Shape Modeling and Geometry Processing
*Exercise 6 - **Skeletal Animation***

INTERACTIVE GEOMETRY LAB

May 17, 2024

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

How to represent rotations?

3D Rotation Representations

- $R \in SO(3)$: 3-by-3 matrix that satisfies $R^T R = I$
- Euler's Angles: three angles to describe the orientation w.r.t. a fixed coordinate system
- Axis-angle representation: unit vector (direction of an axis rotation) + an angle (the magnitude of the rotation about the axis)
- Unit length quaternions: 4-d vector

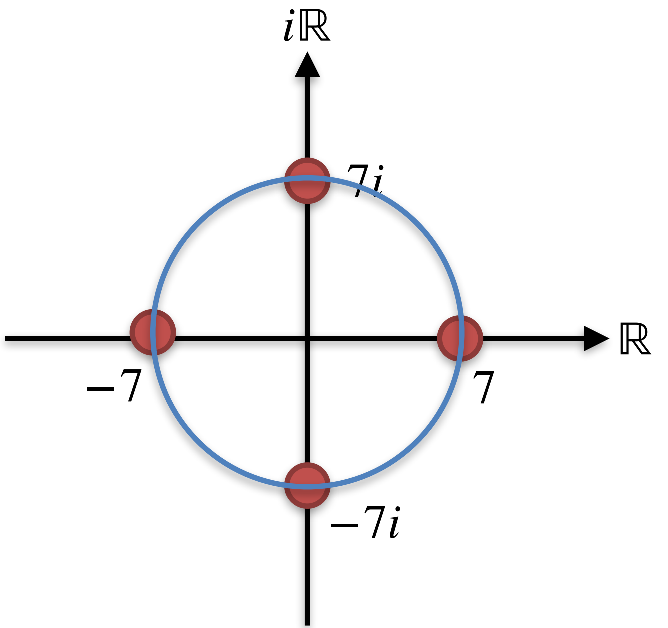
Complex Numbers for Rotations

- Recall the complex numbers
  $z = a + bi, \quad i^2 = -1$
- What's the geometric interpretation of multiplication by i?

  $i^n z$: rotate z by angle $\frac{\pi}{2} n$

- How do we rotate by θ ? Choose $n = \dfrac{2\theta}{\pi}$

  how do we calculate $i^t,\ t \in \mathbb{R}$?



Complex Numbers for Rotations

But how do we calculate $i^t,\ t \in \mathbb{R}$?

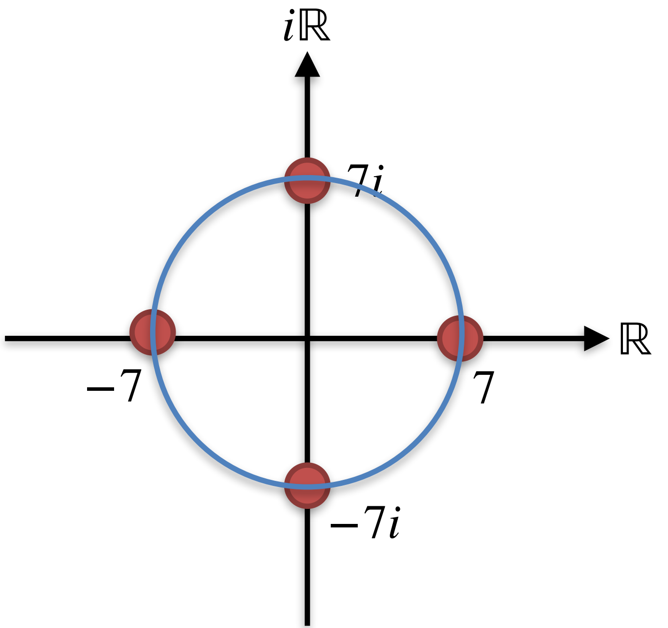
$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

Convert the exponentiation basis and calculate

$$i^t = e^{t \log i}$$

For a rotation by θ:

$$i^{\frac{2\theta}{\pi}} = e^{\frac{2\theta}{\pi} \log i} = e^{i\theta} = \cos\theta + i \sin\theta$$



Quaternions for 3D Rotations

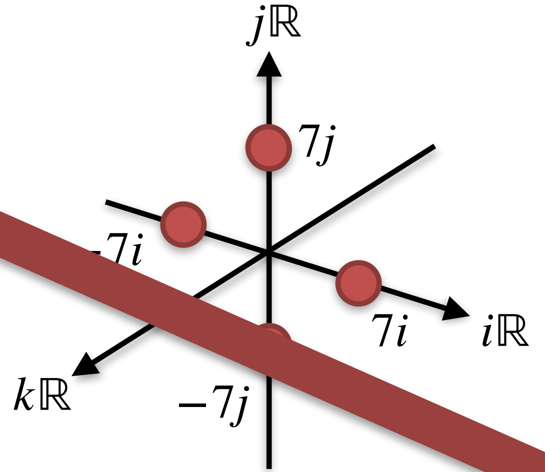
- Define a quaternion:

$$q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k} = a + \overrightarrow{v}$$

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

$$\mathbf{ij} = \mathbf{k}, \quad \mathbf{jk} = \mathbf{i}, \quad \mathbf{ki} = \mathbf{j}$$

What does left multiplication by $\mathbf{k}$ do?



- Can we rotate by calculating $q^t\overrightarrow{v}$?

