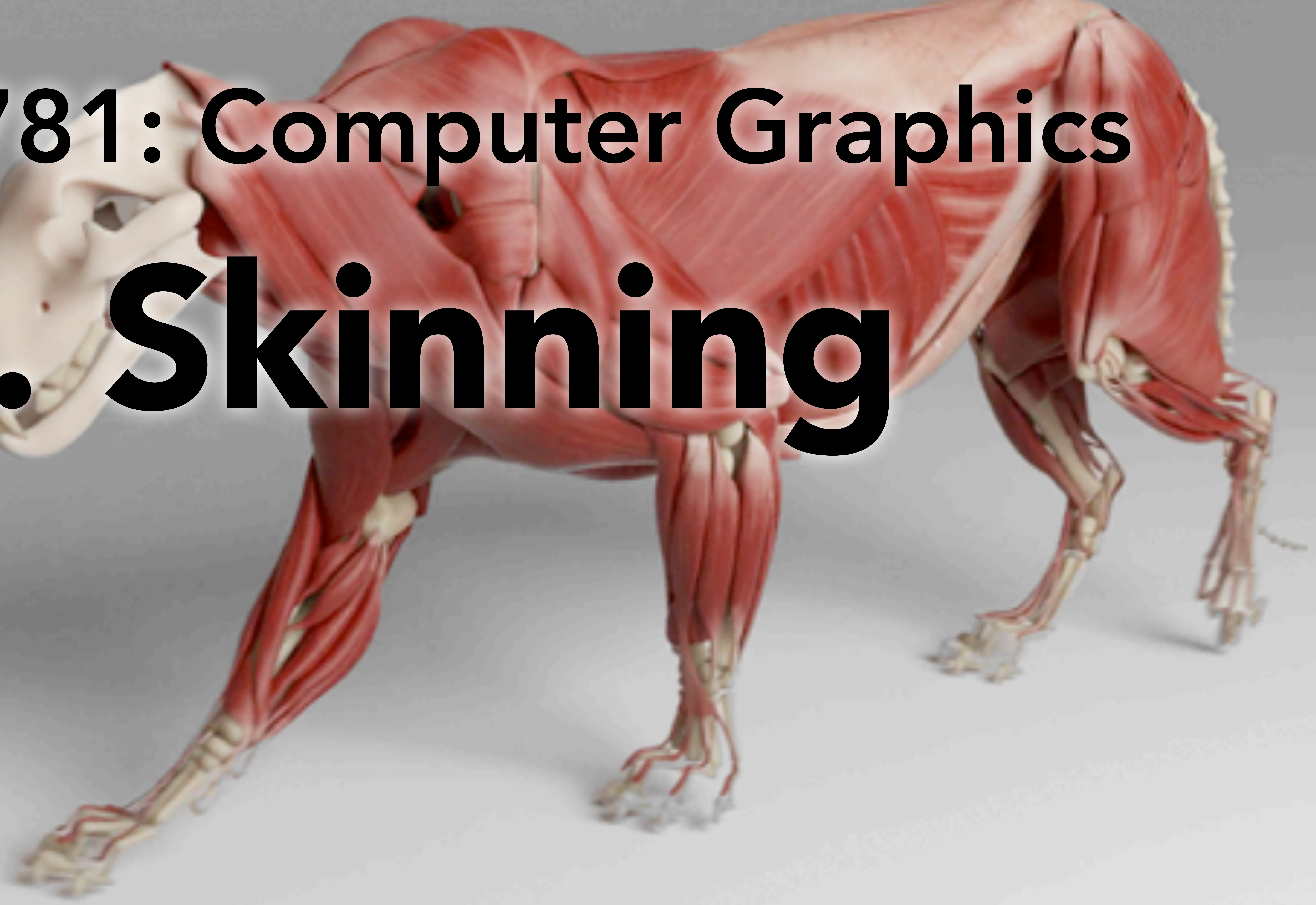
COL781: Computer Graphics

32. Skinning



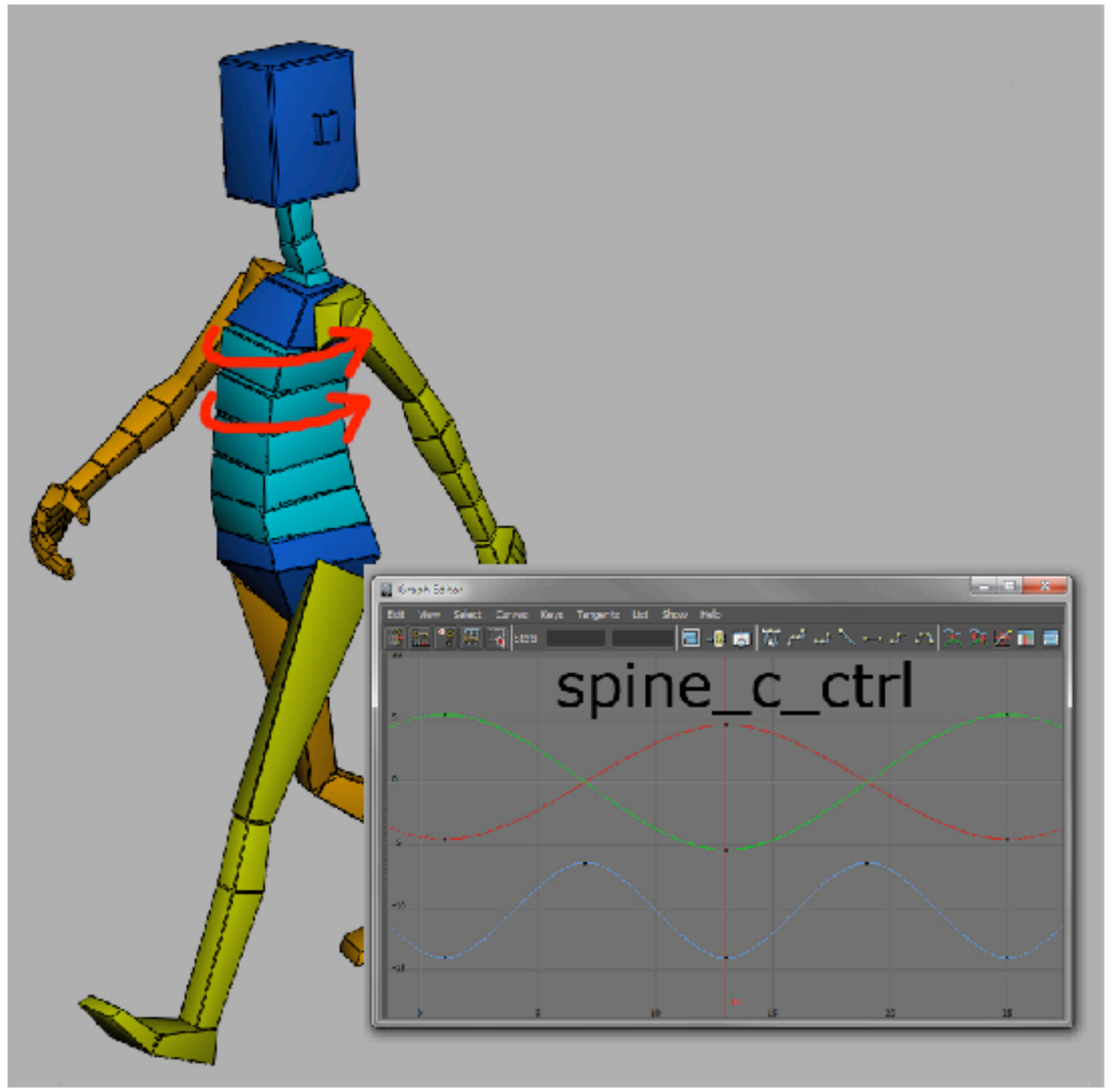
Ziva Dynamics

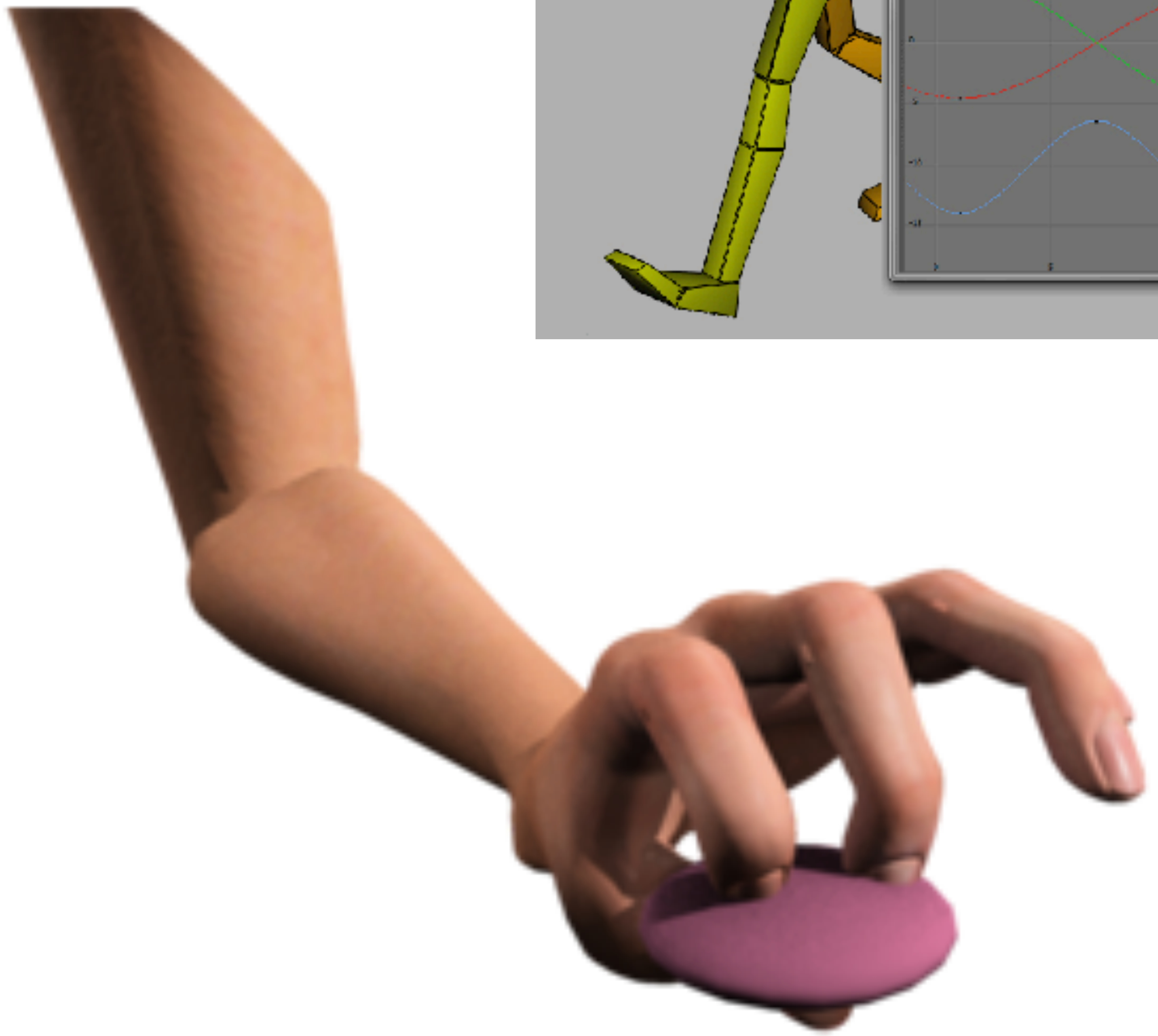
Recap: Skeletal animation

We have a set of animation controls that determine the character's pose:

