Deformation & Skinning
形变与蒙皮

北京大学 前沿计算研究中心

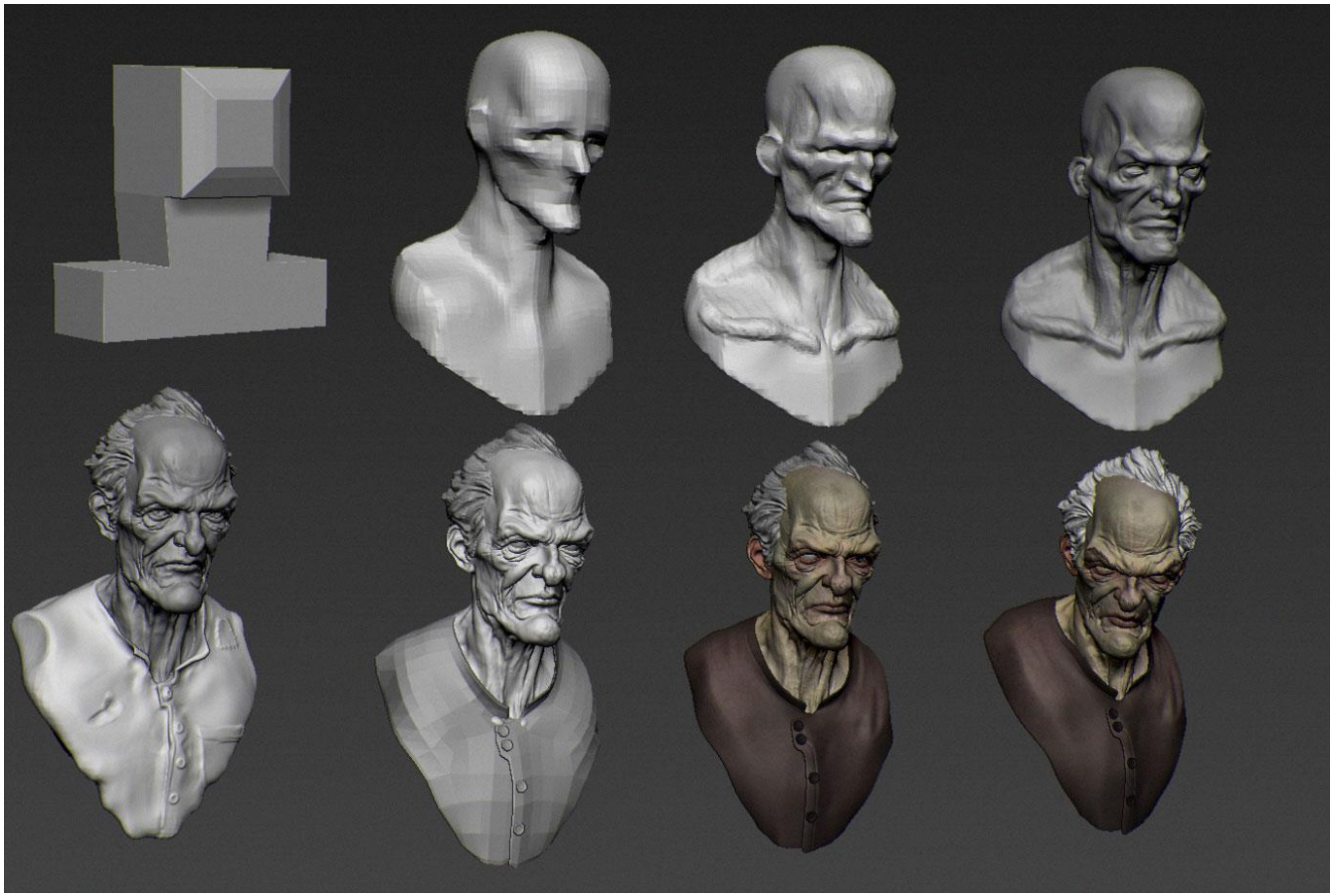
刘利斌

本节内容

- 几何形状的编辑
  - 自由形变 (Free-form Deformation)
  - 基于优化的形变
- 蒙皮形变 (Skinning Deformation)
  - 线性蒙皮
    - Linear-Blend Skinning (LBS)
    - Multi-linear Skinning
  - 非线性蒙皮
    - Dual-quaternion Skinning (DQS)
  - 基于样例的蒙皮
    - Blendshapes
  - 人脸动画

几何形变与编辑

- 直接编辑几何模型顶点/控制点

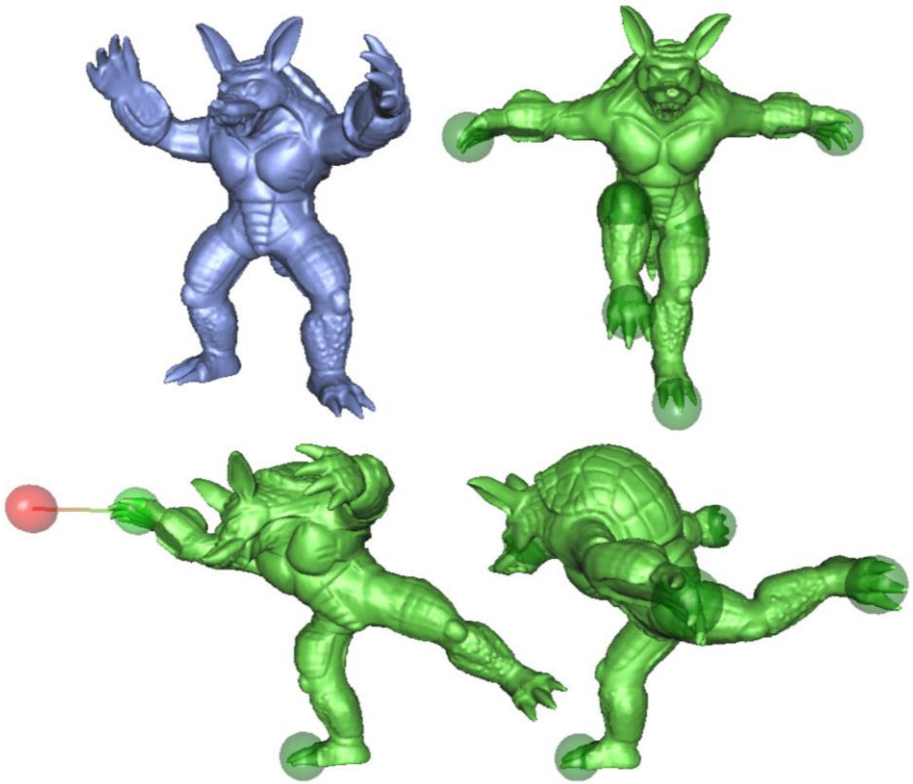


https://khanhha.github.io/portfolio/skeleton-to-mesh-conversion/

几何形变与编辑

- 编辑修改几何形状，同时保留几何模型细节

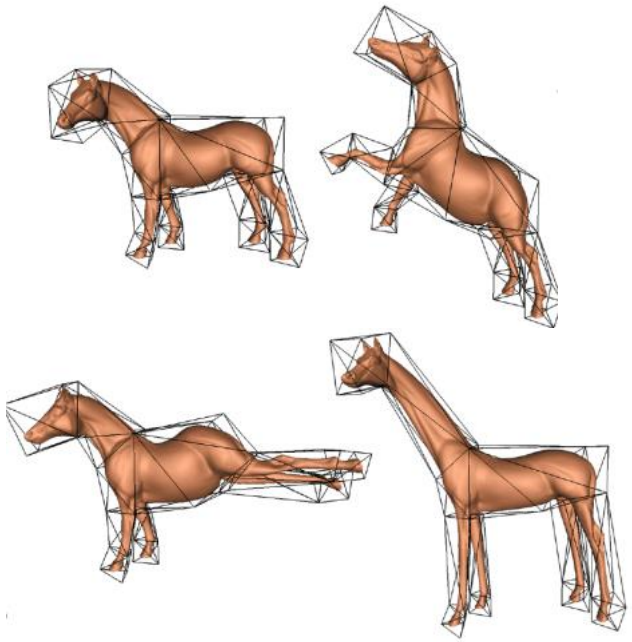
- 一般通过整体性的改变几何模型的参数来实现
  - 网格 -> 顶点
  - NURBS -> 控制点
  - ……



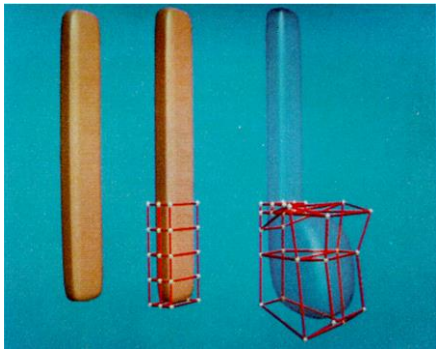
Xiaohan Shi, Kun Zhou, Yiying Tong, Mathieu Desbrun, Hujun Bao, and Baining Guo. 2007. ***Mesh puppetry: cascading optimization of mesh deformation with inverse kinematics***. *ACM Trans. Graph.* 26, 3 (July 2007),

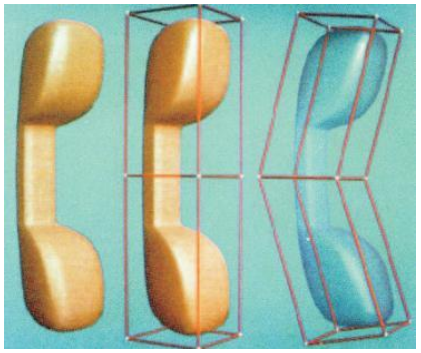
自由变形与Cage-based Deformation

- “扭曲”模型附近的空间来实现变形
  - 将几何模型嵌入到简单网格中
  - 改变简单网格的形状间接的改变模型的形状



Ju, T., Schaefer, S., Warren, J.: ***Mean value coordinates for closed triangular meshes***. In: ACM SIGGRAPH 2005 Papers, SIGGRAPH ’05, pp. 561–566.

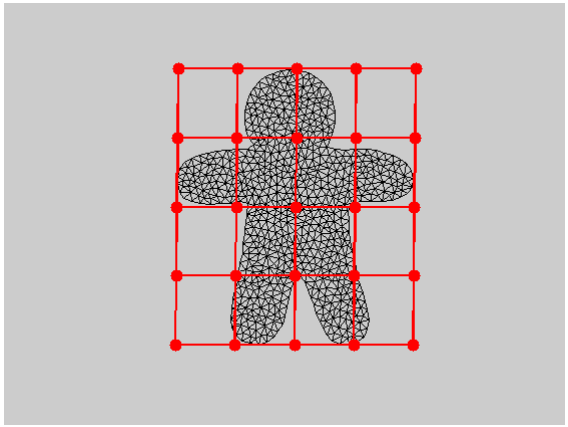




Thomas W. Sederberg and Scott R. Parry. 1986. ***Free-form deformation of solid geometric models***. In Proceedings of the 13th annual conference on Computer graphics and interactive techniques (SIGGRAPH '86).

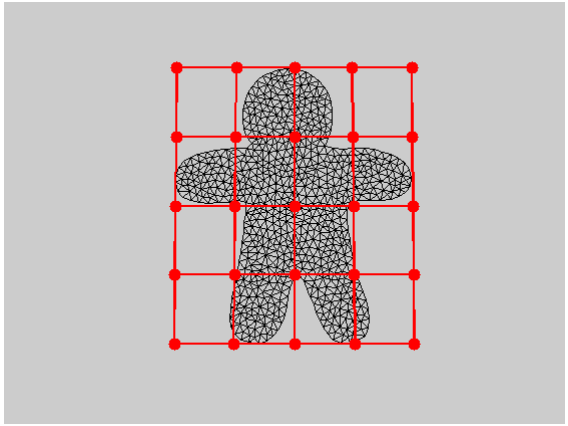
自由变形

- 如何定义嵌入关系?
  - 嵌入的精细网格 $\{x_i\}$
  - 简单网格 $\{p_j\}$



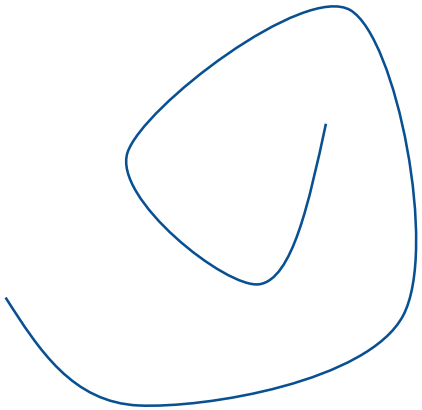
嵌入关系:

$$x_i = \sum_j \omega_{ij} p_j$$

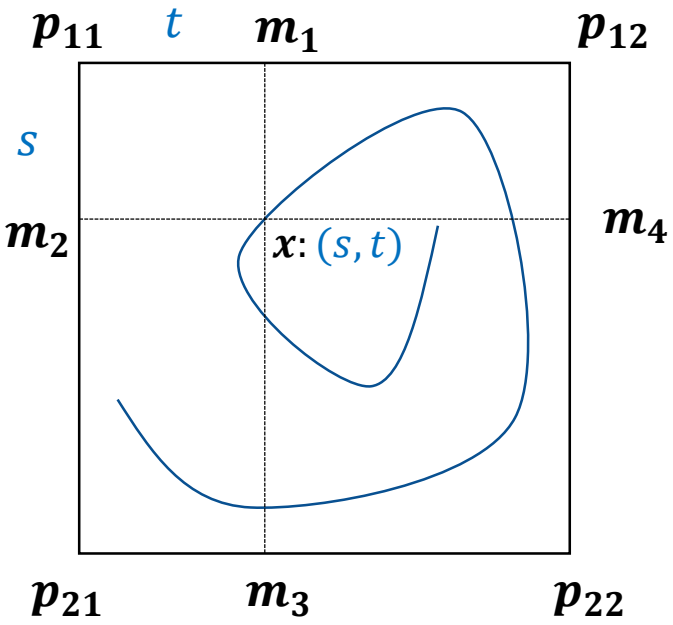


- 如何确定 ω_{ij} ??

线性插值

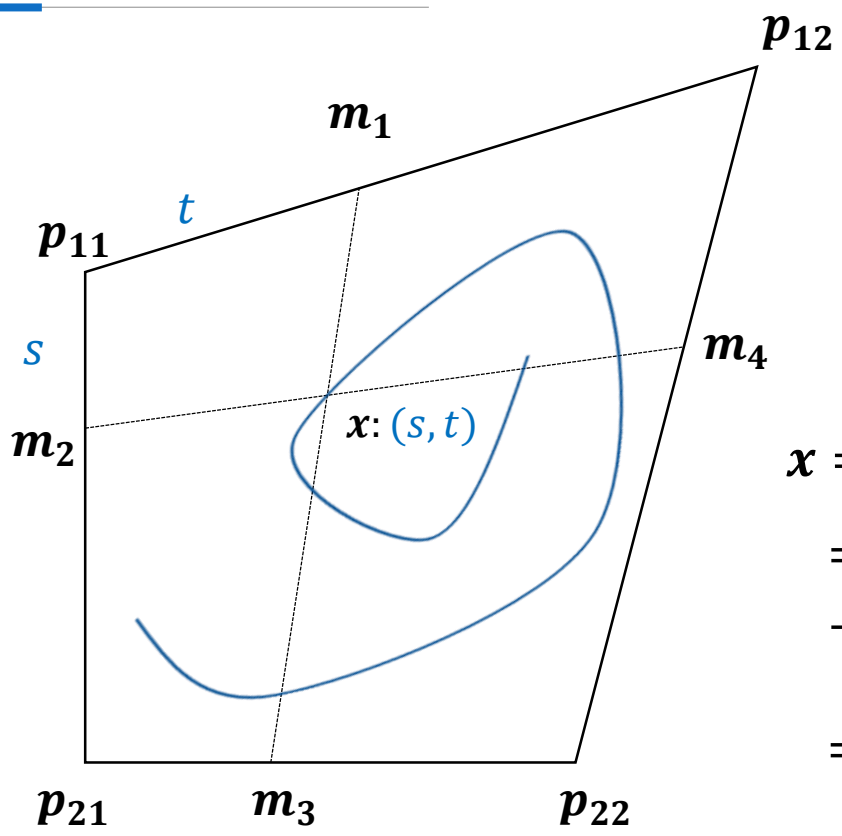


线性插值



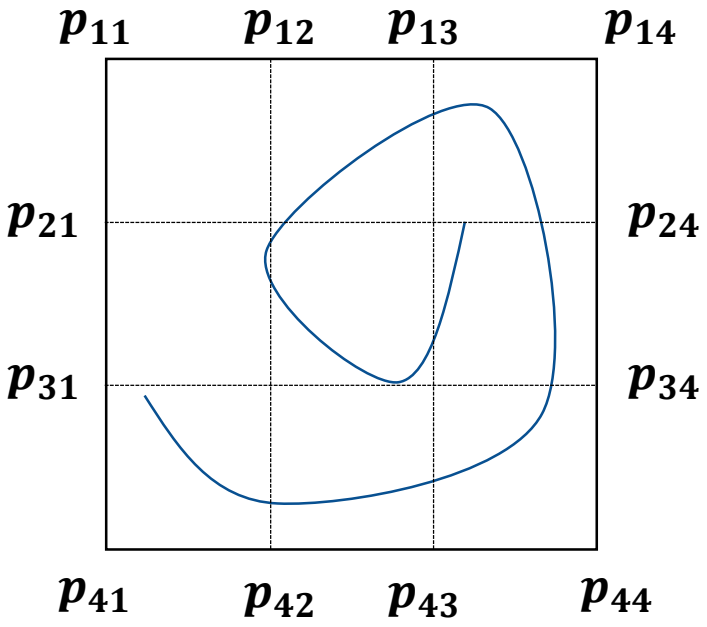
$$\begin{aligned}\boldsymbol{x} &= (1-t)\boldsymbol{m}_2 + t\boldsymbol{m}_4 \\ &= (1-t)\big((1-s)\boldsymbol{p}_{11} + s\boldsymbol{p}_{21}\big) \\ &+ \quad t\ \big((1-s)\boldsymbol{p}_{12} + s\boldsymbol{p}_{22}\big) \\ &= \sum_i \sum_j B_i(s)B_j(t)\boldsymbol{p}_{ij}\end{aligned}$$

线性插值



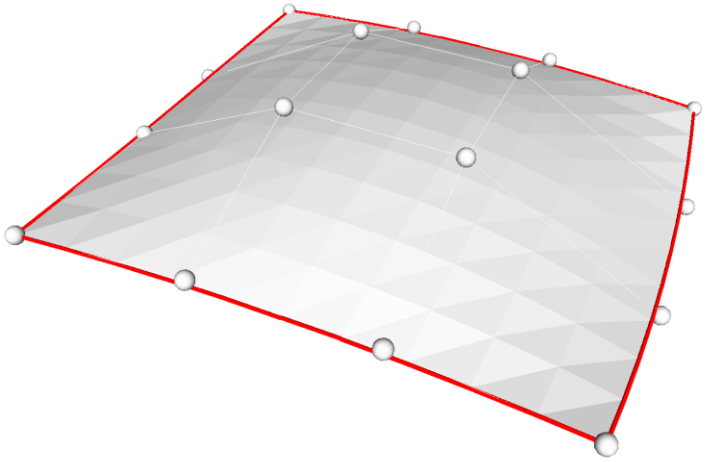
$$\begin{aligned}\boldsymbol{x} &= (1-t)\boldsymbol{m}_2 + t\boldsymbol{m}_4 \\ &= (1-t)\big((1-s)\boldsymbol{p}_{11} + s\boldsymbol{p}_{21}\big) \\ &+ \quad t\ \big((1-s)\boldsymbol{p}_{12} + s\boldsymbol{p}_{22}\big) \\ &= \sum_i \sum_j B_i(s)B_j(t)\boldsymbol{p}_{ij}\end{aligned}$$

