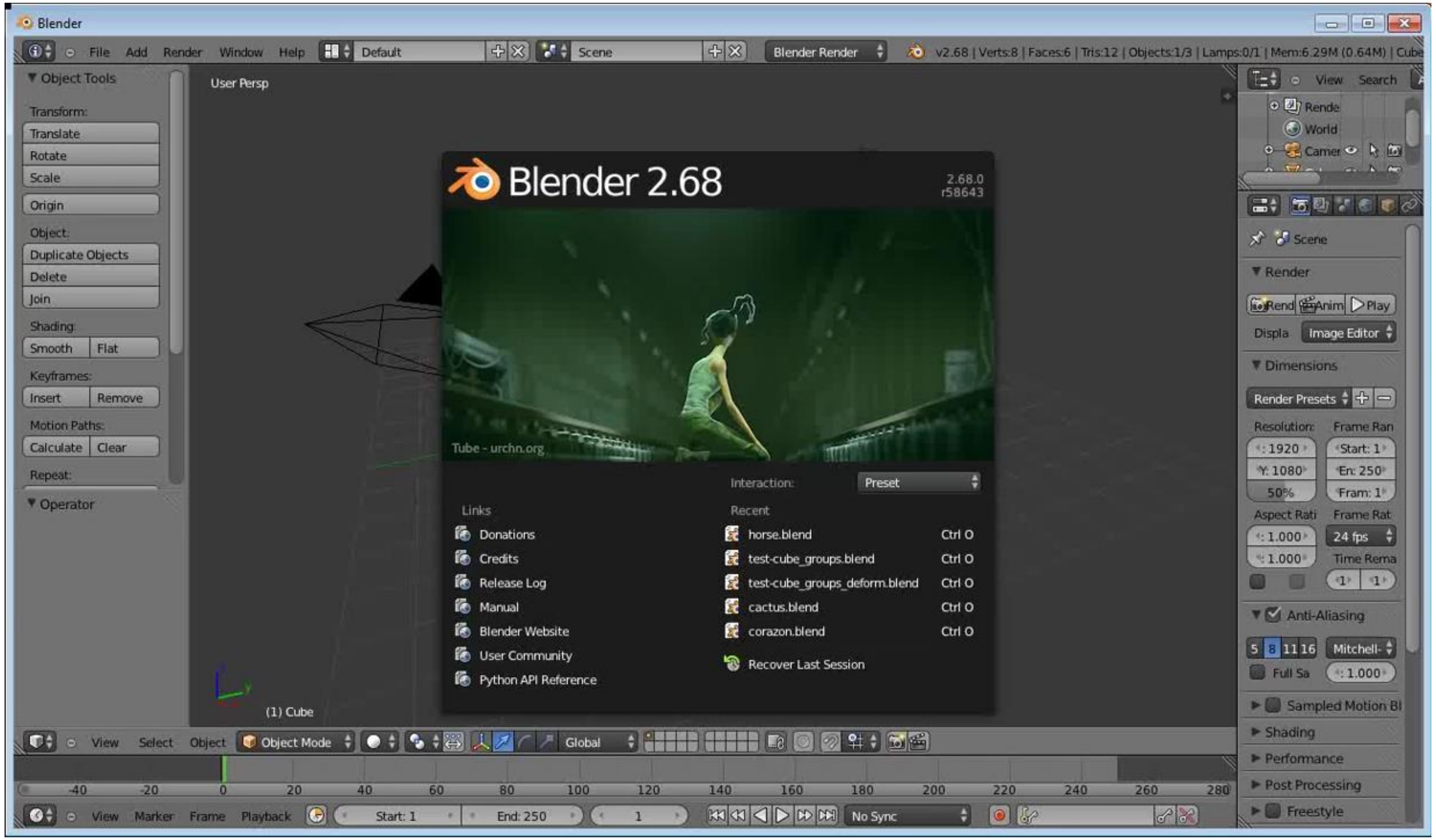


focus.gscept.com

Deformation

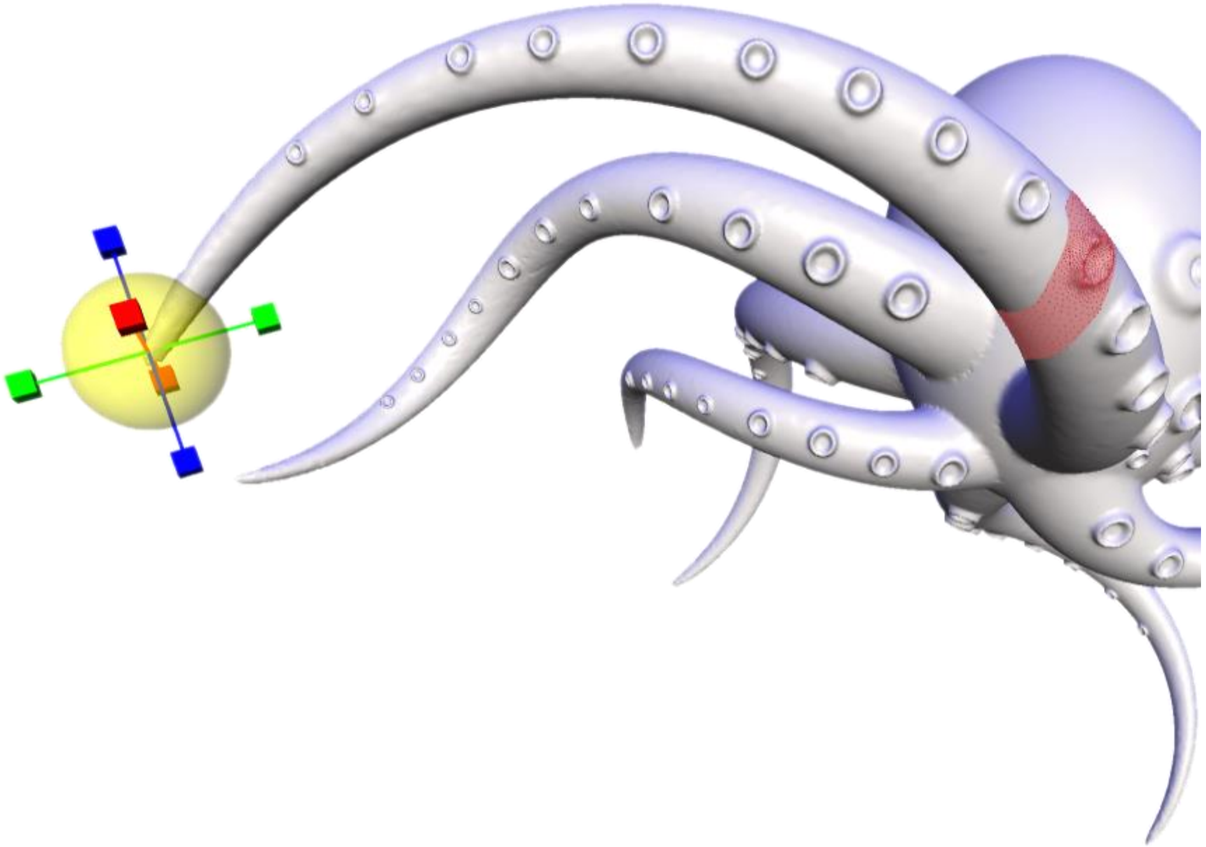
- How to change a character’s pose?
  - Every vertex directly
  - Intuitive computation

https://www.youtube.com/watch?v=oxkf_N-QCNI



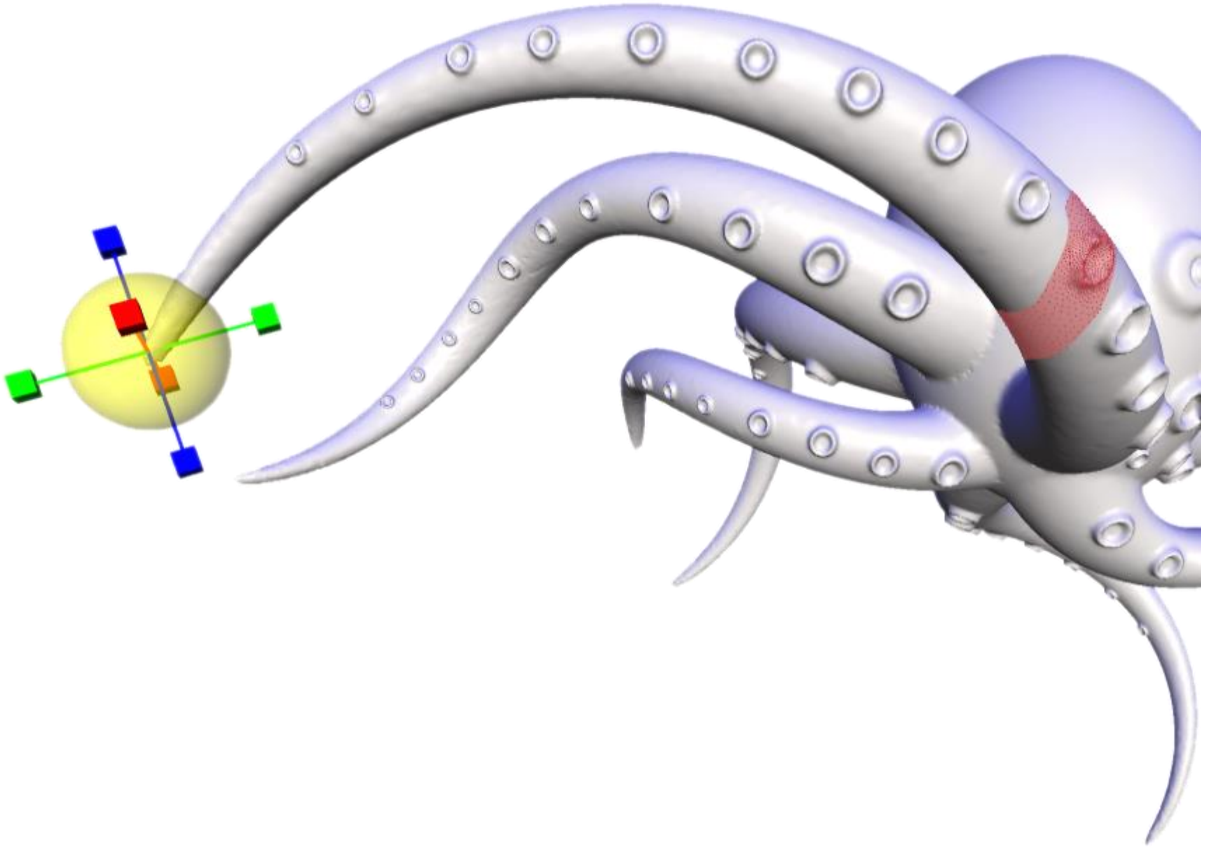
Deformation

- A HUGE variety of methods
  - Laplacian mesh editing
  - ARAP
  - CAGE Base
  - Barycentric coordinates
  - Heat diffusion
  - Variational
  - …



Deformation

- A HUGE variety of methods
  - Laplacian mesh editing
  - ARAP
  - CAGE Base
  - Barycentric coordinates
  - Heat diffusion
  - Variational
  - …

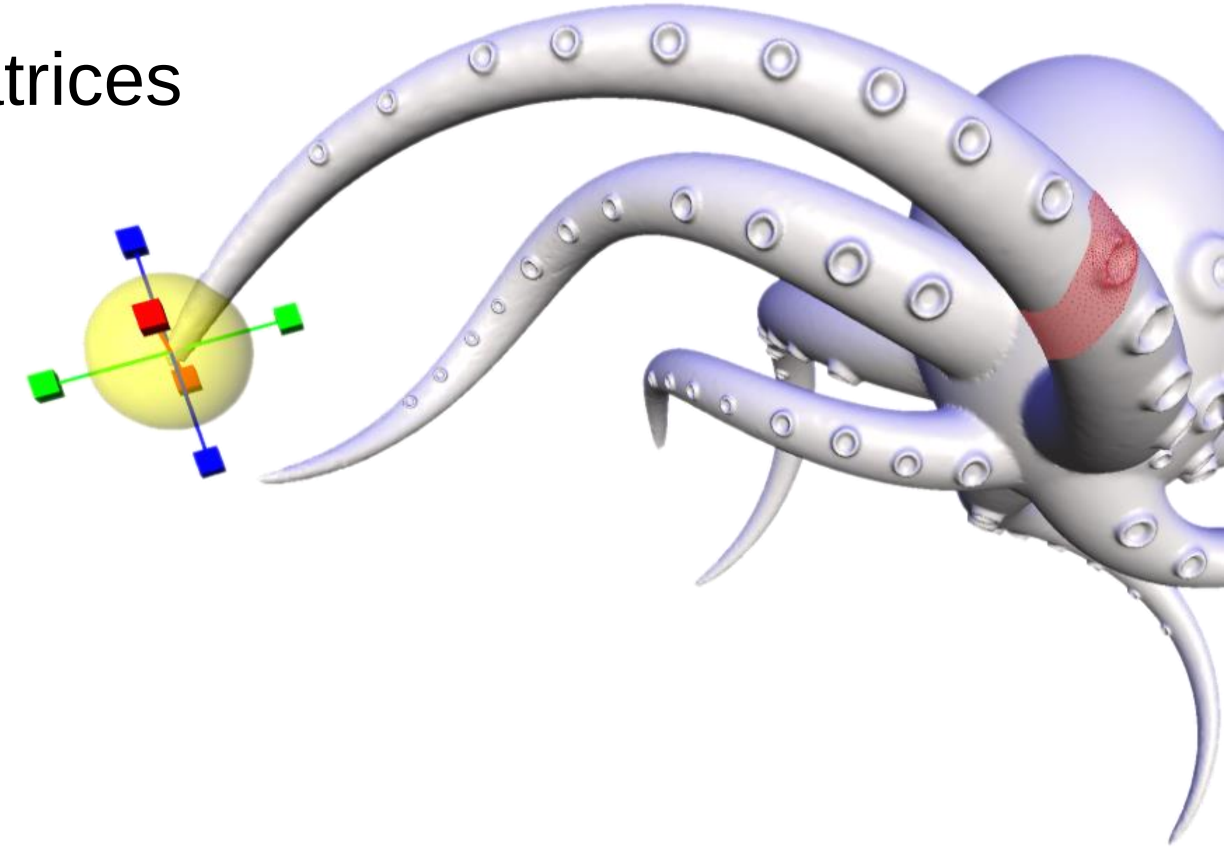


Laplacian Mesh Editing

- Local detail representation – enables detail preservation through various modeling tasks

- Representation with sparse matrices

- Efficient linear surface reconstruction



Overall framework

1. Compute differential representation

$$\delta_i = L(v_i) = v_i - \frac{1}{d_i}\Sigma_{j\in \mathrm{N}(i)} v_j$$

2. Pose modeling constraints

$$v_i' = u_i, \qquad i \in \boldsymbol{C}$$

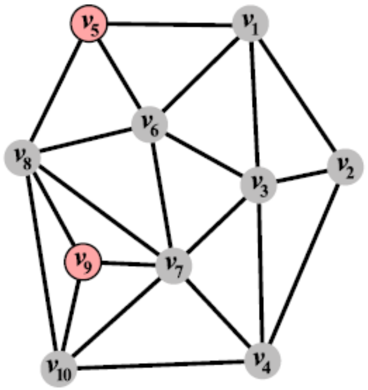
3. Reconstruct the surface – in least-squares sense

$$\begin{pmatrix} L \\ L_c \end{pmatrix} \boldsymbol{V} = \begin{pmatrix} \boldsymbol{\delta} \\ \boldsymbol{U} \end{pmatrix}$$

Differential coordinates?