ONLY if $|q| = 1, \quad q \perp \overrightarrow{v}$

Quaternions for 3D Rotations

- Define quaternion:

$$q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k} = a + \overrightarrow{v}$$

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

$$\mathbf{ij} = \mathbf{k}, \quad \mathbf{jk} = \mathbf{i}, \quad \mathbf{ki} = \mathbf{j}$$

Take two pure imaginary quaternions:

$$u = 0 + \overrightarrow{u}, \quad v = 0 + \overrightarrow{v}$$

Their product satisfies:

$$uv = -\langle \overrightarrow{u}, \overrightarrow{v} \rangle + \overrightarrow{u} \times \overrightarrow{v}$$

$$uv = (u_1\mathbf{i} + u_2\mathbf{j} + u_3\mathbf{k})(v_1\mathbf{i} + v_2\mathbf{j} + v_3\mathbf{k}) =$$

$$\underbrace{-u_1v_1 - u_2v_2 - u_3v_3}_{-\langle \overrightarrow{u}, \overrightarrow{v} \rangle} + \underbrace{(u_2v_3 - u_3v_2)\mathbf{i} + (u_3v_1 - u_1v_3)\mathbf{j} + (u_1v_2 - u_2v_1)\mathbf{k}}_{\overrightarrow{u} \times \overrightarrow{v}}$$

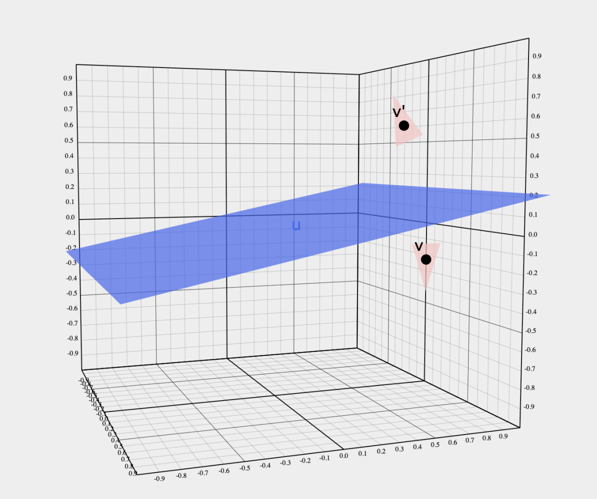
Quaternions for 3D Rotations

$$u = 0 + \vec{u}, \;\; v = 0 + \vec{v} \qquad uv = -\langle \vec{u}, \vec{v} \rangle + \vec{u} \times \vec{v}$$

- What happens if we sandwich v by u? (and $\|u\| = 1$)

$$uvu = \underbrace{\vec{u} \times \vec{v} \times \vec{u}}_{Proj_{u^\perp}(v)} - \underbrace{\langle \vec{u}, \vec{v} \rangle \vec{u}}_{Proj_u(v)}$$

$$= \text{reflect}(\vec{v}, \vec{u}^\perp)$$



- How do we get a rotation from this?

Just reflect again!

Quaternions for 3D Rotations

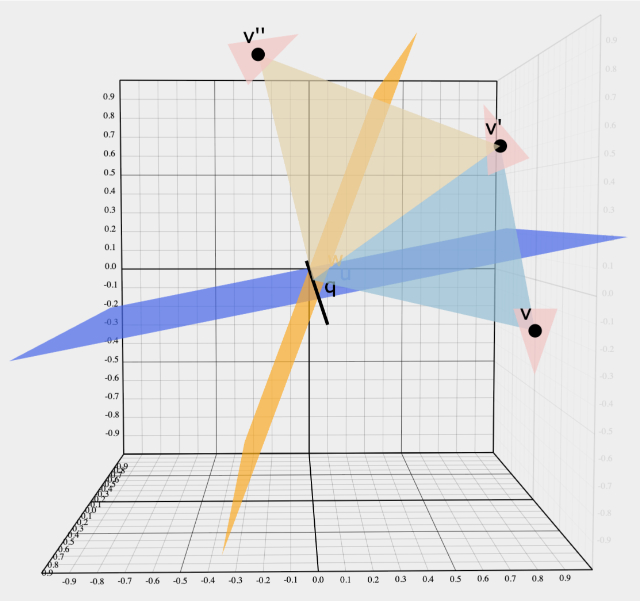
$$uvu = \text{reflect}(\vec{v}, \vec{u}^{\perp})$$

$$w(uvu)w = \text{rotate}(\vec{v}, \vec{u} \times \vec{w}, 2\angle \vec{w}, \vec{u})$$

Define $q = wu$

$$q = \cos\theta + \sin\theta\,\vec{e} \qquad \bar{q} = \cos\theta - \sin\theta\,\vec{e}$$

$$q v \bar{q} = \text{Rotate } \vec{v} \text{ around } \vec{e} \text{ by angle } 2\theta$$



Double Cover

- Both q and $-q$ represents the same 3D rotation.
  - $q = [\cos\frac{\theta}{2}, \sin\frac{\theta}{2}\overrightarrow{e}].$
  - Rotating about $\overrightarrow{e}$ by θ and rotating about $-\overrightarrow{e}$ by $2\pi - \theta$ will give the same rotation, which corresponds to q and $-q$.
  - $v' = qv\bar{q}$
  - Plugging q and $-q$ yields the same v'

How to represent transformations?

Matrix-Form Transformations

- Recall a rigid 3D transformation includes a rotation $R \in \mathbb{SO}(3)$ and a translation $T \in \mathbb{R}^3$
- For point $v = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$, define the homogenous coordinate $\hat{v} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$
- Homogenous transformation representation

$$\hat{R} = \begin{pmatrix} R & T \\ \mathbb{O}_{1\times 3} & 1 \end{pmatrix} \in \mathbb{R}^{4\times 4}$$

```
Eigen::Translation3d T(tx, ty, tz);
Eigen::Quaterniond quat(w, x, y, z);
Eigen::Isometry3d transform = (T*quat);
```

- We have $\hat{R}\hat{v} = \begin{pmatrix} Rv + T \\ 1 \end{pmatrix}$, i.e., we use a 4×4 matrix to represent a rigid transformation in homogenous coordinates

Dual Quaternion

- Define dual quaternion:

$$\hat{q} = q_1 + \epsilon q_2 \qquad q_1, q_2 \in \mathbb{H} \qquad \epsilon^2 = 0 \qquad \epsilon q = q \epsilon$$

  - Points are now represented as $p = 1 + \epsilon \left(x\mathbf{i} + y\mathbf{j} + z\mathbf{k} \right) = 1 + \epsilon \vec{v}$

  - How do we rotate?

$$q_r p \bar{q}_r = q_r (1 + \epsilon \vec{v}) \bar{q}_r = q_r \bar{q}_r + \epsilon \left(q_r \vec{v} \bar{q}_r \right) = 1 + \epsilon \left(q_r \vec{v} \bar{q}_r \right)$$

  - How do we translate?

$$\left(1 + \epsilon \vec{t}\right) \left(1 + \epsilon \vec{v}\right) = 1 + \epsilon \left(\vec{t} + \vec{v}\right) + \epsilon^2 \vec{t}\,\vec{v} = 1 + \epsilon \left(\vec{t} + \vec{v}\right)$$