曲线插值



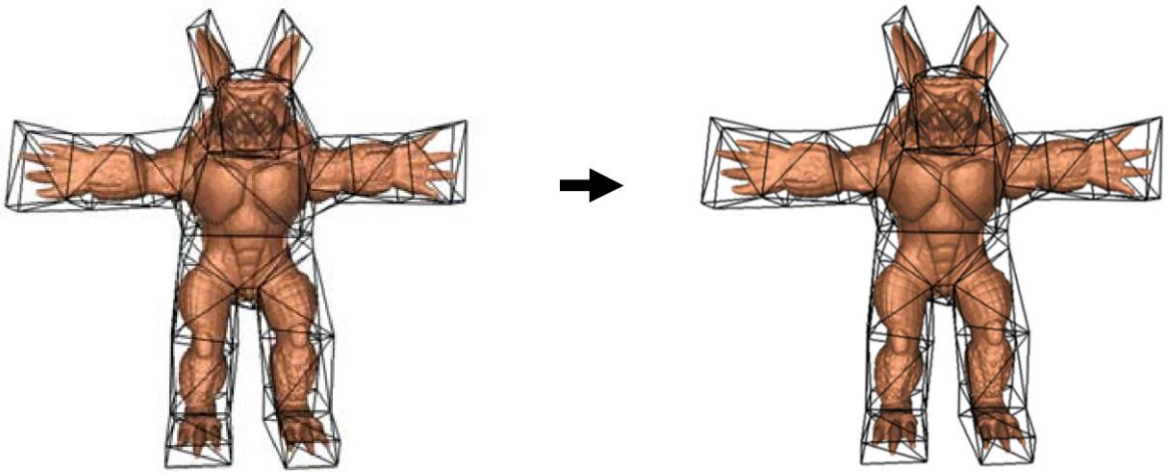
$$x = \sum_{i=0}^{n} \sum_{j=0}^{m} w_{i,j}(u,v)\boldsymbol{p}_{i,j}$$

Bézier Surface, NURBS Surface…

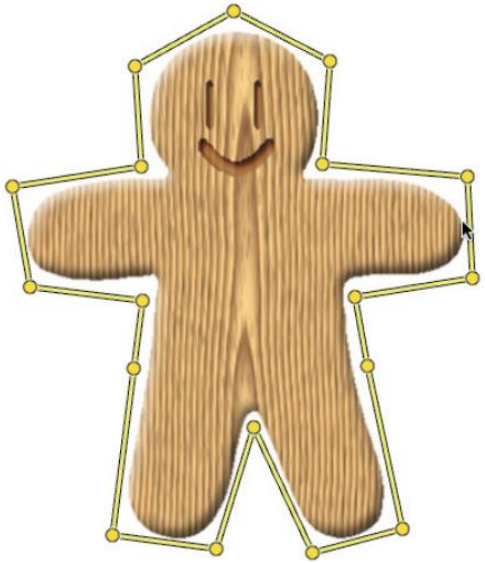


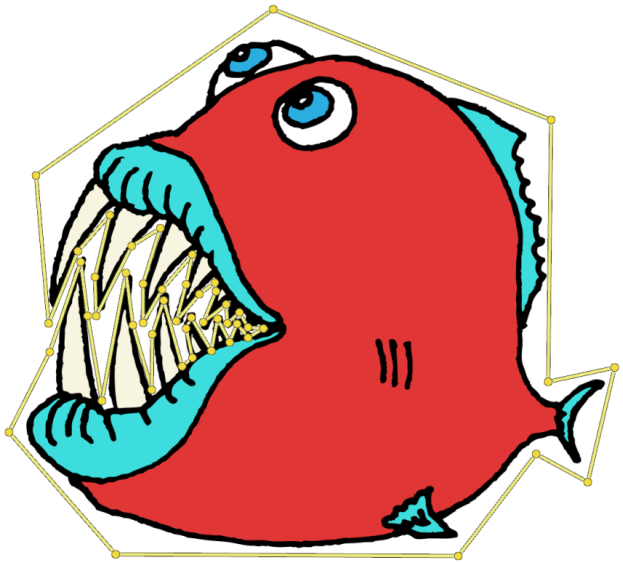
Cage-based Deformation

$$x_i = \sum_j \omega_{ij} p_j$$

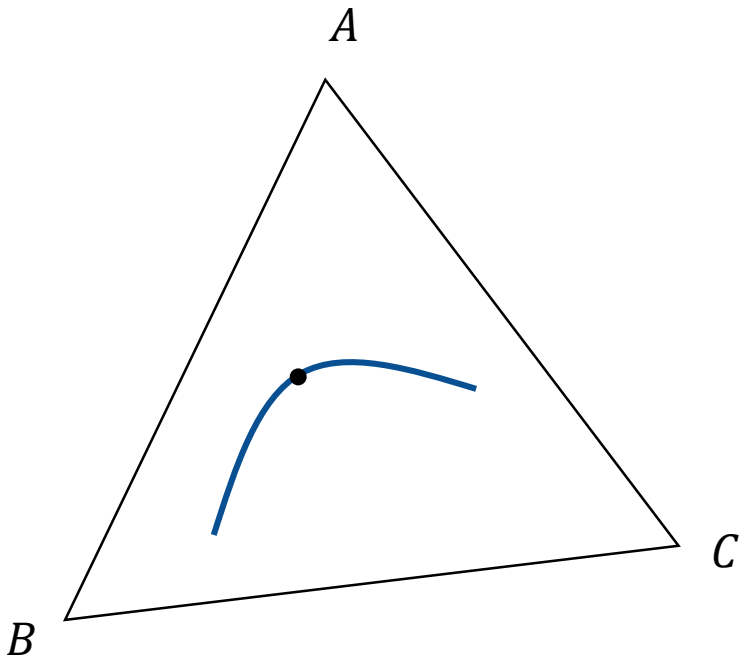


基于Cage计算权重

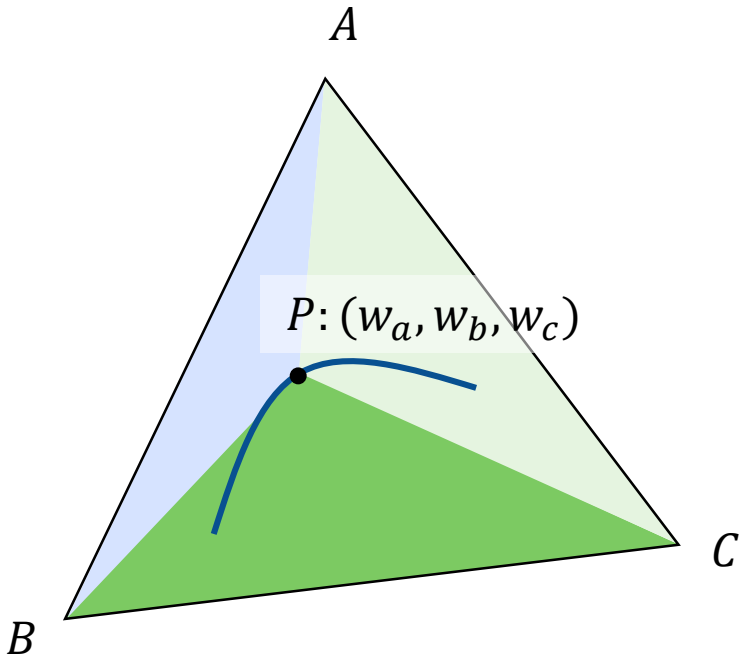




重心坐标



重心坐标



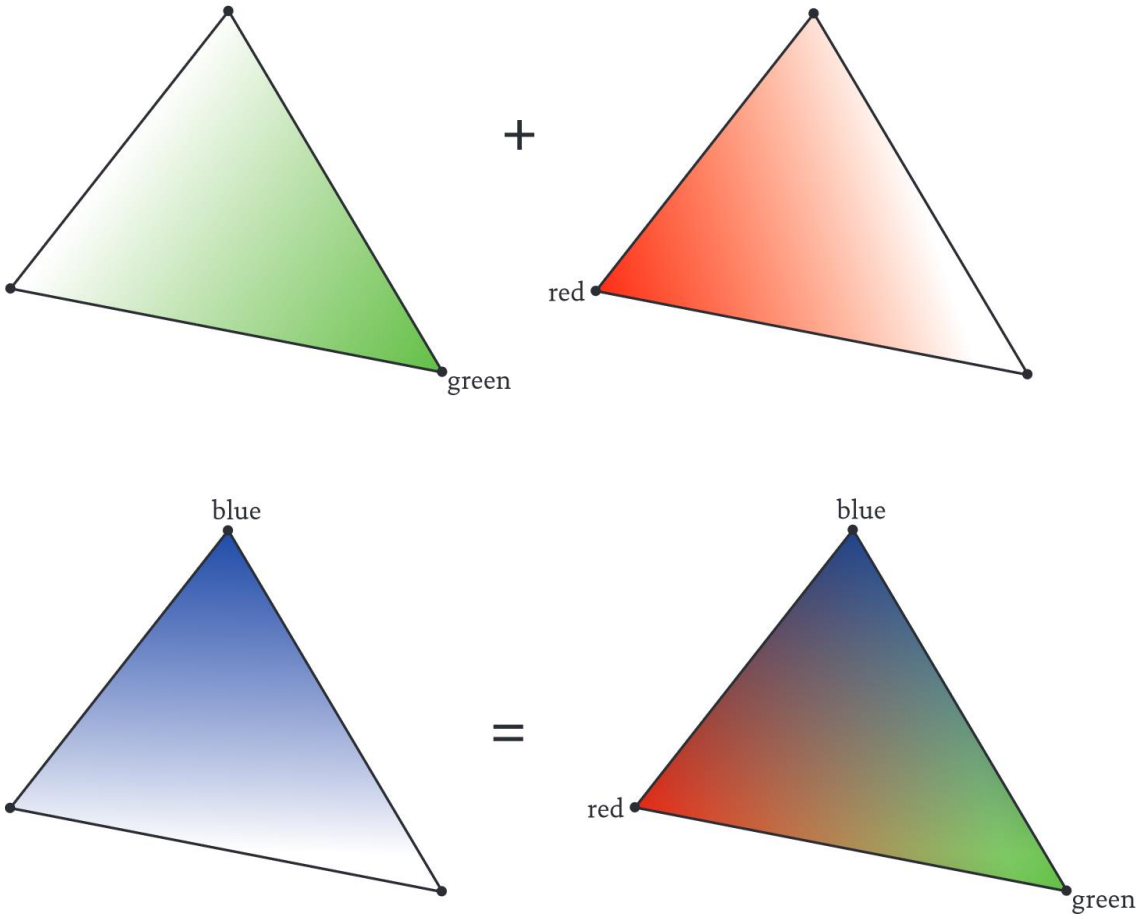
$$P = w_a A + w_b B + w_c C$$

$$w_a = \frac{\Delta PBC}{\Delta ABC}$$

$$w_b = \frac{\Delta PCA}{\Delta ABC}$$

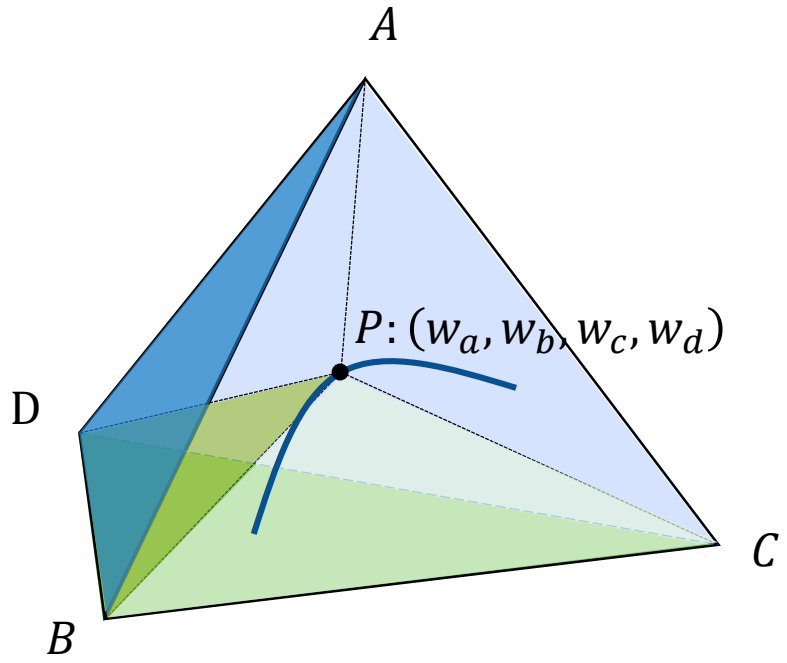
$$w_c = \frac{\Delta PAC}{\Delta ABC}$$

重心坐标



Lidberg, Petter. “Barycentric and Wachspress coordinates in two dimensions: theory and implementation for shape transformations.” (2011).

重心坐标



$$P = w_a A + w_b B + w_c C + w_d D$$

$$w_a = \frac{\Delta PBCD}{\Delta ABCD}$$

$$w_b = \frac{\Delta PCDA}{\Delta ABCD}$$

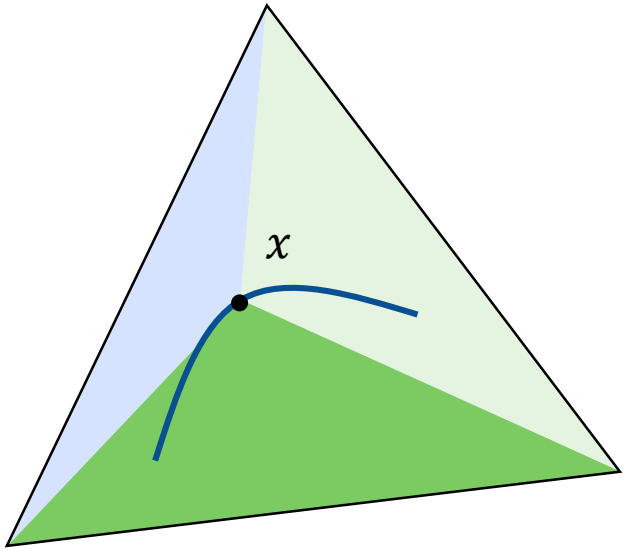
$$w_c = \frac{\Delta PDAB}{\Delta ABCD}$$

$$w_d = \frac{\Delta PABC}{\Delta ABCD}$$

三角形的重心坐标

$$x = \sum_i w_i p_i\,, \qquad \sum_i w_i = 1, \text{且 } w_i \geq 0$$

$$x = \frac{\sum_i w_i' p_i}{\sum_i w_i'}$$



- 可写作 $x = (w_0, w_1, \ldots, w_n)$
- 三角形中的点有唯一重心坐标（why?）
  - 不同坐标组合仅差常系数
  - Homogeneous coordinates
  - 可扩展到N-维单纯形

任意多边形的情况

$$x = \sum_i w_i p_i \,/\, \sum_i w_i$$

- 任意多边形重心坐标不唯一
  - *Wachspress* (WP) coordinates

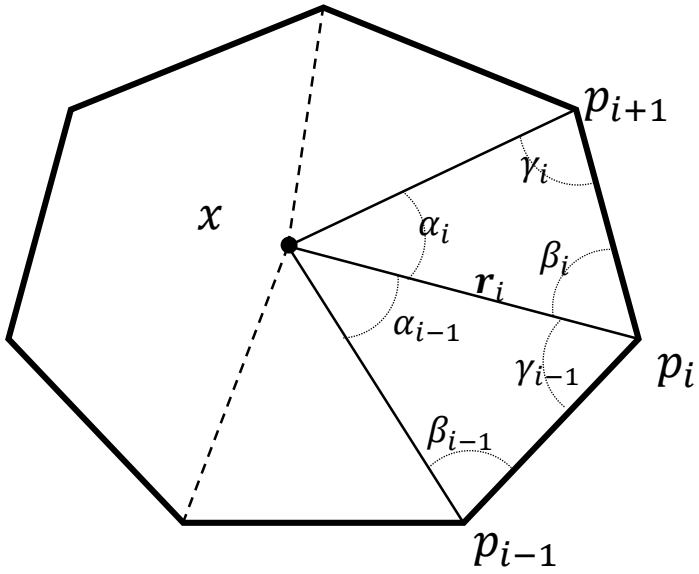
$$w_i = \frac{\cot \gamma_{i-1} + \cot \beta_i}{{r_i}^2}$$

  - *mean value* (MV) coordinates

$$w_i = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{r_i}$$

  - *discrete harmonic* (DH) coordinates

$$w_i = \cot \beta_{i-1} + \cot \gamma_i$$



https://www.inf.usi.ch/hormann/barycentric/

任意多边形的情况

$$x = \sum_i w_i p_i \,/\, \sum_i w_i$$

- 任意多边形重心坐标不唯一
  - *Wachspress* (WP) coordinates

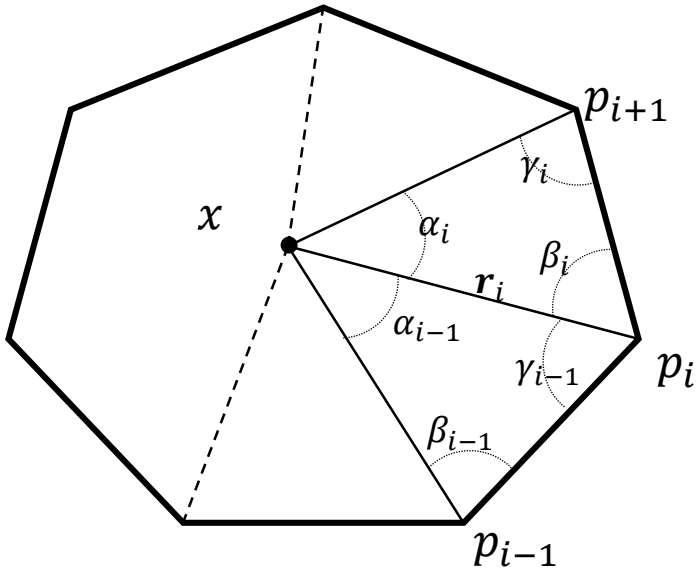
$$w_i = \frac{\cot \gamma_{i-1} + \cot \beta_i}{{r_i}^2}$$

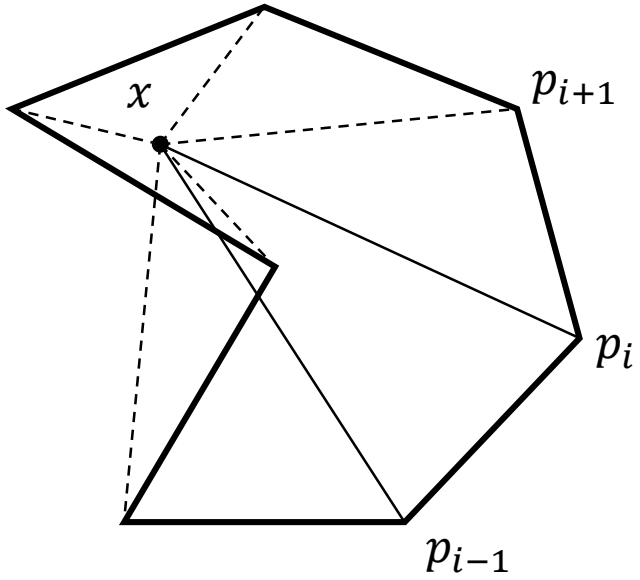
  - *mean value* (MV) coordinates

$$w_i = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{r_i}$$

  - *discrete harmonic* (DH) coordinates

$$w_i = \cot \beta_{i-1} + \cot \gamma_i$$

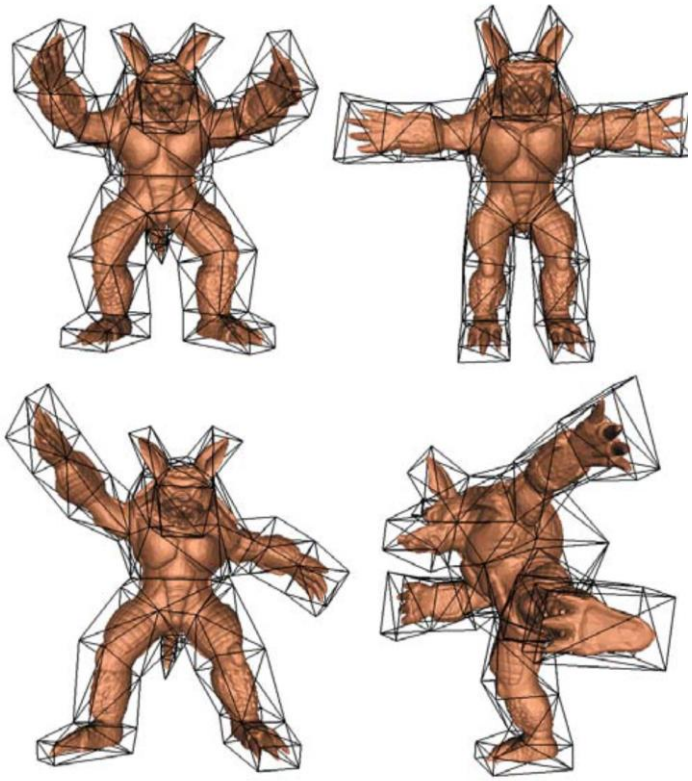




https://www.inf.usi.ch/hormann/barycentric/

任意多边形的情况

- Generalize to 3D Polyhedron is non-trivial
  - See https://www.inf.usi.ch/hormann/barycentric/

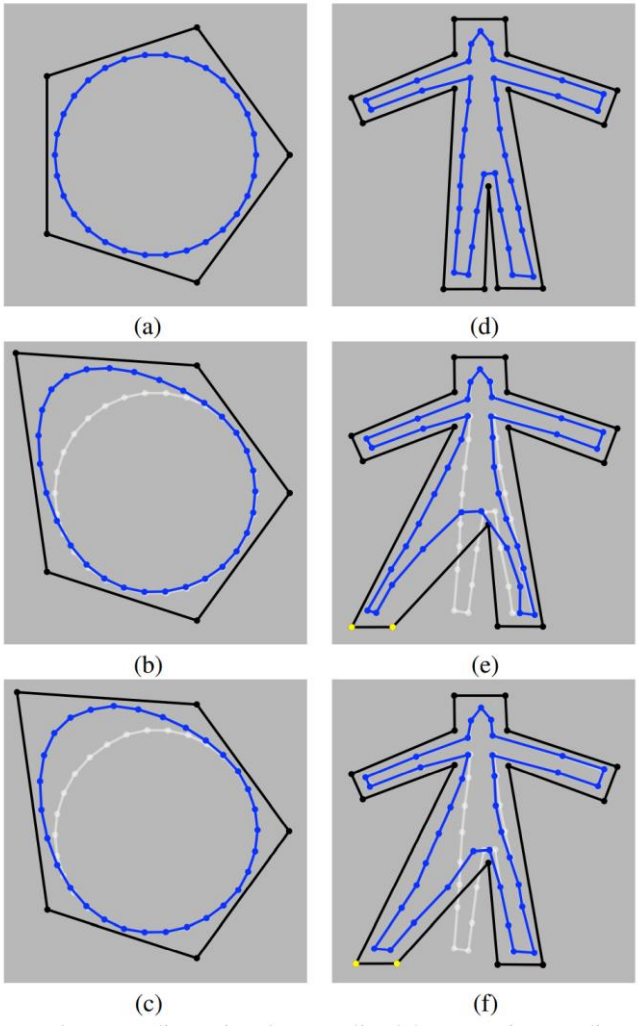


Ju, T., Schaefer, S., Warren, J.: ***Mean value coordinates for closed triangular meshes***. In: ACM SIGGRAPH 2005 Papers, SIGGRAPH '05, pp. 561–566.