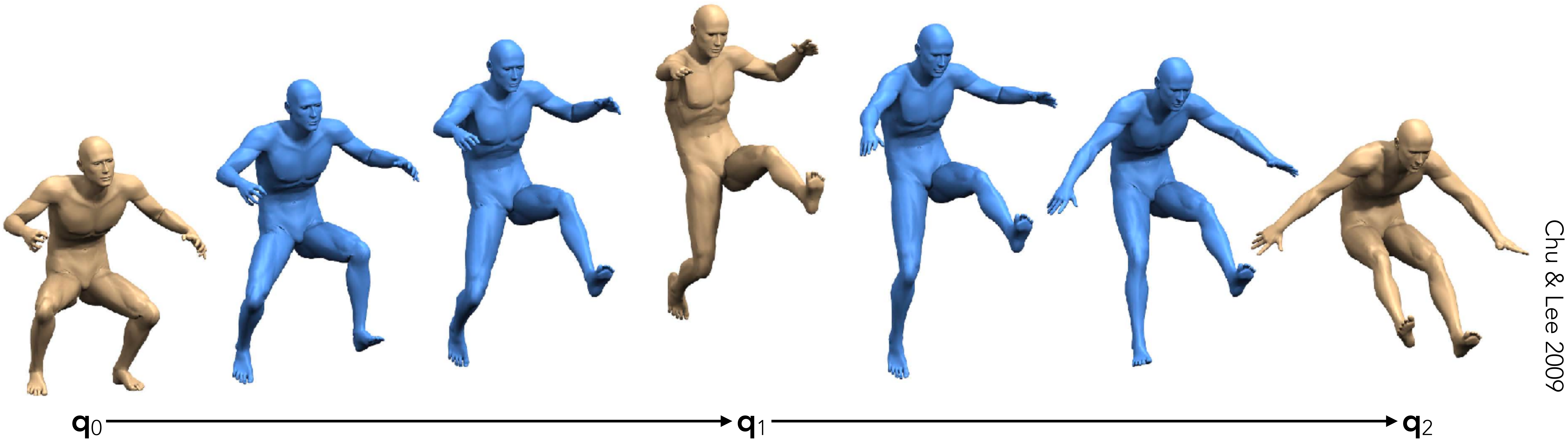
- Forward kinematics: joint angles
- Inverse kinematics: end effector transformations

Based on these we can compute the transformations of all the body parts.





Keyframe animation



Animator specifies character pose (i.e. values of animation controls) at specific keyframes.

How to interpolate to arbitrary times?

Recall **splines**: piecewise polynomial functions with some continuity/differentiability

- Piecewise linear interpolation

$$q(t) = \frac{t_{i+1} - t}{t_{i+1} - t_i} q_i + \frac{t - t_i}{t_{i+1} - t_i} q_{i+1}$$

- Cubic Hermite spline:
  assume $q(t) = at^3 + bt^2 + ct + d$, solve for coefficients so that

$$q(t_i) = q_i,\ q(t_{i+1}) = q_{i+1},$$
$$q'(t_i) = m_i,\ q'(t_{i+1}) = m_{i+1}$$

Closed-form solution:

$$q(t) = (2t^3-3t^2+1)q_i + (t^3-2t^2+t)m_i + (-2t^3+3t^2)q_{i+1} + (t^3-t^2)m_{i+1}$$

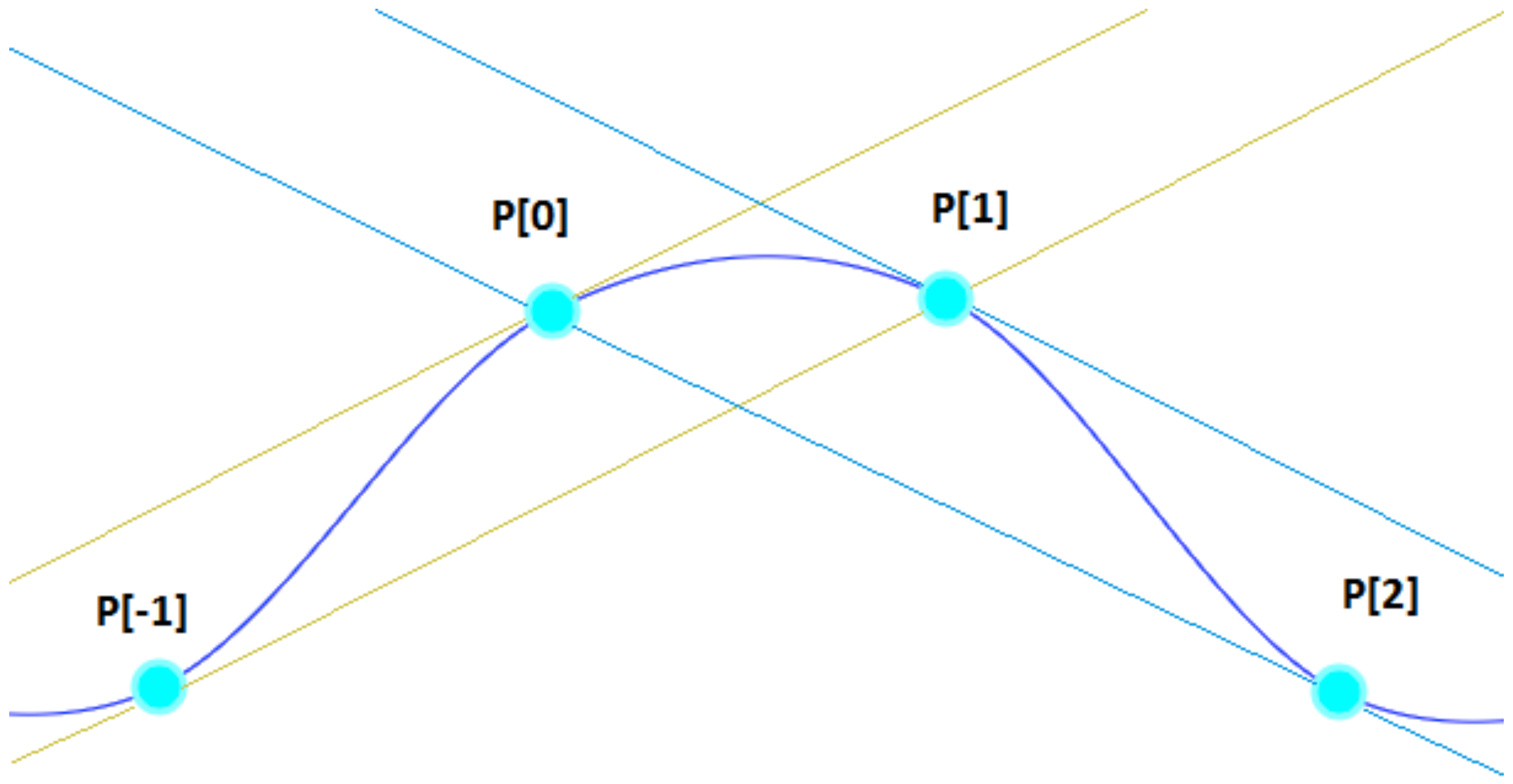
What if derivatives are not given, but still want a C^1 curve?

Catmull-Rom splines

Estimate derivatives from neighbouring points, e.g.:

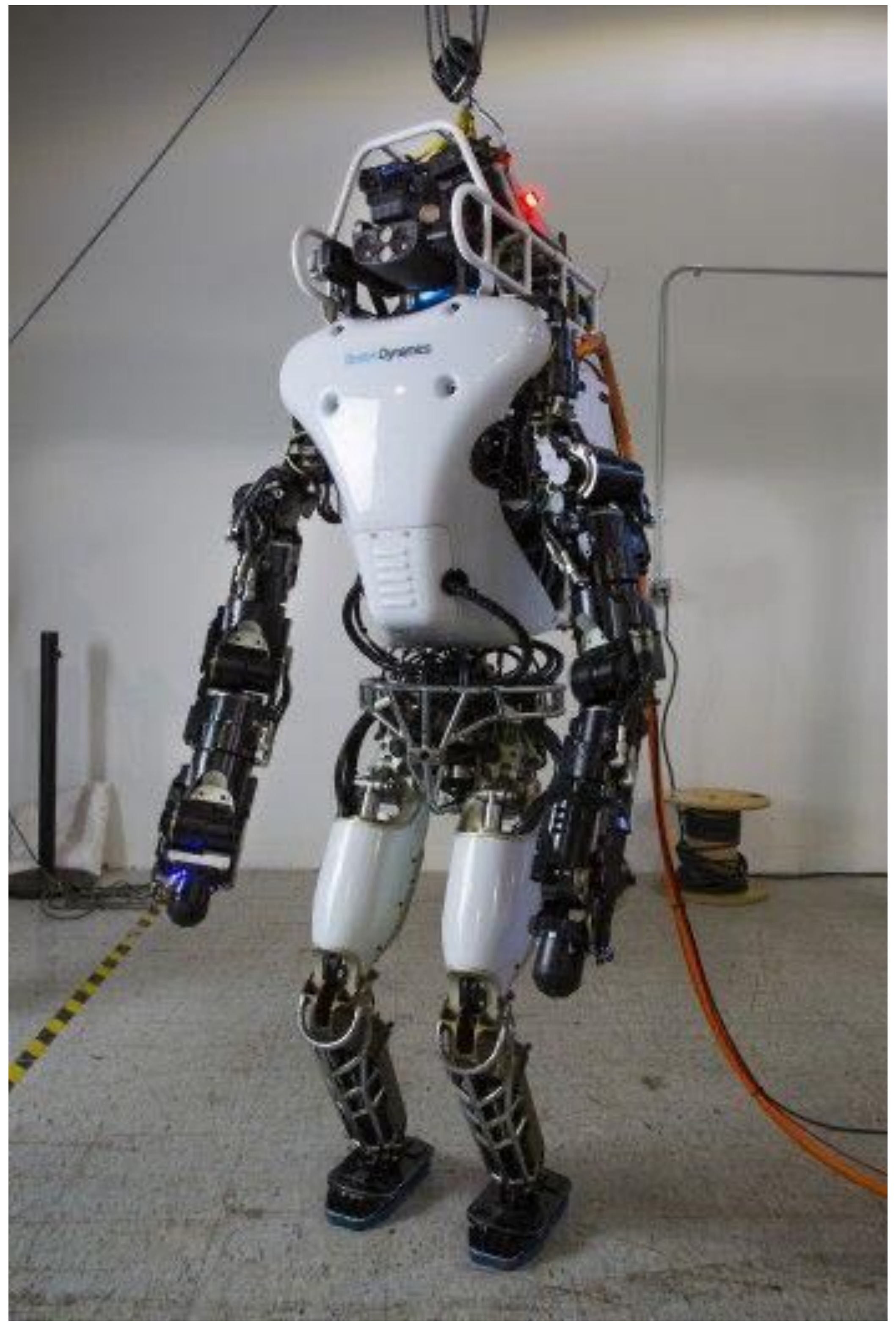
$$m_i = \frac{q_{i+1} - q_{i-1}}{t_{i+1} - t_{i-1}}$$