Dual Quaternion

- Define dual quaternion:

$$\hat{q} = q_1 + \epsilon q_2 \qquad q_1, q_2 \in \mathbb{H} \qquad \epsilon^2 = 0 \qquad \epsilon q = q\epsilon$$

  - Points are now represented as $p = 1 + \epsilon\left(x\mathbf{i} + y\mathbf{j} + z\mathbf{k}\right) = 1 + \epsilon\vec{v}$

  - How do we rotate?

$$q_r p \bar{q}_r = q_r(1 + \epsilon\vec{v})\bar{q}_r = q_r\bar{q}_r + \epsilon\left(q_r\vec{v}\bar{q}_r\right) = 1 + \epsilon\left(q_r\vec{v}\bar{q}_r\right)$$

  - How do we translate?

$$\left(1 + \frac{1}{2}\epsilon\vec{t}\right)\left(1 + \epsilon\vec{v}\right)\left(1 + \frac{1}{2}\epsilon\vec{t}\right) = 1 + \epsilon\left(\vec{t} + \vec{v}\right)$$

Dual Quaternion

- How do we represent rigid transformation?

$$q_r p \bar{q}_r = 1 + \epsilon \left(q_r \vec{v} \bar{q}_r \right) \qquad \left(1 + \frac{1}{2}\epsilon \vec{t}\right)\left(1 + \epsilon \vec{v}\right)\left(1 + \frac{1}{2}\epsilon \vec{t}\right) = 1 + \epsilon \left(\vec{t} + \vec{v}\right)$$

- Rotate by q_r, translate by $\vec{t}$

$$\boxed{p' = \left(1 + \frac{\epsilon}{2}\vec{t}\right) q_r p \bar{q}_r \left(1 + \frac{\epsilon}{2}\vec{t}\right)}$$

- Define

$$\hat{q} = q_r + \frac{1}{2}\epsilon \vec{t} q_r \qquad p' = \hat{q} p \hat{q}*$$

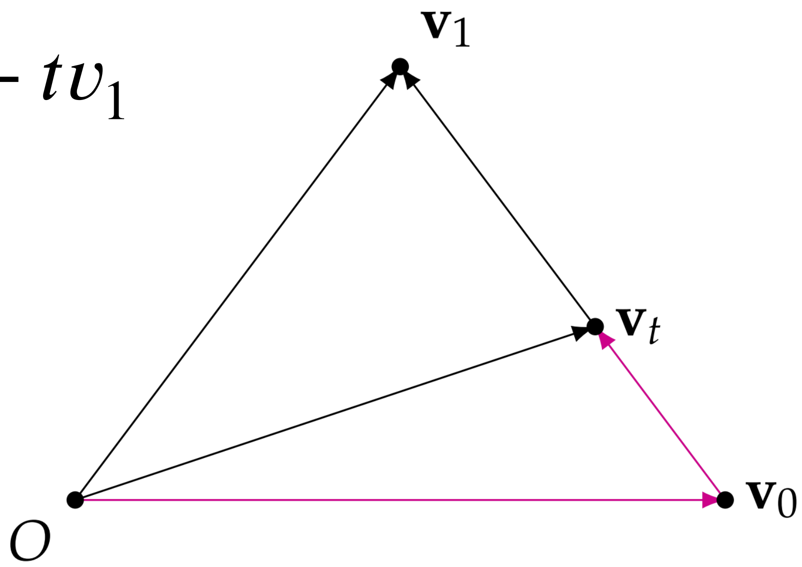
$$\hat{q}* = \bar{q}_1 - \epsilon \bar{q}_2$$

How to interpolate rotations?