更一般的情况

- Cage权重的一些性质:
  - 控制点本身的权重为1
  - 每个点的权重和为1
  - Cage内部权重连续/光滑
  - 权重非负
  - 支持线性插值

$$x_i = \sum_j \omega_{ij} p_j$$

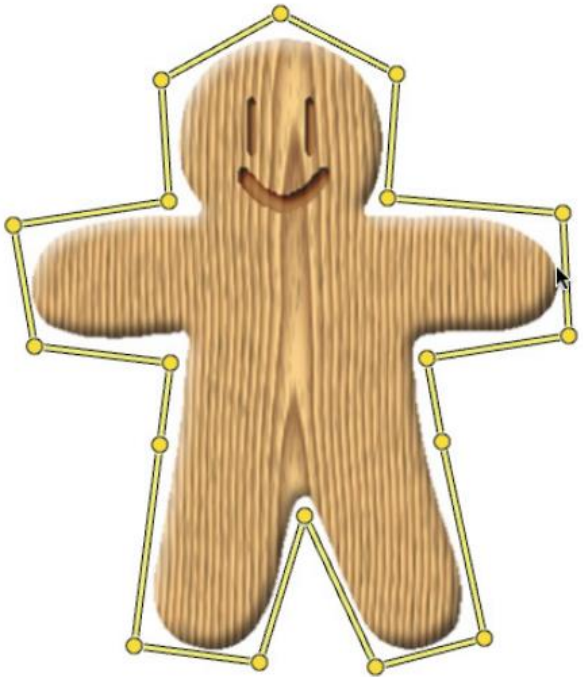


Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. 2007. ***Harmonic coordinates for character articulation***. ACM Trans. Graph. 26, 3 (July 2007), 71–es.

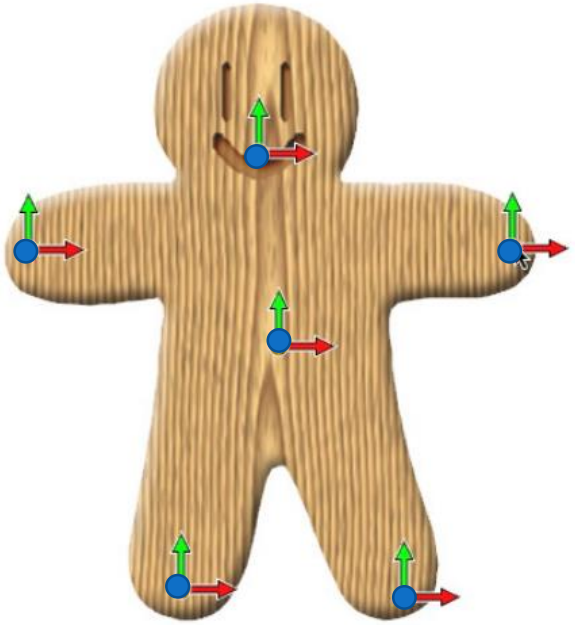
另一种形式的cage

$$x_i = \sum_j \omega_{ij} p_j$$

$$x_i = \sum_j \omega_{ij} p_j \quad ??$$



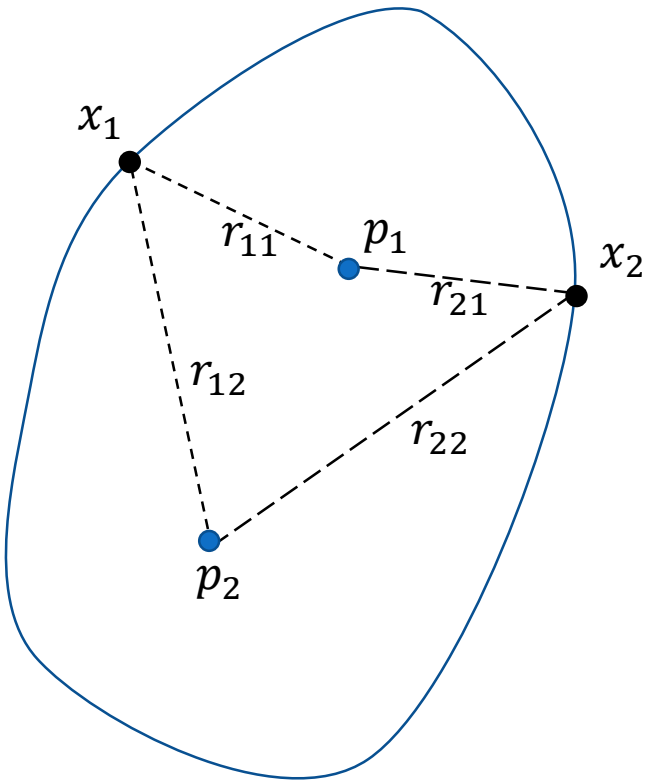
Cage



控制点

Alec Jacobson, Ilya Baran, Jovan Popović, Olga Sorkine-Hornung, ***Bounded Biharmonic Weights for Real-Time Deformation***
https://igl.ethz.ch/projects/bbw/

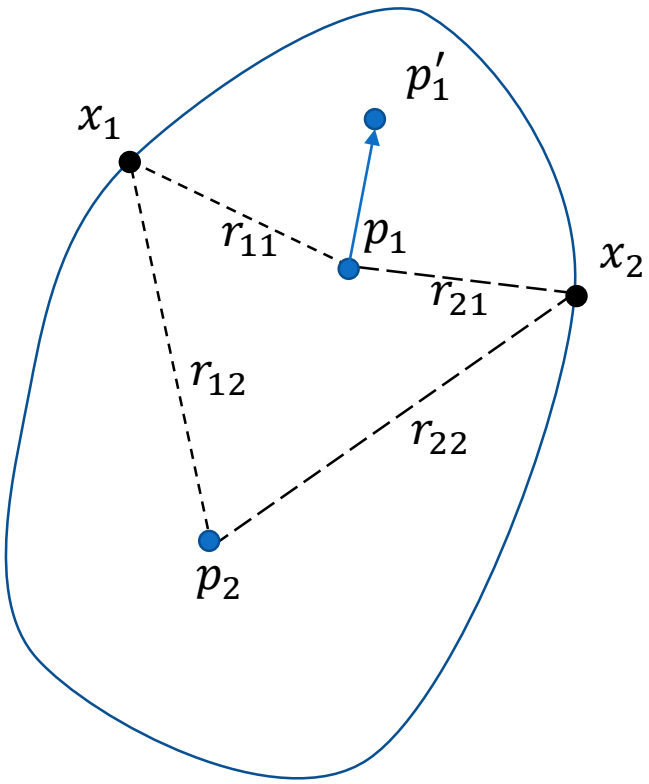
基于控制点的变形



$$x_i = \sum_j \omega_{ij}\left(x_i - p_j + p_j\right)$$

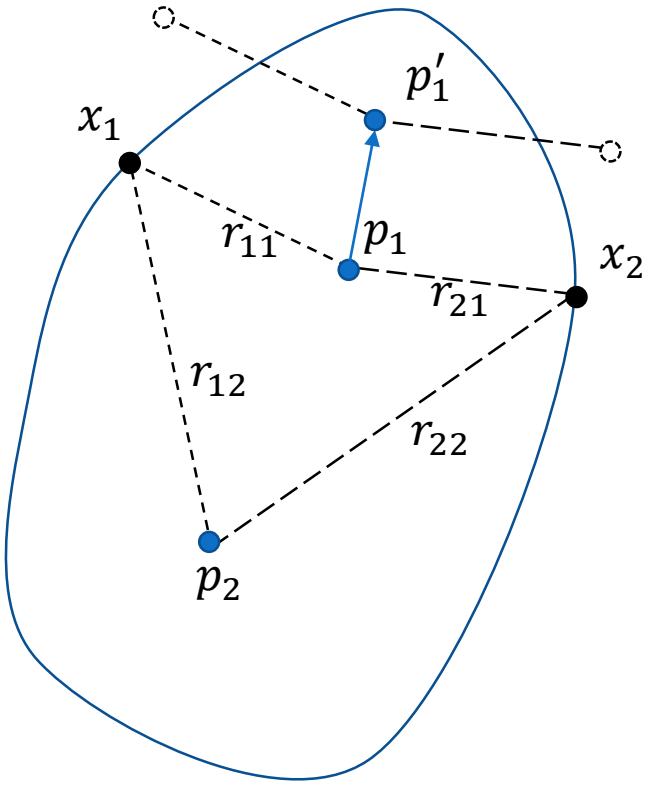
$$= \sum_j \omega_{ij}\left(r_{ij} + p_j\right)$$

基于控制点的变形



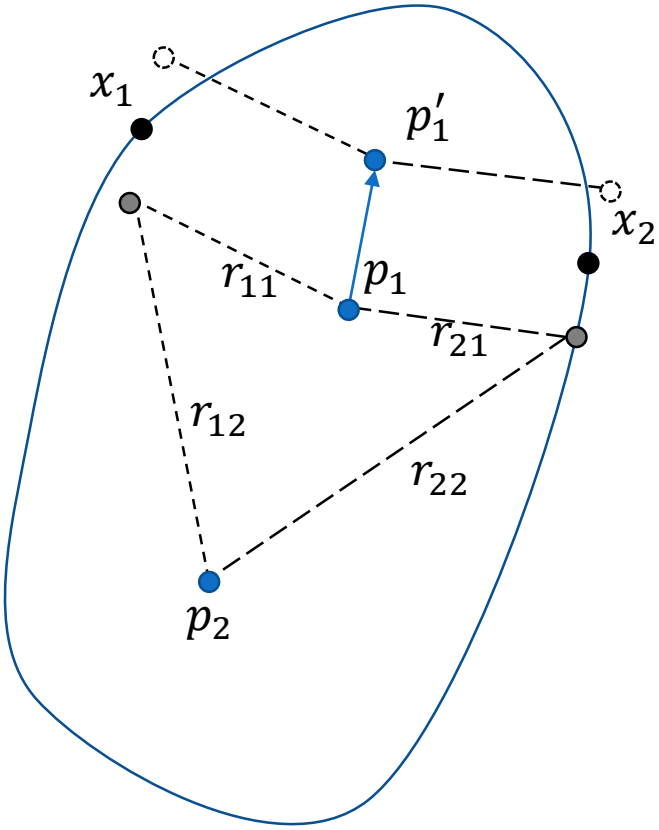
$$x_i' = \sum_j \omega_{ij}\left(r_{ij} + p_j'\right)$$

基于控制点的变形



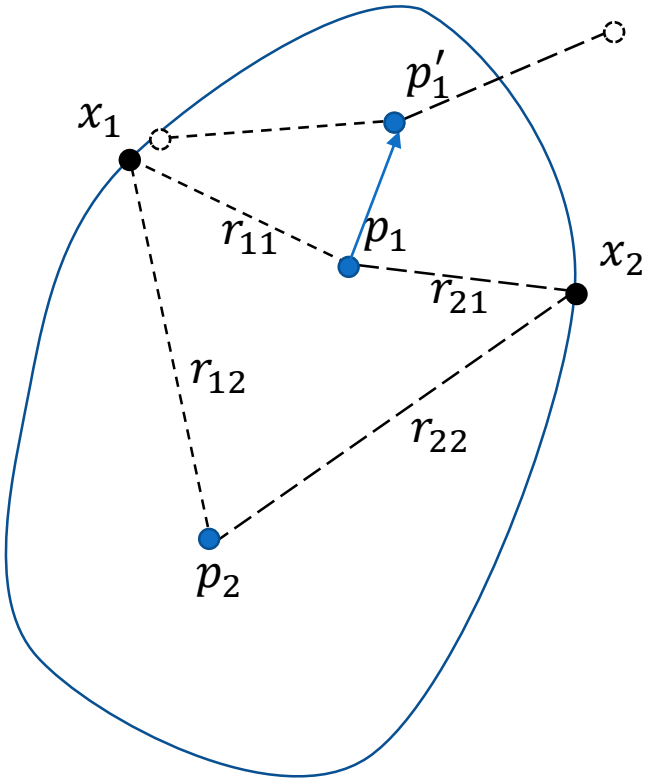
$$x_i' = \sum_j \omega_{ij}\left(r_{ij} + p_j'\right)$$

基于控制点的变形



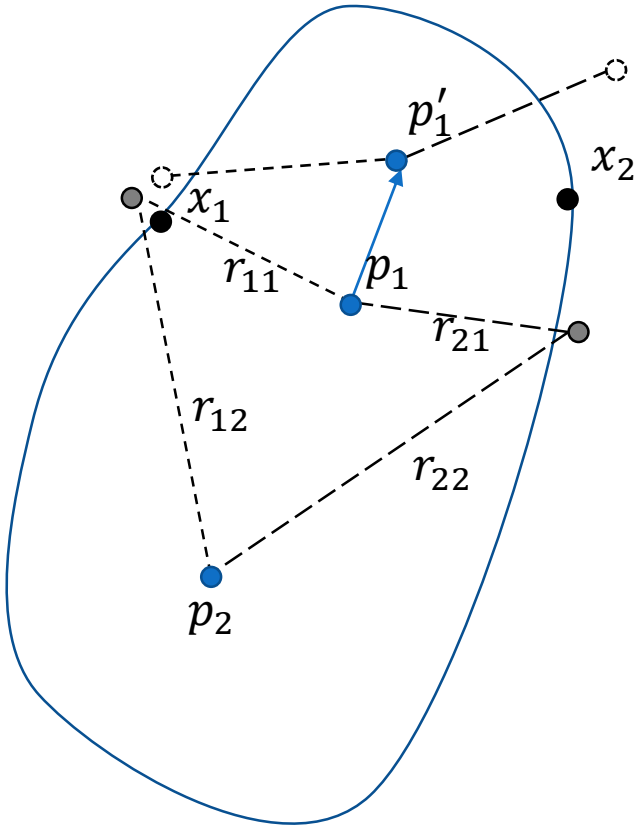
$$x_i' = \sum_j \omega_{ij}\left(r_{ij} + p_j'\right)$$

基于控制点的变形



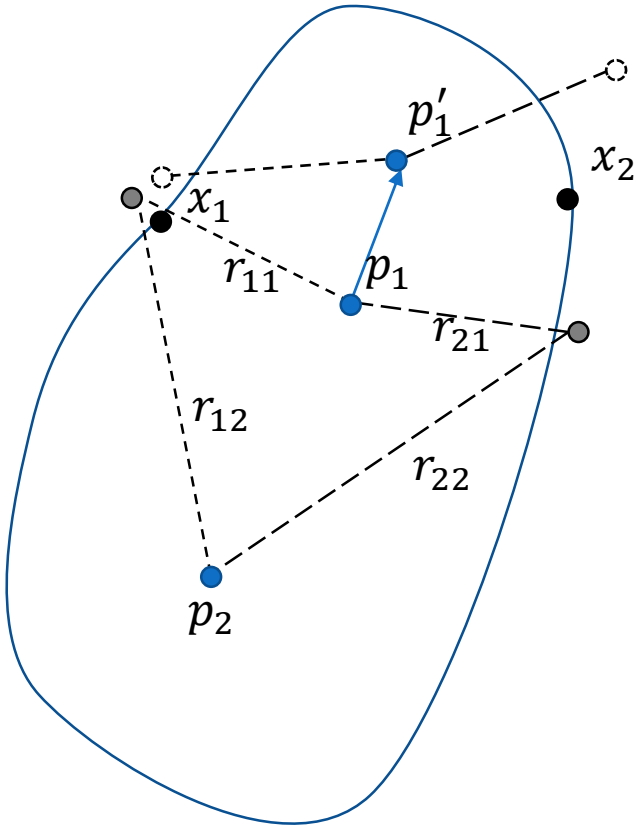
$$x_i' = \sum_j \omega_{ij}\left(R_j r_{ij} + p_j'\right)$$

基于控制点的变形



$$x_i' = \sum_j \omega_{ij}\left(R_j r_{ij} + p_j'\right)$$

基于控制点的变形

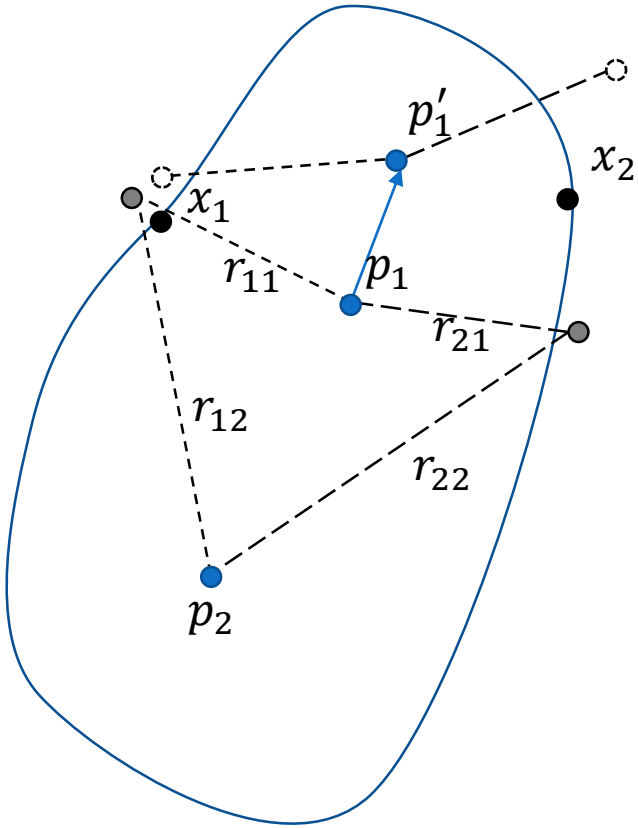


$$x_i' = \sum_j \omega_{ij}\left(R_j r_{ij} + p_j'\right)$$

$$r_{ij} = x_i - p_j$$

$$x_i' = \sum_j \omega_{ij}\left(R_j x_i + t_j\right)$$

基于控制点的变形



$$x_i' = \sum_j \omega_{ij}\left(R_j r_{ij} + p_j'\right)$$

$$r_{ij} = x_i - p_j$$

$$x_i' = \sum_j \omega_{ij}\left(R_j x_i + t_j\right)$$

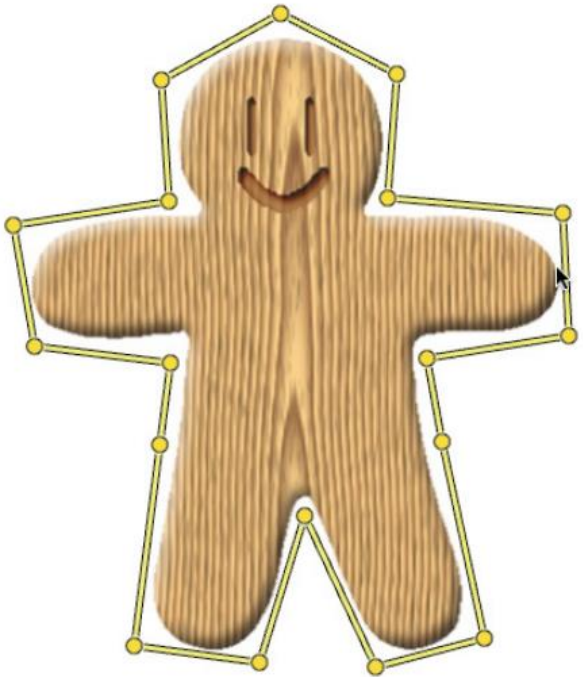
$$T = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} x_i' \\ 1 \end{pmatrix} = \sum_j \omega_{ij} T_j \begin{pmatrix} x_i \\ 1 \end{pmatrix}$$