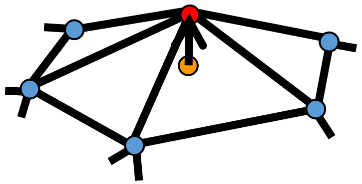
- In matrix form:

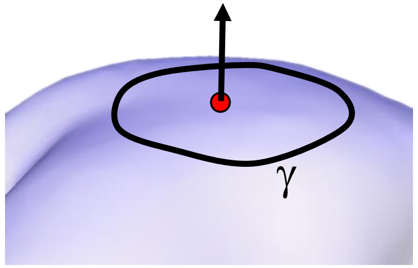
$$L_{ij} = \begin{cases} -w_{ij} & i \neq j \\ \Sigma_{j \in 1_{ring_i}} w_{ij} & i = j \\ 0 & else \end{cases}$$



$$\begin{bmatrix} 4 & -1 & -1 & & -1 & -1 & & & & \\ -1 & 3 & -1 & -1 & & & & & & \\ -1 & -1 & 5 & -1 & & -1 & -1 & & & \\ & -1 & -1 & 4 & & & -1 & & & -1 \\ -1 & & & & 3 & -1 & & -1 & & \\ -1 & & -1 & & & 4 & -1 & -1 & & \\ & & -1 & -1 & & -1 & 6 & -1 & -1 & -1 \\ & & & & -1 & -1 & -1 & 6 & -1 & -1 \\ & & & & & & -1 & -1 & 3 & -1 \\ & & & -1 & & & -1 & -1 & -1 & 4 \end{bmatrix}$$

- They represent the **local** detail / local shape description
  - The direction approximates the normal
  - The size approximates the mean curvature





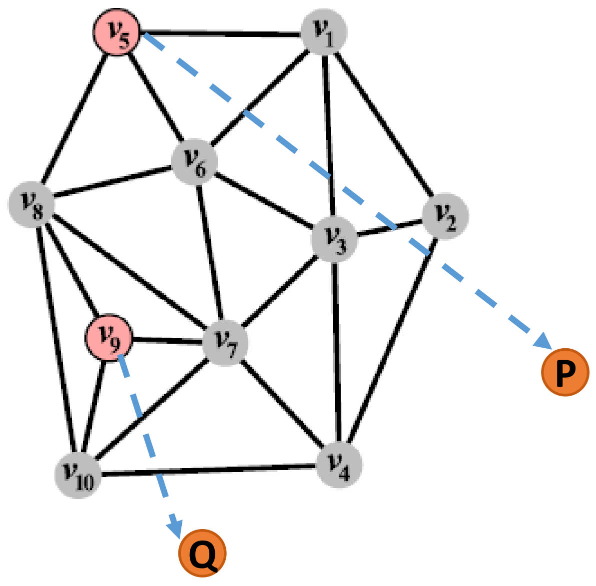
Adding constraints

- In matrix form:

$$L_{ij} = \begin{cases} -w_{ij} & i \neq j \\ \sum_{j \in 1ring_i} w_{ij} & i = j \\ 0 & else \end{cases}$$

4	−1	−1		−1	−1				
−1	3	−1	−1						
−1	−1	5	−1		−1	−1			
	−1	−1	4			−1			−1
−1				3	−1		−1		
−1		−1			4	−1	−1		
		−1	−1		−1	6	−1	−1	−1
				−1	−1	−1	6	−1	−1
						−1	−1	3	−1
			−1			−1	−1	−1	4

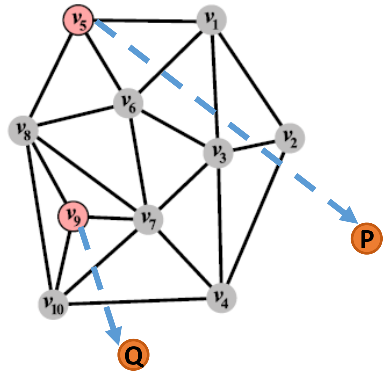
				1					
								1	



Adding constraints

- In matrix form:

$$L_{ij} = \begin{cases} -w_{ij} & i \neq j \\ \Sigma_{j \in 1_{ring_i}} w_{ij} & i = j \\ 0 & else \end{cases}$$



$$\begin{bmatrix} 4 & -1 & -1 & & -1 & -1 & & & & \\ -1 & 3 & -1 & -1 & & & & & & \\ -1 & -1 & 5 & -1 & & -1 & -1 & & & \\ & -1 & -1 & 4 & & & -1 & & & -1 \\ -1 & & & & 3 & -1 & & -1 & & \\ -1 & & -1 & & & 4 & -1 & -1 & & \\ & & -1 & -1 & & -1 & 6 & -1 & -1 & -1 \\ & & & & -1 & -1 & -1 & 6 & -1 & -1 \\ & & & & & & -1 & -1 & 3 & -1 \\ & & & -1 & & & -1 & -1 & -1 & 4 \end{bmatrix} \begin{bmatrix} & & & & 1 & & & & & \\ & & & & & & & & 1 & \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{pmatrix} = \begin{pmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \\ \delta_5 \\ \delta_6 \\ \delta_7 \\ \delta_8 \\ \delta_9 \\ \delta_{10} \\ \boldsymbol{P_x} \\ \boldsymbol{Q_x} \end{pmatrix}$$

$$\begin{bmatrix} 4 & -1 & -1 & & -1 & -1 & & & & \\ -1 & 3 & -1 & -1 & & & & & & \\ -1 & -1 & 5 & -1 & & -1 & -1 & & & \\ & -1 & -1 & 4 & & & -1 & & & -1 \\ -1 & & & & 3 & -1 & & -1 & & \\ -1 & & -1 & & & 4 & -1 & -1 & & \\ & & -1 & -1 & & -1 & 6 & -1 & -1 & -1 \\ & & & & -1 & -1 & -1 & 6 & -1 & -1 \\ & & & & & & -1 & -1 & 3 & -1 \\ & & & -1 & & & -1 & -1 & -1 & 4 \end{bmatrix} \begin{bmatrix} & & & & 1 & & & & & \\ & & & & & & & & 1 & \end{bmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \end{pmatrix} = \begin{pmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \\ \delta_5 \\ \delta_6 \\ \delta_7 \\ \delta_8 \\ \delta_9 \\ \delta_{10} \\ \boldsymbol{P_y} \\ \boldsymbol{Q_y} \end{pmatrix}$$

Example

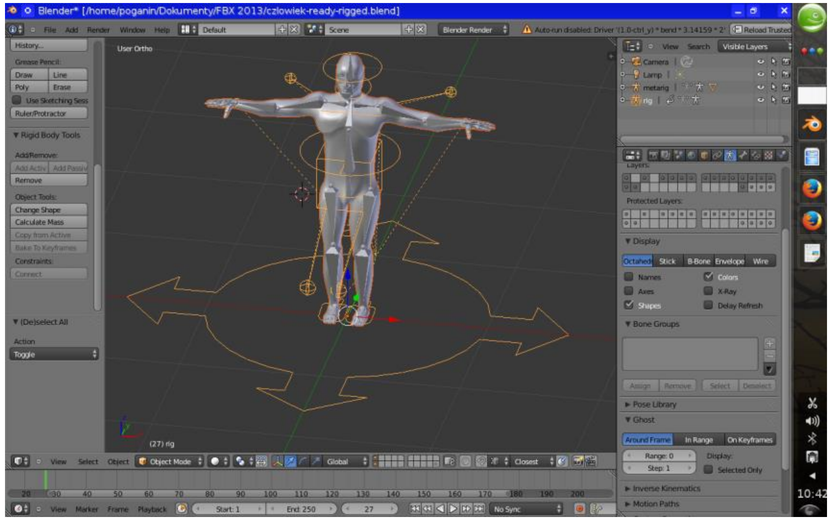
Laplacian Mesh Editing

A short editing session
with the *Octopus*

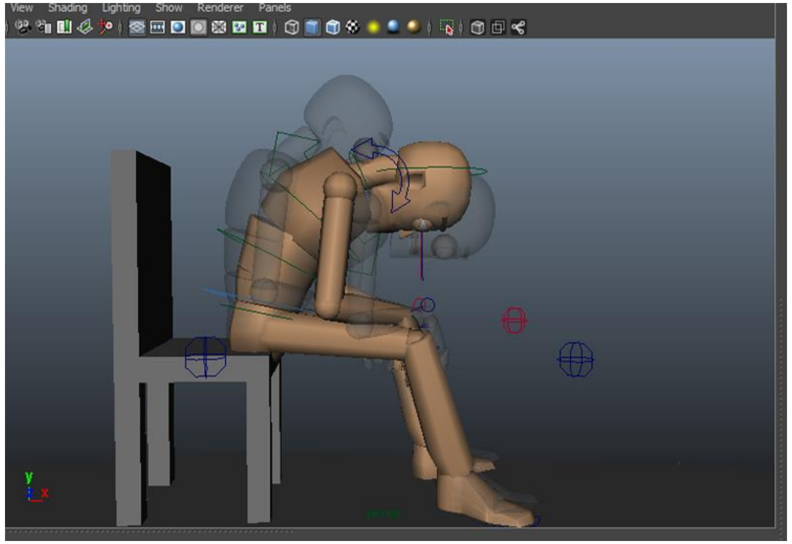
Character Animation Methods

- Modeling (manipulation)
  - Deformation
  - Blendshapes
  - Skeletons

- Interpolation
  - Key-framing
  - Kinematics
  - Motion Capture



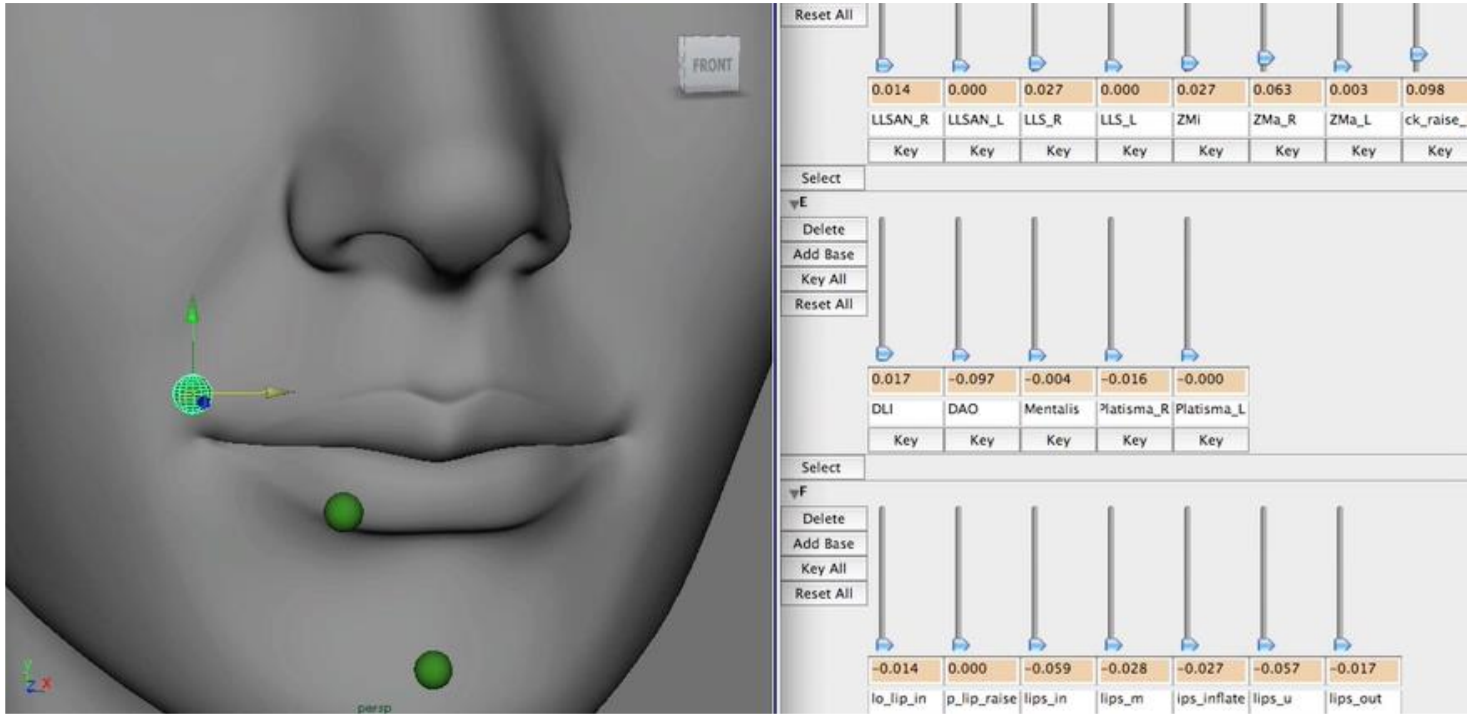
https://blenderartists.org/



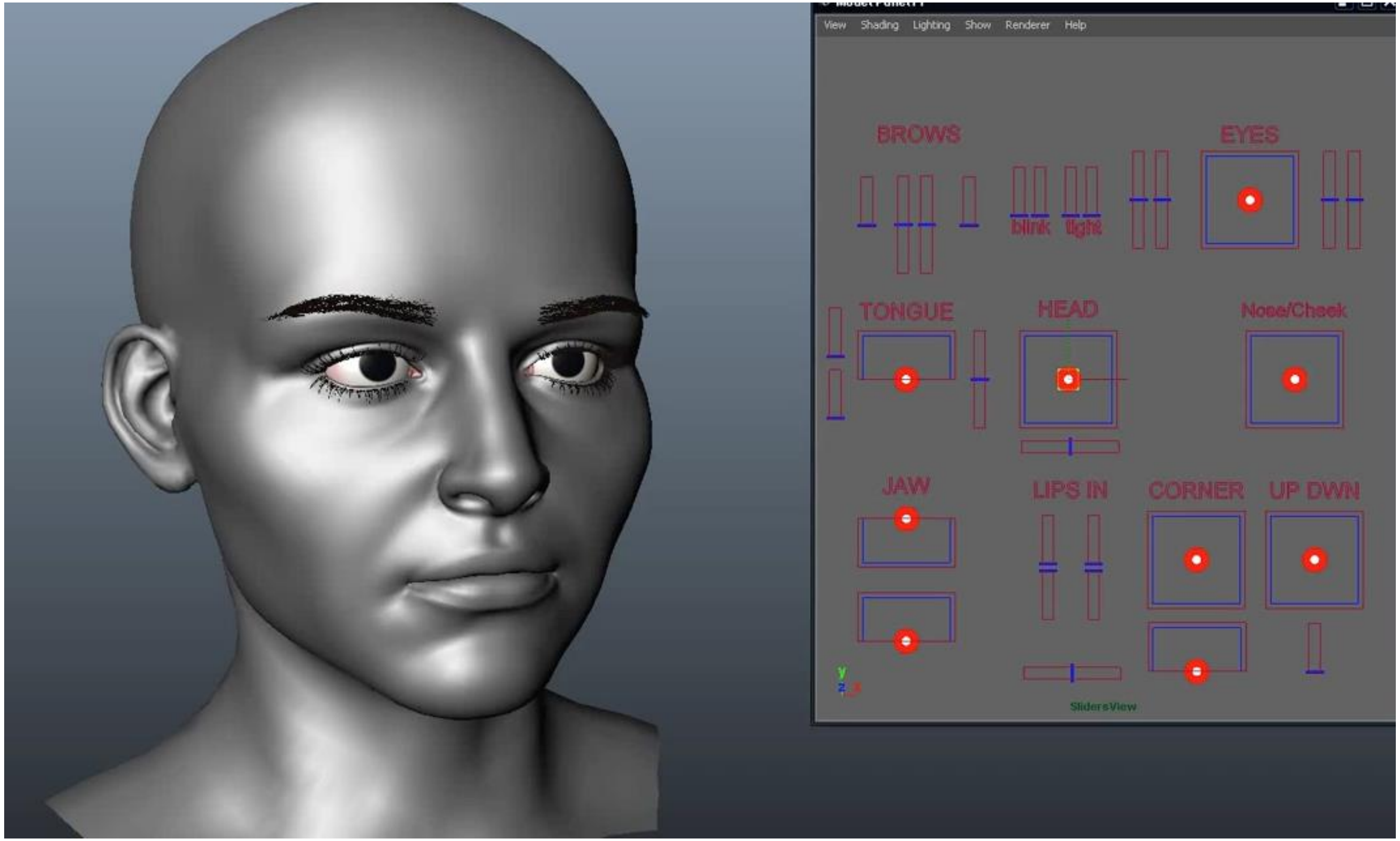
focus.gscept.com

Blendshapes

- Blendshapes are an approximate semantic parameterization
- Linear blend of predefined poses