Lerp, NLerp, SLerp

- Lerp: Linear Interpolation

$$v_t = \mathrm{Lerp}(v_0, v_1, t) = (1 - t)v_0 + tv_1$$

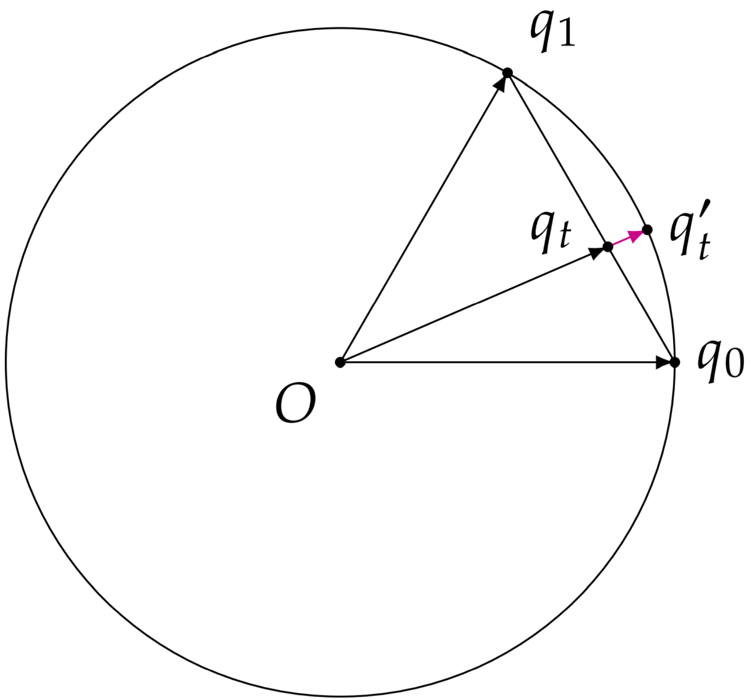


Lerp, NLerp, SLerp

- Nlerp: Normalized Lerp

$$q_t = \mathrm{Lerp}(q_0, q_1, t)$$

$$q_t' = \frac{q_t}{\|q_t\|}$$



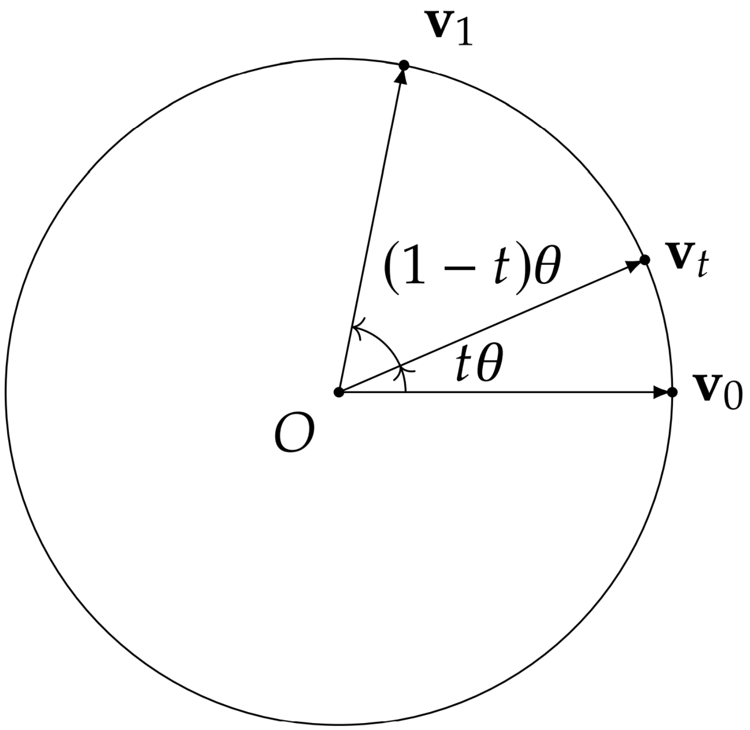
Lerp, NLerp, SLerp

- Slerp: Spherical Lerp

$$q_0 \longrightarrow q_1$$

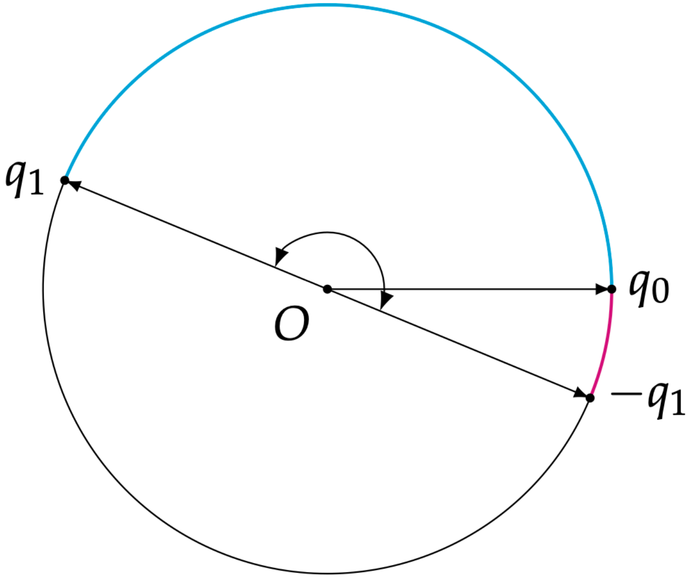
$$q_0 \underbrace{(q_0^{-1} q_1)^t}_{q_0 \to q_1} \qquad t \in [0,1]$$

$$= q_0 \exp(t \log(q_0^{-1} q_1))$$



Interpolation and Double Cover

- Check if they are on the same hemisphere before interpolating!



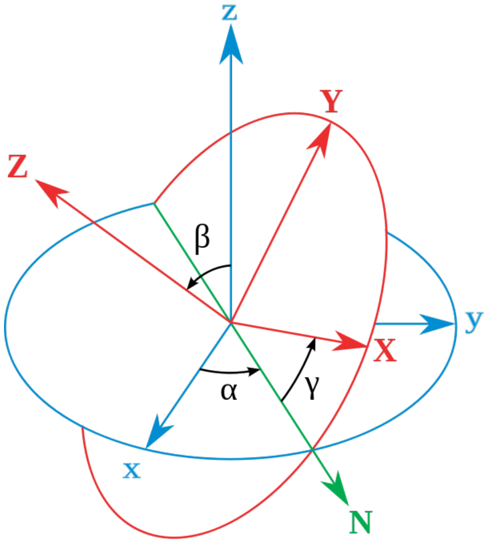
Tasks in Assignment 6

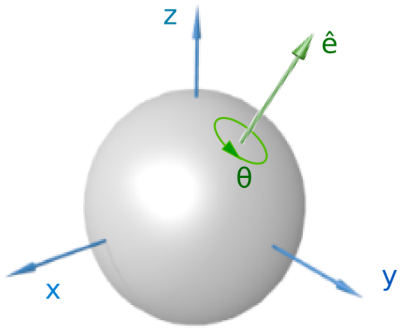
Task 1: input

- Understanding the problem and loading the relevant data:
  - The rest shape (.obj)
  - The rigging (.skel: a graph)
  - The handles for each bone (.dmat: a matrix)
  - The rotations per reference frame (.dmat)
  - The global position per frame (.dmat)