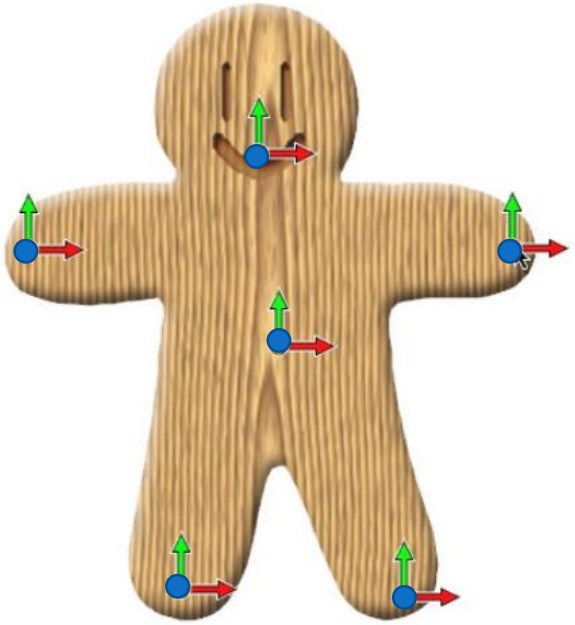
另一种形式的cage

$$x_i = \sum_j \omega_{ij} p_j$$

$$x_i' = \sum_j \omega_{ij} T_j x_i$$



Cage

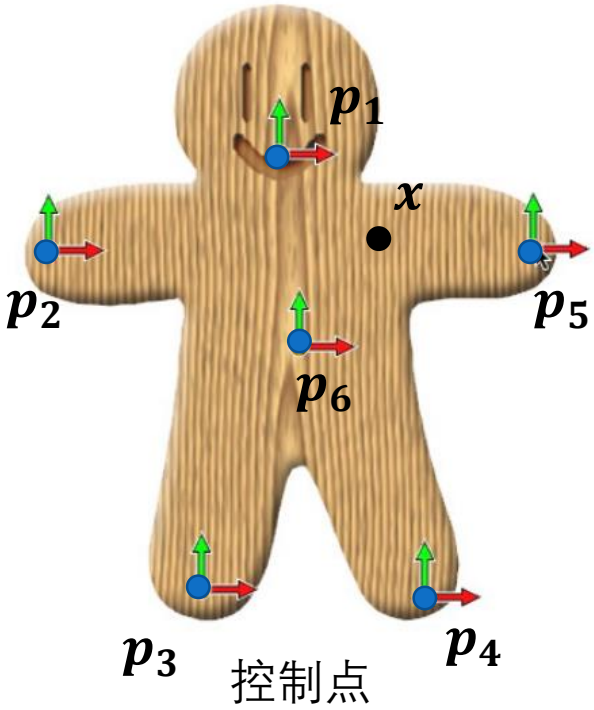


控制点

Alec Jacobson, Ilya Baran, Jovan Popović, Olga Sorkine-Hornung, ***Bounded Biharmonic Weights for Real-Time Deformation***
https://igl.ethz.ch/projects/bbw/

控制点权重的计算

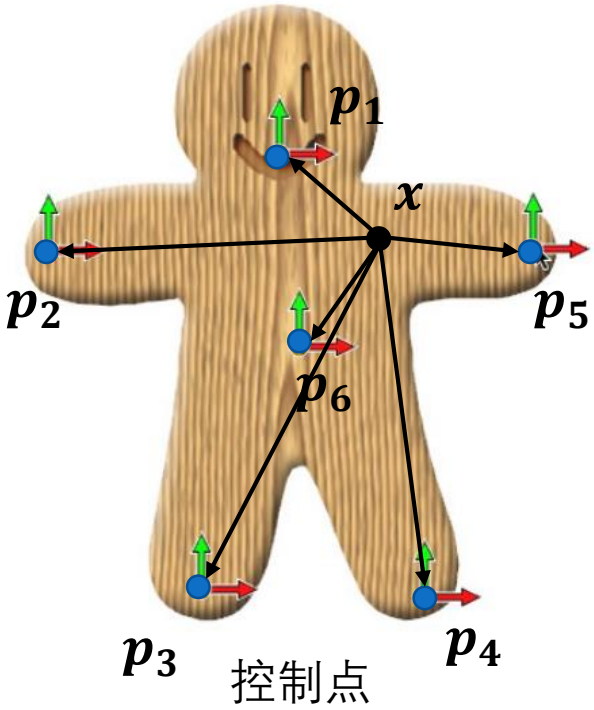
$$x_i' = \sum_j \omega_{ij} T_j x_i$$



控制点

控制点权重的计算

$$x_i' = \sum_j \omega_{ij} T_j x_i$$



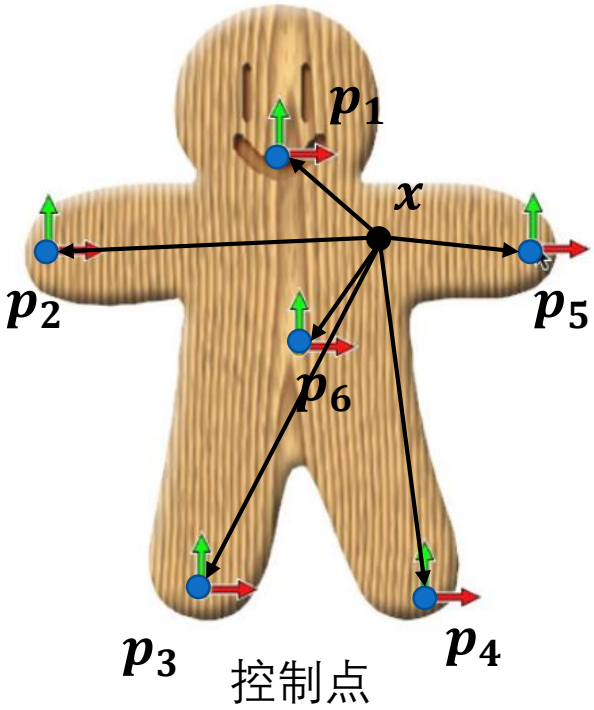
基于距离的权重

欧氏距离：$d_j = \|\boldsymbol{x} - \boldsymbol{p}_j\|$

权重：$w_{ij} = \begin{cases} 1, & d_j < d_k, \forall k \neq j \\ 0, & \text{otherwise} \end{cases}$

控制点权重的计算

$$x_i' = \sum_j \omega_{ij} T_j x_i$$



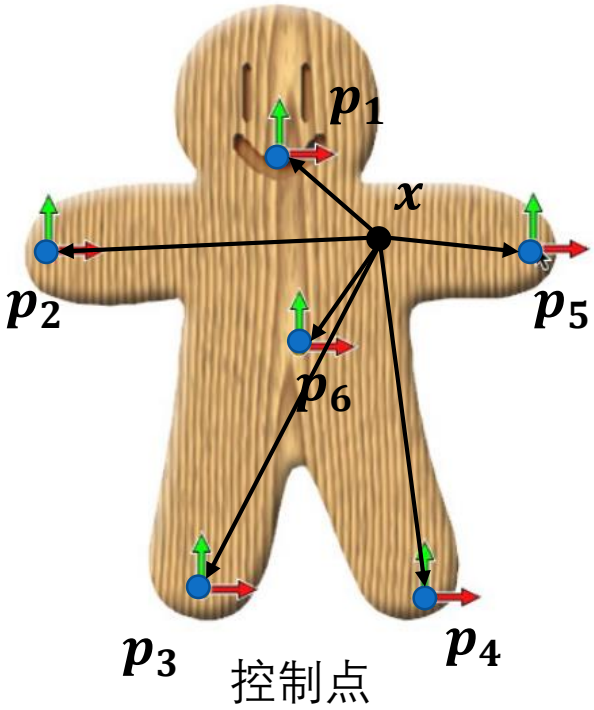
基于距离的权重

欧氏距离：$d_j = \|x - p_j\|$

权重：$w_{ij} = \frac{1}{d_j^p}$

控制点权重的计算

$$x_i' = \sum_j \omega_{ij} T_j x_i$$



基于距离的权重

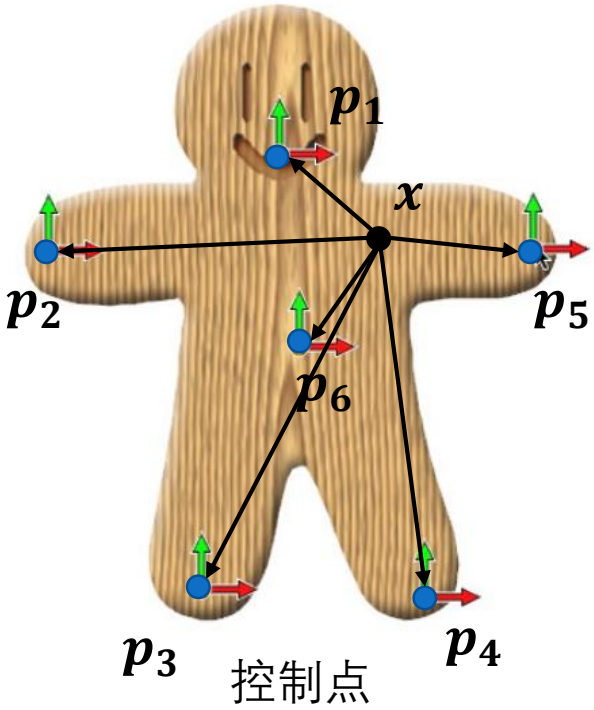
欧氏距离：$d_j = \|x - p_j\|$

权重：$\hat{w}_{ij} = \frac{1}{d_j^p}$

$$w_{ij} = \frac{\hat{w}_{ij}}{\sum_j \hat{w}_{ij}}$$

控制点权重的计算

$$x_i' = \sum_j \omega_{ij} T_j x_i$$



基于距离的权重

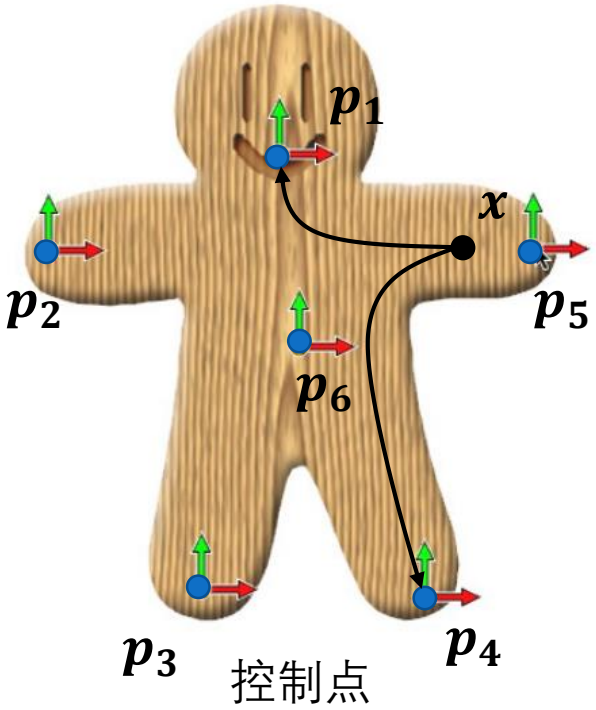
欧氏距离：$d_j = \|x - p_j\|$

权重：$\hat{w}_{ij} = \frac{1}{d_j^p + \varepsilon}$

$$w_{ij} = \frac{\hat{w}_{ij}}{\sum_j \hat{w}_{ij}}$$

控制点权重的计算

$$x_i' = \sum_j \omega_{ij} T_j x_i$$



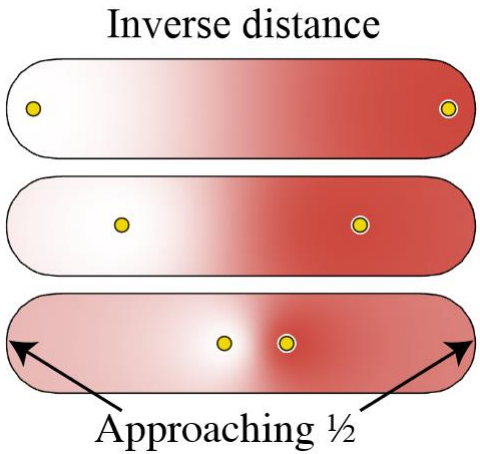
基于距离的权重

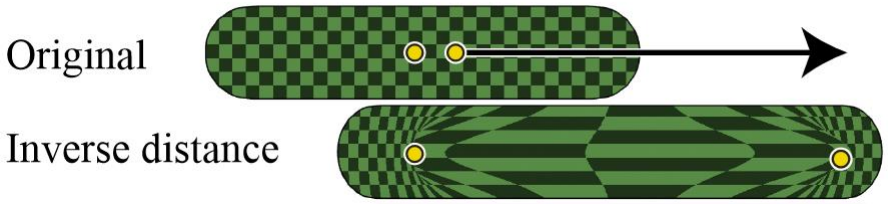
几何距离：$d_j = d(x, \boldsymbol{p}_i)$

权重：$\hat{w}_{ij} = \frac{1}{d_j^p + \varepsilon}$

$$w_{ij} = \frac{\hat{w}_{ij}}{\sum_j \hat{w}_{ij}}$$

控制点权重的计算





Alec Jacobson, Zhigang Deng, Ladislav Kavan, and J. P. Lewis. 2014. ***Skinning: real-time shape deformation****. In ACM SIGGRAPH 2014 Courses (SIGGRAPH '14)*

控制点权重的计算

- 基于能量优化的控制点权重
  - 权重在控制点处为1，平滑过渡到整个空间
  - 平滑能量函数

Common name	Least squares description	Energy	PDE
Dirichlet	f should be as constant as possible	$\int_\Omega \|\nabla f\|^2 dV$	$\Delta f = 0$
Laplacian	f should be as harmonic as possible	$\int_\Omega (\Delta f)^2 dV$	$\Delta^2 f = 0$
Laplacian gradient	Δf should be as constant as possible	$\int_\Omega \|\nabla \Delta f\|^2 dV$	$\Delta^3 f = 0$
...			

控制点权重的计算

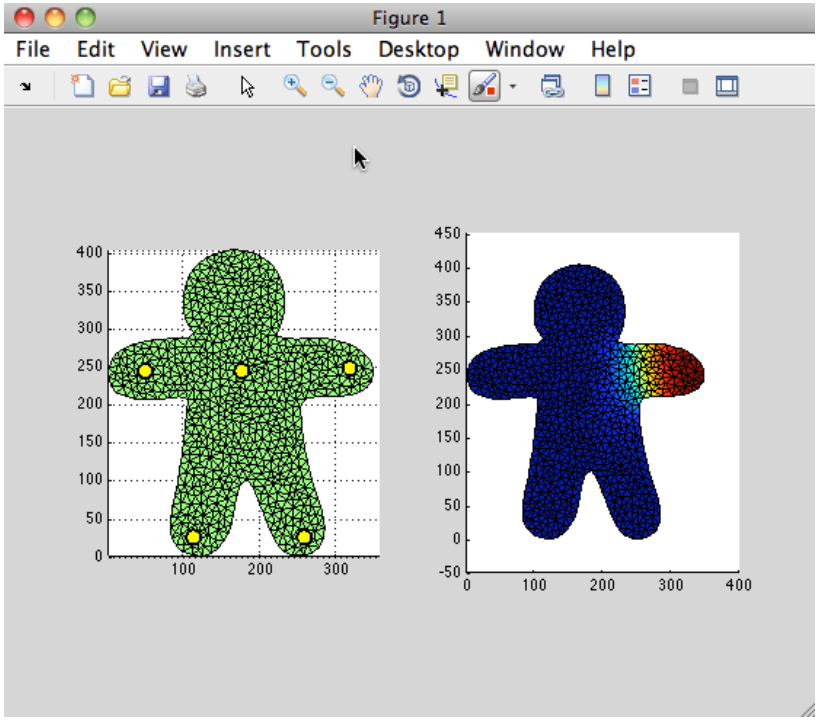
Bounded Biharmonic Weights

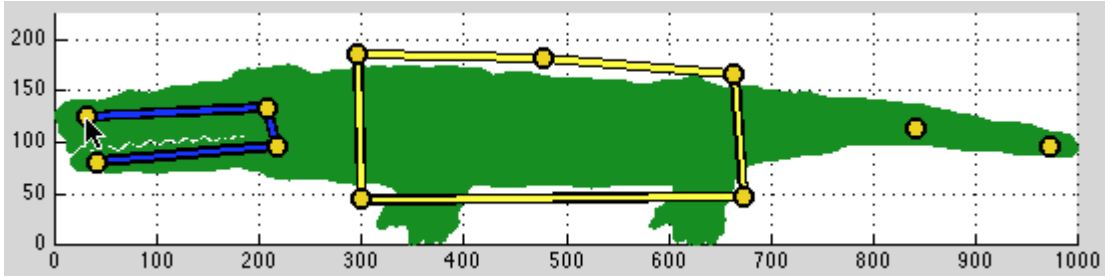
$$\underset{w_1,\ldots,w_m}{\operatorname{argmin}} \sum_{j=1}^{m} \frac{1}{2} \int_{\Omega} (\Delta w_j(\mathbf{v}))^2 \, dV,$$

$$\text{subject to } w_j(\mathbf{v}) = \varphi_j(\mathbf{v}) \; \forall \mathbf{v} \in H,$$

$$w_j(\mathbf{v}) > 0,$$

$$\sum_{j=1}^{m} w_j(\mathbf{v}) = 1.$$

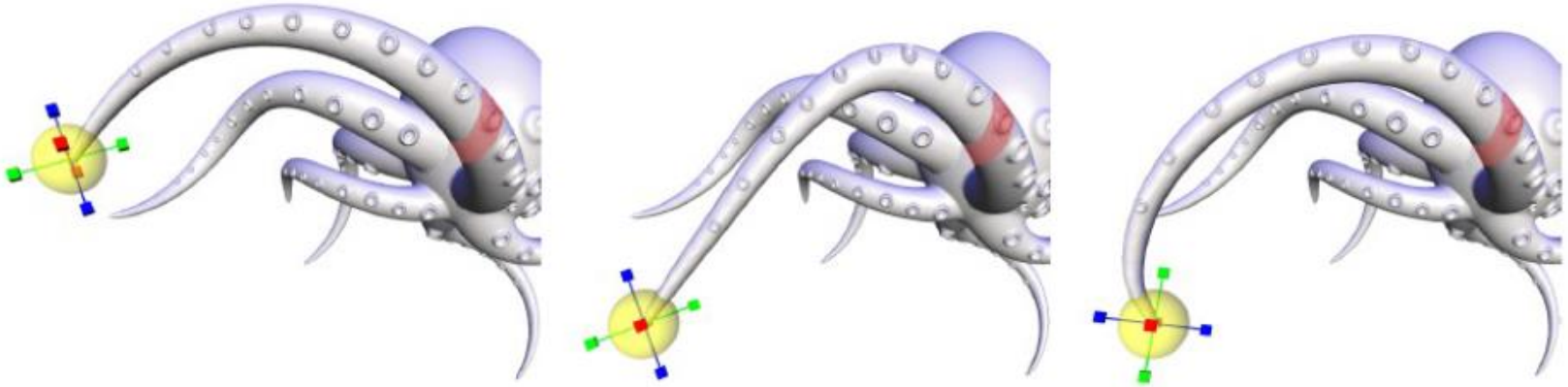




Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine-Hornung. 2014. ***Bounded biharmonic weights for real-time deformation***. *Commun. ACM* 57, 4 (April 2014)