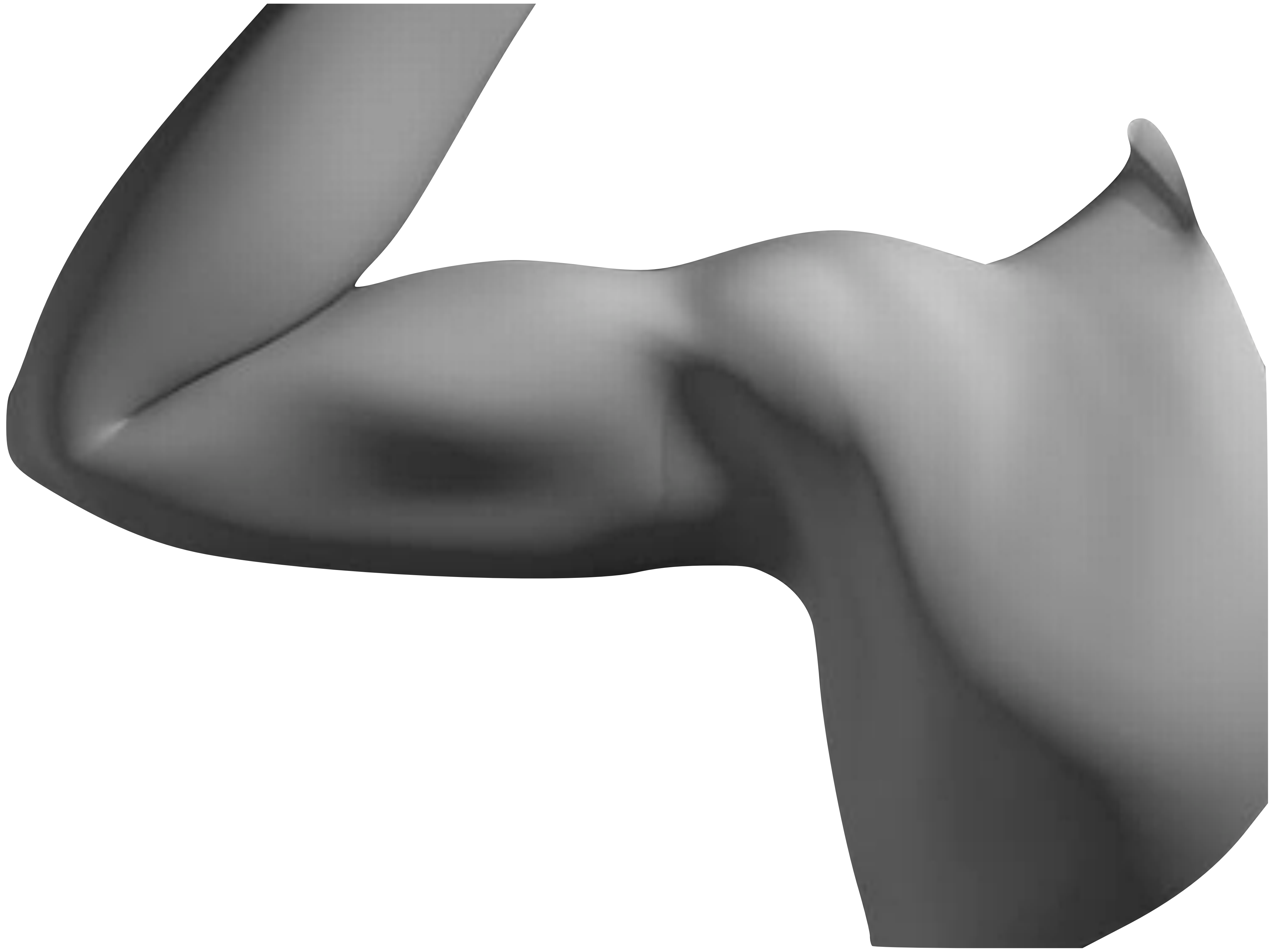
In fact, any estimate that only uses data at points $i-1$, i, $i+1$ will give C^1 continuity. (Why?)



Rigid bone transformations may be sufficient for robots and toys with rigid parts.

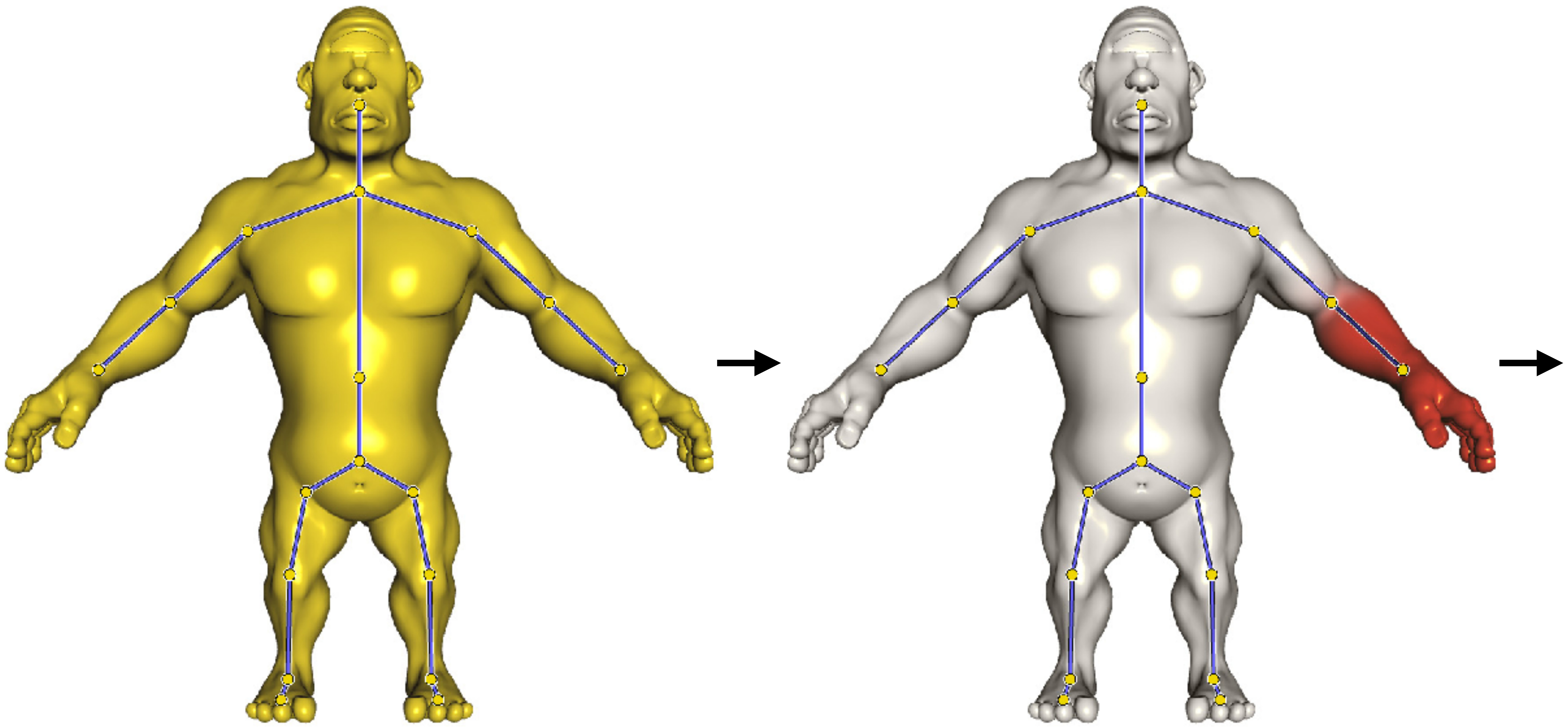
What about organic characters?





J.P. Lewis

Skinning



Skeleton → Skinning weights → Deformed shape

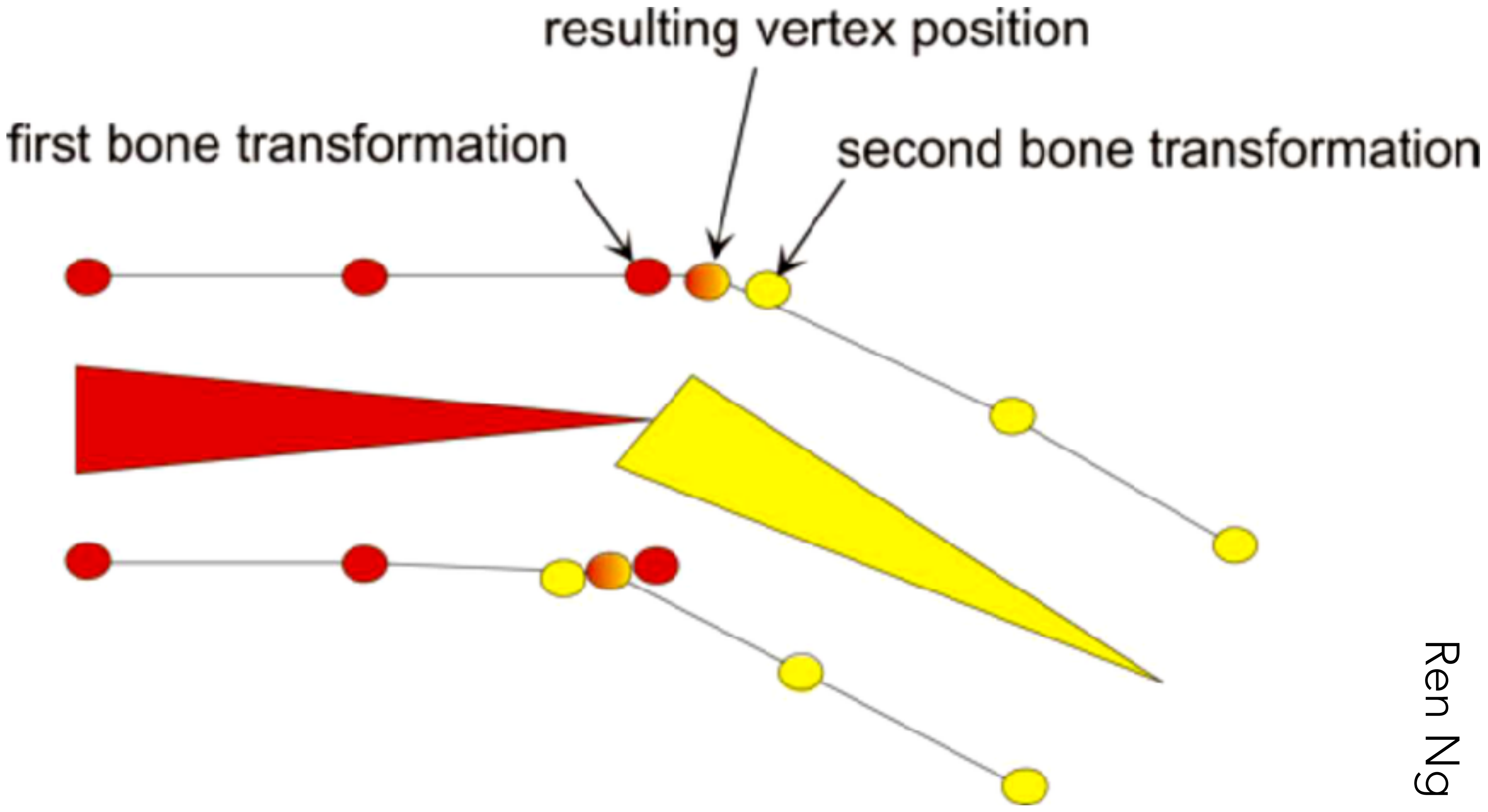
Alec Jacobson

Each vertex $\mathbf{v}_i$ may be affected by transformations from multiple bones.