Task 2: Theory

- Understand the different rotation representations





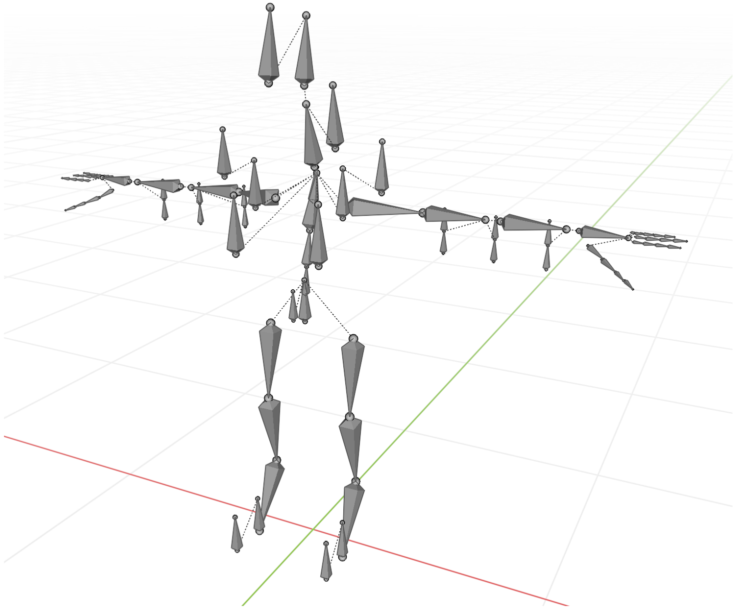
Source of images: wikipedia

Task 3: Skeletal animation

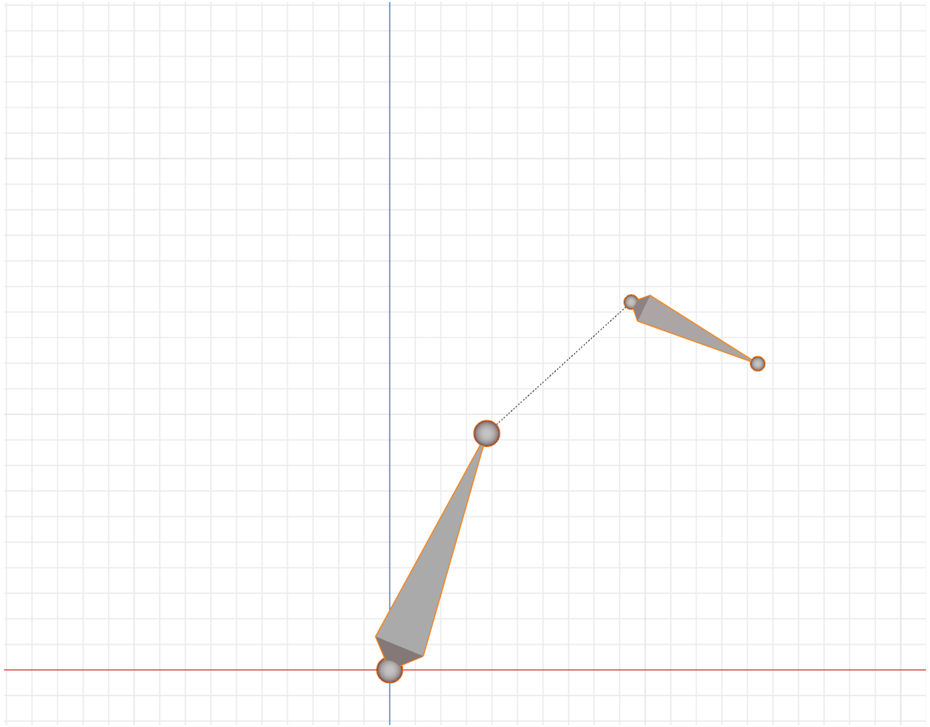
- Move the rigging using **forward kinematics**
- Compute global rotations $\hat{R}_k^{(l)}$ and translations $\hat{T}_k^{(l)}$
  - For bone k at frame l
- Compute new position $x_k^{(l)} = \hat{R}_k^{(l)} x_k + T_k^{(l)}$

Forward kinematics

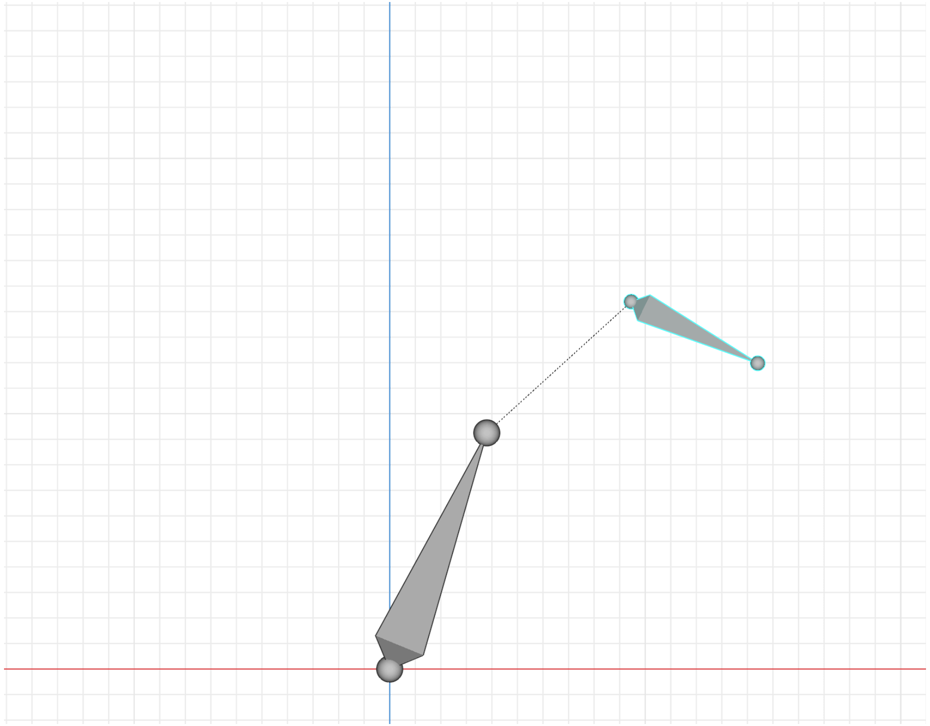
The skeleton is not necessarily connected!