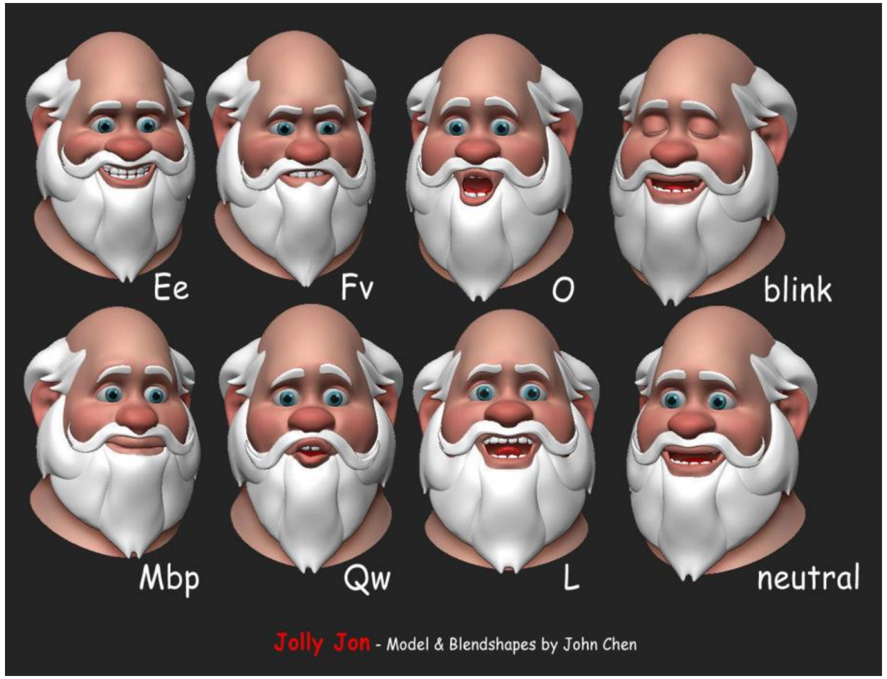
Blendshapes



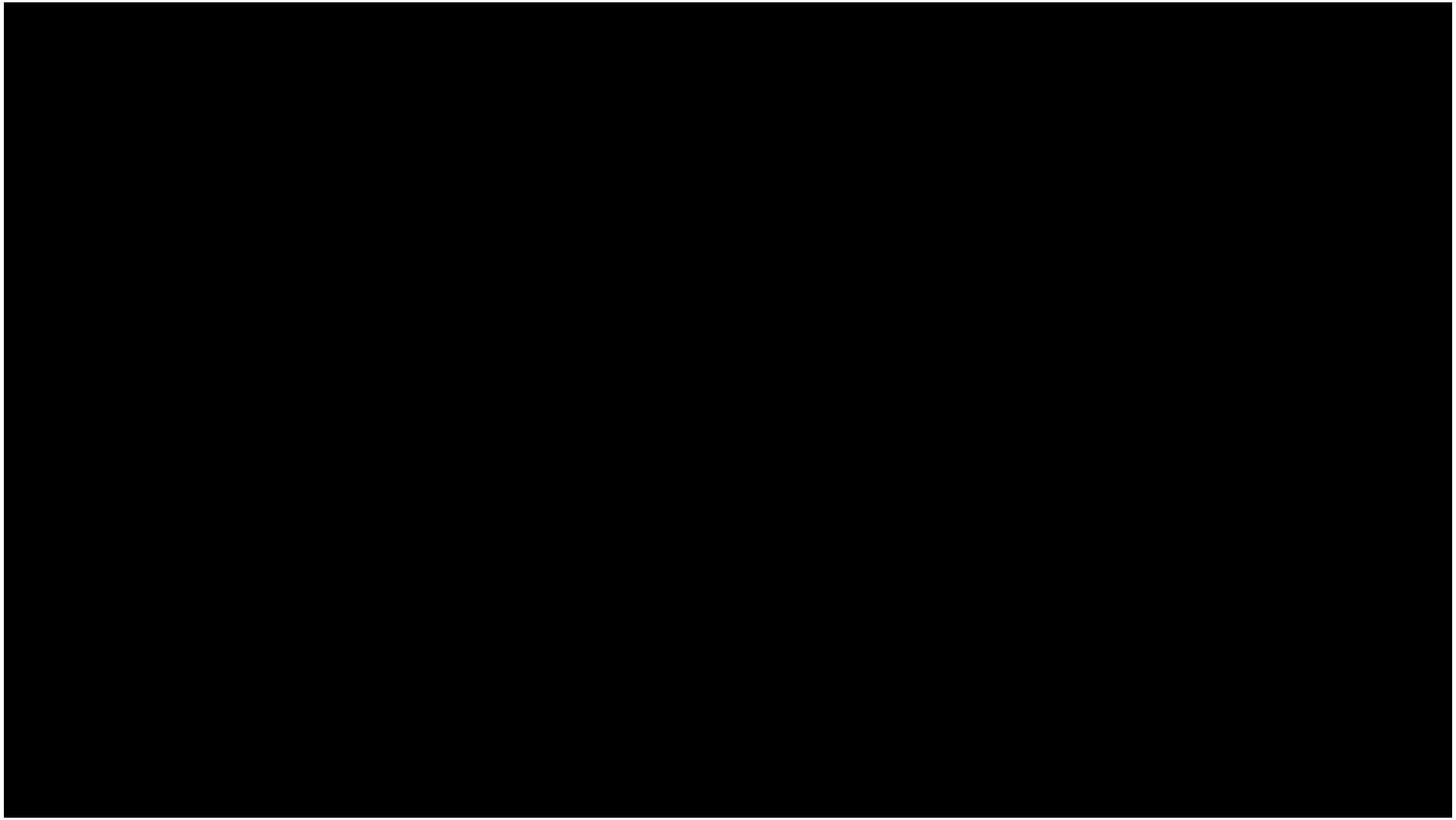
https://www.youtube.com/watch?v=KPDfMpuK2fQ

Blendshapes

- Usually used for difficult to pose complex deformations
  - Such faces
- Given:
  - A mesh $M = (V, E)$ with m vertices
  - n configurations of the same mesh, $M_b = (V_b, E), b = 1 \dots n$
- A new configuration is simply:
  - $M' = (\Sigma_{b=1\dots n} \mathrm{w_b V_b}, \mathrm{E})$
- Delta formulation:
  - $M' = (\Sigma_{b=1\dots n} V_0 + \mathrm{w_b}(\mathrm{V_b} - \mathrm{V_0}), \mathrm{E})$
  - A bit more convenient
- M_0 - the rest pose, w_b blend weights



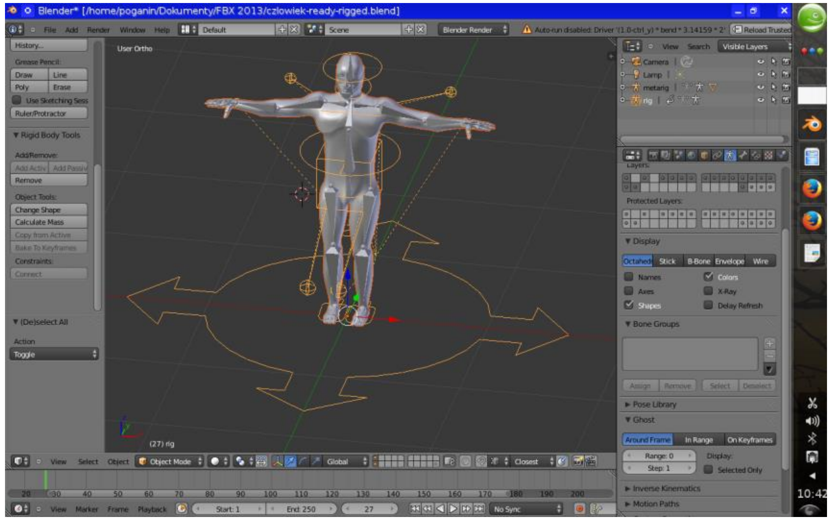
Blendshapes



https://www.youtube.com/watch?v=zvUfiKQI5jQ

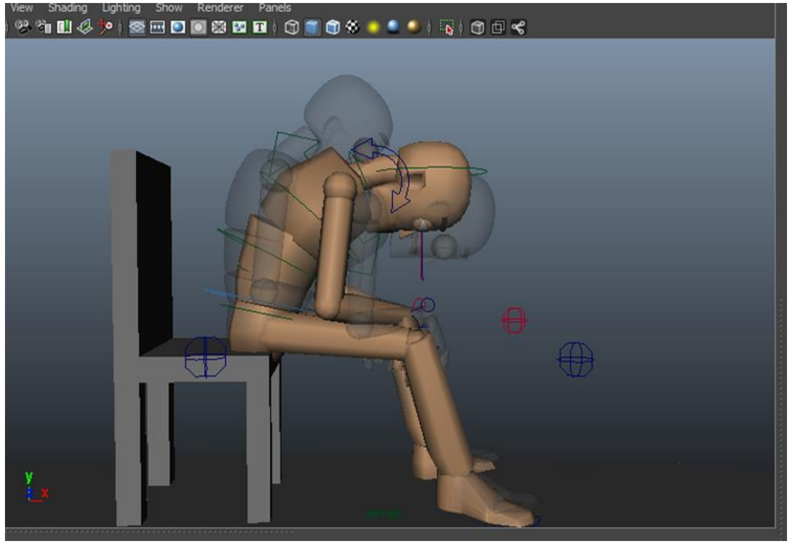
Character Animation Methods

- Modeling (manipulation)
  - Deformation
  - Blendshapes
  - Skeletons



https://blenderartists.org/

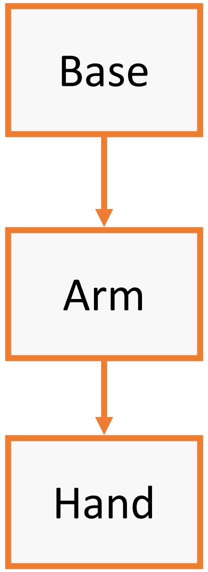
- Interpolation
  - Key-framing
  - Kinematics
  - Motion Capture



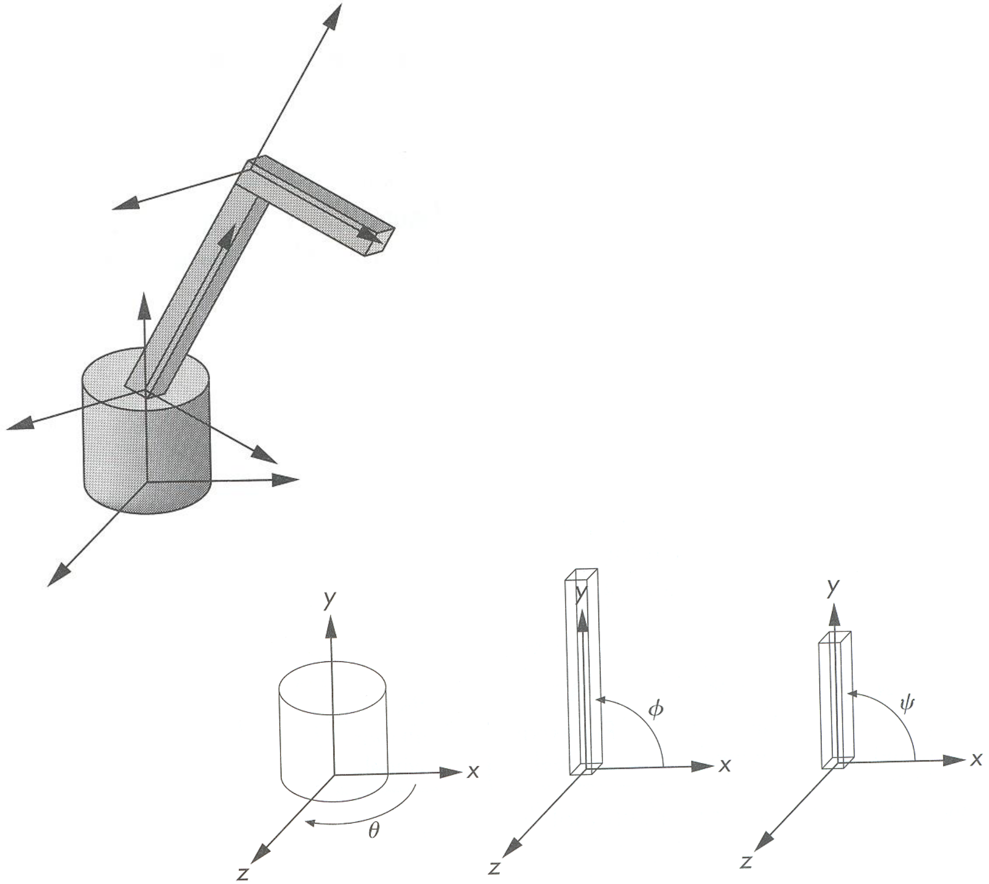
focus.gscept.com

Articulated Figures

- Character poses described by set of rigid bodies connected by “joints”



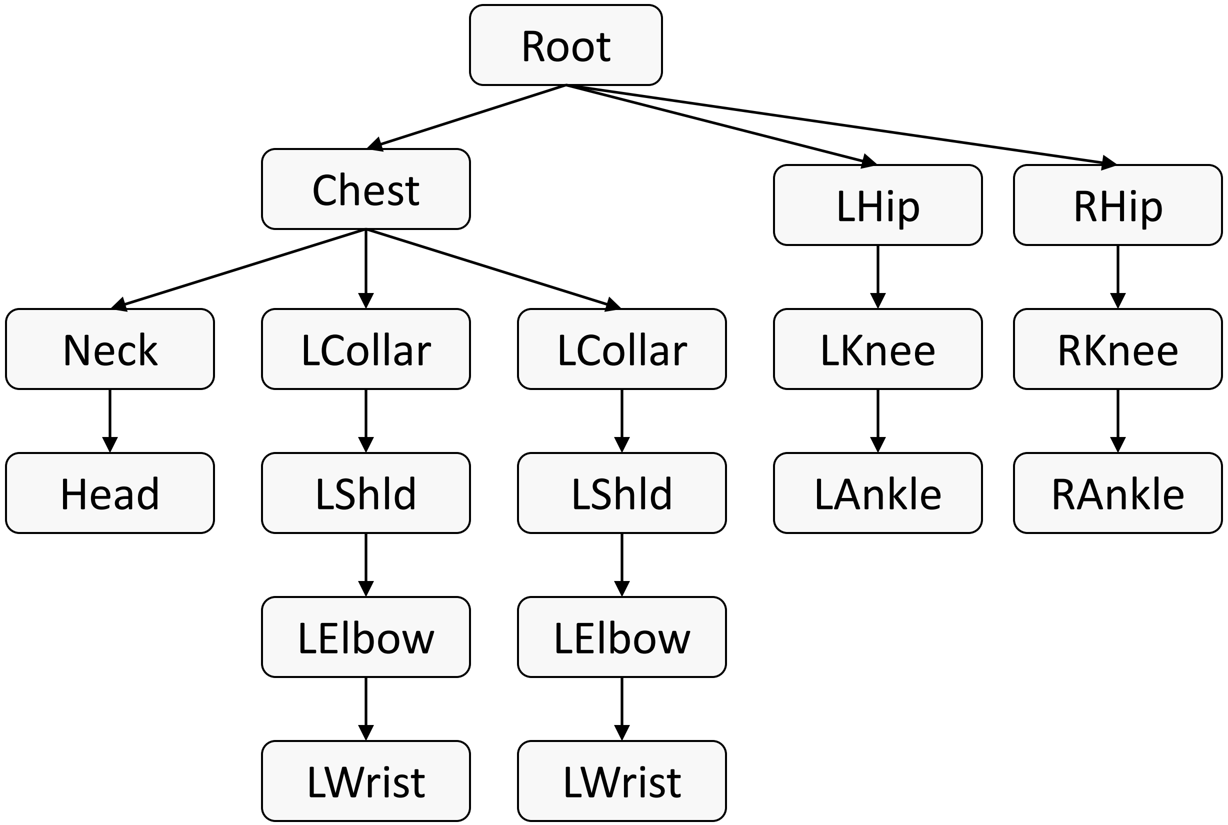
Scene Graph

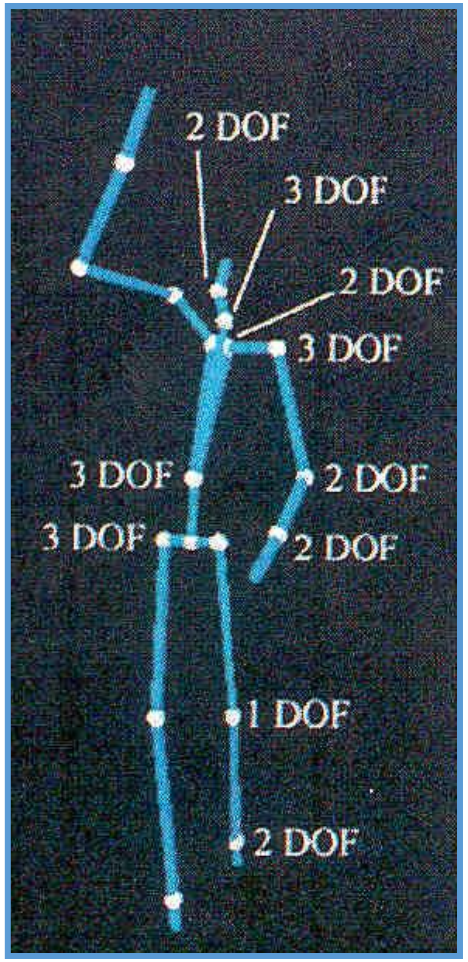


Angel Figures 8.8 & 8.9

Articulated Figures

- Well-suited for humanoid characters

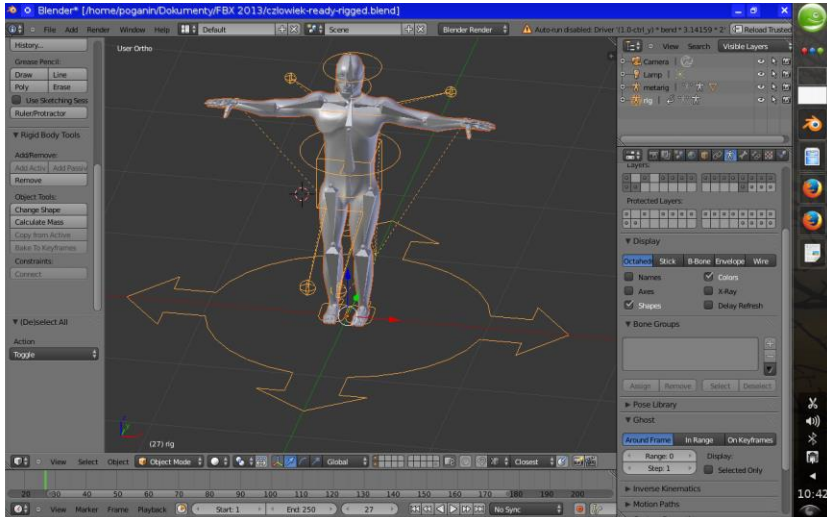




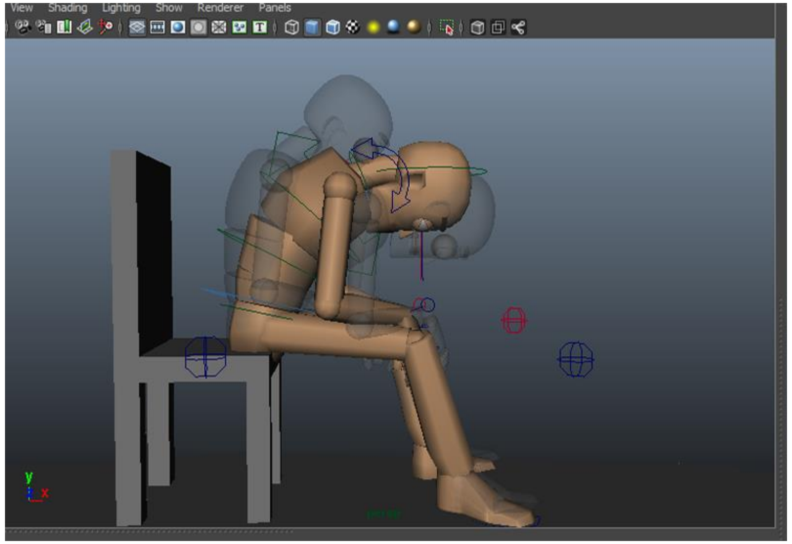
Rose et al. `96

Character Animation Methods

- Modeling (manipulation)
  - Deformation
  - Blendshapes
  - Skeletons
- Interpolation
  - Key-framing
  - Kinematics
  - Motion Capture