Linear blend skinning: Final position is weighted average

$$\mathbf{v}_i' = \sum_{\text{bone } j} w_{ij} \mathbf{T}_j \begin{bmatrix} \mathbf{v}_i \\ 1 \end{bmatrix}$$

Of course, for a weighted average, we should have $\sum_j w_{ij} = 1$ for each vertex



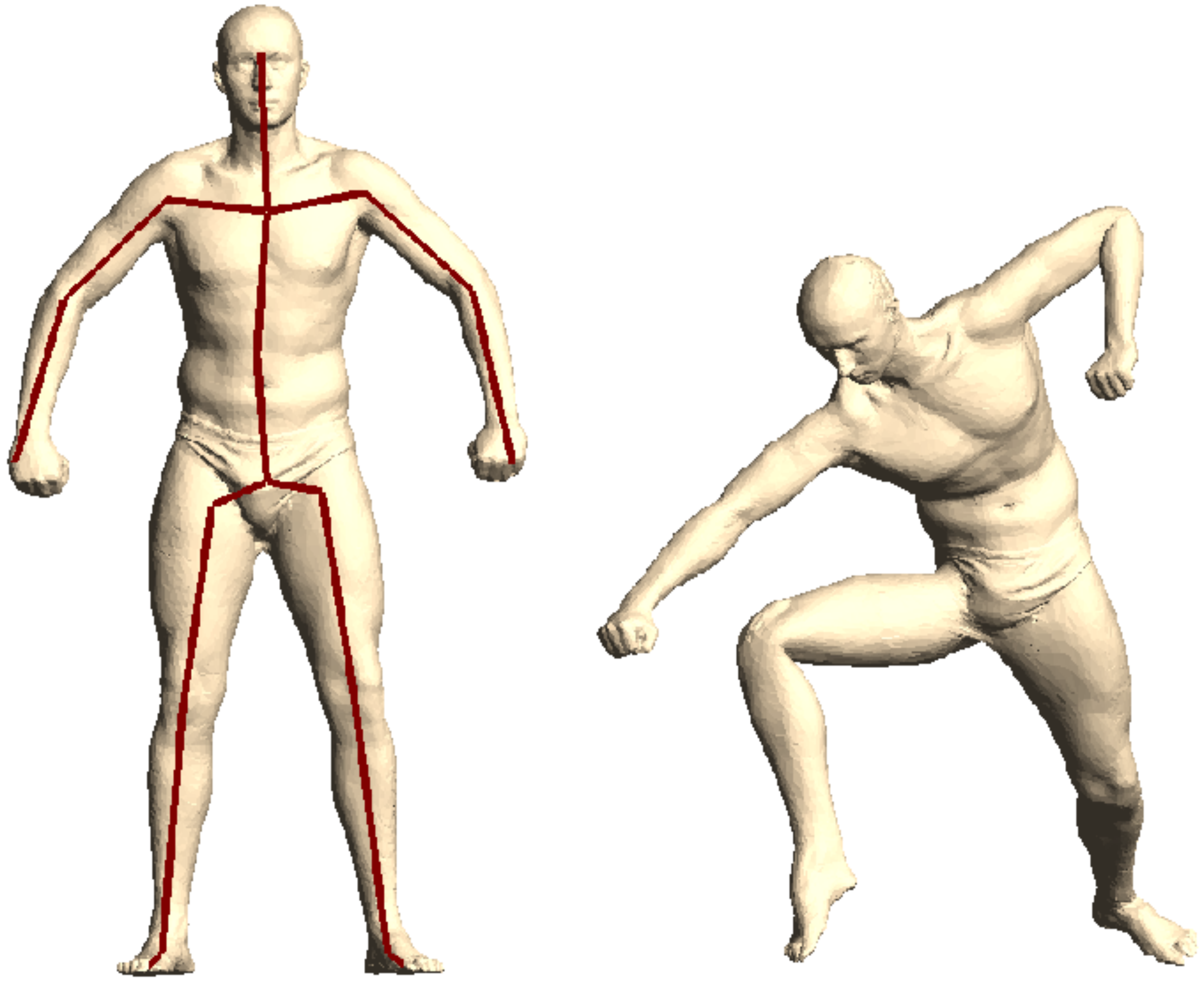
Ren Ng

Wait, to apply bone transformation $\mathbf{T}_j$, we need to have vertex $\mathbf{v}_i$ in the bone's coordinate frame…

$$\mathbf{v}_i' = \sum_{\text{bone } j} w_{ij} \mathbf{T}_j \mathbf{B}_j^{-1} \begin{bmatrix} \mathbf{v}_i \\ 1 \end{bmatrix}$$

- $\mathbf{B}_j$: bone transformation in bind pose
- $\mathbf{T}_j$: bone transformation in deformed pose

From now on, let's just call the product $\mathbf{T}_j$

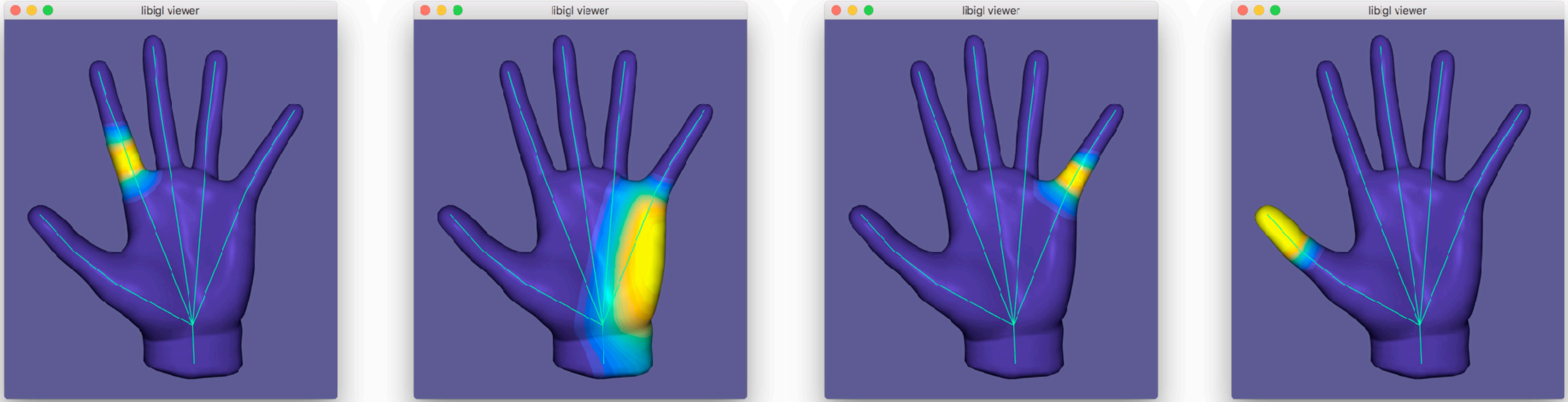


Bind pose

Deformed pose

Baran & Popovic 2007

Example: Skinning weights



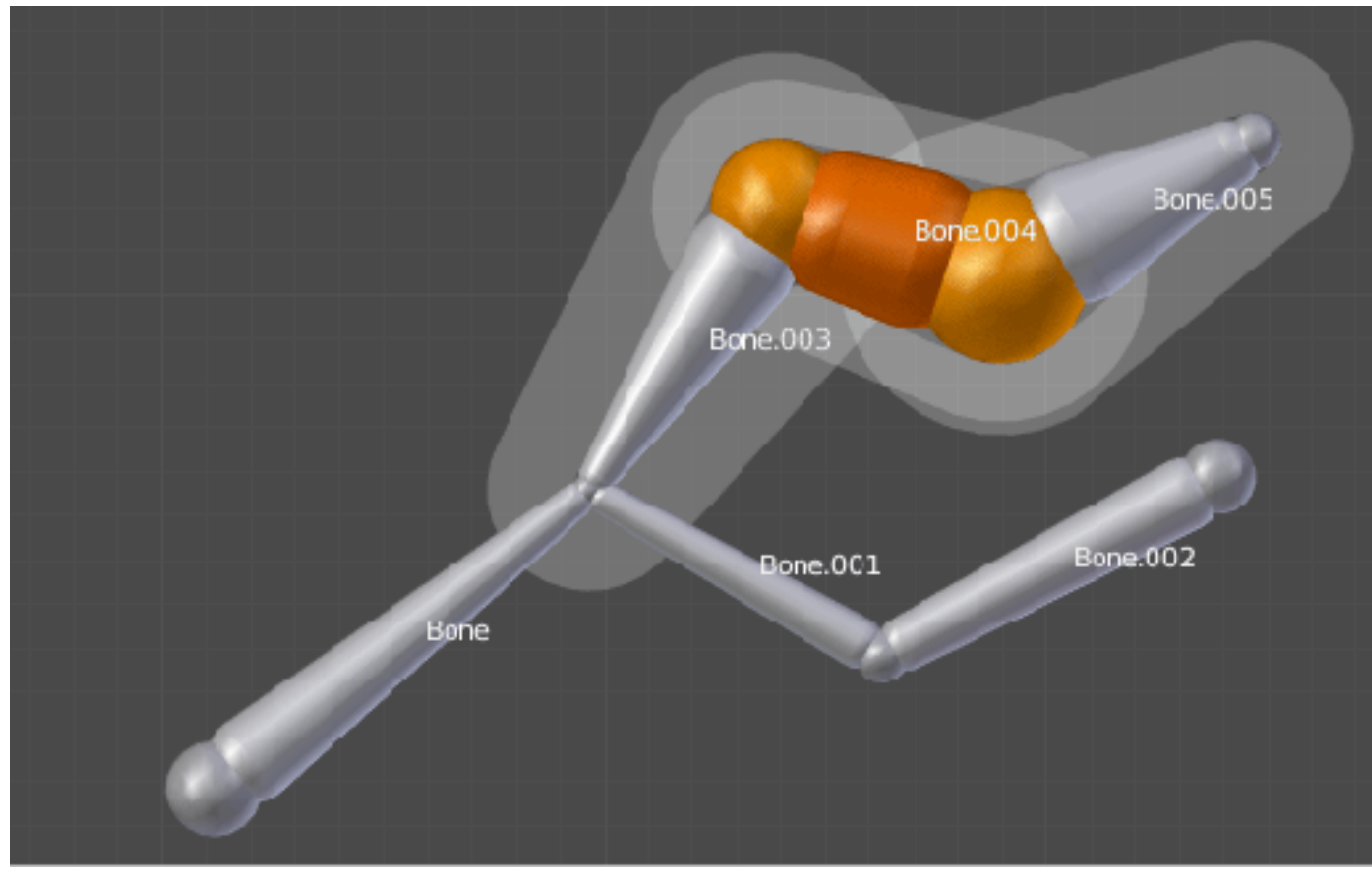
libigl tutorial

How to get the weights?