作为优化问题的几何形变

- 更加直接的形变编辑
  - 直接操作几何模型上的点
  - 将修改平滑地施加在整个几何模型上
  - 构造优化问题求解

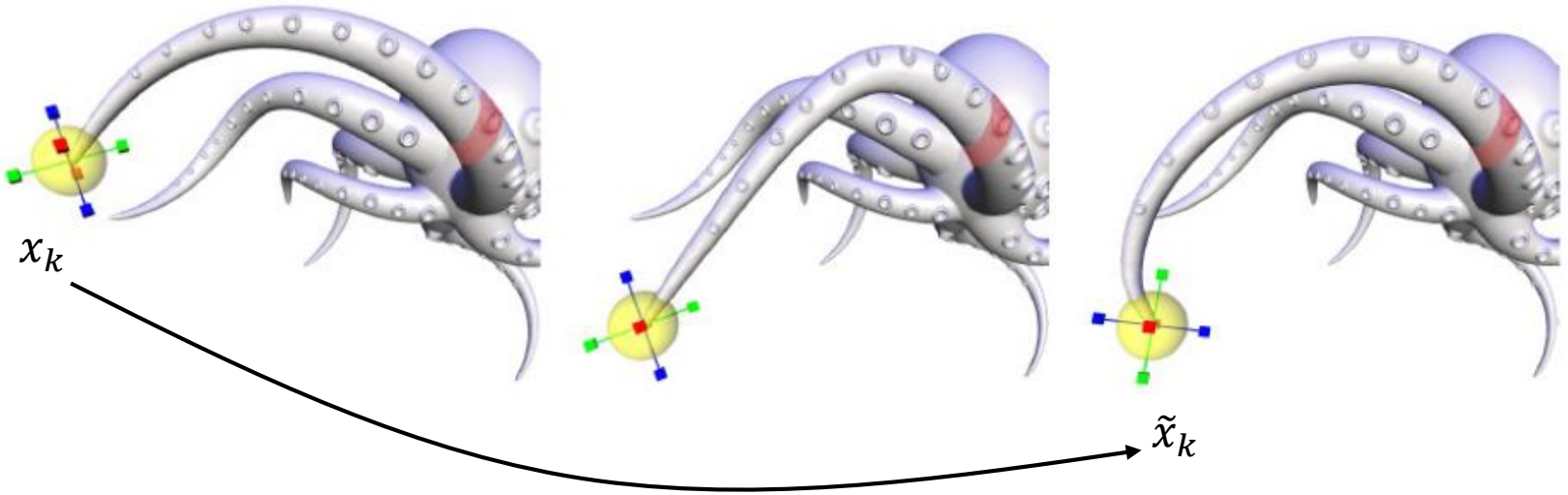


作为优化问题的几何形变

$$\min_{X'} \sum_{k} \|x'_k - \tilde{x}_k\|^2 + E(X, X')$$

编辑约束

平滑能量



作为优化问题的几何形变

$$\min_{X'} \sum_{k} \|x'_k - \tilde{x}_k\|^2 + E(X, X')$$

编辑约束

平滑能量

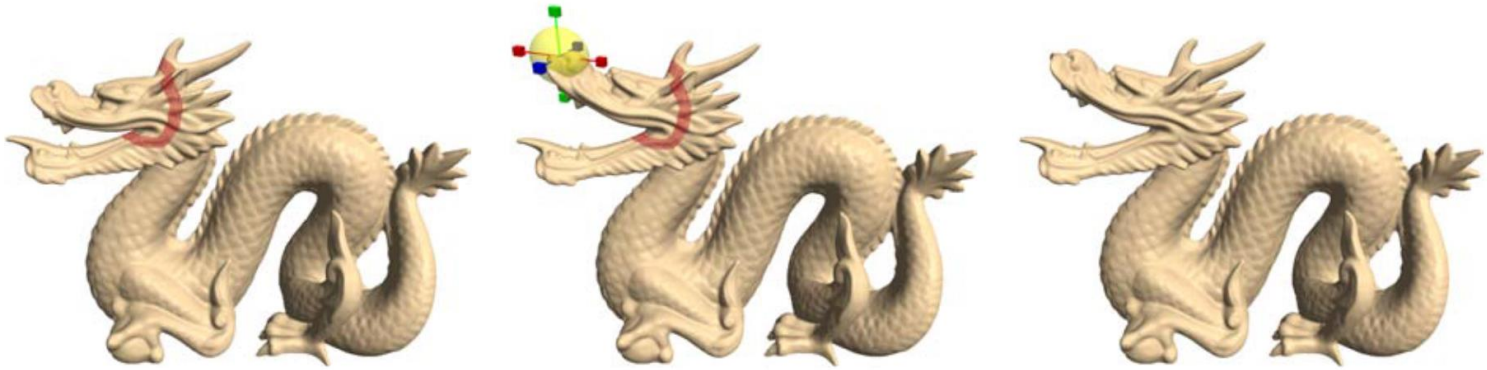
Common name	Least squares description	Energy	PDE
Dirichlet	f should be as constant as possible	$\int_\Omega \|\nabla f\|^2 dV$	$\Delta f = 0$
Laplacian	f should be as harmonic as possible	$\int_\Omega (\Delta f)^2 dV$	$\Delta^2 f = 0$
Laplacian gradient	Δf should be as constant as possible	$\int_\Omega \|\nabla \Delta f\|^2 dV$	$\Delta^3 f = 0$
...			

Laplacian Mesh Editing

编辑约束

平滑能量

$$\min_{X'} \sum_{k} \|x'_k - \tilde{x}_k\|^2 + \sum_{i} \|T_i(X')\mathcal{L}(x_i) - \mathcal{L}(x'_i)\|^2$$



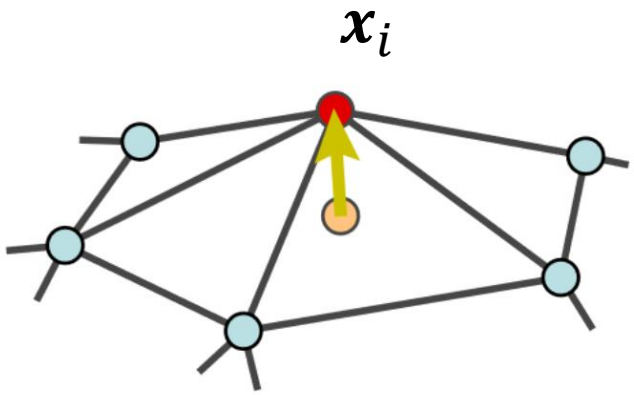
O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. 2004. ***Laplacian surface editing***. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing* (SGP '04), Association for Computing Machinery, New York, NY, USA, 175–184.

Laplacian坐标

$$\min_{X'} \sum_k \|x'_k - \tilde{x}_k\|^2 + \sum_i \|T_i(X')\mathcal{L}(x_i) - \mathcal{L}(x'_i)\|^2$$

$$\mathcal{L}(x) = \Delta \boldsymbol{f}(x)$$

Laplace-Beltrami operator



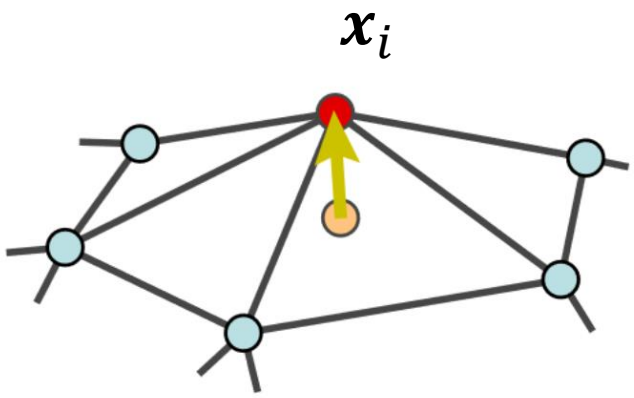
$\mathcal{N}_i$: 1-neighbors

Laplacian坐标

$$\min_{X'} \sum_k \|x'_k - \tilde{x}_k\|^2 + \sum_i \|T_i(X')\mathcal{L}(x_i) - \mathcal{L}(x'_i)\|^2$$

$$\mathcal{L}(\boldsymbol{x}_i) = \boldsymbol{x}_i - \frac{1}{d_i}\sum_{j\in\mathcal{N}_i} \boldsymbol{x}_j$$

“Umbrella” operator

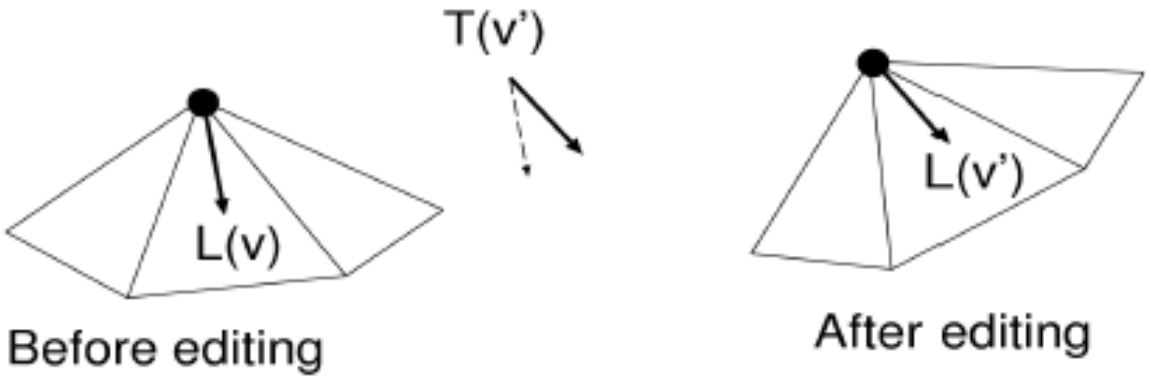


$\mathcal{N}_i$: 1-neighbors

Laplacian坐标

$$\min_{X'} \sum_k \|x'_k - \tilde{x}_k\|^2 + \sum_i \|T_i(X')\mathcal{L}(x_i) - \mathcal{L}(x'_i)\|^2$$

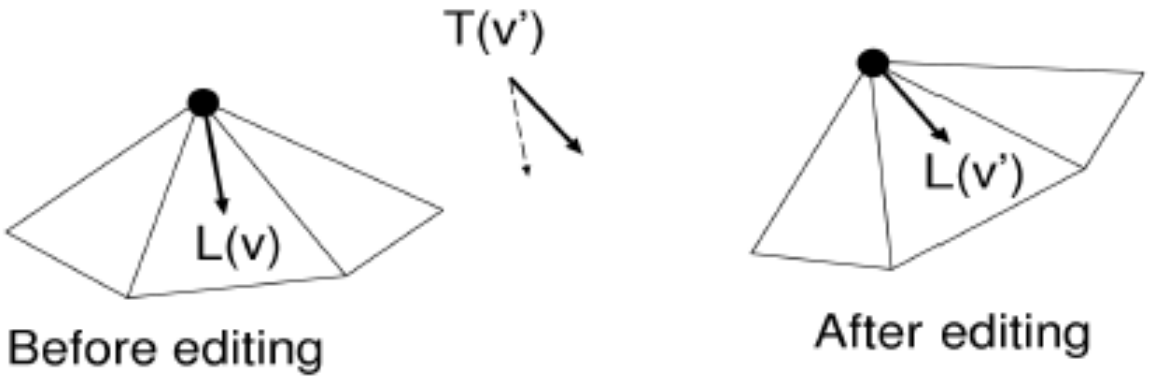
- 我们希望变换过后每个点的Laplacian坐标不变



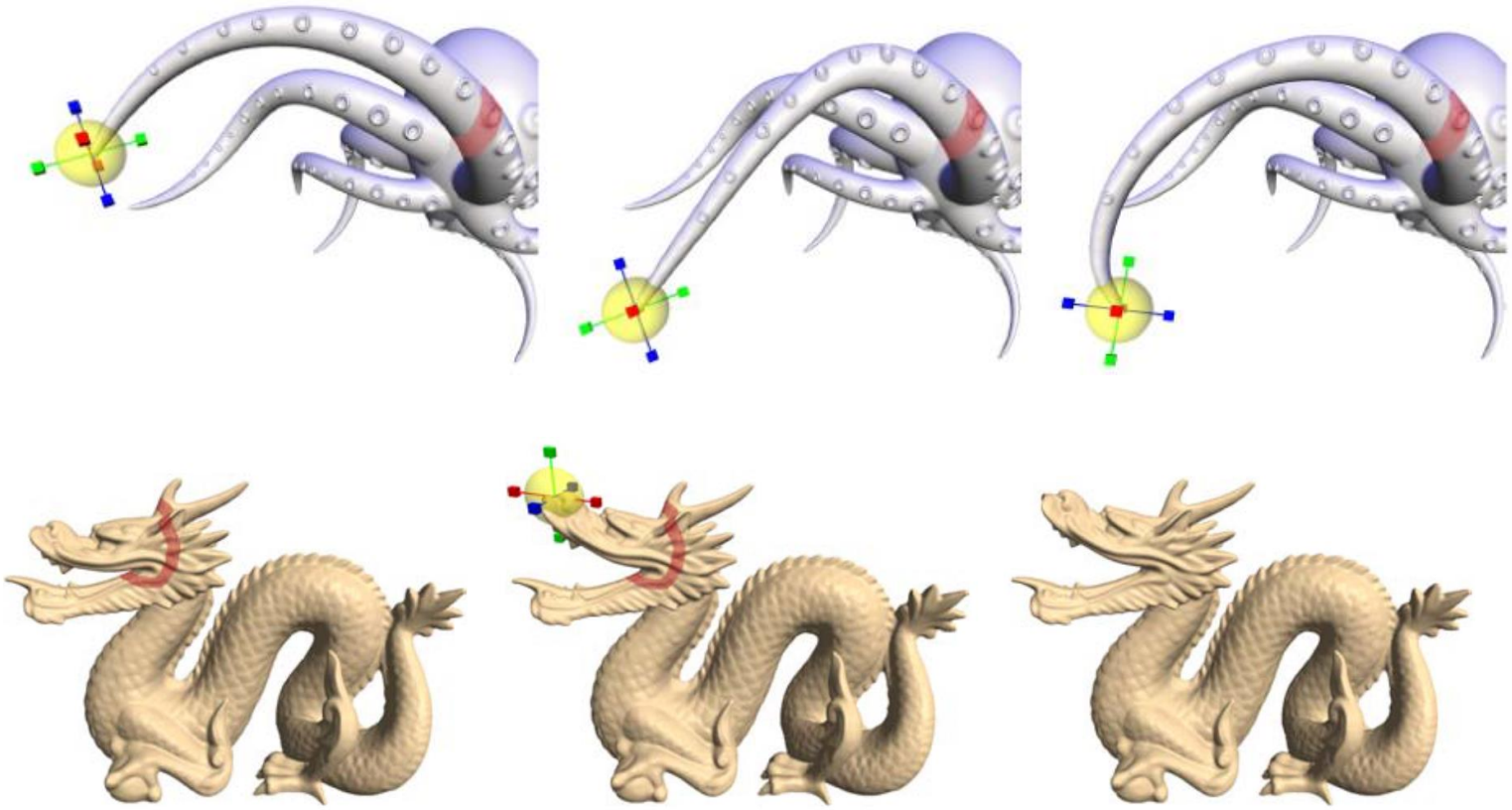
Laplacian坐标

$$\min_{X',T_i} \sum_k \|x'_k - \tilde{x}_k\|^2 + \sum_i \|T_i(X')\mathcal{L}(x_i) - \mathcal{L}(x'_i)\|^2$$

- 我们希望变换过后每个点的Laplacian坐标不变



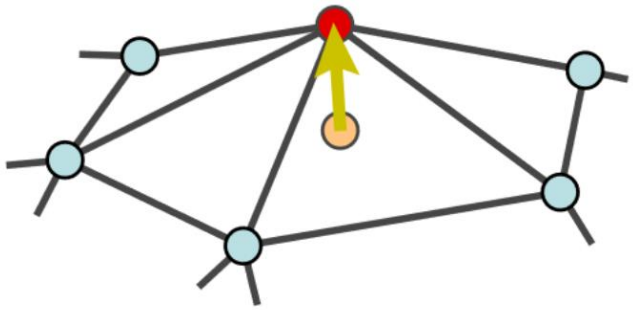
Laplacian Mesh Editing

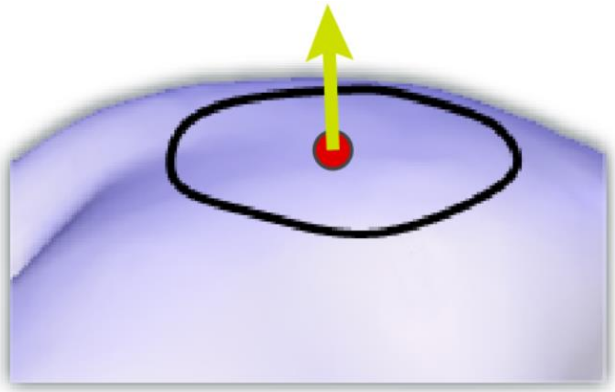


O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. 2004. ***Laplacian surface editing***. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing* (SGP '04), Association for Computing Machinery, New York, NY, USA, 175–184.