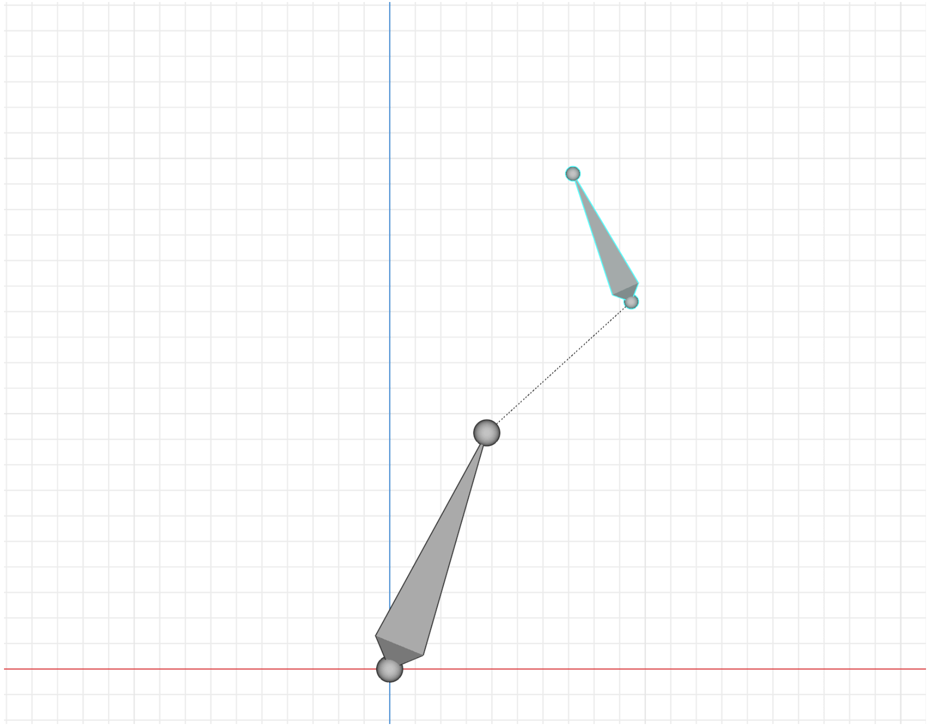
Forward kinematics



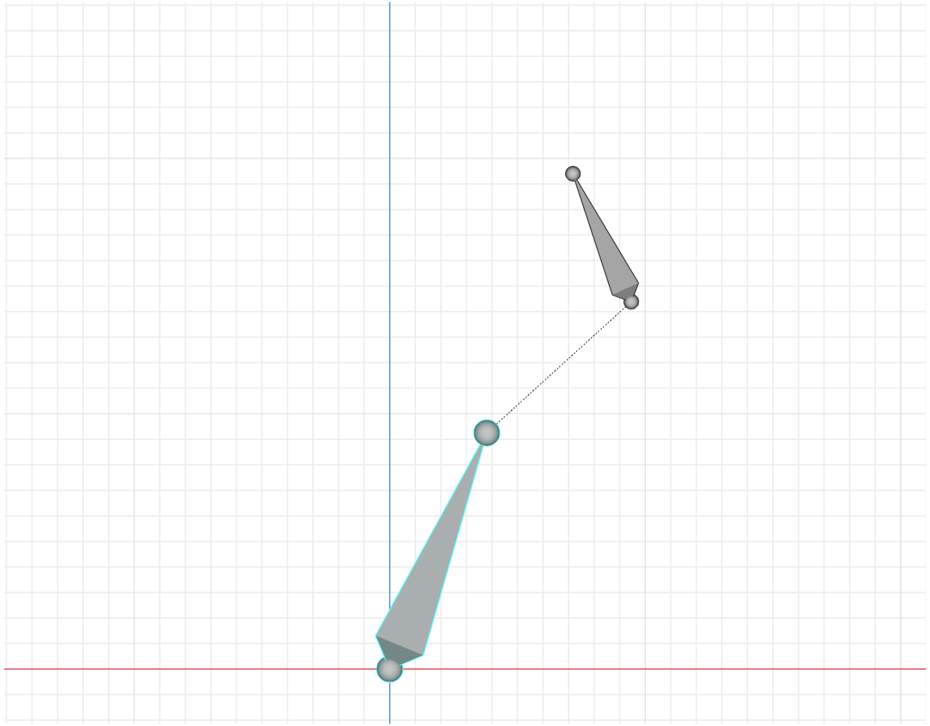
Forward kinematics



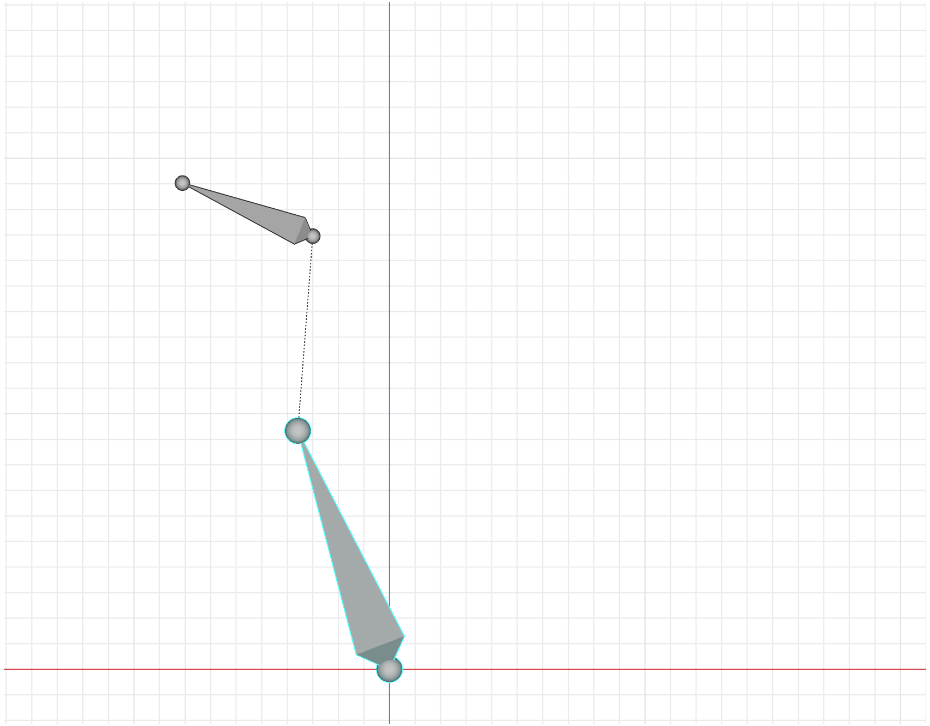
Forward kinematics



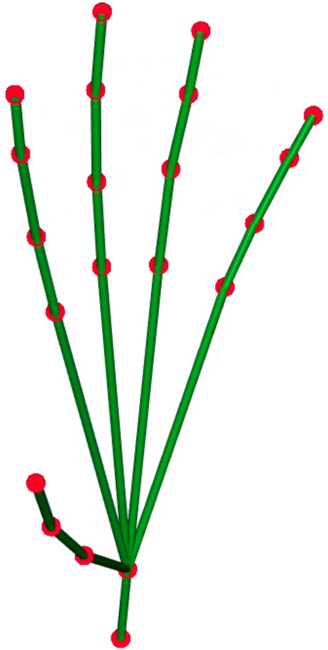
Forward kinematics



Forward kinematics



Task 3: Skeletal animation

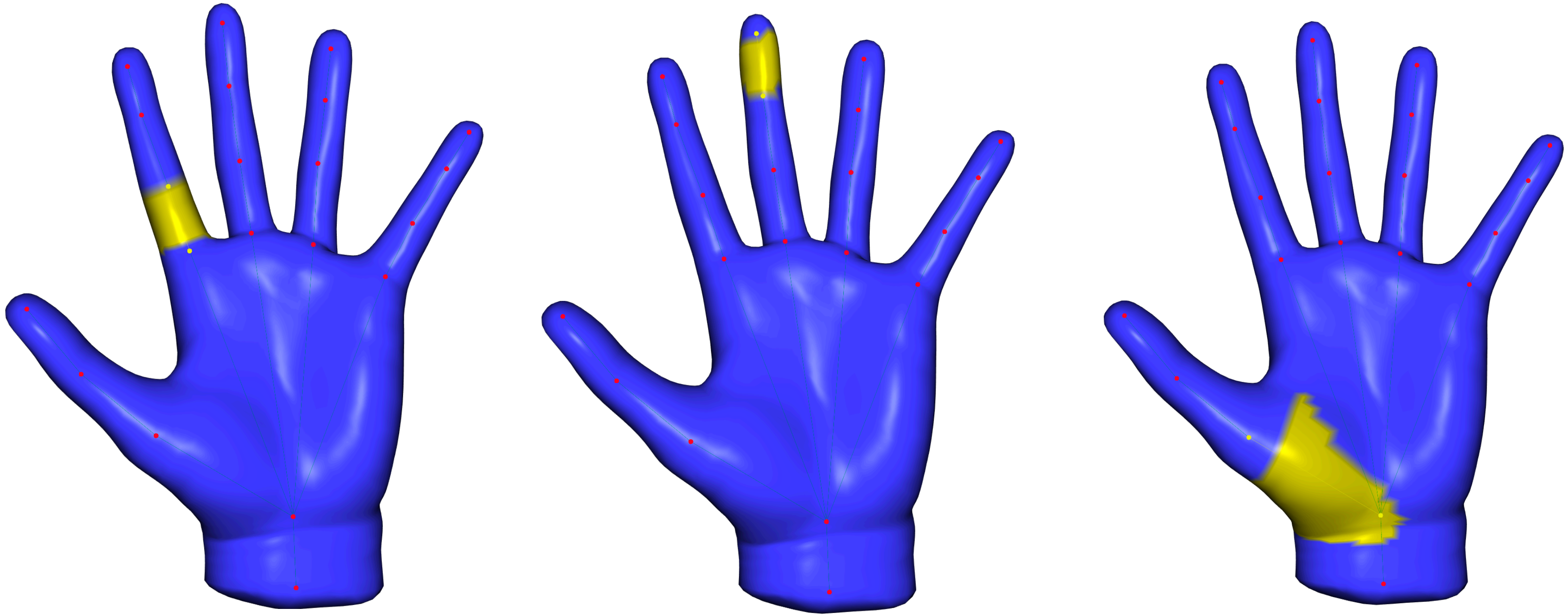


Task 4: Harmonic weight