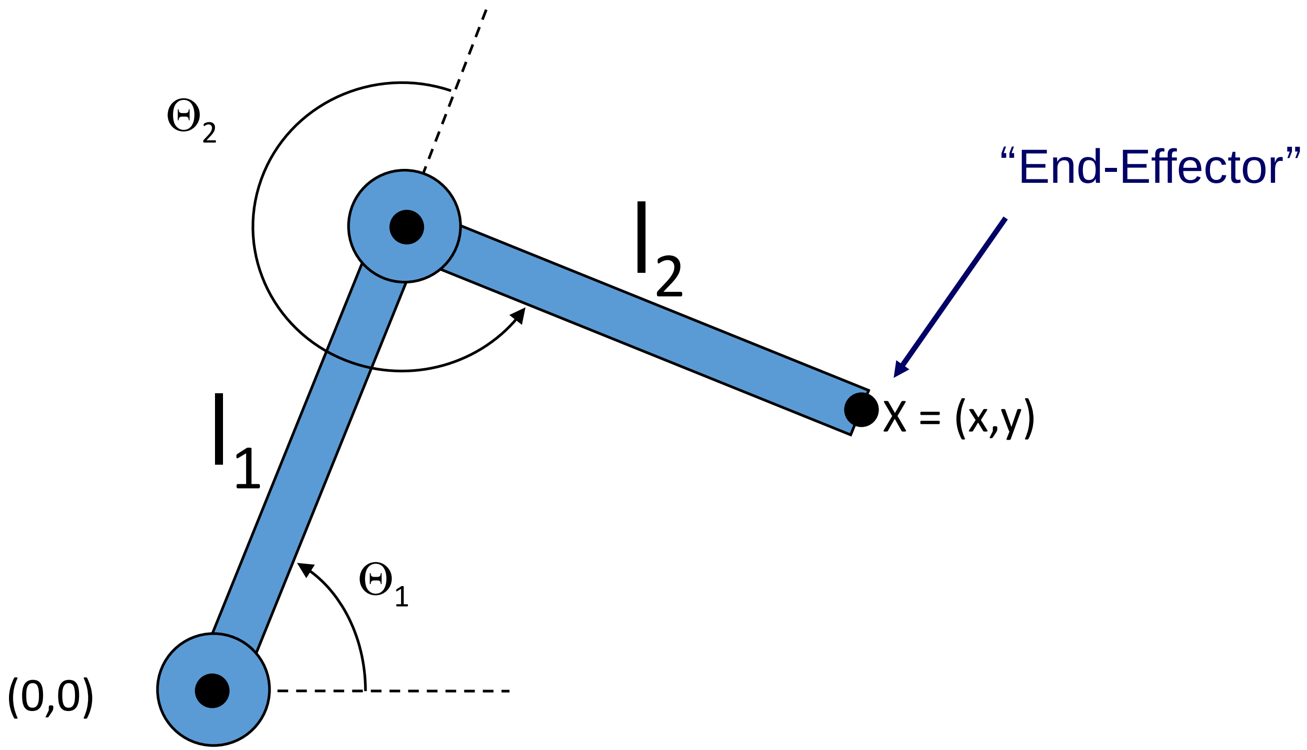
https://blenderartists.org/



focus.gscept.com

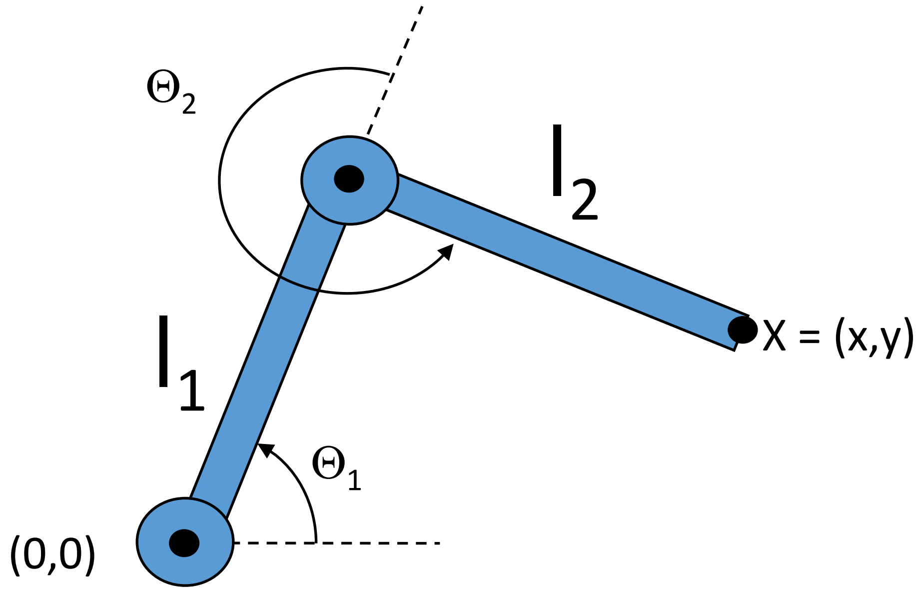
Forward Kinematics

- Describe motion of articulated character



Forward Kinematics

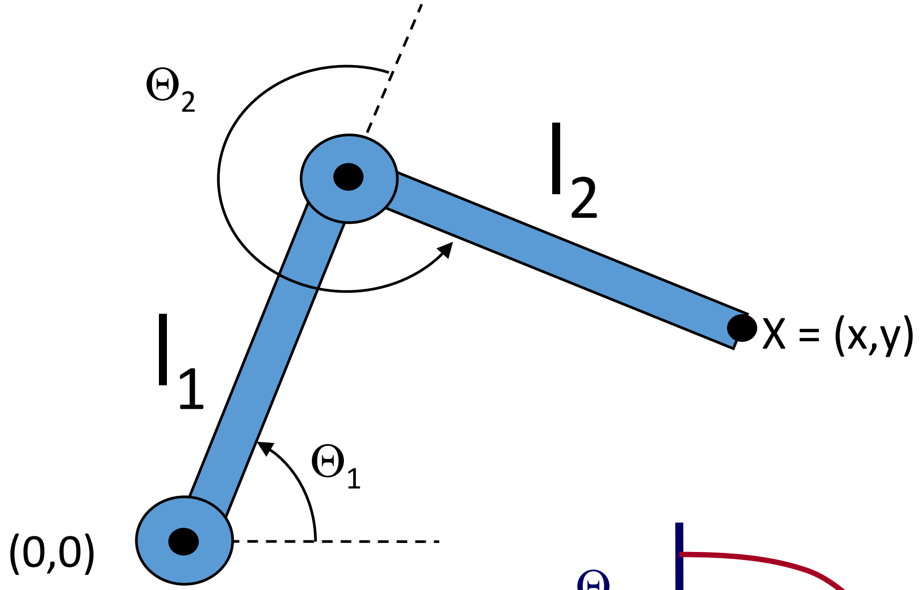
- Animator specifies joint angles: Θ_1 and Θ_2
- Computer finds positions of end-effector: X

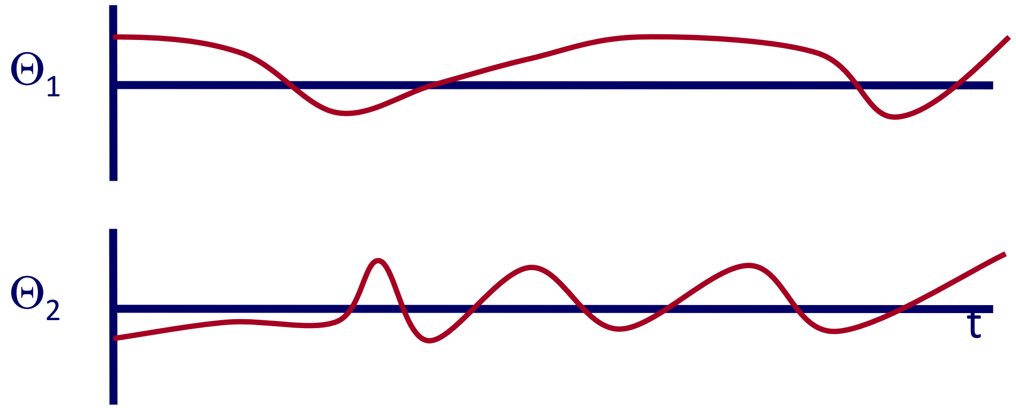


$$X = (l_1 \cos\Theta_1 + l_2 \cos(\Theta_1 + \Theta_2), l_1 \sin\Theta_1 + l_2 \sin(\Theta_1 + \Theta_2))$$

Forward Kinematics Parameterization

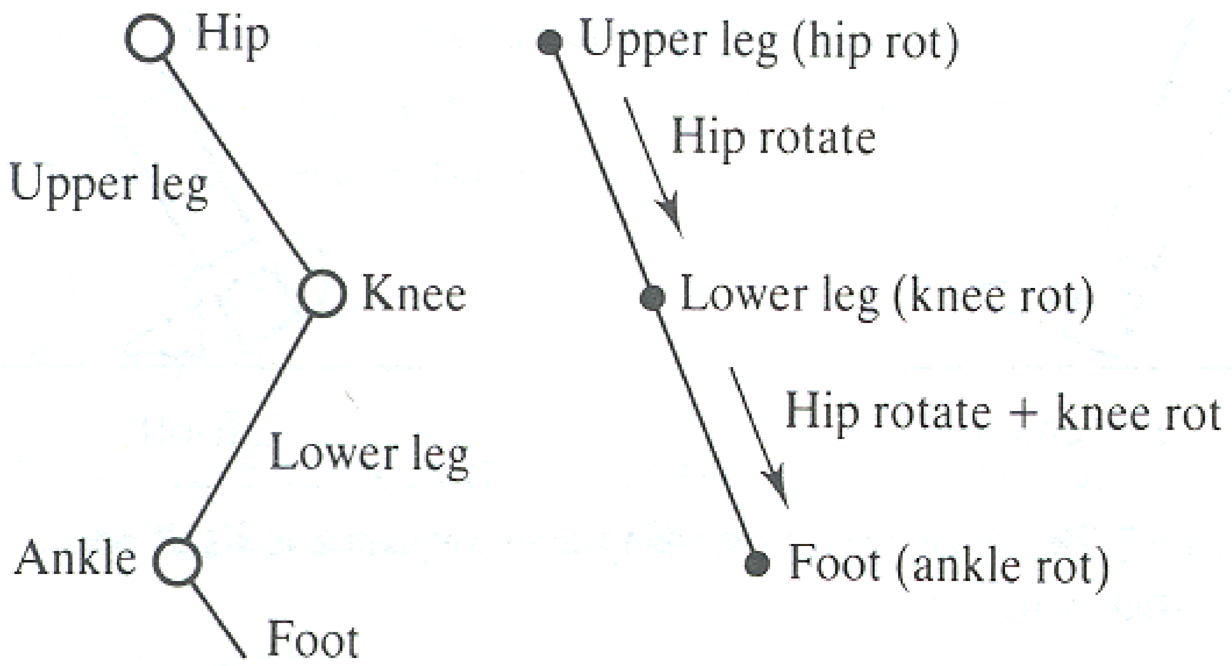
- Joint motions specified e.g. by spline curves





Example: Walk Cycle

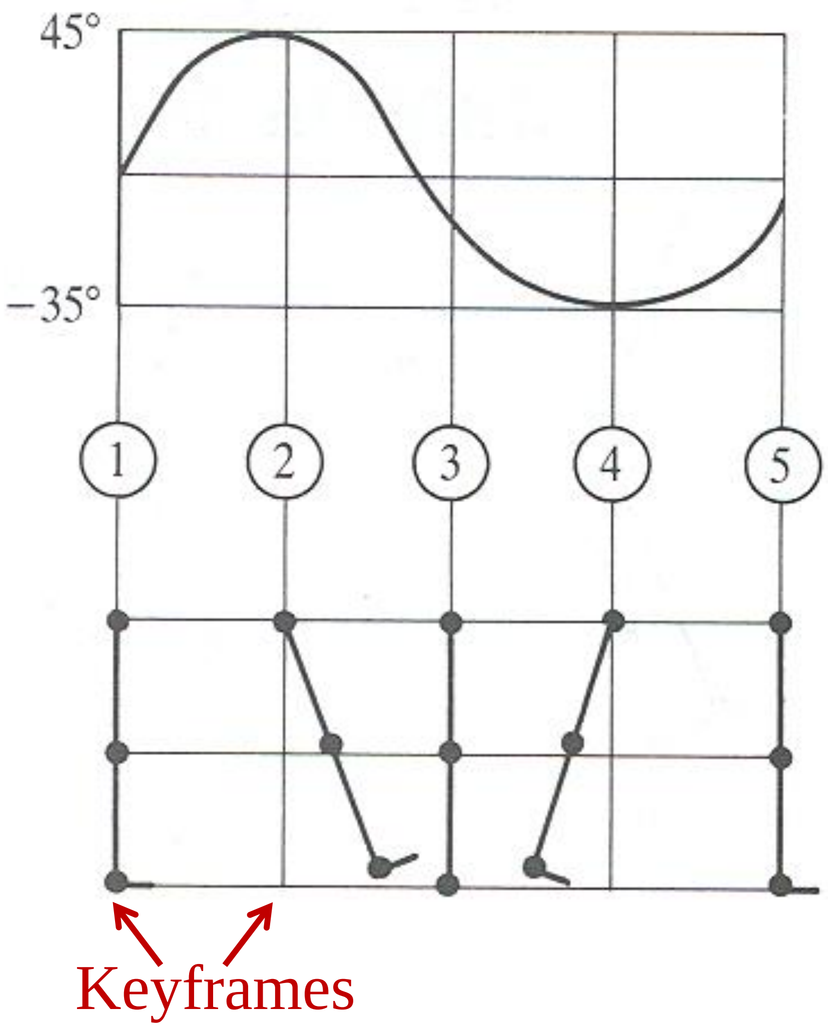
- Articulated figure:



Watt & Watt

Example: Walk Cycle

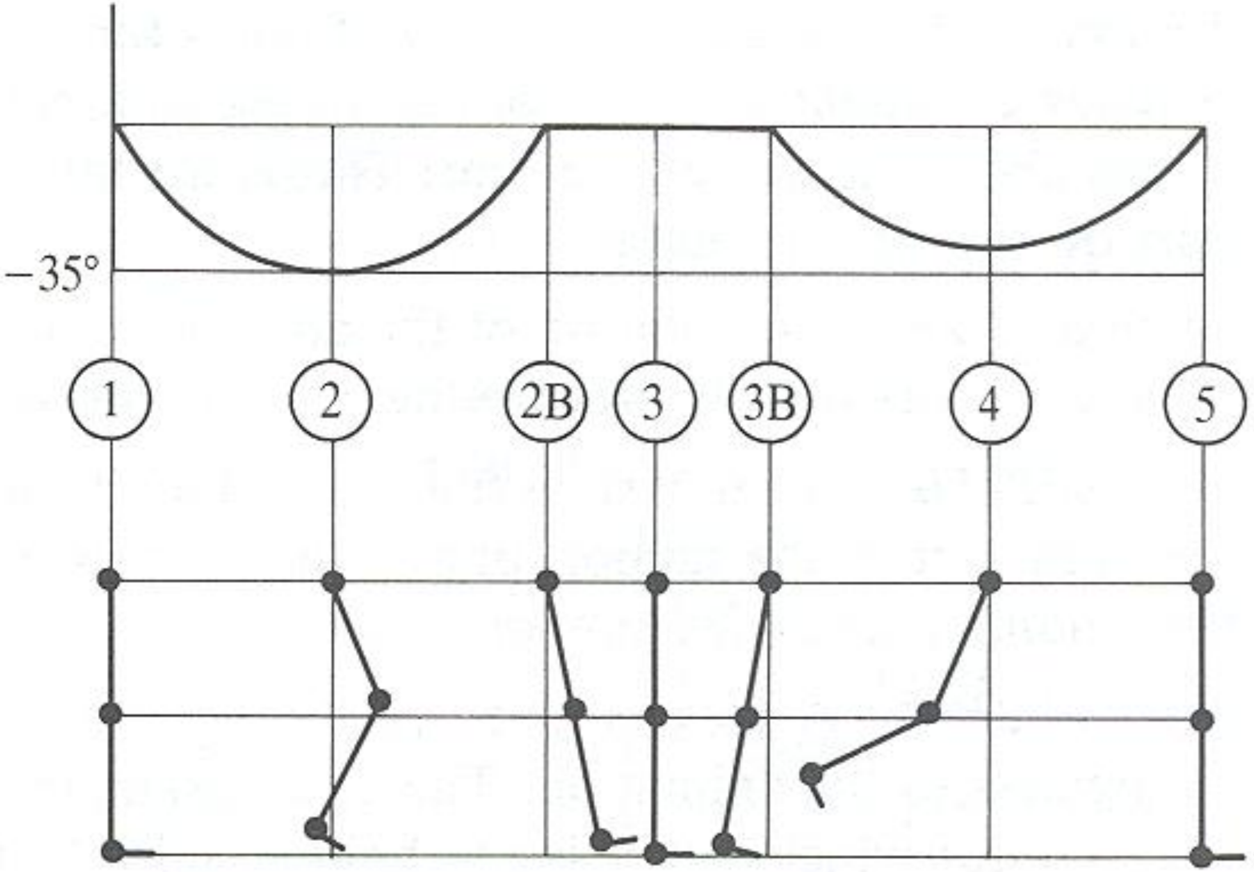
- Hip joint orientation:



Watt & Watt

Example: Walk Cycle

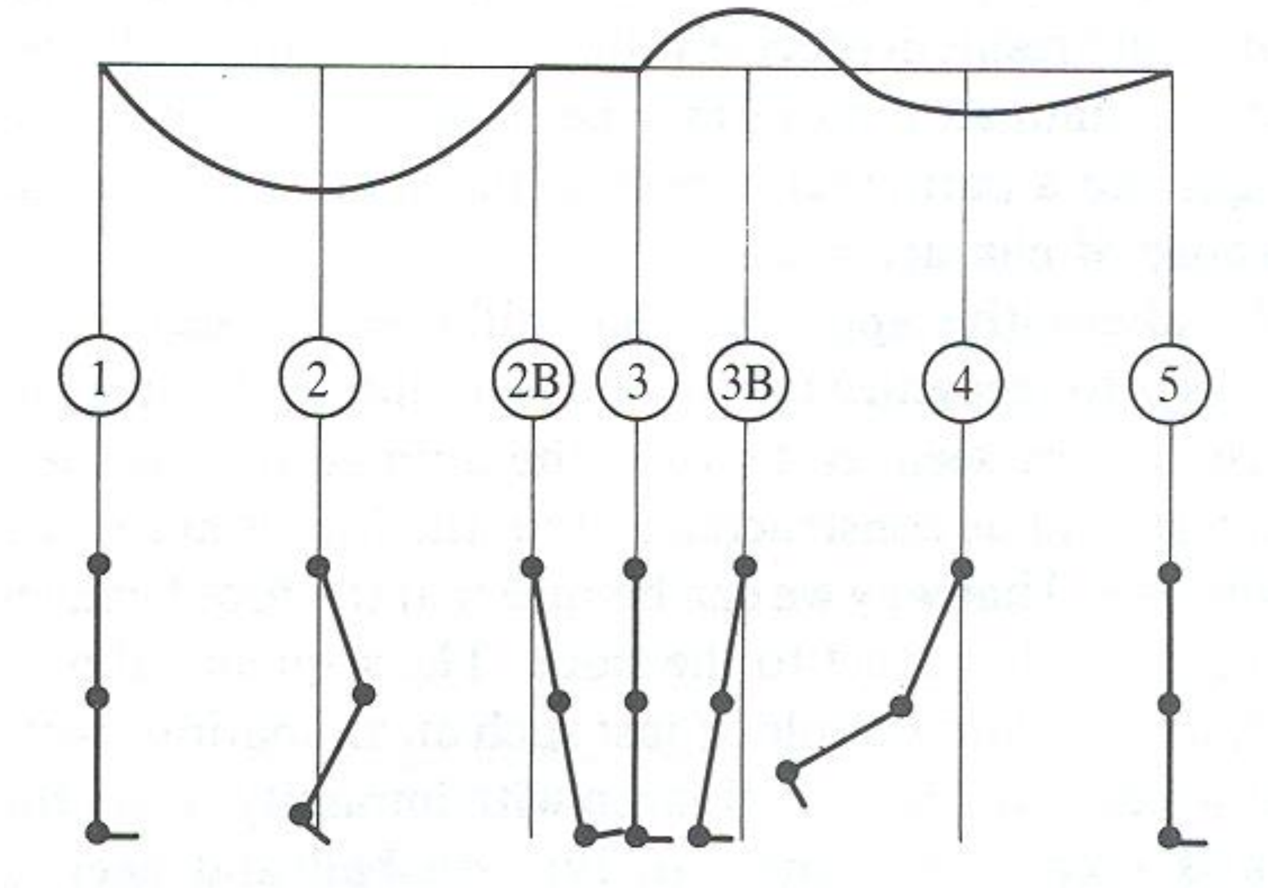
- Knee joint orientation:



Watt & Watt

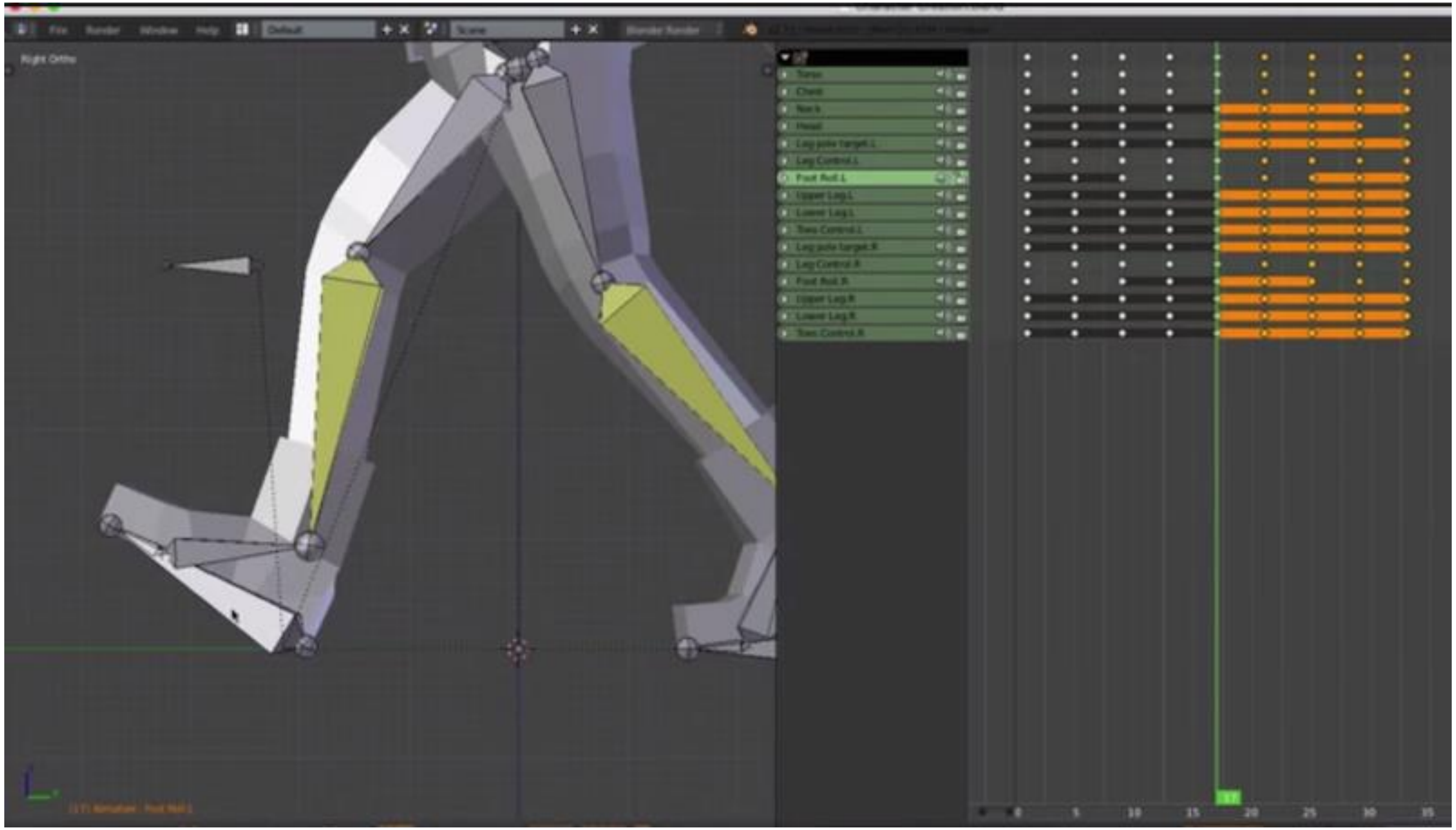
Example: Walk Cycle

- Ankle joint orientation:



Watt & Watt

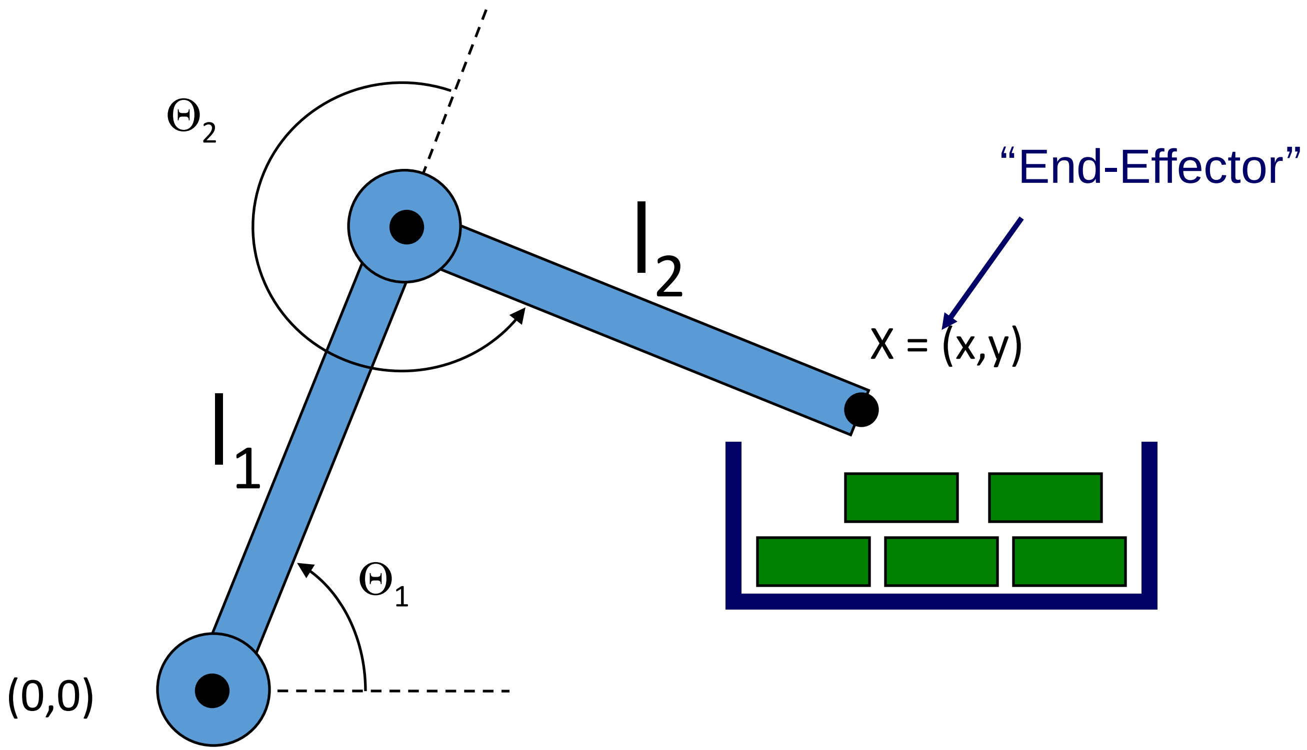
Example: walk cycle



https://www.youtube.com/watch?v=DuUWxUitJos

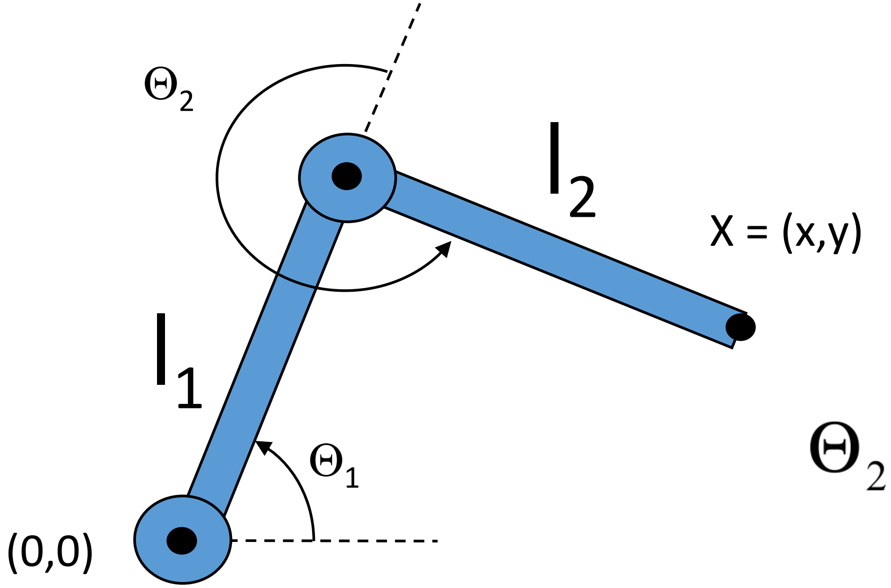
Inverse Kinematics

- What if animator knows position of "end-effector"?



Inverse Kinematics

- Animator specifies end-effector positions: X
- Computer finds joint angles: Θ_1 and Θ_2:

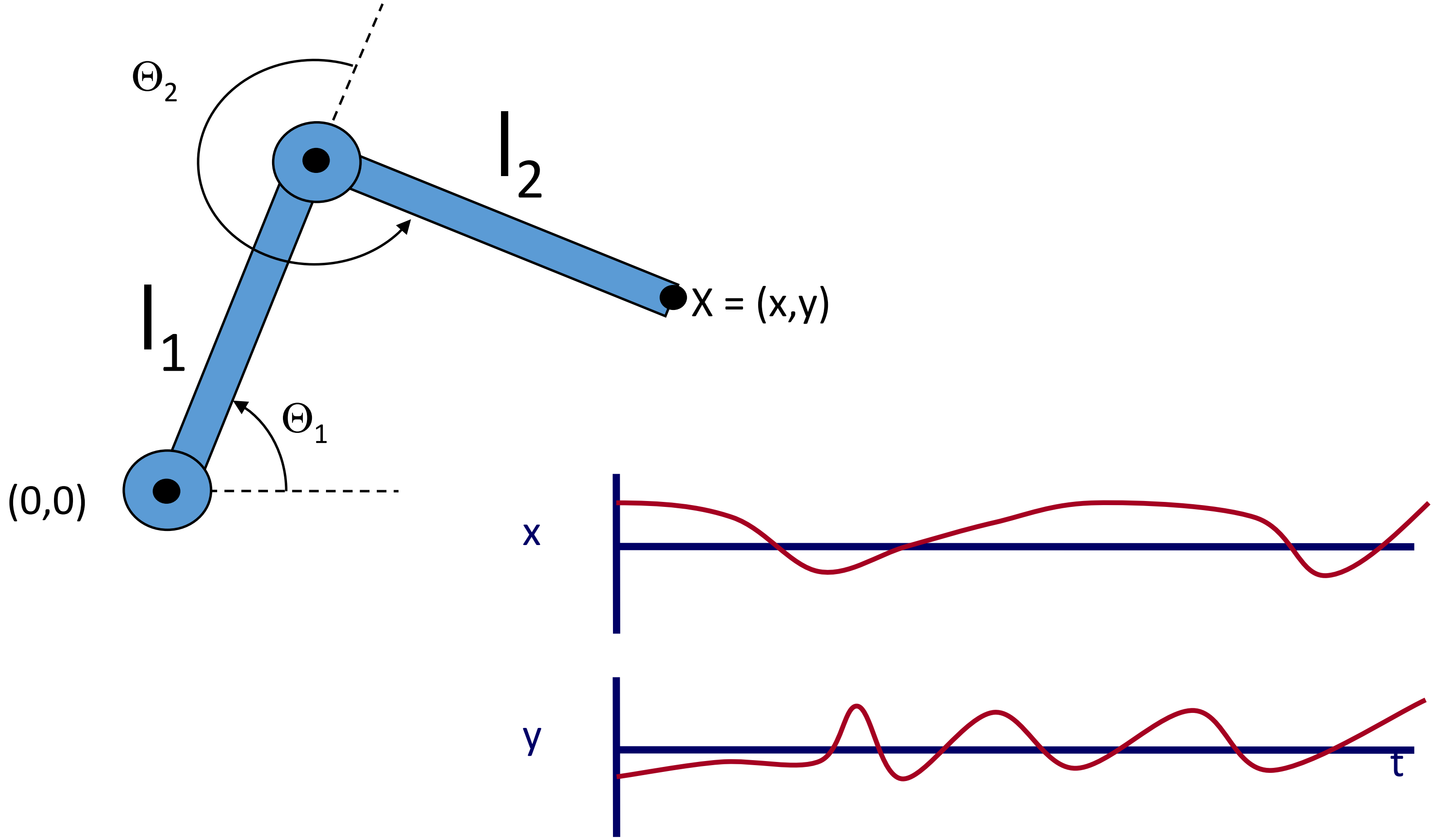


$$\Theta_2 = \cos^{-1}\left(\frac{x^2 + x^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right)$$

$$\Theta_1 = \frac{-(l_2 \sin(\Theta_2) x + (l_1 + l_2 \cos(\Theta_2)) y}{(l_2 \sin(\Theta_2)) y + (l_1 + l_2 \cos(\Theta_2)) x}$$