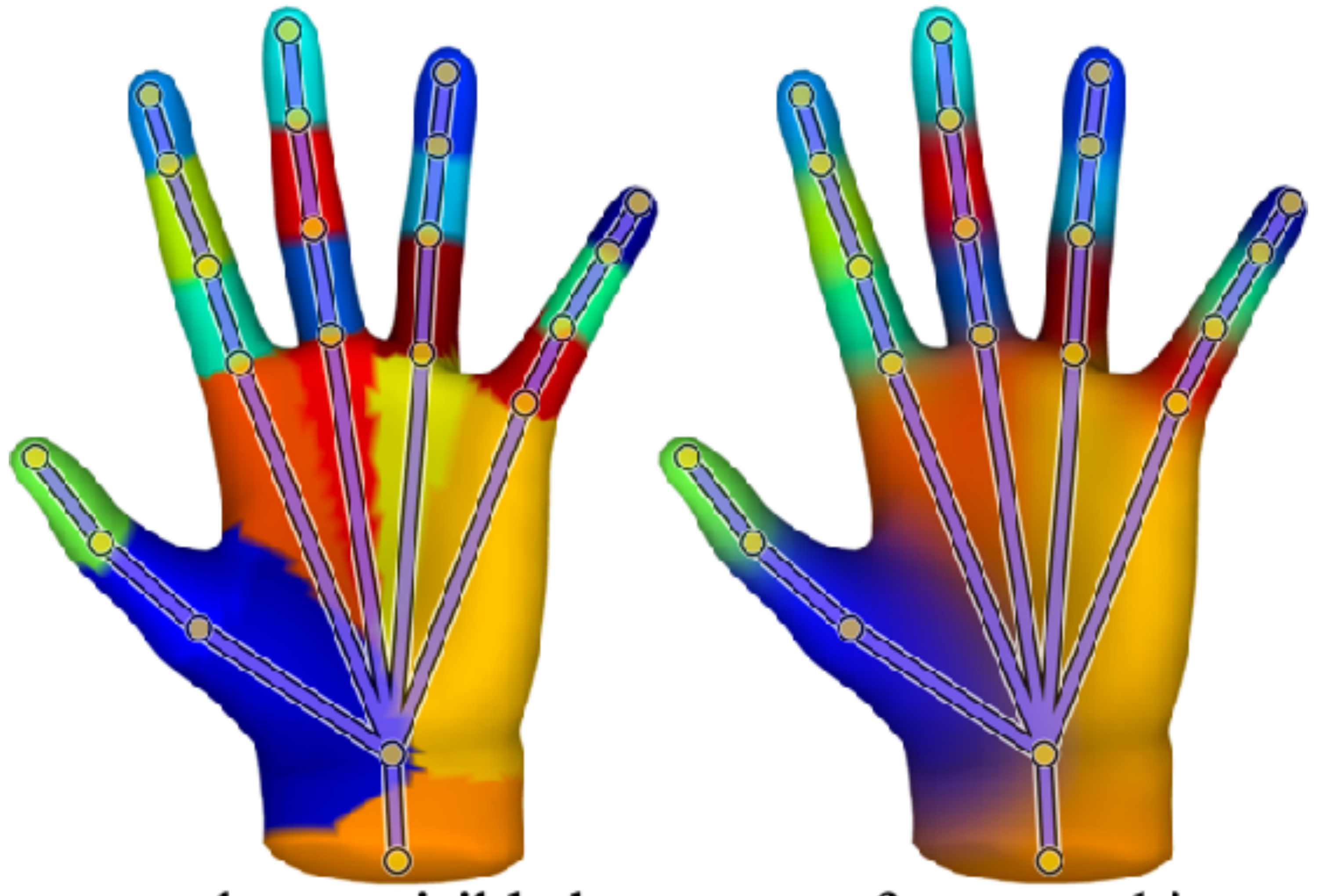
One common way: User paints them manually on the mesh!

Various automatic methods:

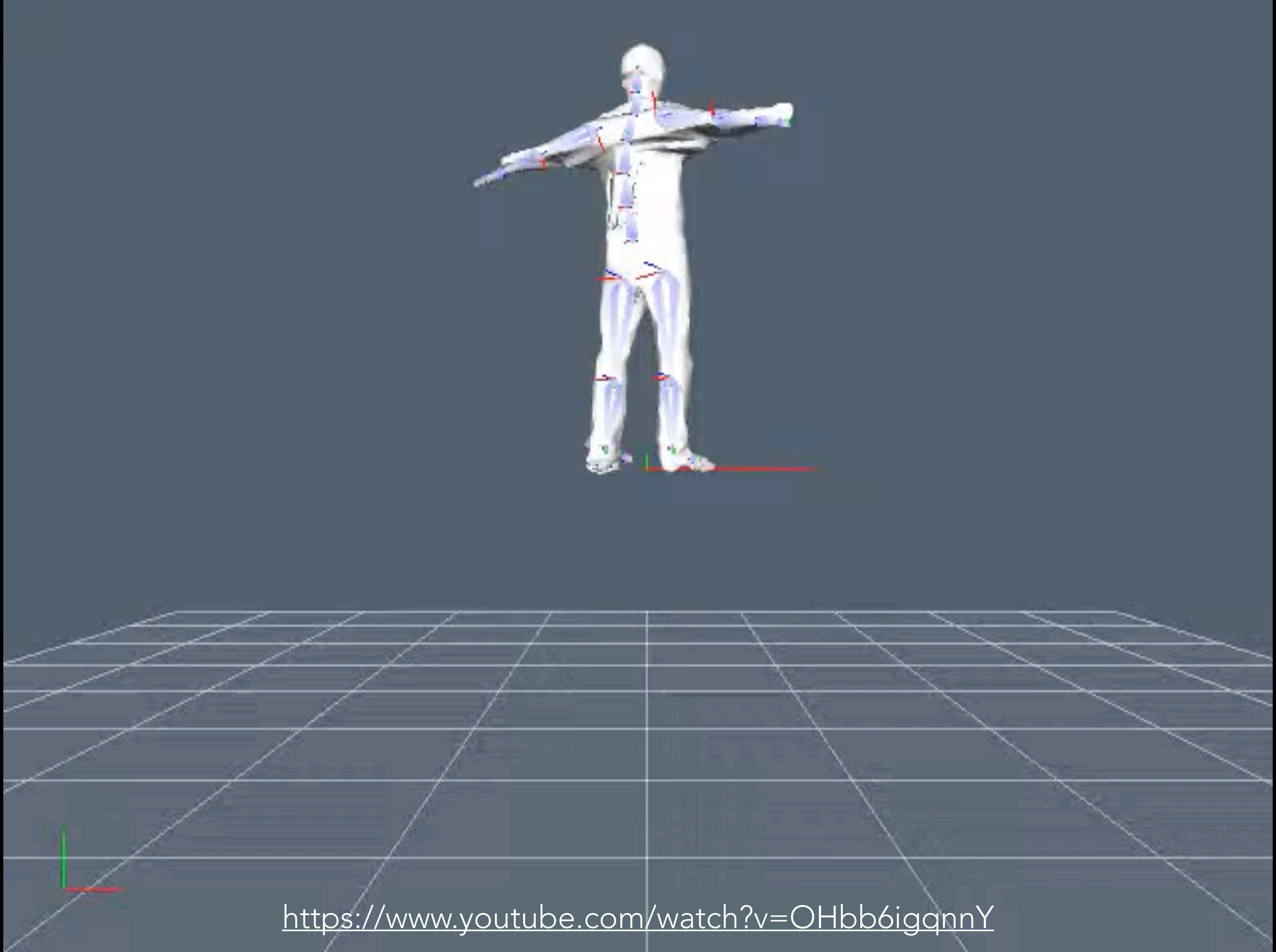
- **Envelopes**: manually specified region of influence
- **Bone heat** (Baran & Popovic 2007): assign vertices to closest visible bone, then apply smoothing (heat diffusion)
- Weights optimized for smoothness, locality, monotonicity, etc., e.g. bounded biharmonic weights (Jacobson et al. 2011)