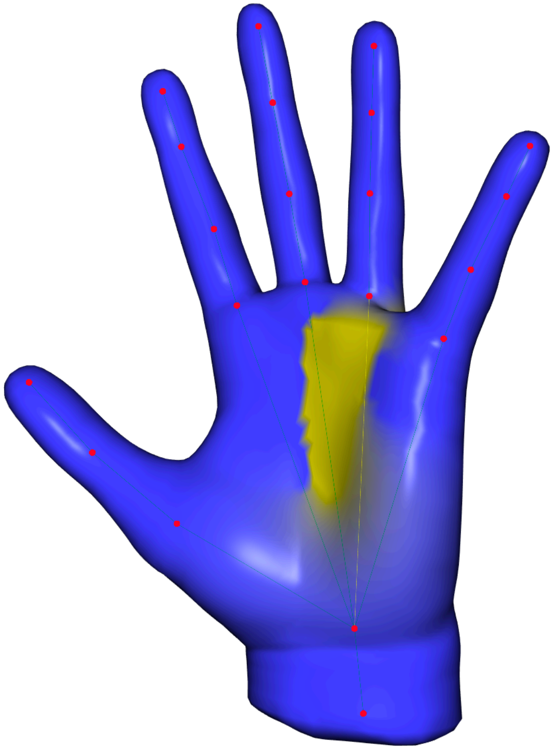
- Goal: weight the influence w_k of each bones on the mesh vertices
- Some have total influence (handles): $w_k(v) = 1$
- 1- determine the handles
- 2- spread the weights via Laplace equation:

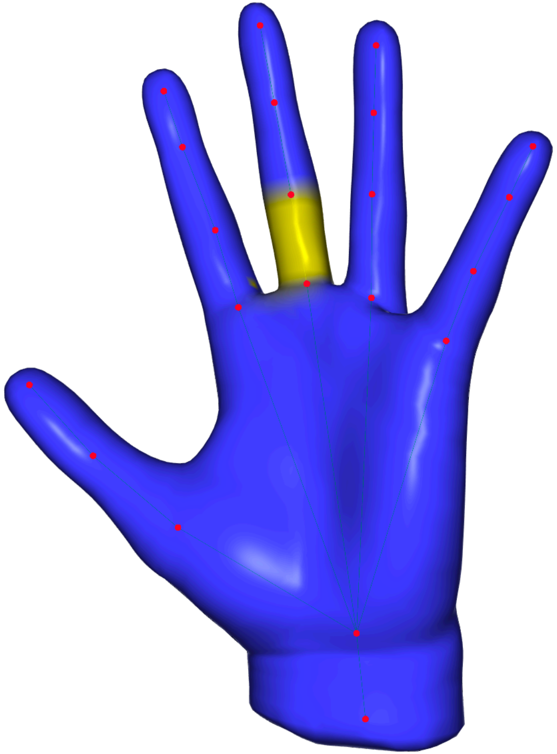
$$Lw_k = 0 \quad \text{subject to} \quad w_k(v) = 1 \, \forall v \in H_k, w_k(v) = 0 \, \forall v \in H_j \text{ where } j \neq k$$