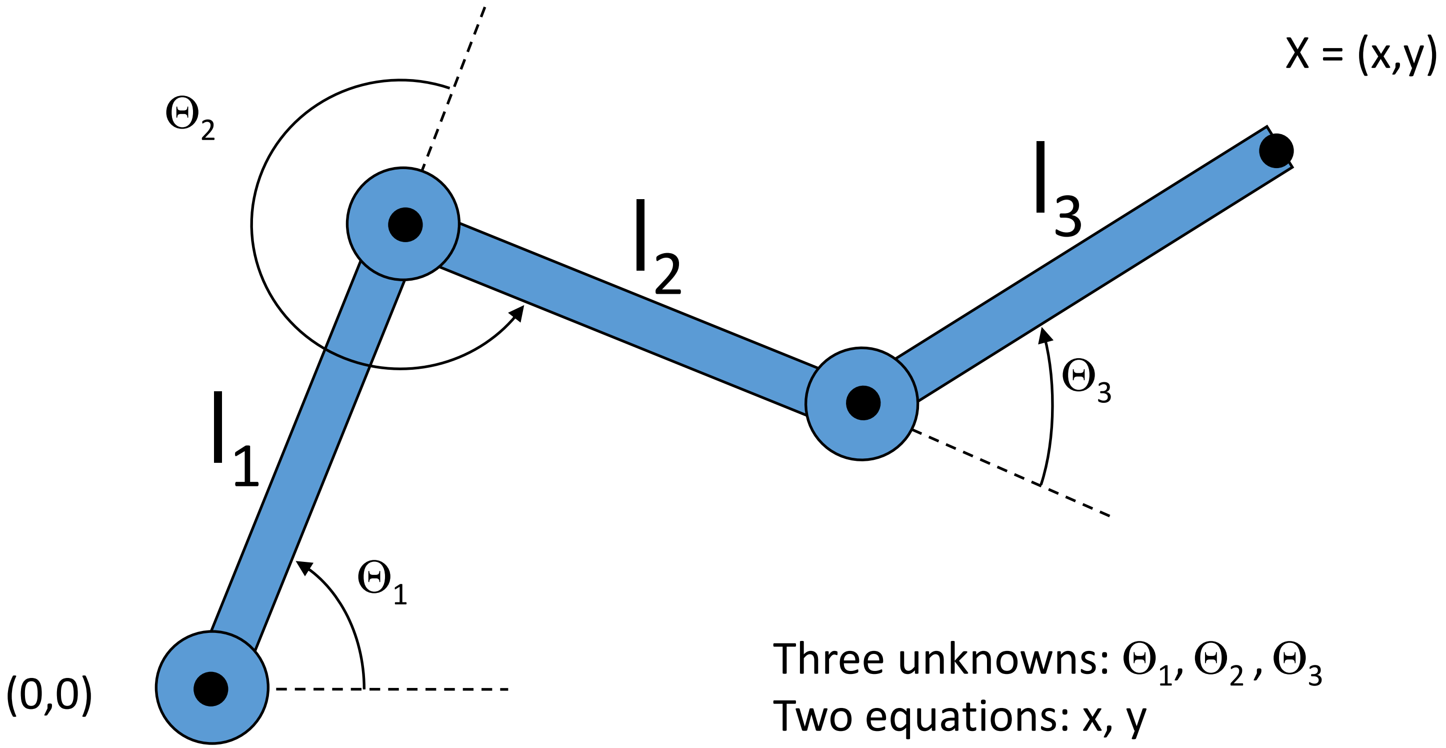
Inverse Kinematics

- End-effector postions can be specified by spline curves



Inverse Kinematics

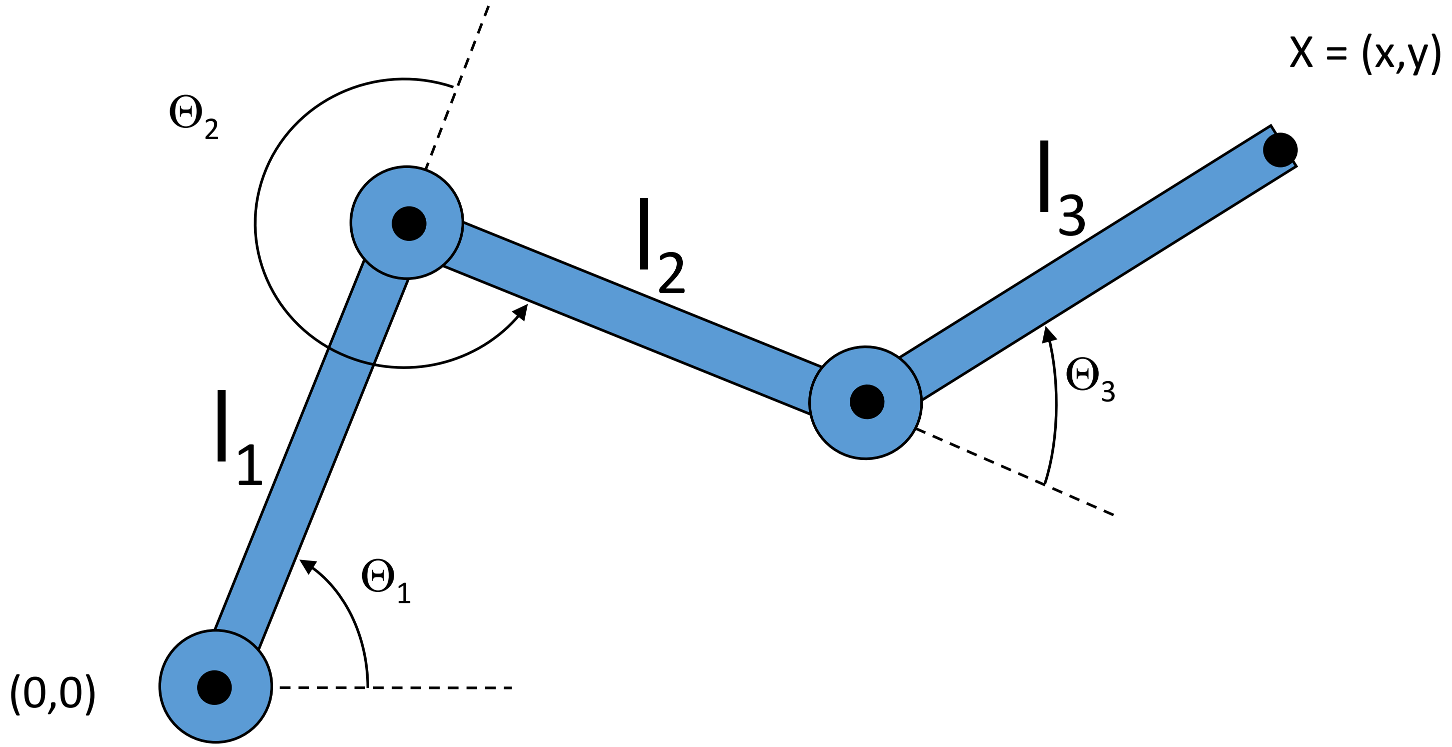
- Problem for more complex structures
  - System of equations is usually under-constrained
  - Multiple solutions



Three unknowns: $\Theta_1, \Theta_2, \Theta_3$
Two equations: x, y

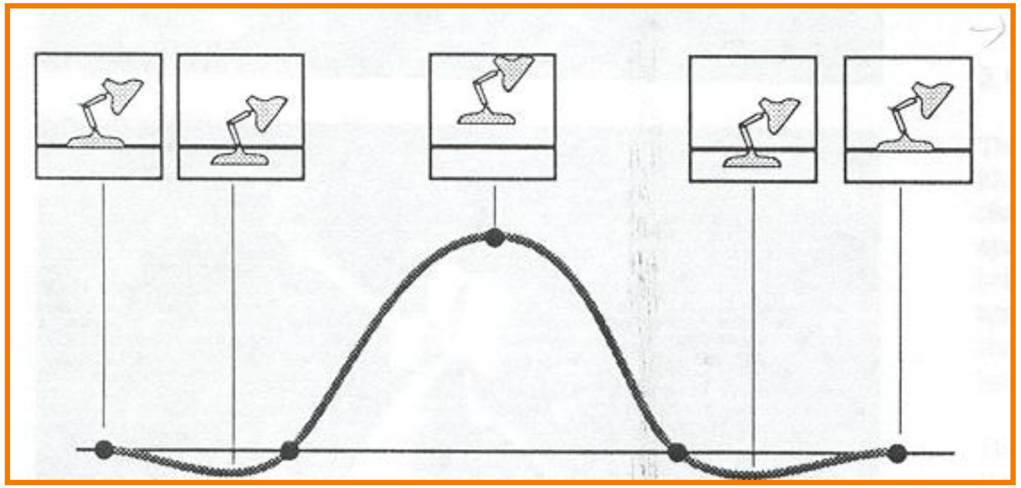
Inverse Kinematics

- Solution for more complex structures:
  - Find best solution (e.g., minimize energy in motion)
  - Non-linear optimization



Kinematics

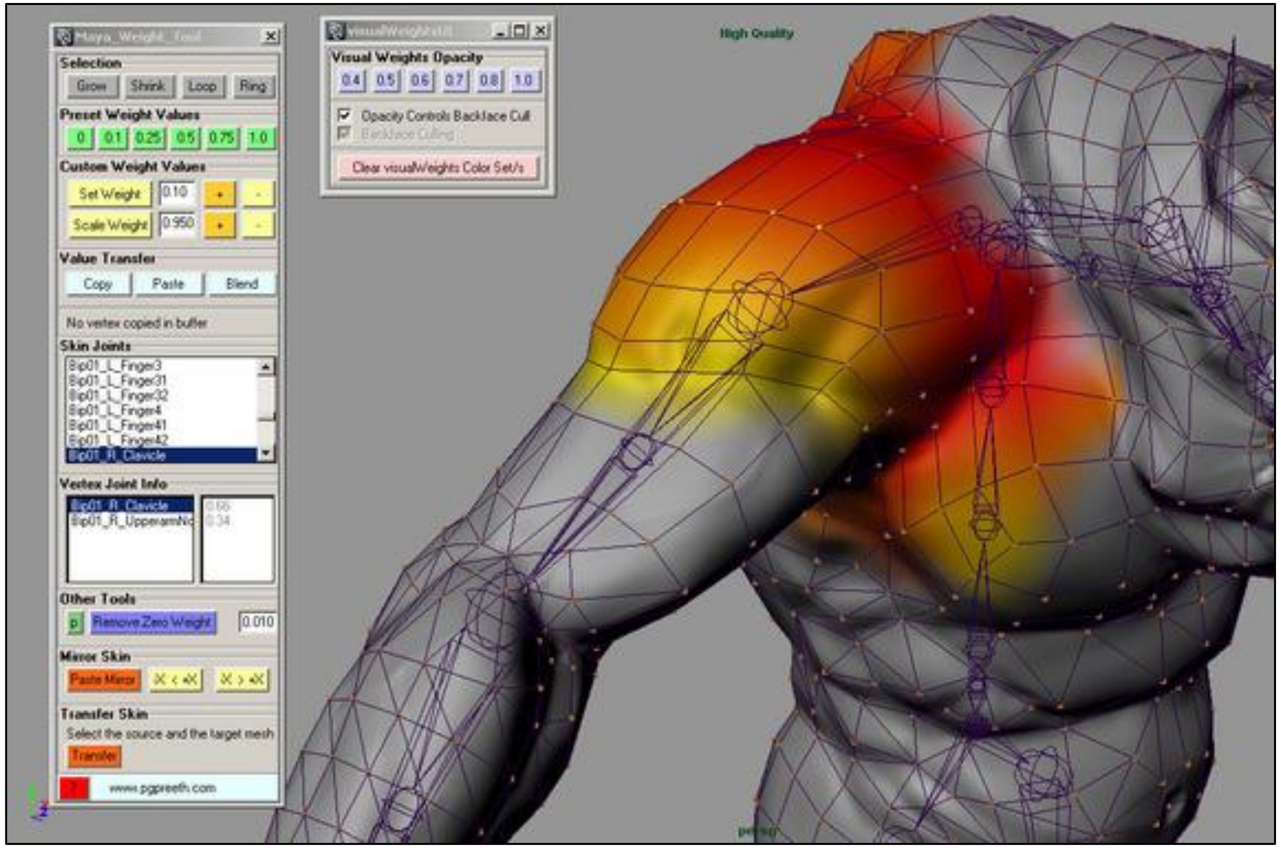
- Advantages
  - Simple to implement
  - Complete animator control
- Disadvantages
  - Motions may not follow physical laws
  - Tedious for animator



Lasseter `87

Beyond Skeletons…

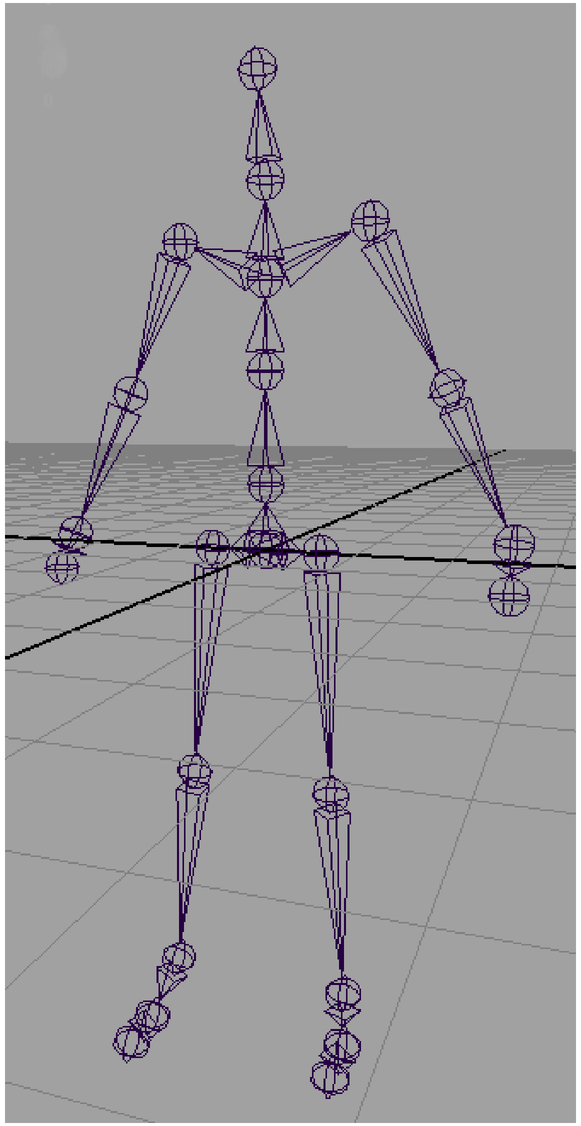
- Skinning



creativecrash.com

Kinematic Skeletons

- Hierarchy of transformations ("bones")
  - Changes to parent affect all descendent bones
- So far: bones affect objects in scene or parts of a mesh
  - Equivalently, each point on a mesh acted upon by one bone
  - Leads to discontinuities when parts of mesh animated
- Extension: each point on a mesh acted upon by more than one bone



Linear Blend Skinning

- Each vertex of skin potentially influenced by all bones
  - Normalized weight vector $w^{(v)}$ gives influence of each bone transform
  - When bones move, influenced vertices also move

- Computing a transformation T_v for a skinned vertex
  - For each bone
    - Compute global bone transformation T_b from transformation hierarchy
  - For each vertex
    - Take a linear combination of bone transforms
    - Apply transformation to vertex in original pose

$$T_v = \sum_{b \in B} w_b^{(v)} T_b$$

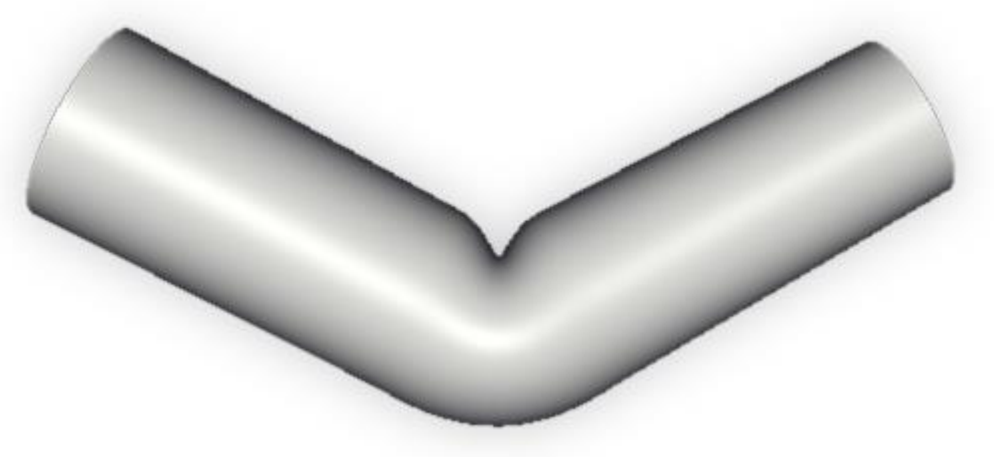
- Equivalently, transformed vertex position is weighted combination of positions transformed by bones

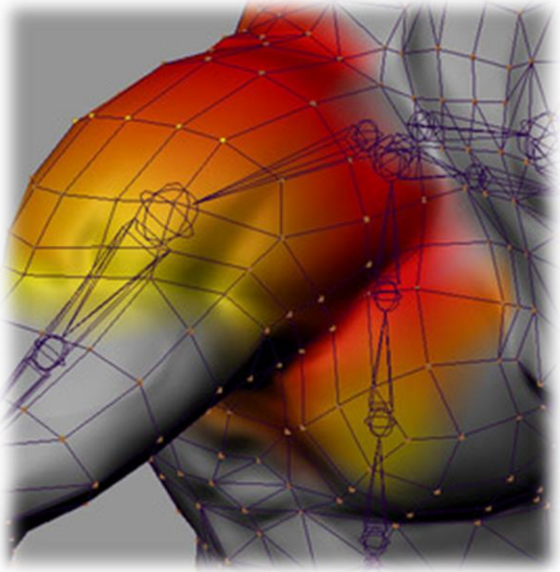
$$v_{transformed} = \sum_{b \in B} w_b^{(v)} \left(T_b v \right)$$

Assigning Weights: “Rigging”

- Painted by hand
- Automatic: function of relative distances to nearest bones

  - Smoothness of skinned surface depends on smoothness of weights!