Blender manual

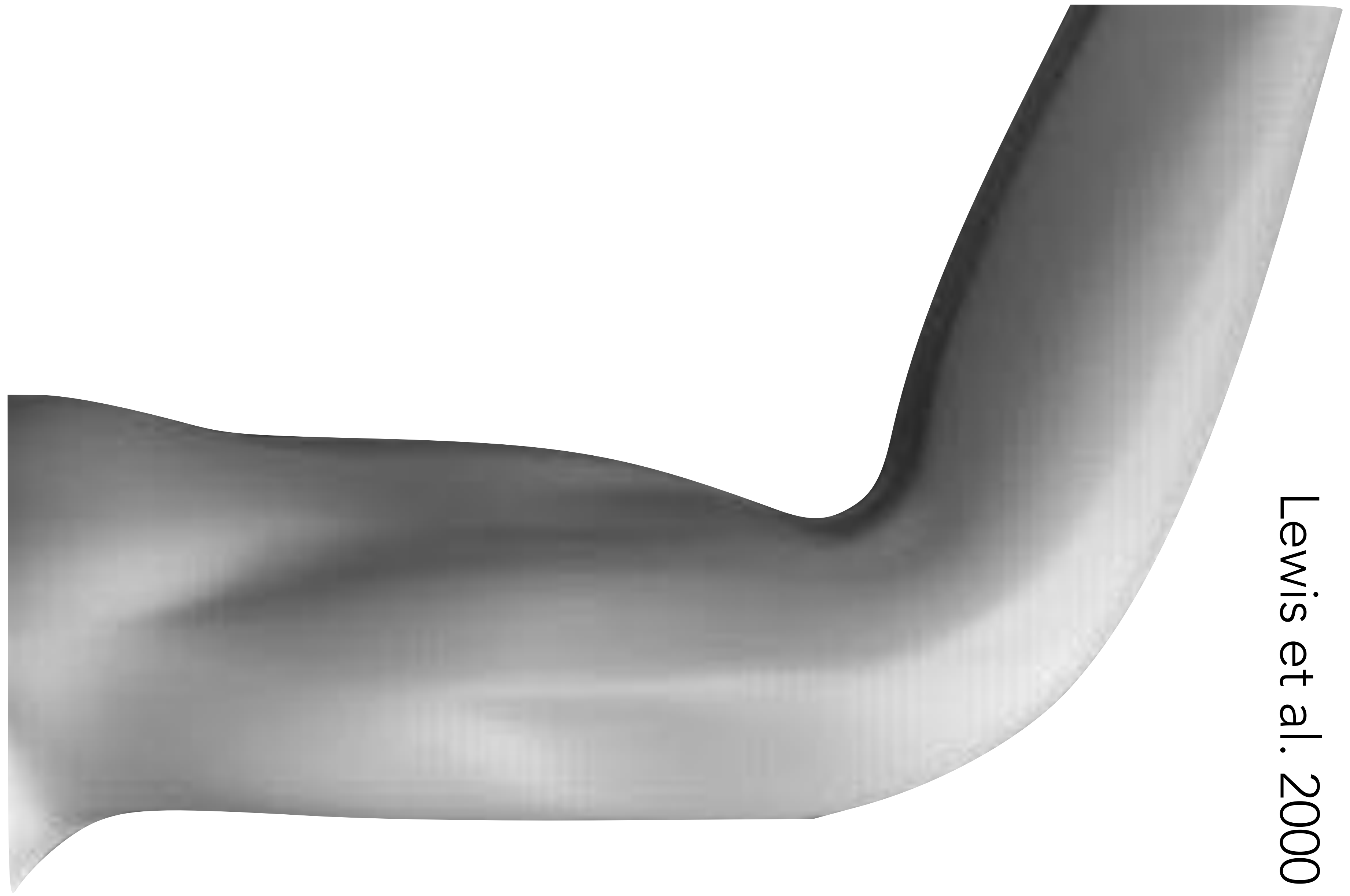


Jacobson 2014



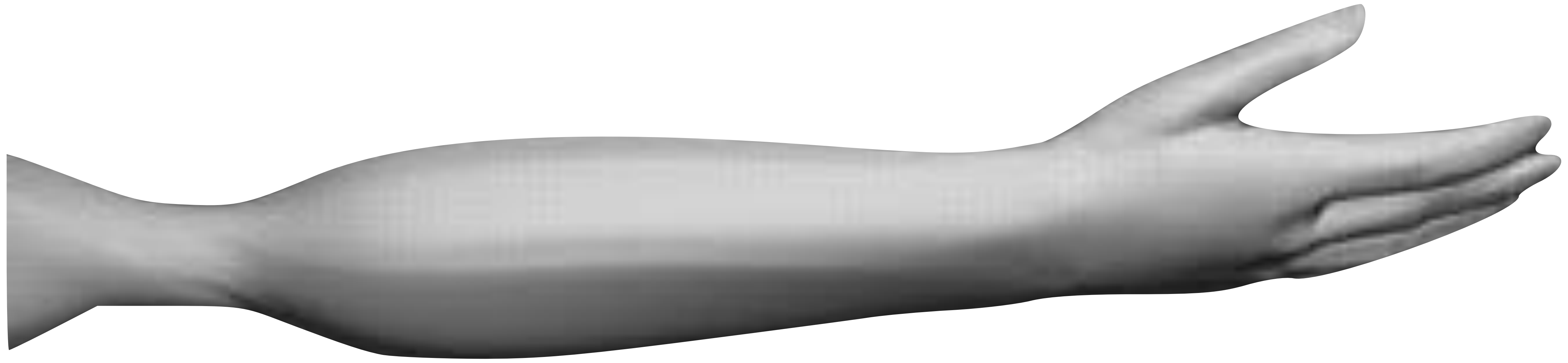
https://www.youtube.com/watch?v=OHbb6igqnnY

harrumphoid



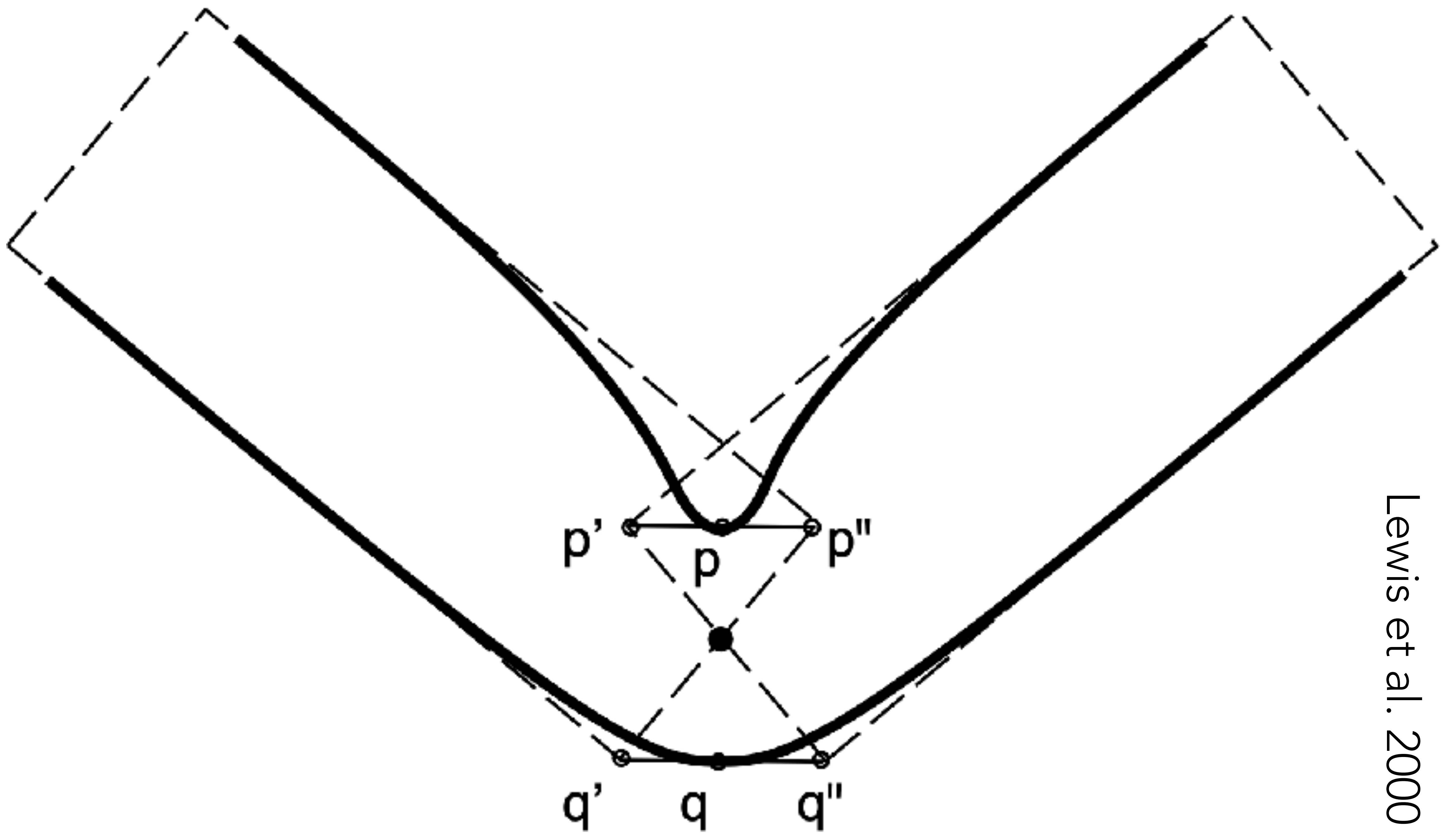
Lewis et al. 2000

Joint collapse



Candy wrapper effect

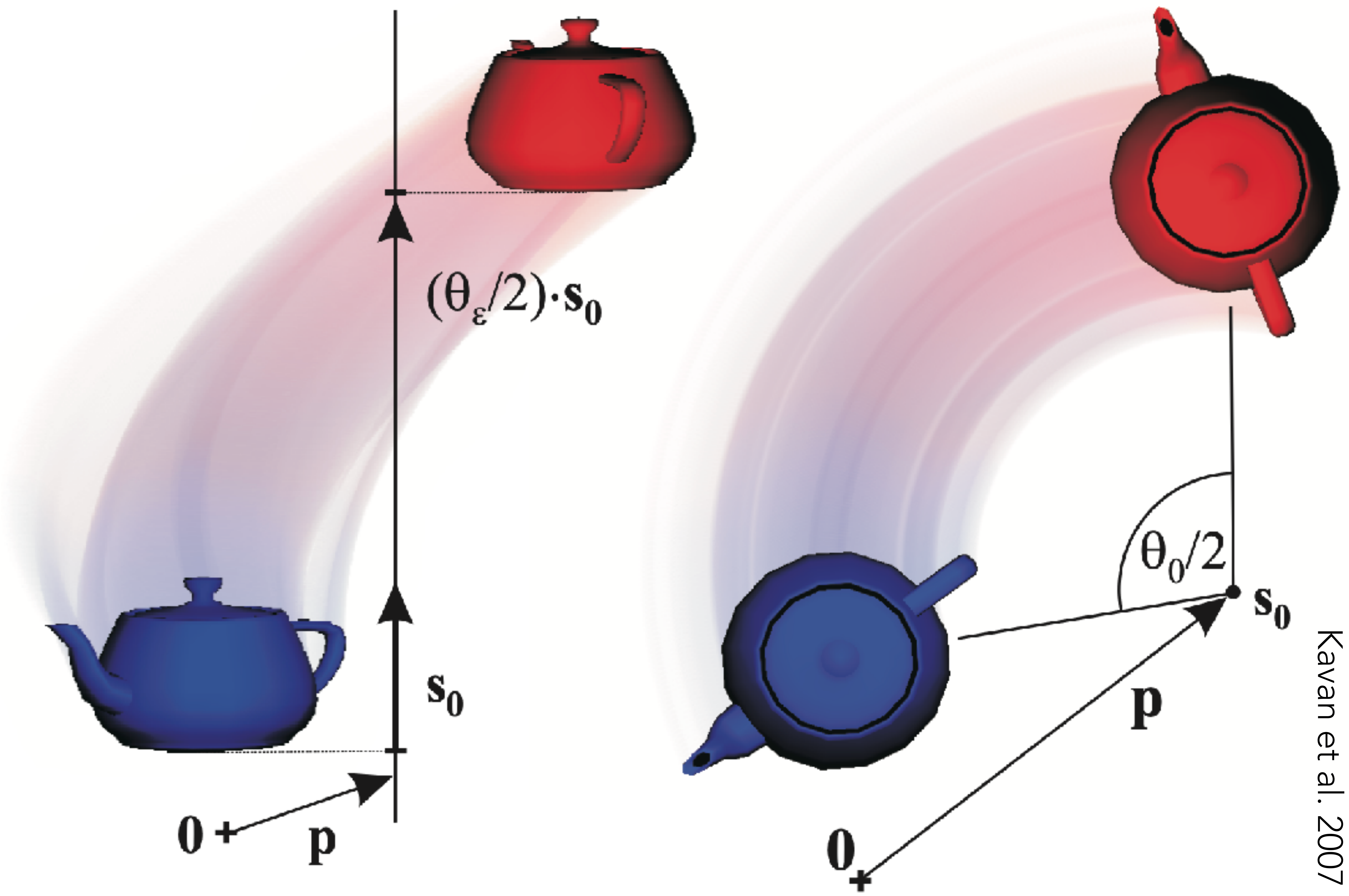
Problems with linear blending



Lewis et al. 2000

Root cause: linearly averaging two rigid transformations does not give a rigid transformation!

How to interpolate rigid transformations (translation + rotation)?