Laplacian坐标的含义

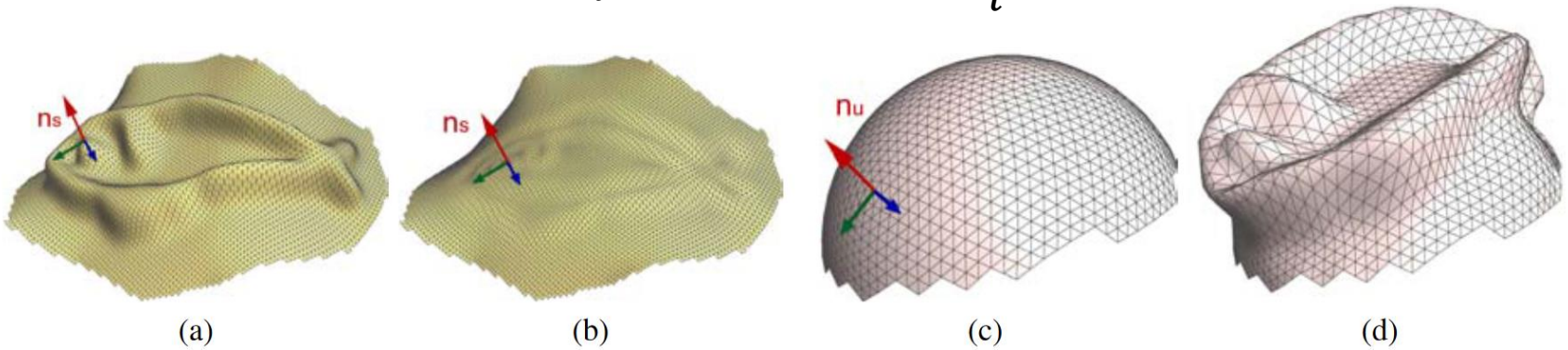
- 几何表面**局部**的细节
  - 方向 ≈ 法向
  - 大小 ≈ 曲率

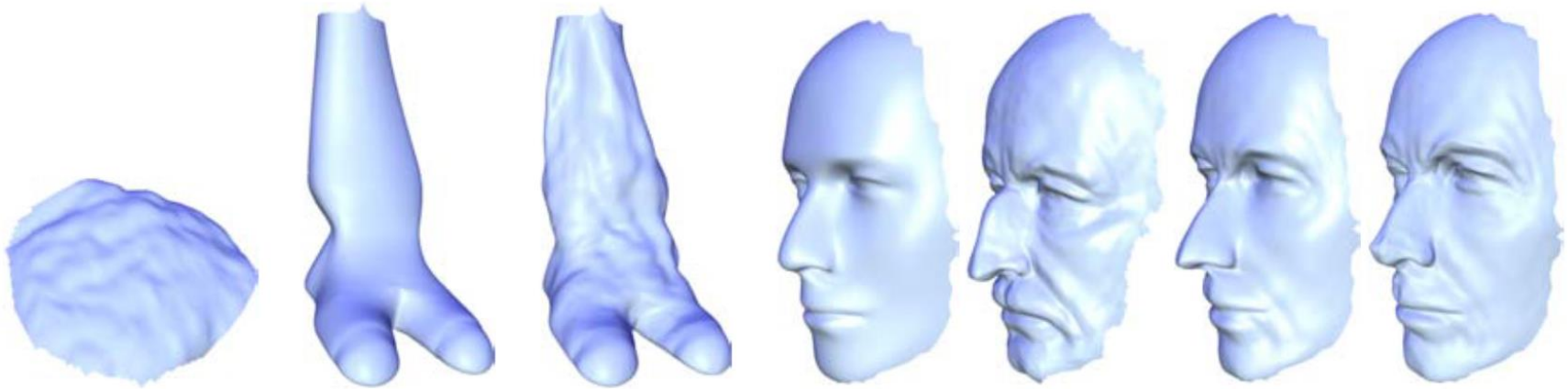




Coating Transfer

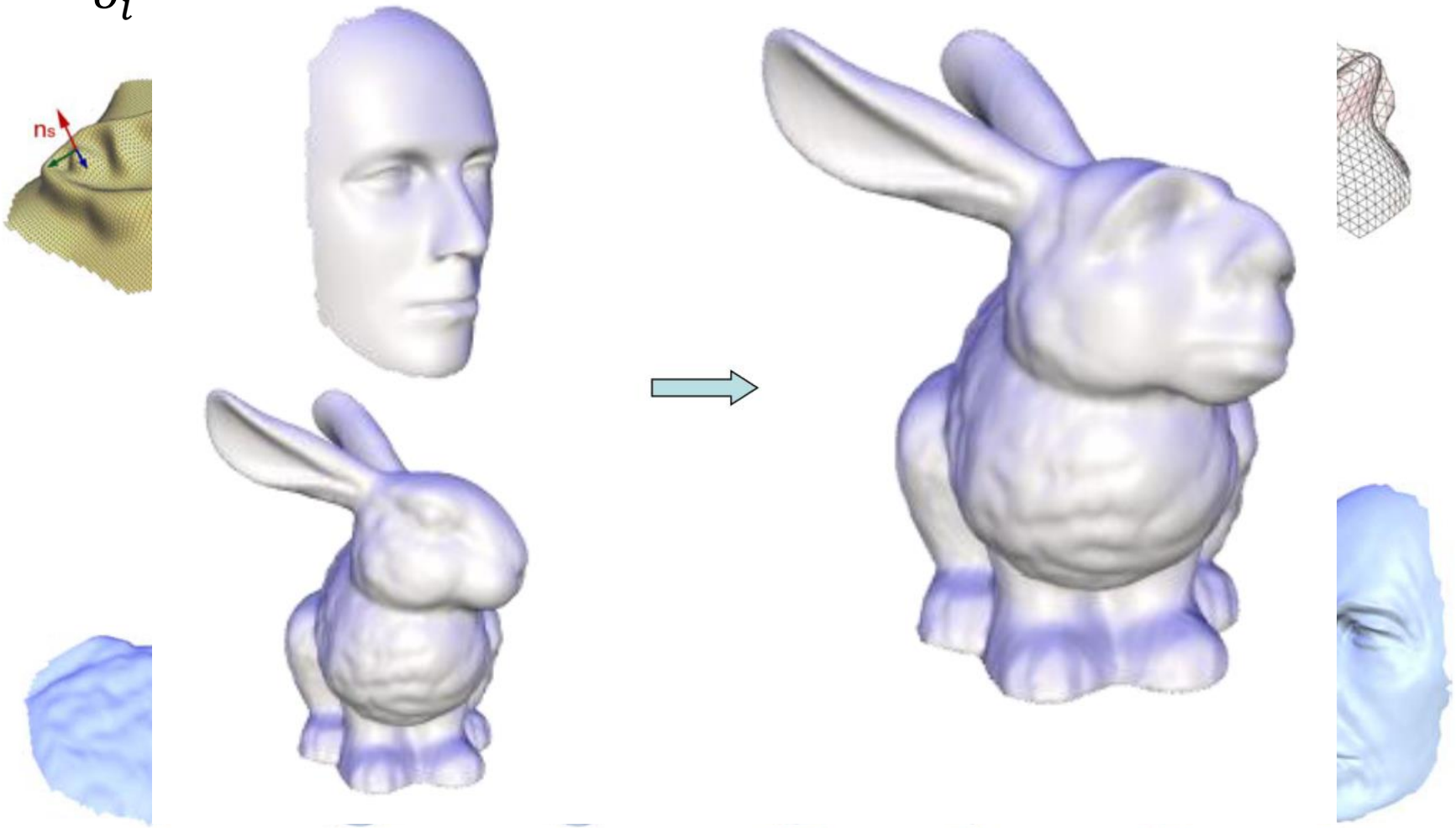
$$\delta_i = \mathcal{L}(x_i) - \widehat{\delta}_i + \widehat{\delta}'_i = \delta'_i$$





Coating Transfer

δ_i



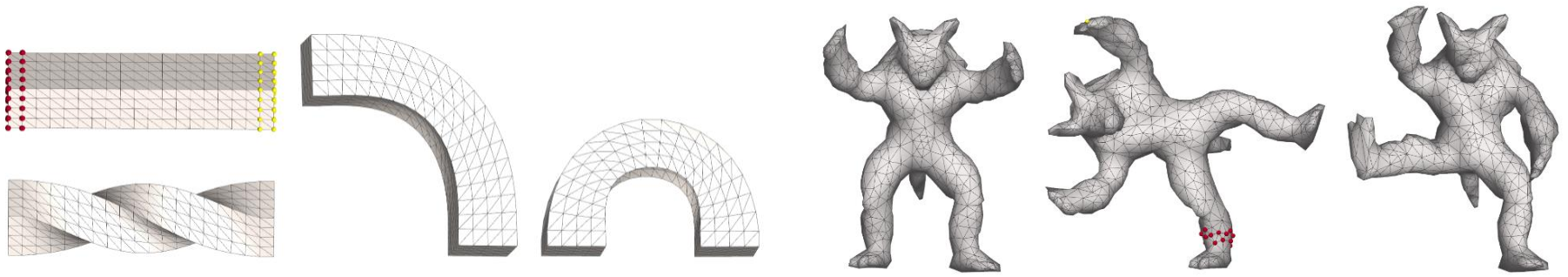
As Rigid As Possible Deformation

编辑约束

$$\min_{X'} \sum_k \|x'_k - \tilde{x}_k\|^2 + E(X, X')$$

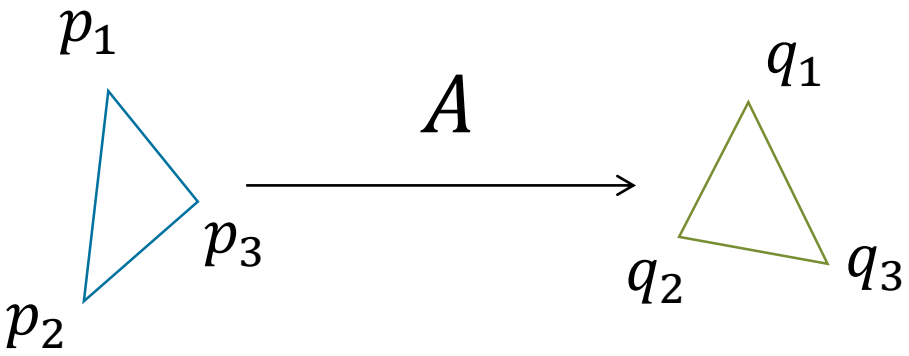
平滑能量项

尽可能减少非“刚性”形变 ➔ 减少不必要的缩放



Olga Sorkine and Marc Alexa. 2007. ***As-rigid-as-possible surface modeling***. In *Proceedings of the fifth Eurographics symposium on Geometry processing* (SGP ’07), Eurographics Association, Goslar, DEU, 109–116.

As Rigid As Possible Deformation

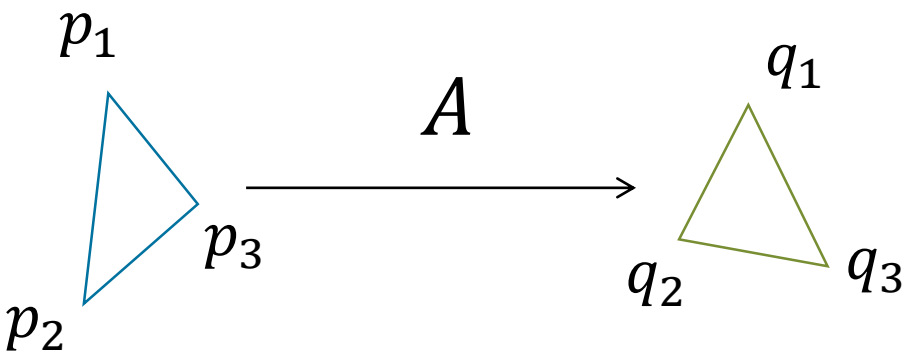


$$\boldsymbol{q}_i = A\boldsymbol{p}_i + \boldsymbol{t}$$

$$A = U\Sigma V^T = (UV^T)(VDV^T) = RS$$

How to find the “closest” rotation between two point sets?
See: https://igl.ethz.ch/projects/ARAP/svd_rot.pdf

As Rigid As Possible Shape Interpolation



Interpolation between two shapes

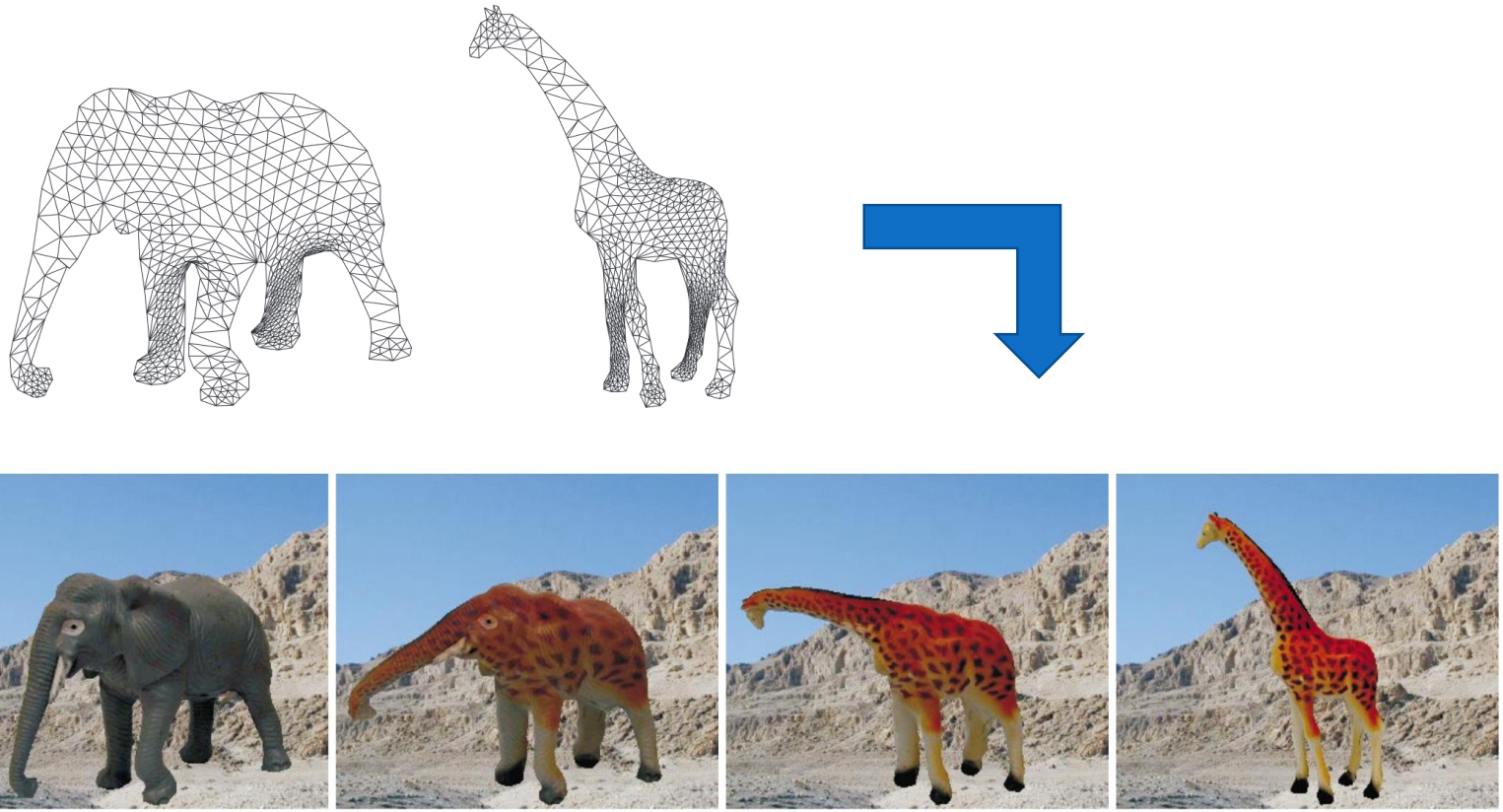
$$\boldsymbol{q}_i(t) = \mathrm{A}(t)\boldsymbol{p}_i + \boldsymbol{t}$$

$$\mathrm{A}(t) = R(t)\big((1-t)I + tS\big)$$

顶点线性插值

刚性变换插值

As Rigid As Possible Shape Interpolation



Marc Alexa, Daniel Cohen-Or, and David Levin. 2000. ***As-rigid-as-possible shape interpolation***. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '00)

As Rigid As Possible Deformation

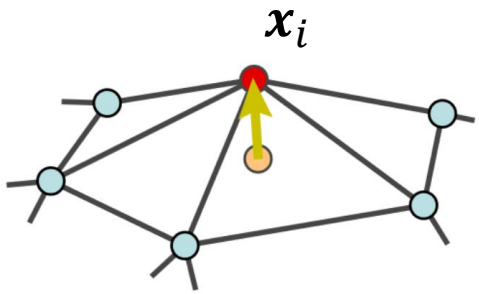
$$\min_{X'} \sum_k \|x'_k - \tilde{x}_k\|^2 + E(X, X')$$

编辑约束

平滑能量项

尽可能减少非“刚性”形变 ➔ 减少不必要的缩放

$$E(X, X') = \sum_{j \in \mathcal{N}_i} w_{ij} \left\| (x'_i - x'_j) - R_i (x_i - x_j) \right\|^2$$



$\mathcal{N}_i$: 1-neighbors

As Rigid As Possible Deformation

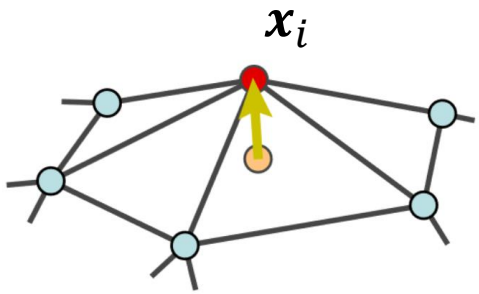
$$\min_{X', \boldsymbol{R_i}} \sum_k \|x'_k - \tilde{x}_k\|^2 + E(X, X')$$

编辑约束

平滑能量项

尽可能减少非“刚性”形变 ➔ 减少不必要的缩放

$$E(X, X') = \sum_{j \in \mathcal{N}_i} w_{ij} \left\| (x'_i - x'_j) - R_i (x_i - x_j) \right\|^2$$



$\mathcal{N}_i$: 1-neighbors

As Rigid As Possible Deformation

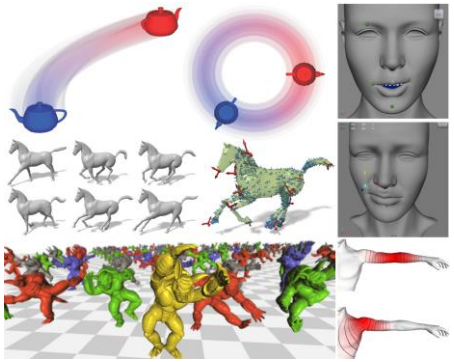
Olga Sorkine and Marc Alexa. 2007. ***As-rigid-as-possible surface modeling***. In *Proceedings of the fifth Eurographics symposium on Geometry processing* (SGP ’07), Eurographics Association, Goslar, DEU, 109–116.

蒙皮变形

Skinning

What is covered

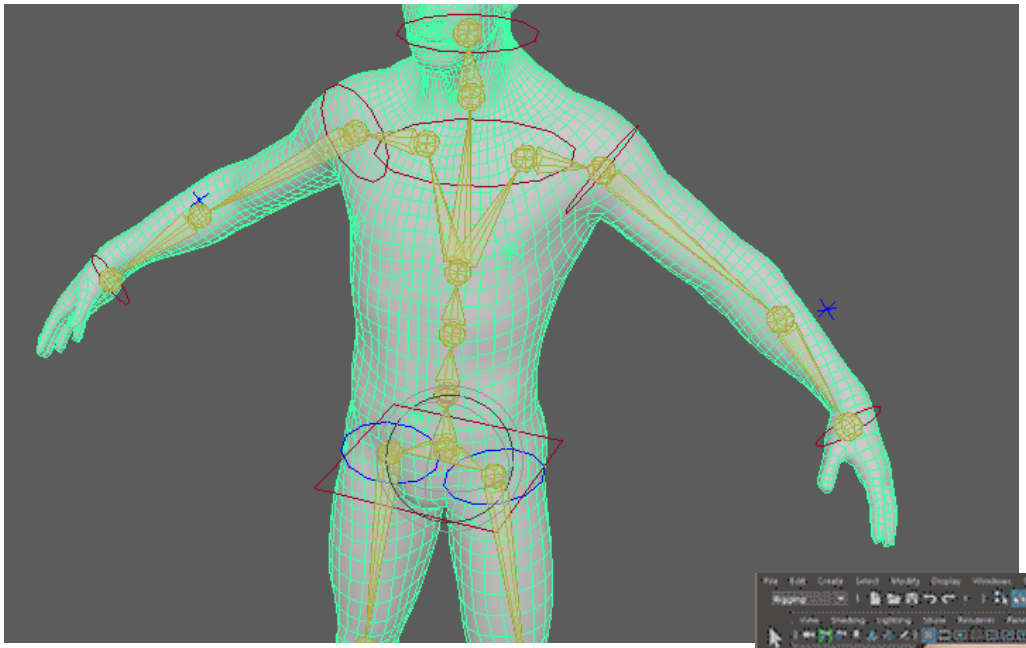
- How to deform a mesh according to skeletal animation?
  - Linear Blend Skinning (LBS)
    - Multi-linear Skinning
  - Non-linear Skinning
    - Dual-Quaternion Skinning (DQS)
- How to compute weights?
- Example-based methods?