Assigning Weights: “Rigging”

- Painted by hand
- Automatic: function of relative distances to nearest bones
  - Smoothness of skinned surface depends on smoothness of weights!
  - Other problems with extreme deformations
    - Many solutions

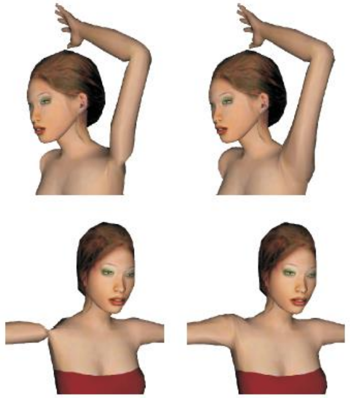
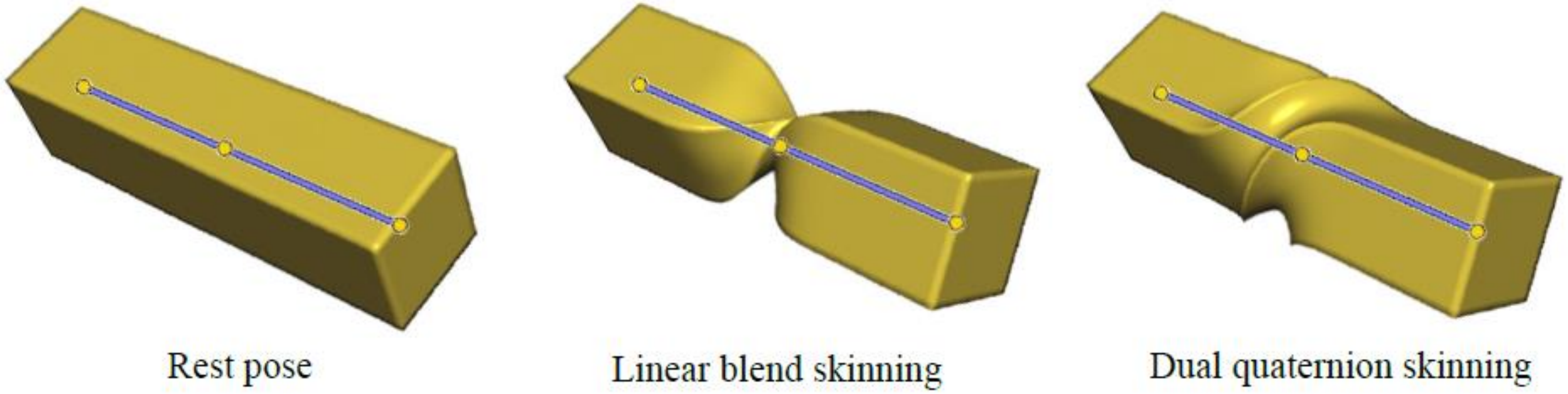


Figure 14: Comparison of linear (left) and dual quaternion (right) blending. Dual quaternions preserve rigidity of input transformations and therefore avoid skin collapsing artifacts.

[Kavan et al. SG '08]



[Kavan, SG'14 course]

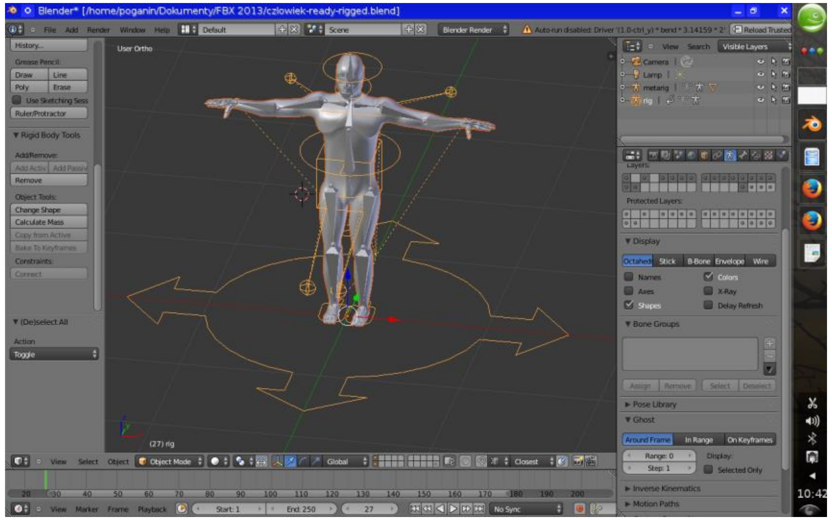
Assigning Weights: “Rigging”

- Painted by hand
- Automatic: function of relative distances to nearest bones
    - Smoothness of skinned surface depends on smoothness of weights!
    - Other problems with extreme deformations

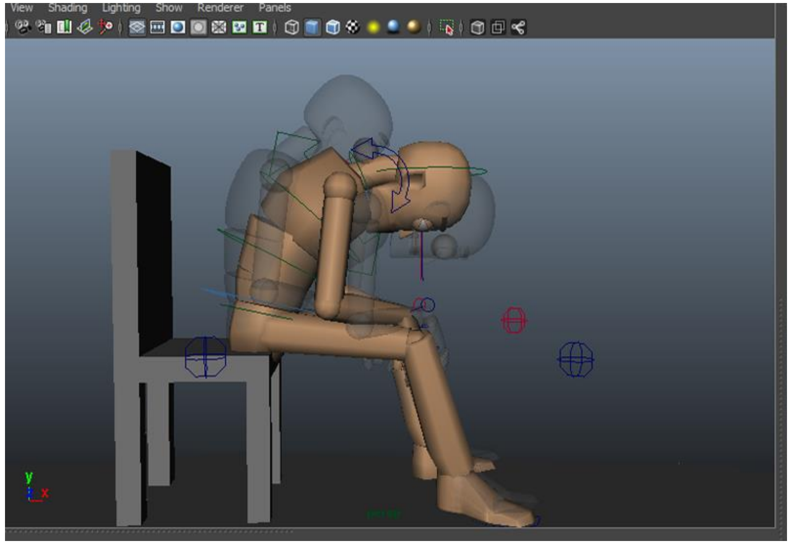
Oztireli2013: https://graphics.ethz.ch/publications/papers/paperOzt13.php

Character Animation Methods

- Modeling (manipulation)
  - Deformation
  - Blendshapes
  - Skeletons
- Interpolation
  - Key-framing
  - Kinematics
  - Motion Capture



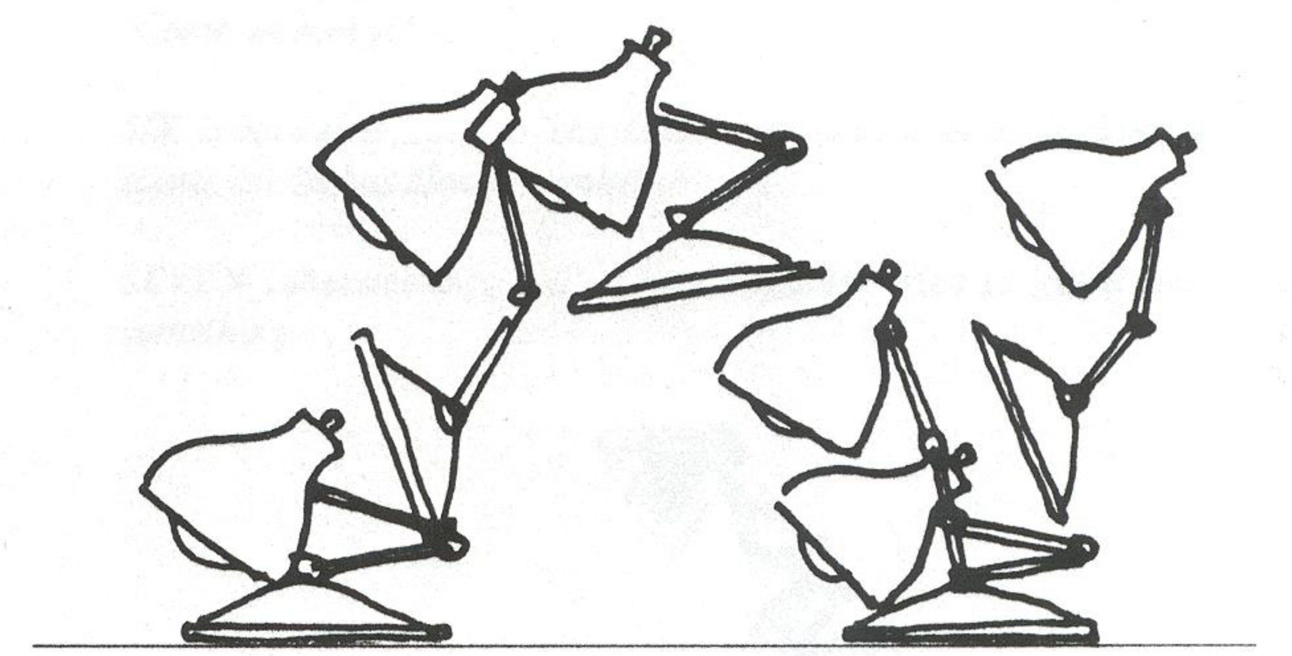
https://blenderartists.org/



focus.gscept.com

Keyframe Animation

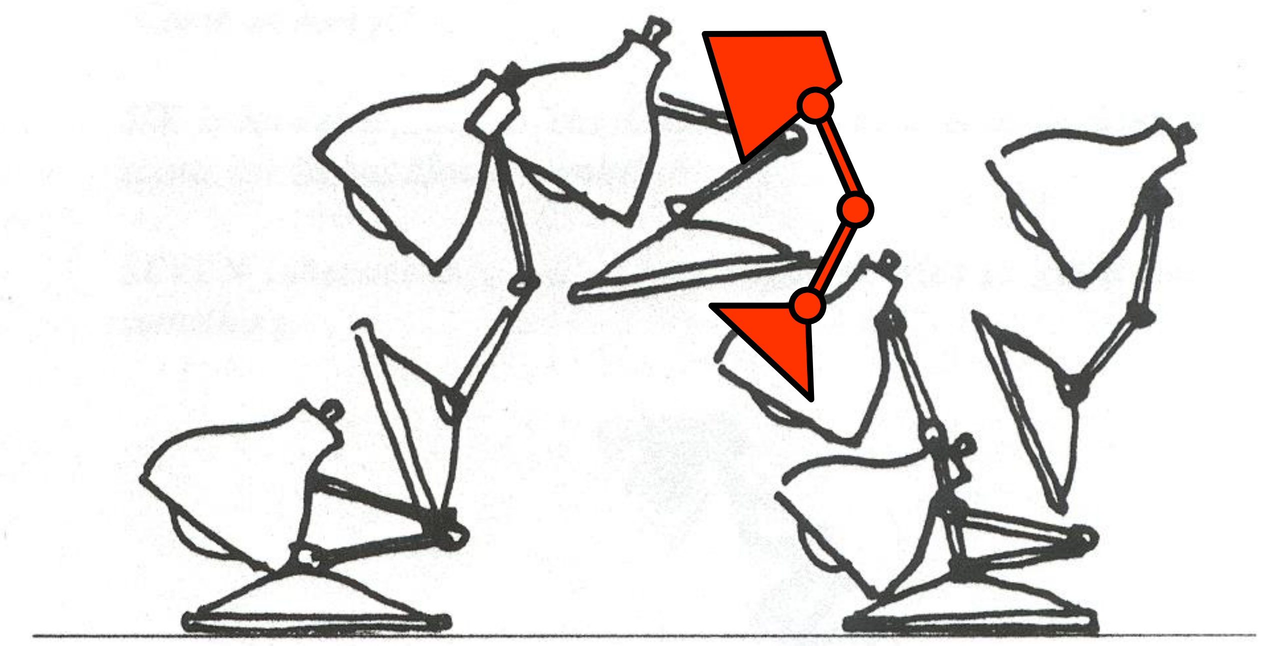
- Define character poses at specific time steps called “keyframes”



Lasseter `87

Keyframe Animation

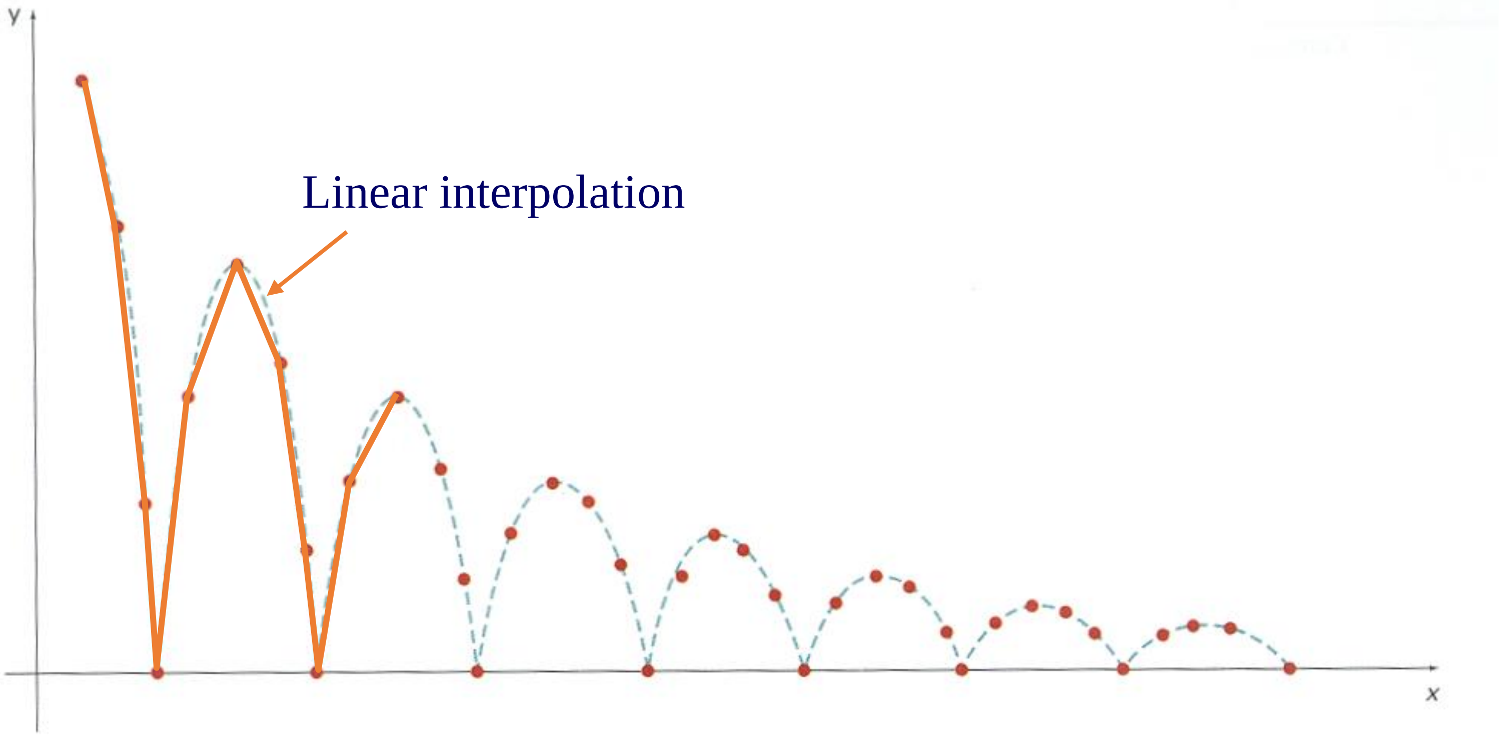
- Interpolate variables describing keyframes to determine poses for character in between