Kavan et al. 2007

Recap: Quaternions

Quaternions are quantities of the form $\mathbf{q} = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ where

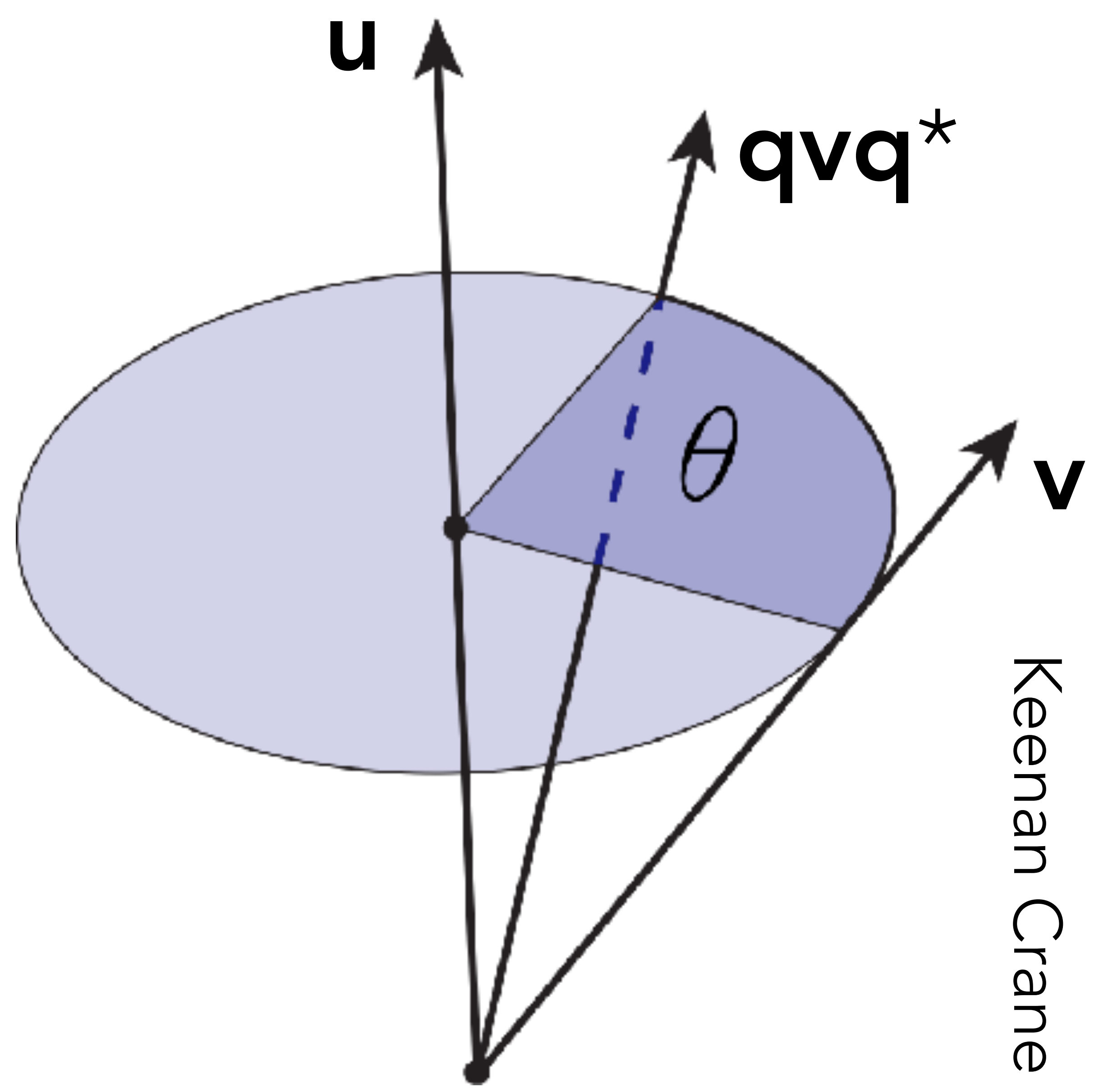
$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$$

Useful to separate into scalar part and vector part: $\mathbf{q} = (a, \mathbf{u})$

Any unit quaternion represents a 3D rotation.
Rotation by angle θ about axis $\mathbf{u}$: $\mathbf{q} = (\cos(\theta/2), \mathbf{u}\sin(\theta/2))$

How to rotate a vector $\mathbf{v} = (x, y, z) \in \mathbb{R}^3$?

- Interpret as a purely imaginary quaternion $\mathbf{v} = 0 + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$.
- Then the rotated vector is $\mathbf{qvq}^*$



Keenan Crane

Dual quaternions

A dual quaternion is a quantity of the form

$$\hat{\mathbf{q}} = \mathbf{q}_0 + \varepsilon \mathbf{q}_\varepsilon$$

where $\mathbf{q}_0$ and $\mathbf{q}_\varepsilon$ are quaternions, and $\varepsilon^2 = 0$.

- Multiplication: just drop all ε^2 terms

$$(\mathbf{p}_0 + \varepsilon \mathbf{p}_\varepsilon)(\mathbf{q}_0 + \varepsilon \mathbf{q}_\varepsilon) = \mathbf{p}_0\mathbf{q}_0 + \varepsilon(\mathbf{p}_0\mathbf{q}_\varepsilon + \mathbf{p}_\varepsilon\mathbf{q}_0)$$

- Dual conjugate: $\overline{\hat{\mathbf{q}}} = \mathbf{q}_0 - \varepsilon \mathbf{q}_\varepsilon$, quaternion conjugate: $\hat{\mathbf{q}}^* = \mathbf{q}_0^* + \varepsilon \mathbf{q}_\varepsilon^*$

- Norm: $\|\hat{\mathbf{q}}\| = \sqrt{\hat{\mathbf{q}}^*\hat{\mathbf{q}}} = \|\mathbf{q}_0\| + \epsilon \dfrac{\langle \mathbf{q}_0, \mathbf{q}_\epsilon \rangle}{\|\mathbf{q}_0\|}$

Any unit dual quaternion represents a rigid transformation.

- Rotation by quaternion $\mathbf{q}$: $\hat{\mathbf{q}} = \mathbf{q} + \varepsilon \mathbf{0}$
- Translation by vector $\mathbf{t}$: $\hat{\mathbf{t}} = 1 + \frac{\epsilon}{2}\mathbf{t}$
- Rotation then translation: $\hat{\mathbf{t}}\hat{\mathbf{q}} = \mathbf{q} + \frac{\epsilon}{2}\mathbf{t}\mathbf{q}$

How to transform a point $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$?

- Interpret as a dual quaternion $\hat{\mathbf{p}} = 1 + \varepsilon(xi + yj + zk)$
- Then the transformed point is $\hat{\mathbf{q}}\hat{\mathbf{p}}\overline{\hat{\mathbf{q}}^*}$

Dual quaternion skinning

Linear blend skinning:

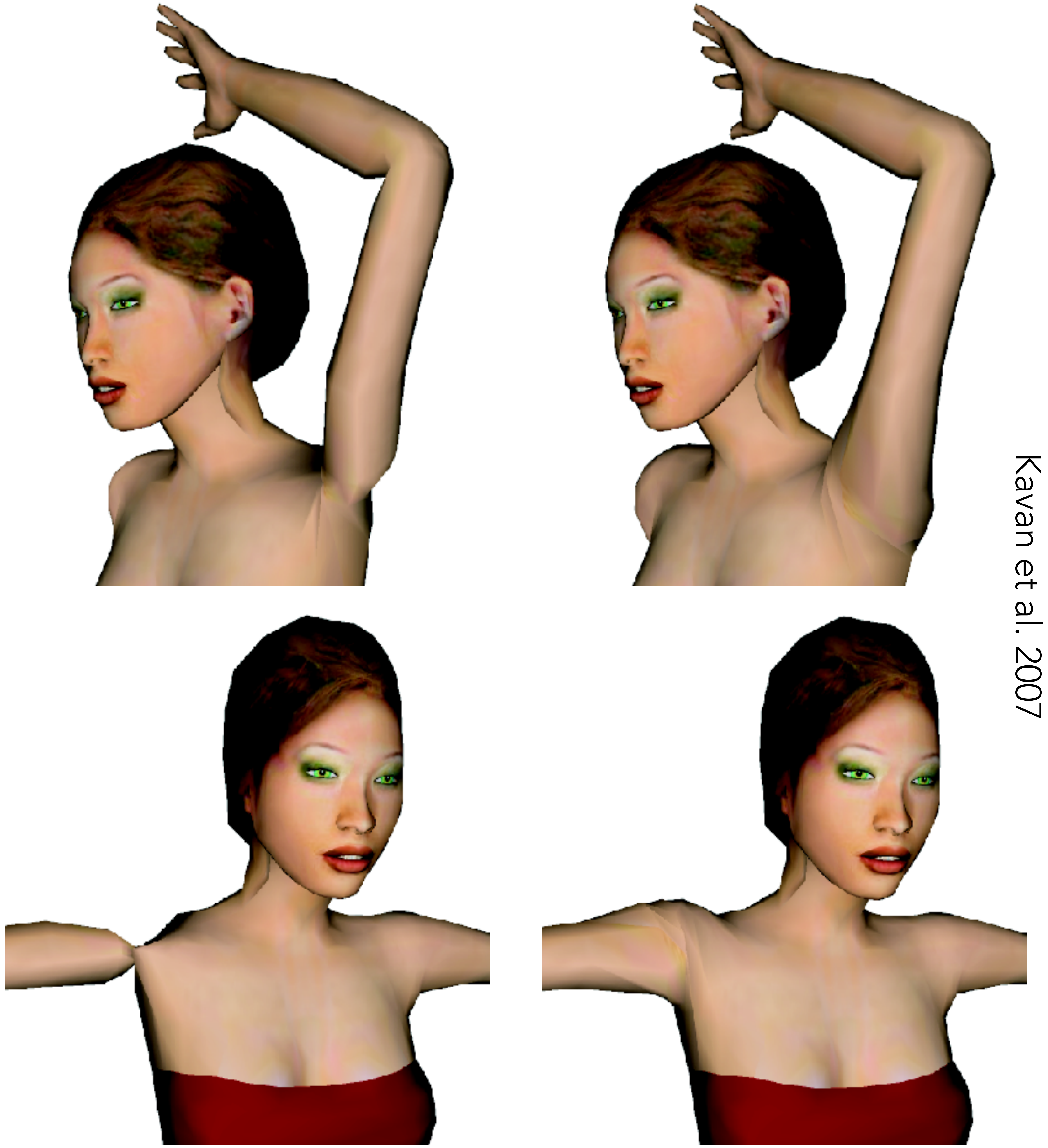
$$\mathbf{v}' = \sum_{\text{bone } j} w_j \mathbf{T}_j \begin{bmatrix} \mathbf{v} \\ 1 \end{bmatrix} = \left(\sum_{\text{bone } j} w_j \mathbf{T}_j \right) \begin{bmatrix} \mathbf{v} \\ 1 \end{bmatrix}$$

Dual quaternion skinning:

$$\hat{\mathbf{q}} = \frac{\sum_j w_j \hat{\mathbf{q}}_j}{\| \sum_j w_j \hat{\mathbf{q}}_j \|}$$

$$\mathbf{v}' = \hat{\mathbf{q}} \hat{\mathbf{v}} \overline{\hat{\mathbf{q}}^*}$$

Normalization ensures that final transformation is still rigid!