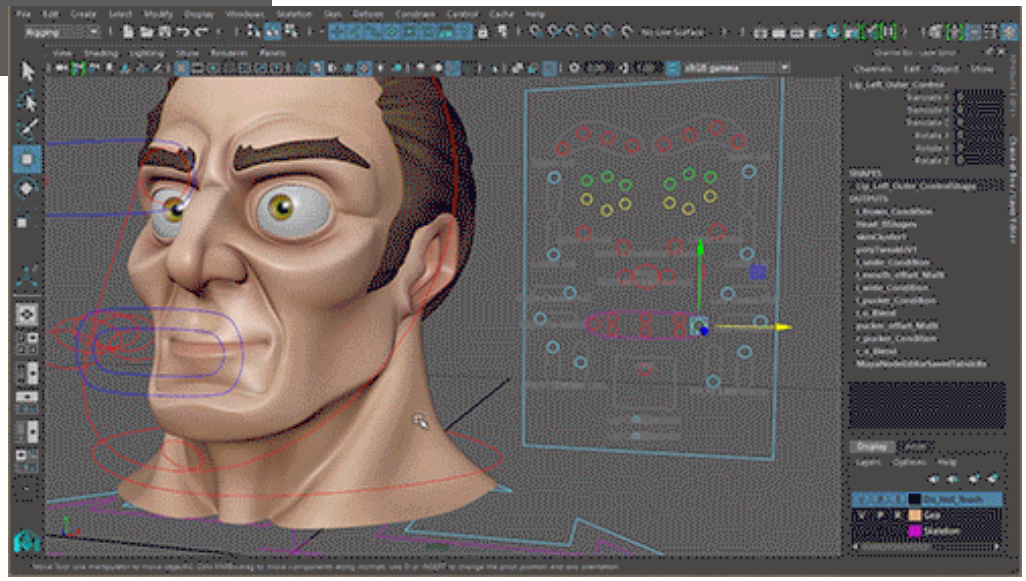


Many images are from: https://skinning.org/

Alec Jacobson, Zhigang Deng, Ladislav Kavan, and J. P. Lewis. 2014. ***Skinning: real-time shape deformation****. In ACM SIGGRAPH 2014 Courses (SIGGRAPH '14)*

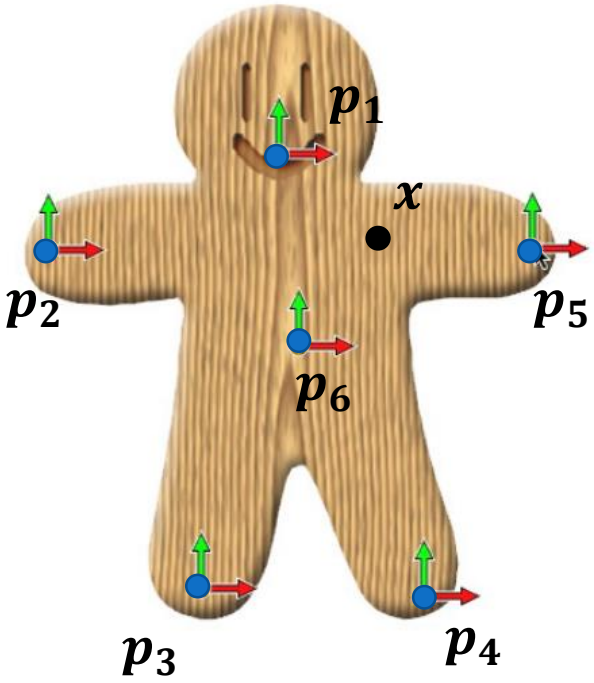
Skinning





Recall: deformation with control points

$$x_i' = \sum_j \omega_{ij} T_j x_i$$

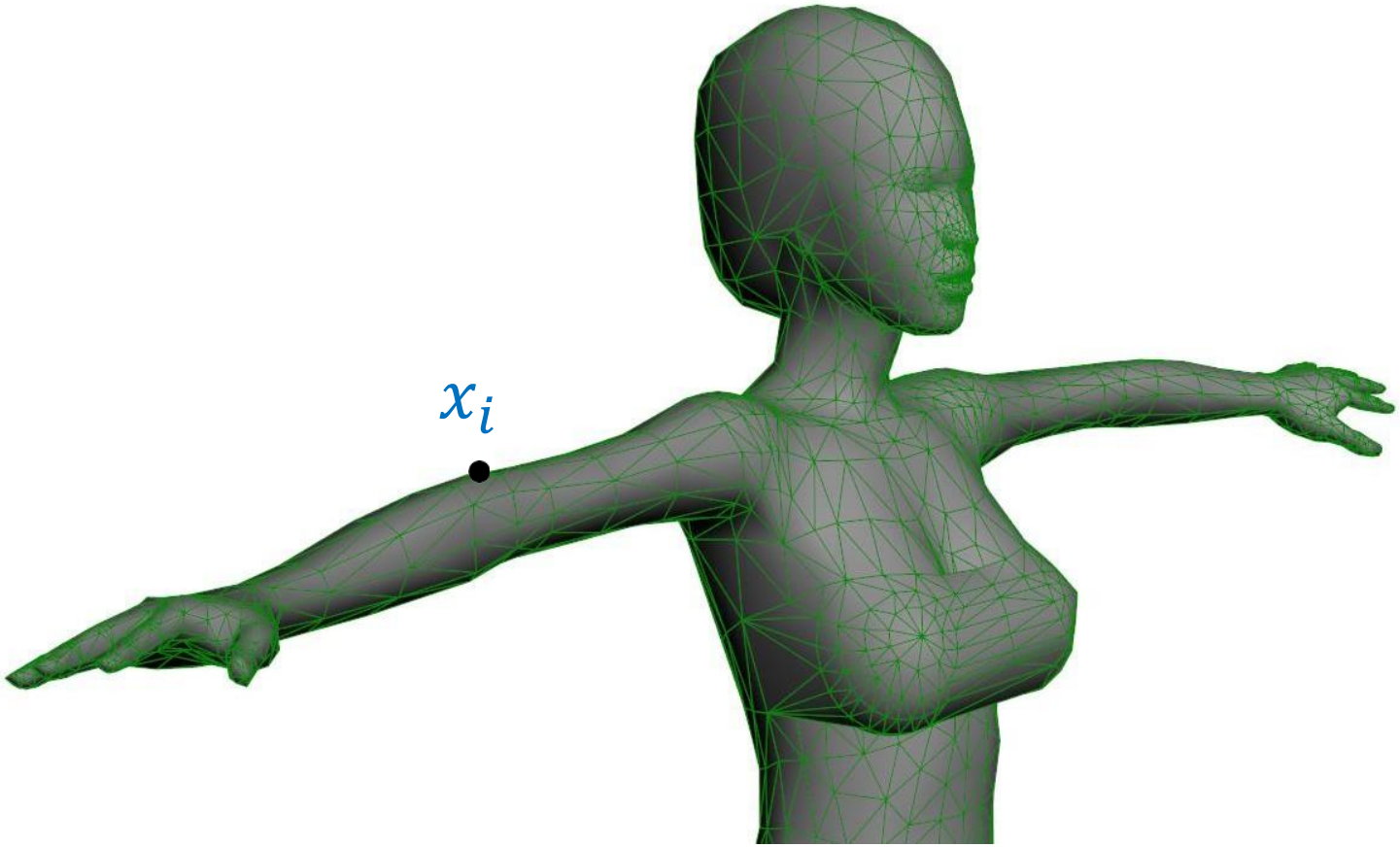


Basic Setup

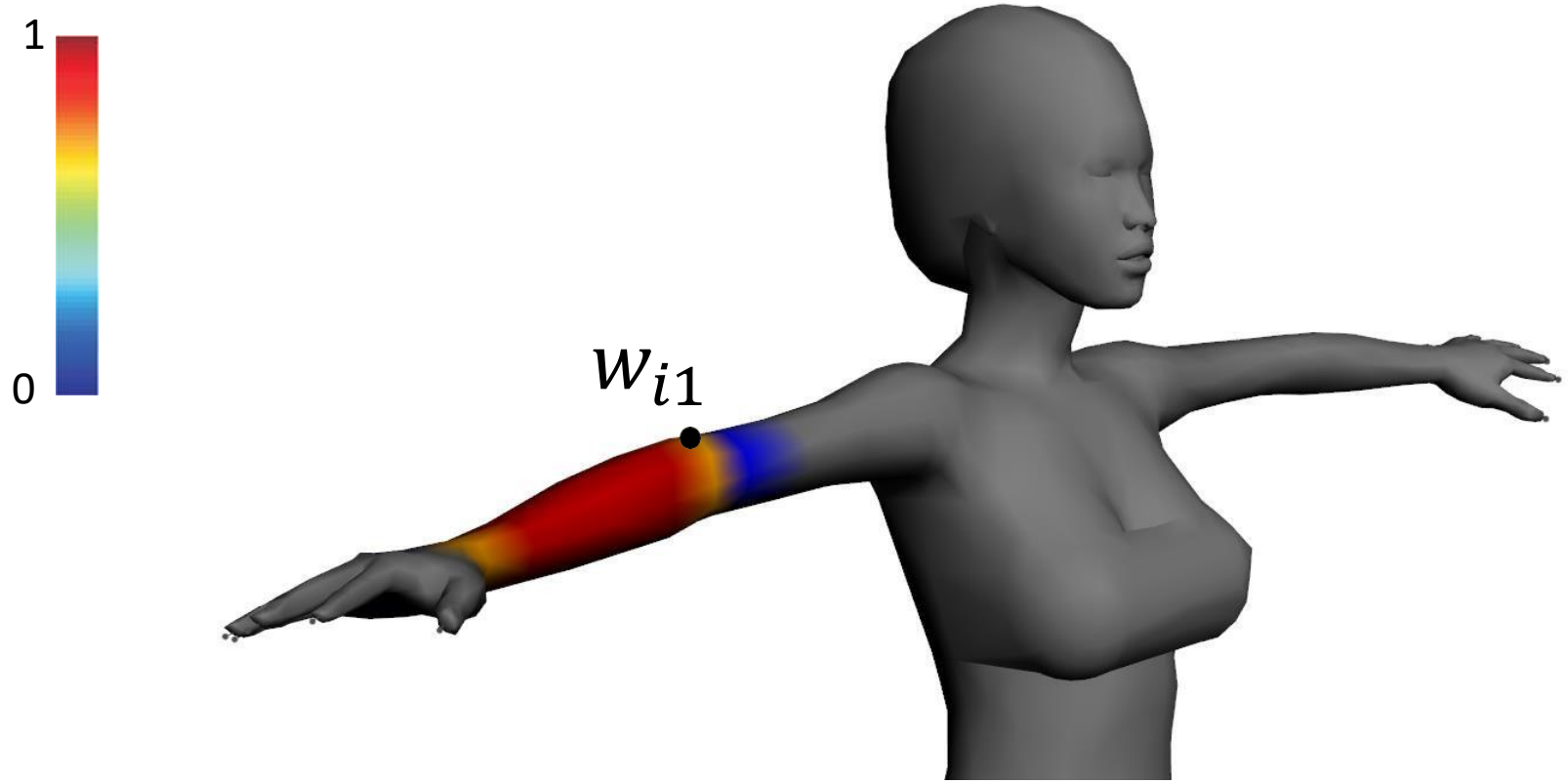
- Rest pose
- Skinning weights
- Skinning transformation