Task 4: Handle selection



Task 4: Harmonic weight



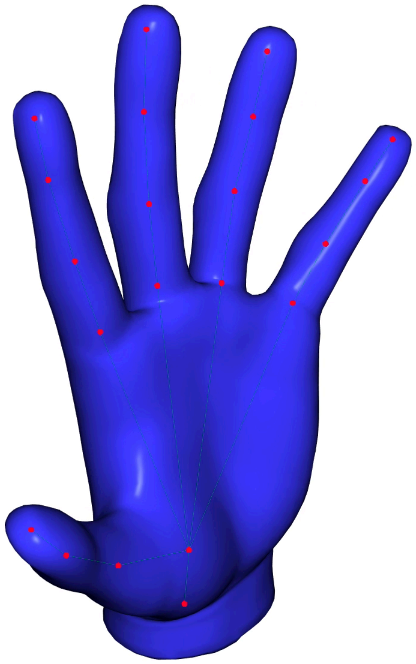


Task 5: Linear Blending Skinning

- For each bone, we know the rotation $R_k^{(l)}$ and translation $T_k^{(l)}$
- We apply them to the mesh according to the harmonic weights

$$v_i^{(l)} = \sum_{k=1}^{K} w_k(i)\left(R_k^{(l)} v_i^{(0)} + T_k^{(l)}\right)$$

Task 5: Linear Blending Skinning



Task 6: Dual Quaternion Skinning

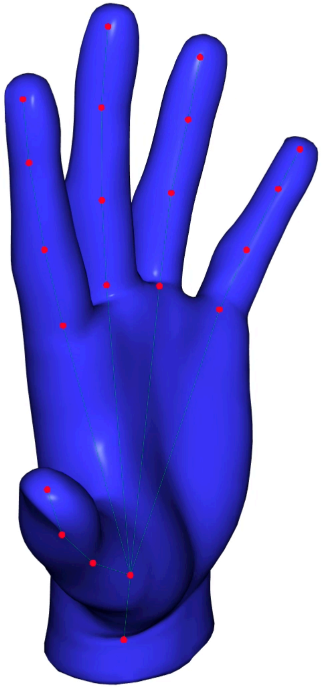
- Problem with LBS: weighted sum of rotations might not be a rotation

- Idea: normalize the result to get a unit dual quaternion (NLERP)

$$q_i^{(l)} = \frac{\sum_{k=1}^{K} w_k(i) q_k^{(l)}}{\left\| \sum_{k=1}^{K} w_k(i) q_k^{(l)} \right\|}$$

- DO NOT directly use igl::dqs()
- Always check for if quaternions are on the same hemisphere.

Task 6: Dual Quaternion Skinning



Task 7: Per-face linear Blending

- Goal: rotate the faces instead of the vertices
- 1- compute the weights $h_k(f) = \dfrac{\sum_{i \in \mathcal{N}_f} w_k(v_i)}{|N_f|}$ for each face
- 2- use averaging to compute the new rotation

$$\tilde{q}_f^{(l)} = \arg\max_{q \in \mathbb{S}^3} q^T \left(\sum_{k=1}^{K} h_k(f) q_k^{(l)} q_k^{(l)T} \right) q$$

- 3- **Solving:** system is over-constrained. Poisson stiching.

$$G \begin{pmatrix} v_1^{(l)} \\ \cdots \\ v_n^{(l)} \end{pmatrix} = \begin{pmatrix} \tilde{R}_1^{(l)} \\ \cdots \\ \tilde{R}_m^{(l)} \end{pmatrix}, G \in \mathbb{R}^{3m \times n} \text{ is the gradient matrix}$$

- Rarely used in practice, but useful to see how powerful Poisson stiching.

Task 8: Context-aware deformation

- Problem with LBS: might result in unlikely deformation
- Idea: use “reference animation”
- You are given a set of example deformations with corresponding meshes and rotations

Task 8: Context-aware deformation

- 1- Unpose the example shapes.

$$\bar{v}_{j,i} = T^{-1}\left(P_j, w_k(i)\right) v_{j,i}$$

- 2- Compute the displacement

$$\delta_{j,i} = \bar{v}_{j,i} - v_i^{(0)}$$

- 3- Apply the displacement to the animated shape

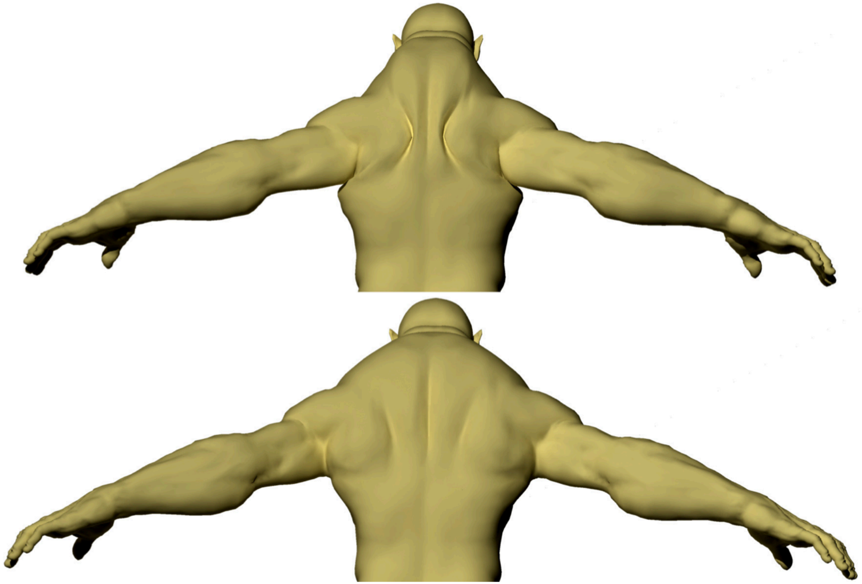
$$v_i = T\left(P, w_k(i)\right)\left(v_i^{(0)} + \sum_{j=1}^{J} a_j\left(P\right)\delta_{j,i}\right)$$

Task 8: Context-aware deformation

- Computation of $a_j(P)$, the weight of the j^{th} example pose to P:
  - $\sum_{j=1}^{J} a_j(P) = 1$
  - $\forall j, a_j\left(P_j\right) = 1,\ \forall i \neq j\ a_j\left(P_i\right) = 0$
  - $a_j(P)$ is continuous w.r.t. the pose
- Choose a RBF function ϕ and find the corresponding $c_{j,t}$ such that

$a_j\left(P\right) = \sum_{t=1}^{J} c_{j,t}\phi\left(\|P - P_t\|\right)$ –> solve the system corresponding to constraints

Task 8: Context-aware deformation



LBS

Context-aware

Weber et al (2007),
Context-Aware Skeletal Shape Deformation

Thank You