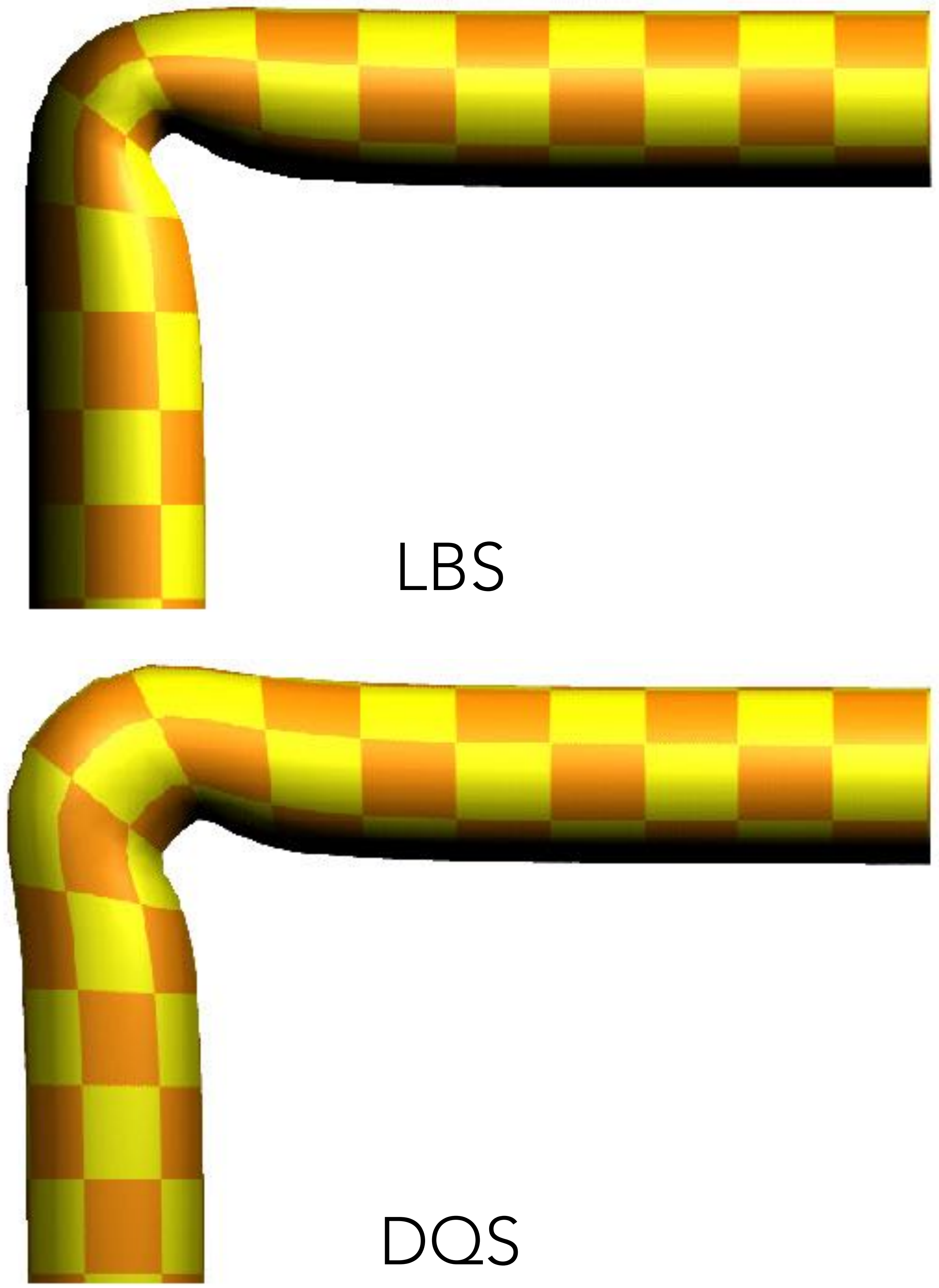
Kavan et al. 2007

Linear blend skinning

Dual quaternion skinning

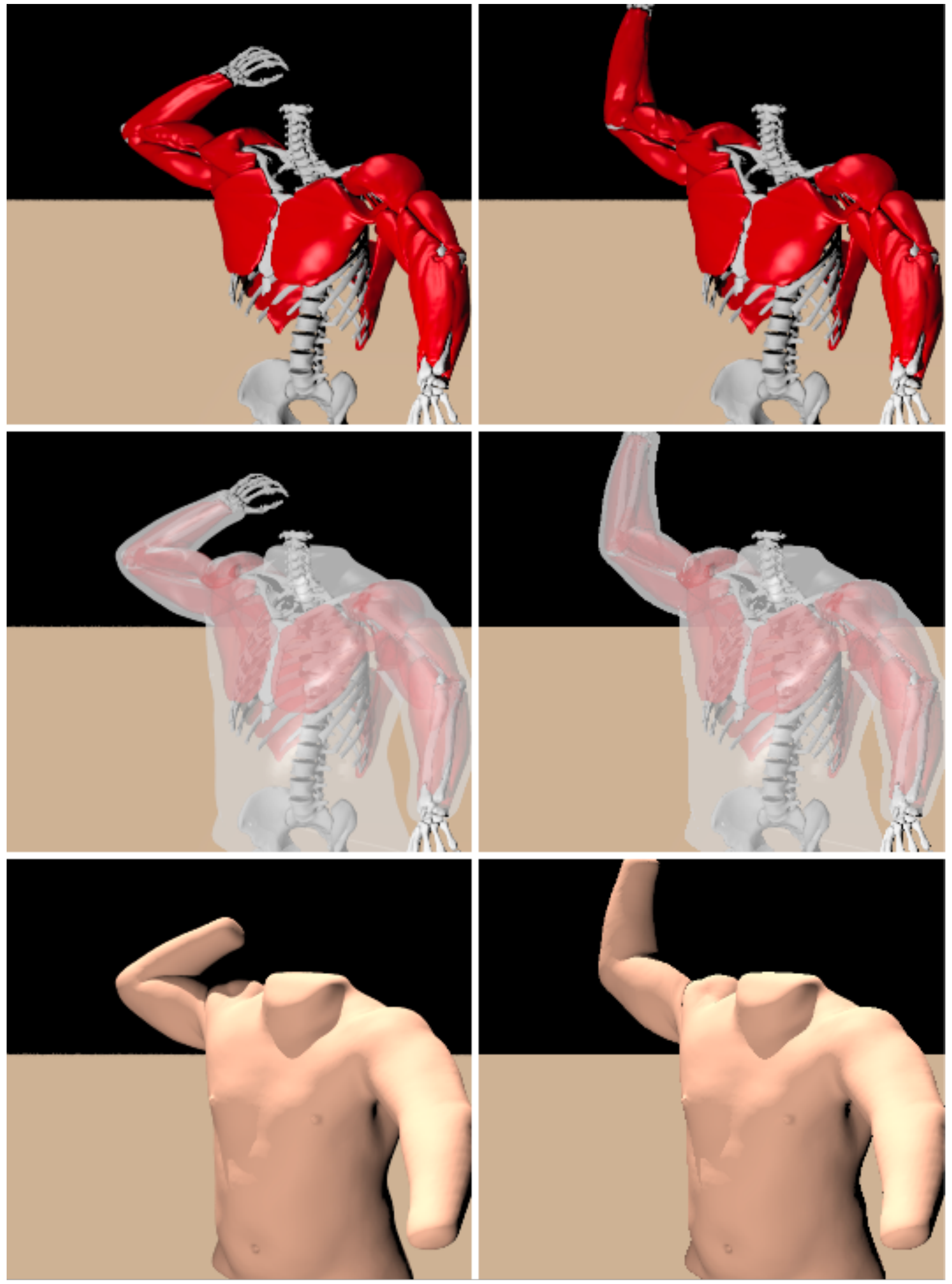
Dual quaternion skinning is not the end of the story!

- Pose space deformation
- Elasticity-inspired deformers
- Implicit skinning
- Optimized centers of rotation
- Direct delta mush
- …

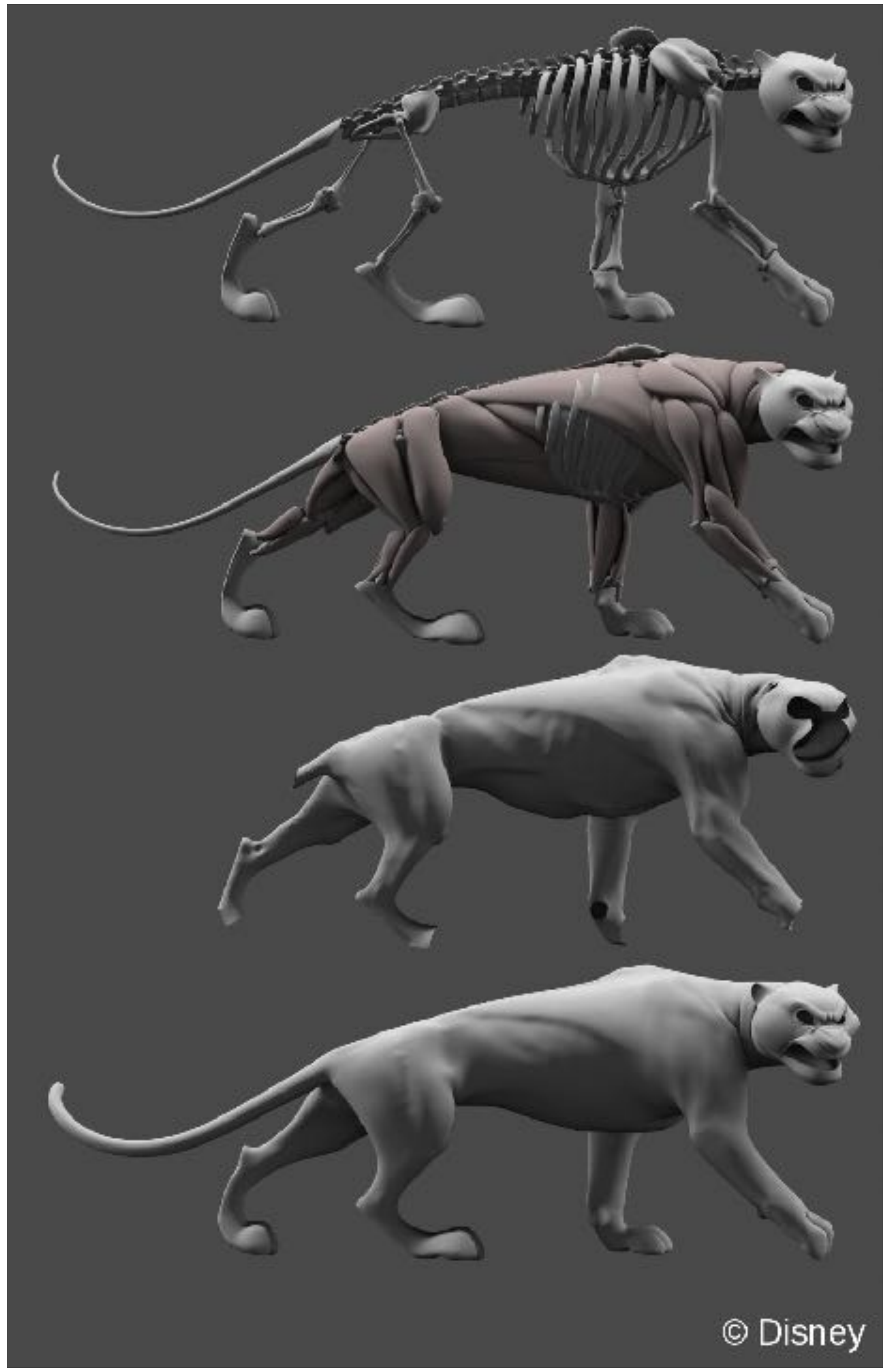


Ladislav Kavan

Beyond geometric skinning



Teran et al. 2005



Milne et al. 2016