Lasseter `87

Keyframe Animation

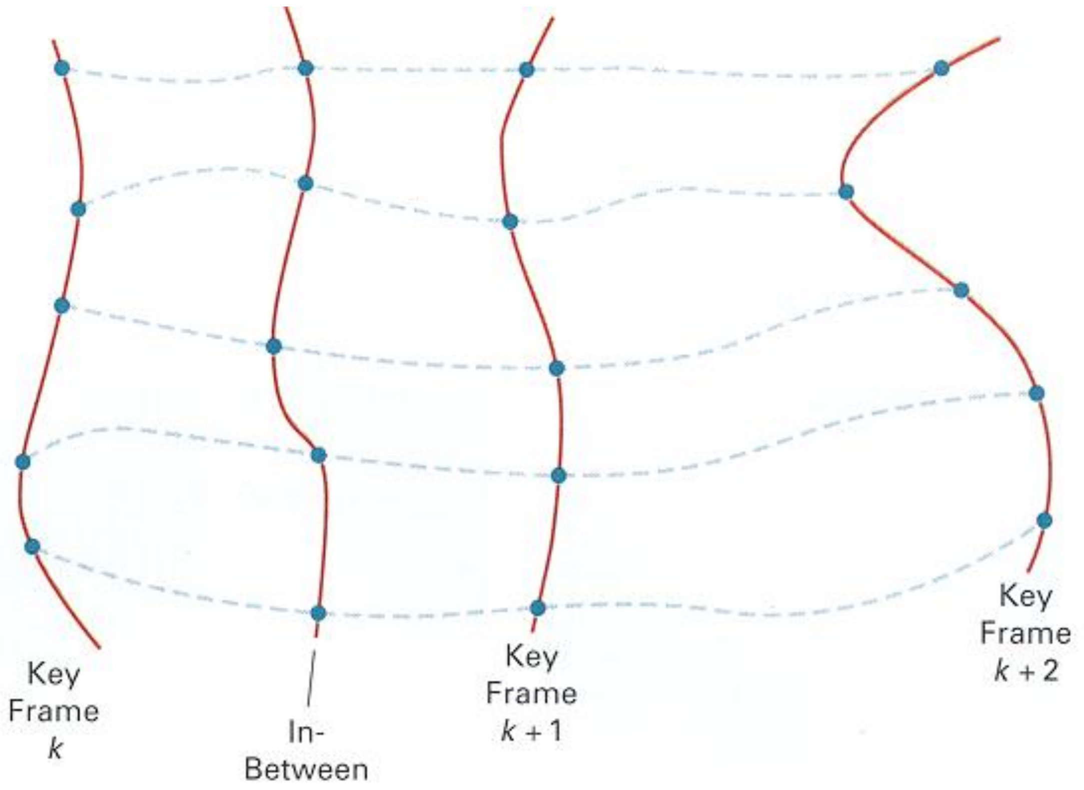
- Inbetweening:
  - Linear interpolation - usually not enough continuity



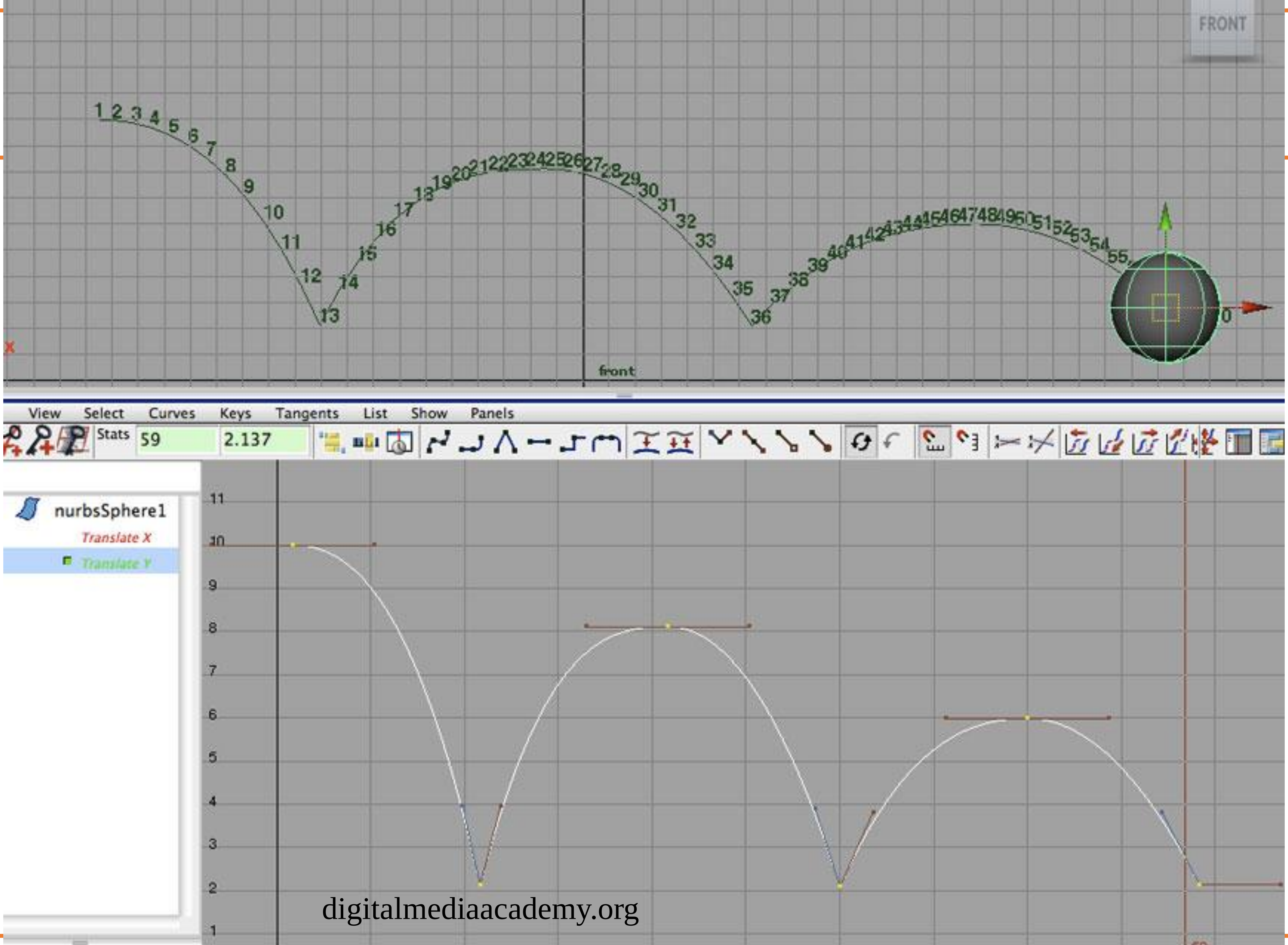
H&B Figure 16.16

Keyframe Animation

- Inbetweening:
  - Spline interpolation - maybe good enough



H&B Figure 16.11

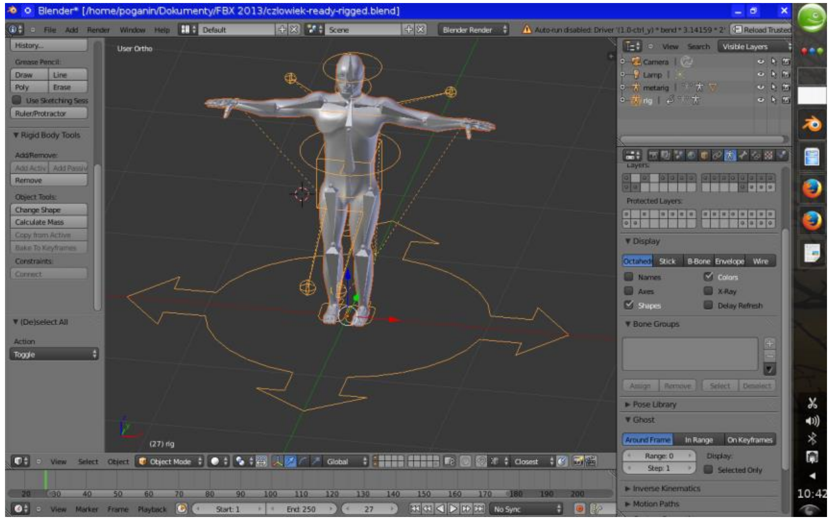


digitalmediaacademy.org

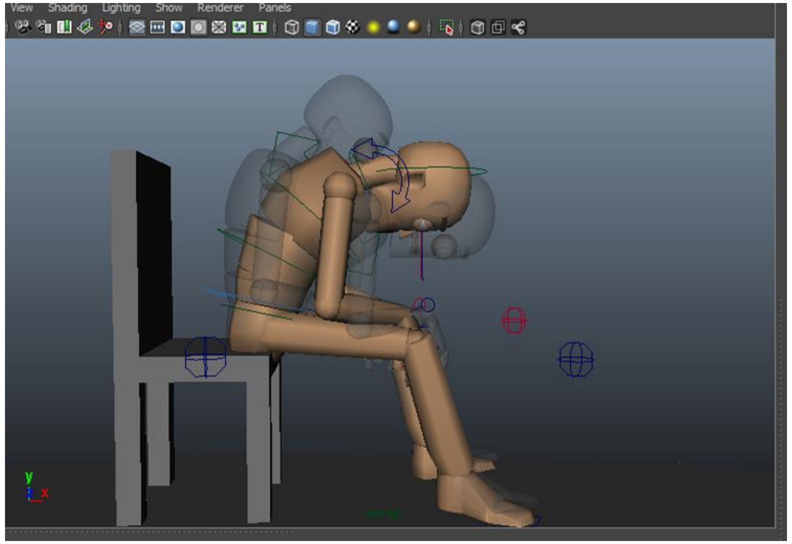
Character Animation Methods

- Modeling (manipulation)
  - Deformation
  - Blendshapes
  - Skeletons

- Interpolation
  - Key-framing
  - Kinematics
  - Motion Capture



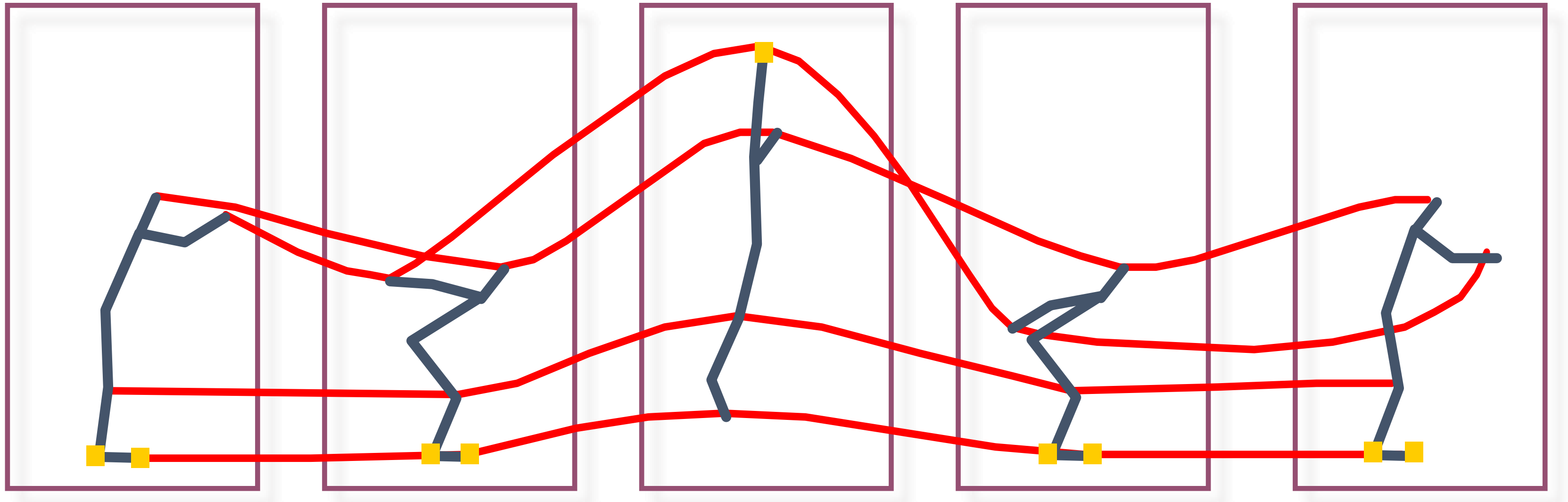
https://blenderartists.org/



focus.gscept.com

Motion Capture

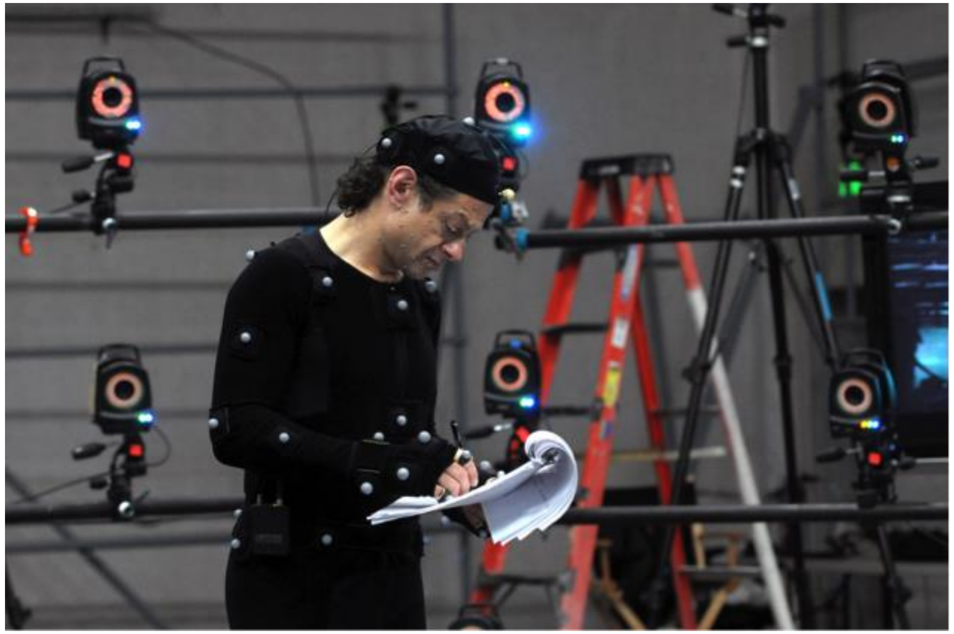
- Measure motion of real characters and then simply “play it back” with kinematics



Captured Motion

Motion Capture

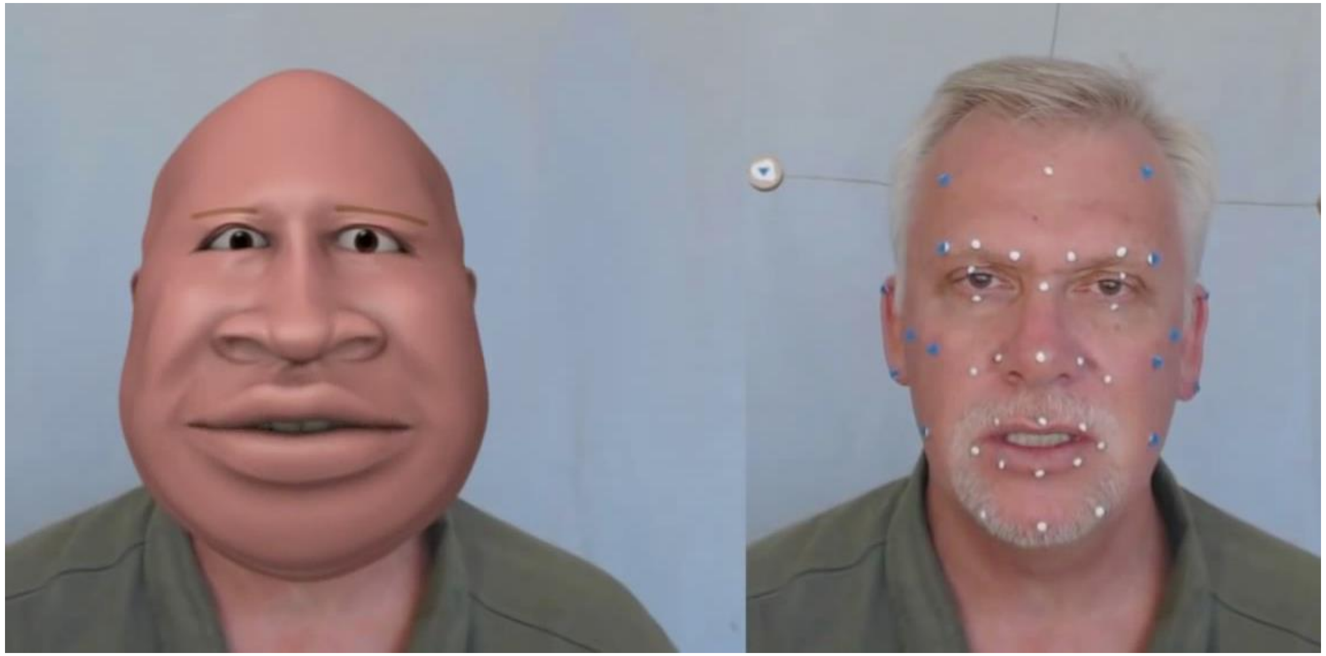
- Measure human motion
- Play back with kinematics



https://www.youtube.com/watch?v=MVvDw15-3e8

Motion Capture

- Could be applied on different parameters
  - Skeleton Transformations
  - Direct mesh deformation
- Advantage:
  - Physical realism
- Challenge:
  - Animator control



Summary

- Kinematics
  - **Animator specifies poses (joint angles or positions) at keyframes** and computer determines motion by kinematics and **interpolation**
- Dynamics
  - Animator specifies **physical attributes**, constraints, and starting conditions and computer determines motion by **physical simulation**
- Motion Capture
  - Compute **captures motion of real character** and provides tools for animator to edit it