$$x_i' = \sum_{j=1}^{m} \omega_{ij} T_j x_i$$

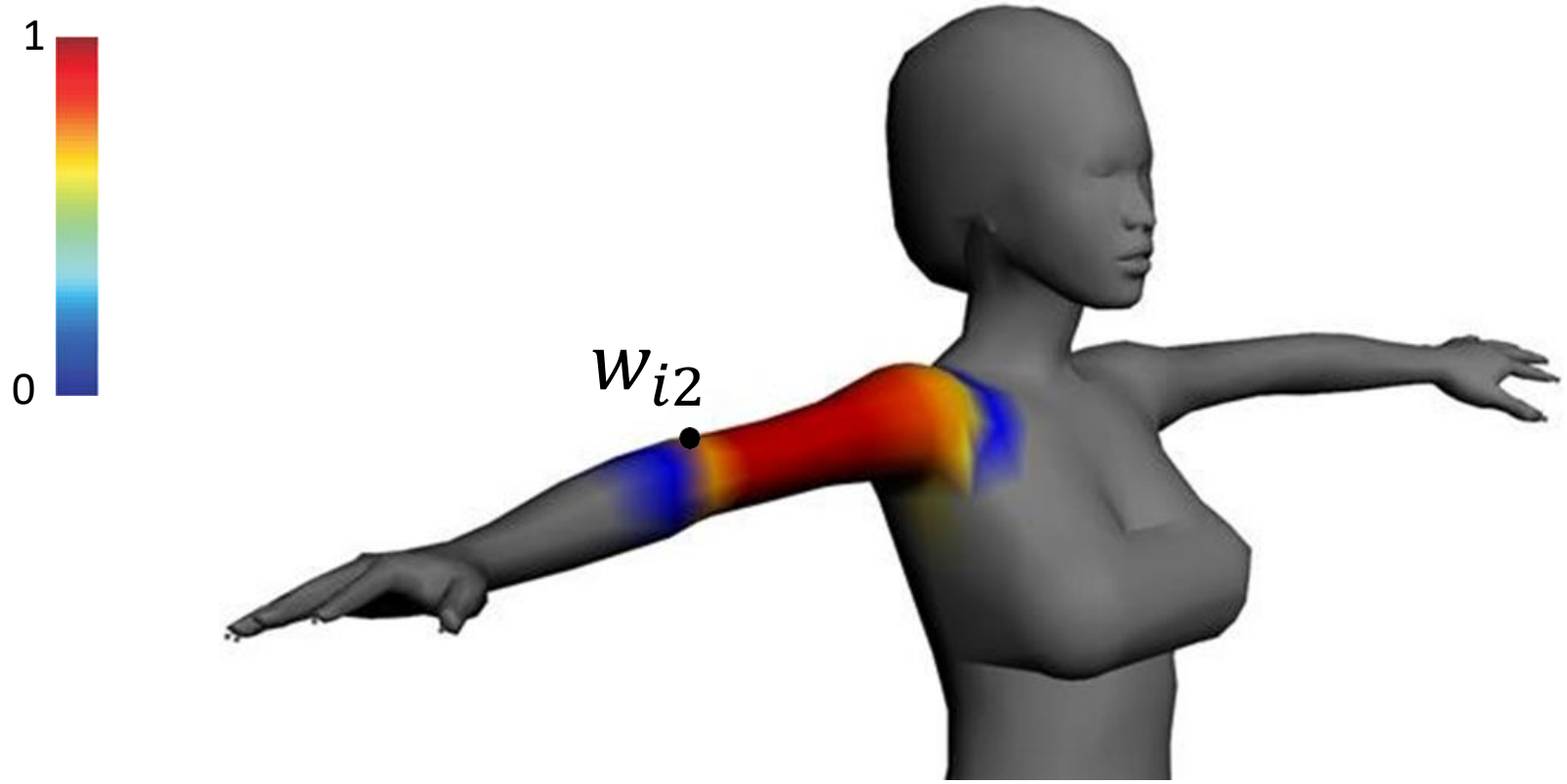
Basic Setup: Rest Pose/Bind Pose



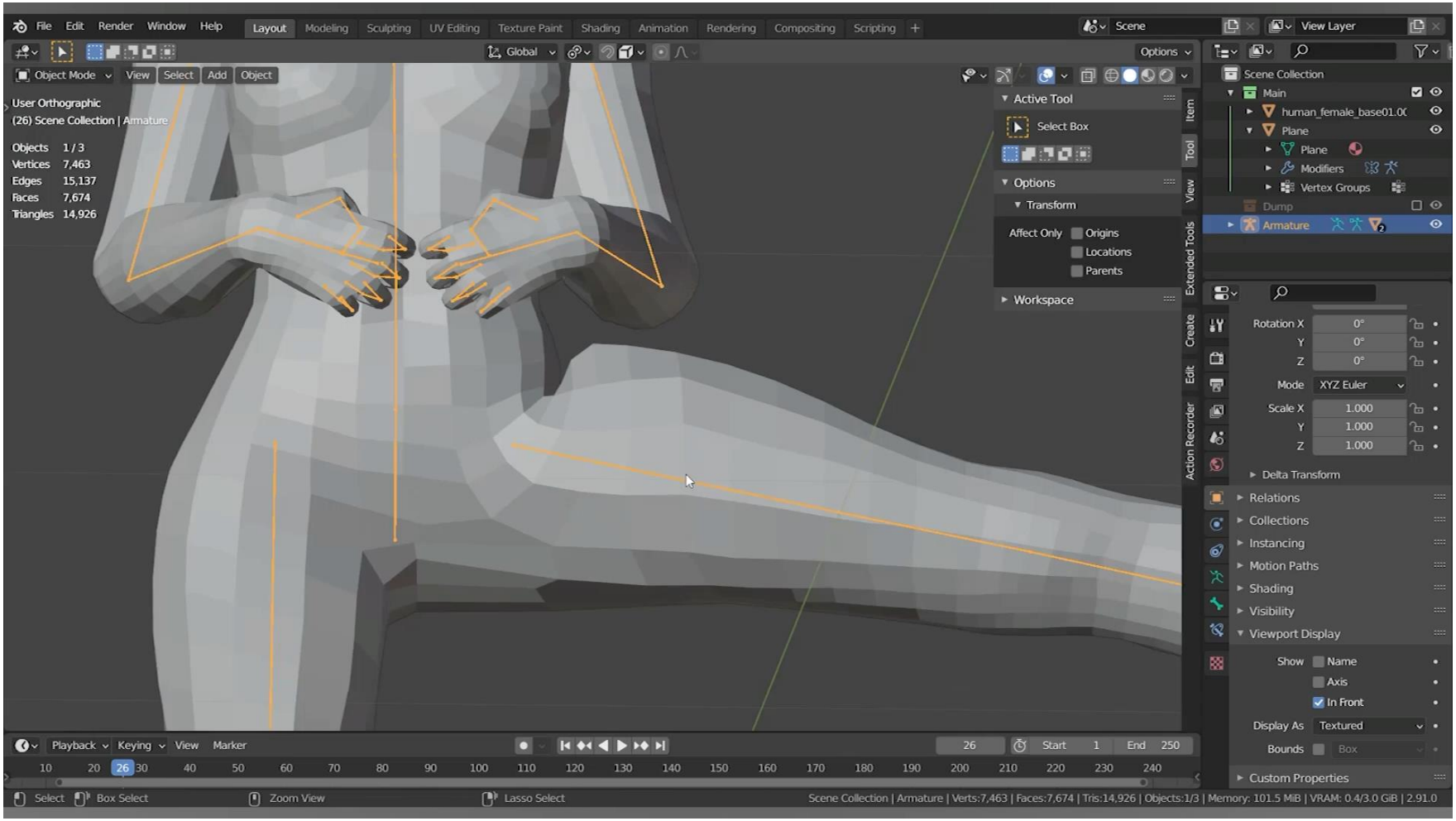
Basic Setup: Skinning Weights



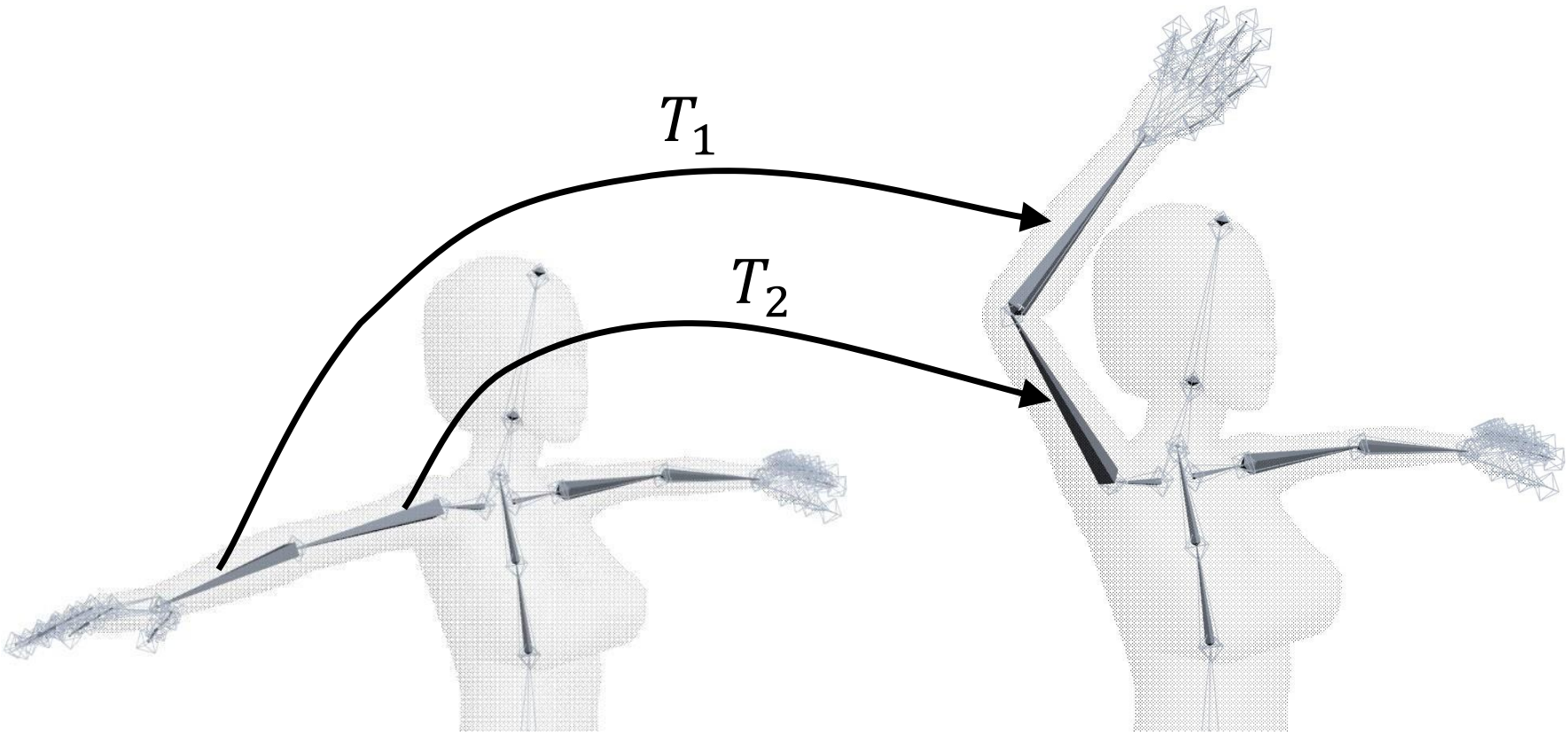
Basic Setup: Skinning Weights



Basic Setup: Skinning Weights

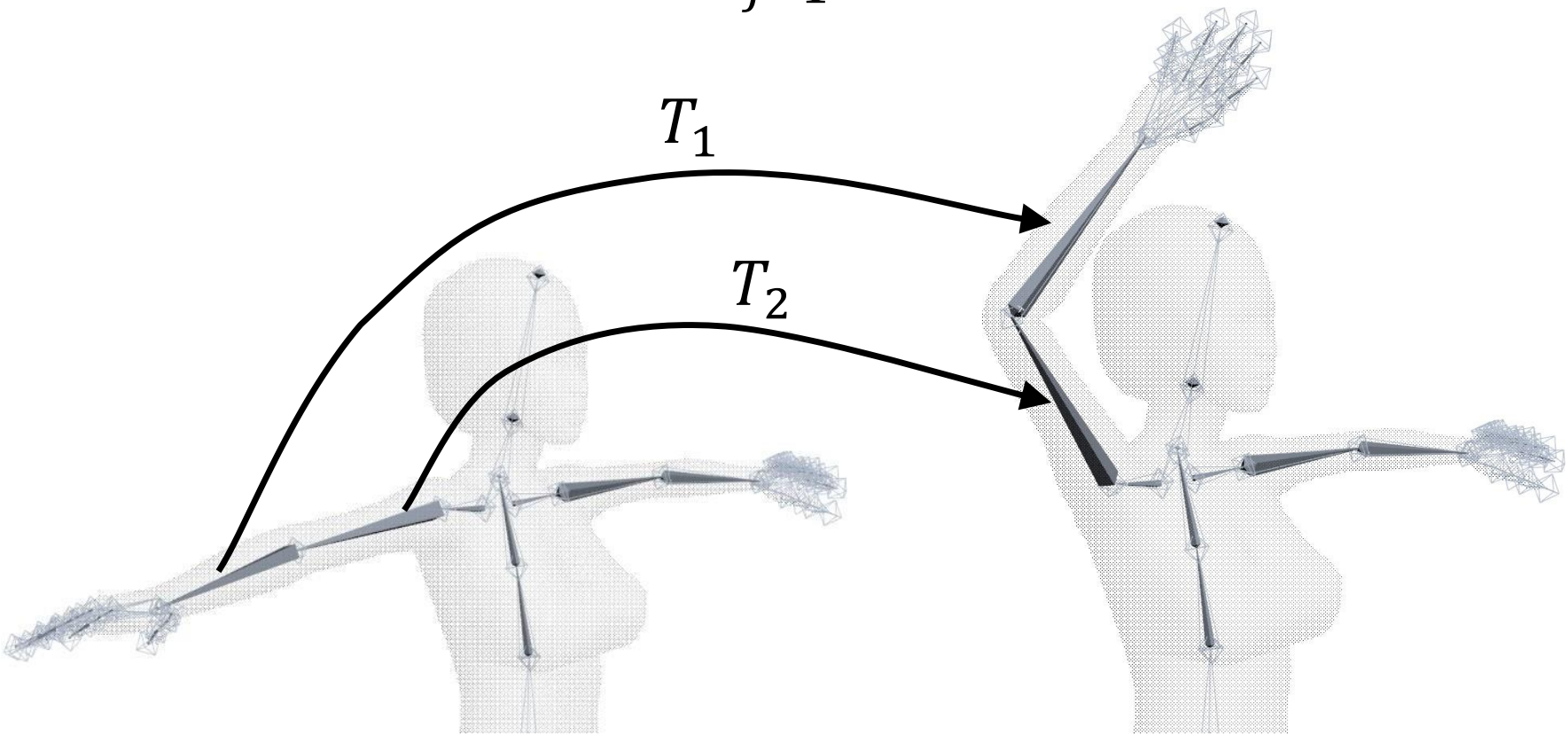


Basic Setup: Skinning Transformation



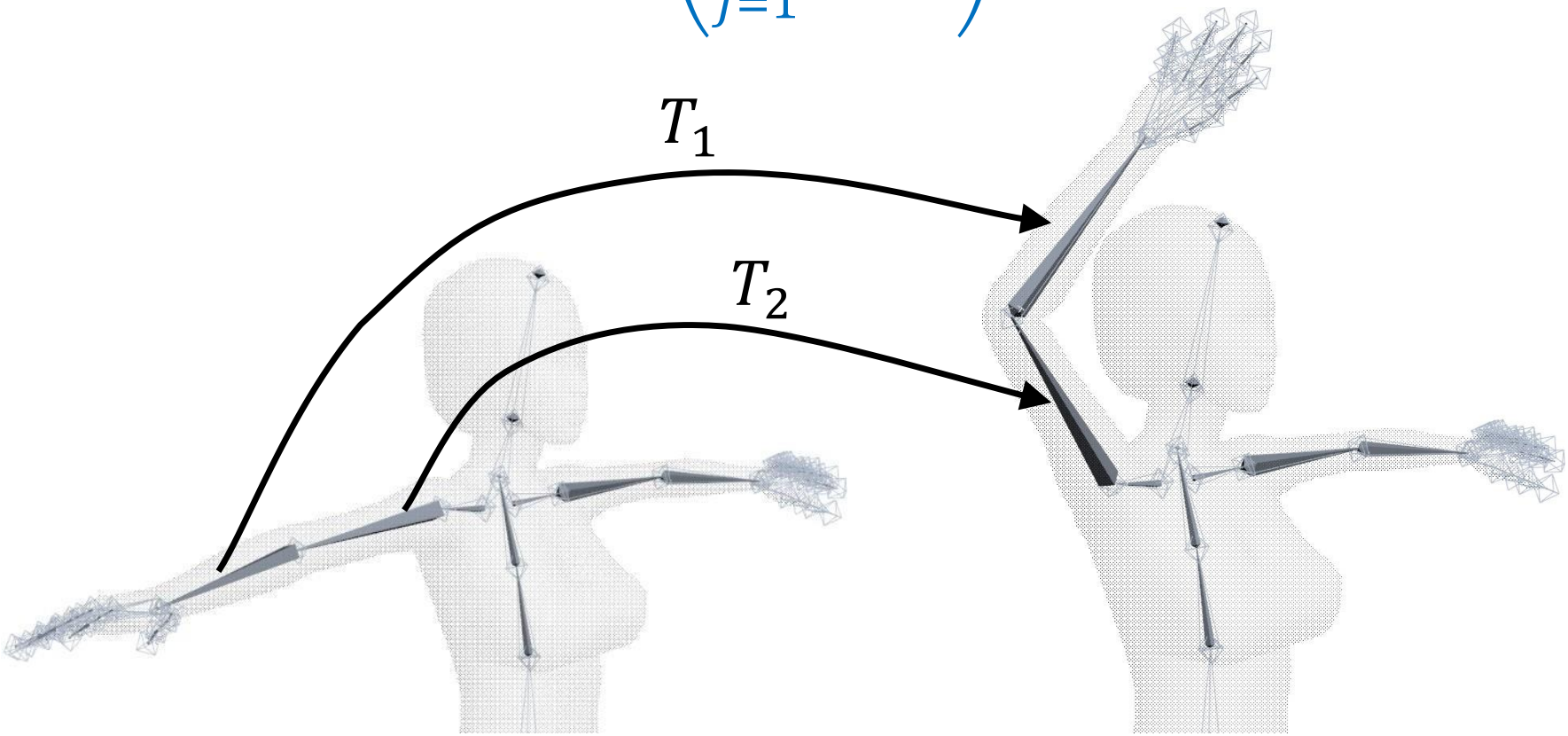
Linear Blend Skinning (LBS)

$$x_i' = \sum_{j=1}^{m} \omega_{ij} T_j x_i$$



Linear Blend Skinning (LBS)

$$x_i' = \left(\sum_{j=1}^{m} \omega_{ij} T_j \right) x_i$$



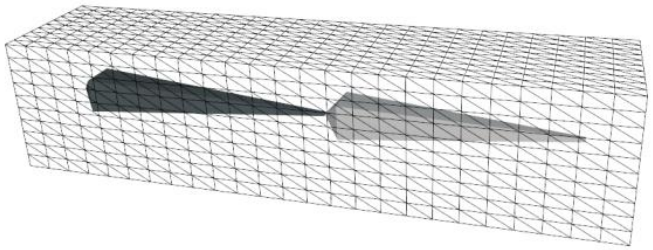
Linear Blend Skinning (LBS)

$$x_i' = \left(\sum_{j=1}^{m} \omega_{ij} T_j \right) x_i$$

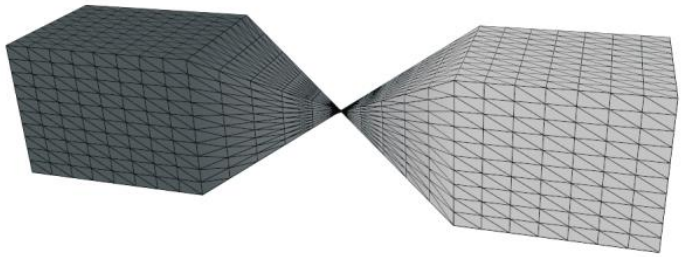
- Used widely in the industry
- Efficient and GPU-friendly
  - Games like it

Halo 3

Candy-Wrapper Artifact



Rest pose



Linear blend skinning

Candy-Wrapper Artifact

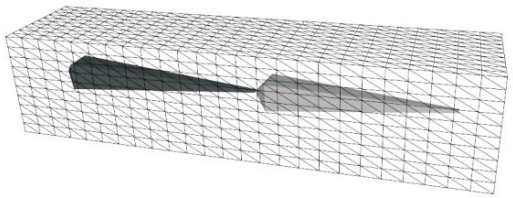
$$x_i' = \left(\sum_{j=1}^{m} \omega_{ij} T_j \right) x_i$$

Consider $T = (R \quad t)$ and

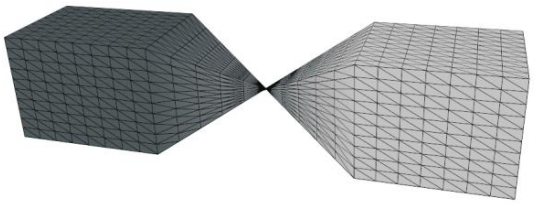
$$\mathbf{R}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{R}_2 = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Advanced skinning methods

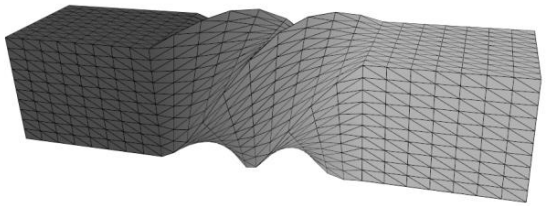
- Multi-linear Skinning
- Nonlinear Skinning
  - Dual-quaternion Skinning (DQS)



Rest pose



Linear blend skinning



Dual quaternion skinning

Multi-linear Skinning

- Multi-weight enveloping [Wang and Phillips 2002]
- Animation Space [Merry et al. 2006]

Multi-linear Skinning

$$x' = Tx$$

$$T = [R \mid u] \in \mathbb{R}^{3\times 4}$$

$$v' = Xt$$

$v' \in \mathbb{R}^{3n}$: stack of all vertices x'

$t \in \mathbb{R}^{12m}$: stack of all transformation T

$X \in \mathbb{R}^{3n\times 12m}$: skinning matrix

Multi-linear Skinning

The structure of matrix $\boldsymbol{X}$

$$\boldsymbol{X} \in \mathbb{R}^{3n \times 12m} \qquad \boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_{11} & \cdots & \boldsymbol{X}_{1m} \\ \vdots & \ddots & \vdots \\ \boldsymbol{X}_{n1} & \cdots & \boldsymbol{X}_{nm} \end{bmatrix}$$

$$\boldsymbol{X}_{ij} \in \mathbb{R}^{3 \times 12}$$

Multi-linear Skinning

Matrix vectorization

$$\mathrm{vec}(\boldsymbol{A}) = (a_{11}, a_{12}, a_{13}, \dots, a_{mn})^T$$

Kronecker product

$$\boldsymbol{A} \in \mathbb{R}^{p\times q}, \qquad \boldsymbol{B} \in \mathbb{R}^{r\times s}$$

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}B & \cdots & a_{1q}B \\ \vdots & \ddots & \vdots \\ a_{p1}B & \cdots & a_{pq}B \end{bmatrix} \in \mathbb{R}^{pr\times qs}$$

$$\mathrm{vec}(\mathbf{AB}) = (\boldsymbol{I}_s \otimes \boldsymbol{A})\,\mathrm{vec}(\boldsymbol{B}) = \left(\boldsymbol{B}^T \otimes \boldsymbol{I}_p\right)\mathrm{vec}(\boldsymbol{A})$$

Multi-linear Skinning

$$x_i' = \sum_{j=1}^{m} \omega_{ij} T_j x_i$$

$$x_i' = \mathrm{vec}(x_i') = \sum_{j=1}^{m} \mathrm{vec}\left(T_j \omega_{ij} x_i\right) = \sum_{j=1}^{m} \left(\omega_{ij} \boldsymbol{x}_i \otimes I\right) \mathrm{vec}\left(T_j\right)$$

$$= \sum_{j=1}^{m} X_{ij}^{LBS} t_j \; = X_i^{LBS} t$$

LBS in matrix form

$$X_{ij}^{LBS} = [w_{ij}x_{i1}\mathbf{I}_3 \quad w_{ij}x_{i2}\mathbf{I}_3 \quad w_{ij}x_{i3}\mathbf{I}_3 \quad w_{ij}\mathbf{I}_3]$$

$$\mathbf{I}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i3} \end{bmatrix}$$

$$\begin{bmatrix} w_{ij}x_{i1} & 0 & 0 & w_{ij}x_{i2} & 0 & 0 & w_{ij}x_{i3} & 0 & 0 & w_{ij} & 0 & 0 \\ 0 & w_{ij}x_{i1} & 0 & 0 & w_{ij}x_{i2} & 0 & 0 & w_{ij}x_{i3} & 0 & 0 & w_{ij} & 0 \\ 0 & 0 & w_{ij}x_{i1} & 0 & 0 & w_{ij}x_{i2} & 0 & 0 & w_{ij}x_{i3} & 0 & 0 & w_{ij} \end{bmatrix}$$

LBS has one weight per vertex/bone pair

Multi-linear Skinning

$$X_{ij} = \begin{bmatrix} * & * & * & * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * & * & * & * & * \end{bmatrix}$$

$$\in \mathbb{R}^{3\times 12}$$

We can have 36 weights per vertex/bone pair in the most general form

Multi-weight Enveloping [Wang and Phillips 2002]

$$X_{ij}^{MWE} =$$

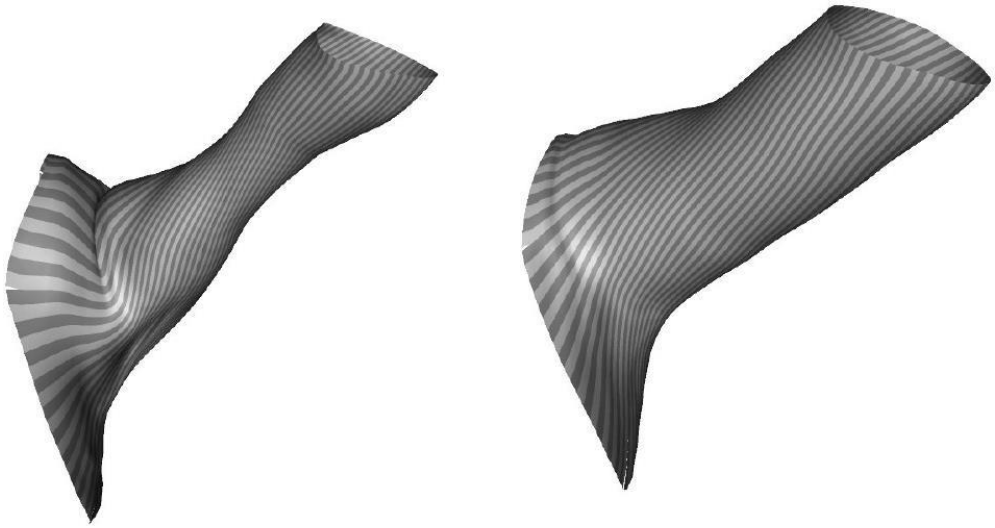
$$\begin{bmatrix} w_{ij}^{1}x_{i1} & 0 & 0 & w_{ij}^{4}x_{i2} & 0 & 0 & w_{ij}^{7}x_{i3} & 0 & 0 & w_{ij}^{10} & 0 & 0 \\ 0 & w_{ij}^{2}x_{i1} & 0 & 0 & w_{ij}^{5}x_{i2} & 0 & 0 & w_{ij}^{8}x_{i3} & 0 & 0 & w_{ij}^{11} & 0 \\ 0 & 0 & w_{ij}^{3}x_{i1} & 0 & 0 & w_{ij}^{6}x_{i2} & 0 & 0 & w_{ij}^{9}x_{i3} & 0 & 0 & w_{ij}^{12} \end{bmatrix}$$

12 weights per vertex/bone

Multi-weight Enveloping [Wang and Phillips 2002]

More powerful

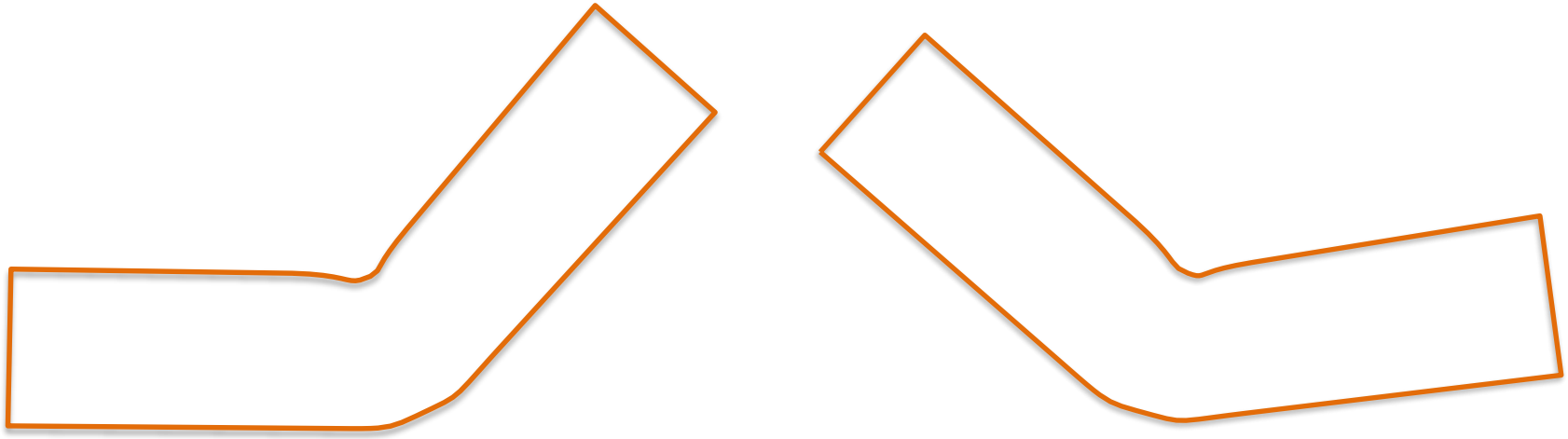
Weights trained from examples



LBS MWE

Animation Space [Merry et al. 2006]

World-space rotation invariance.



Animation Space [Merry et al. 2006]

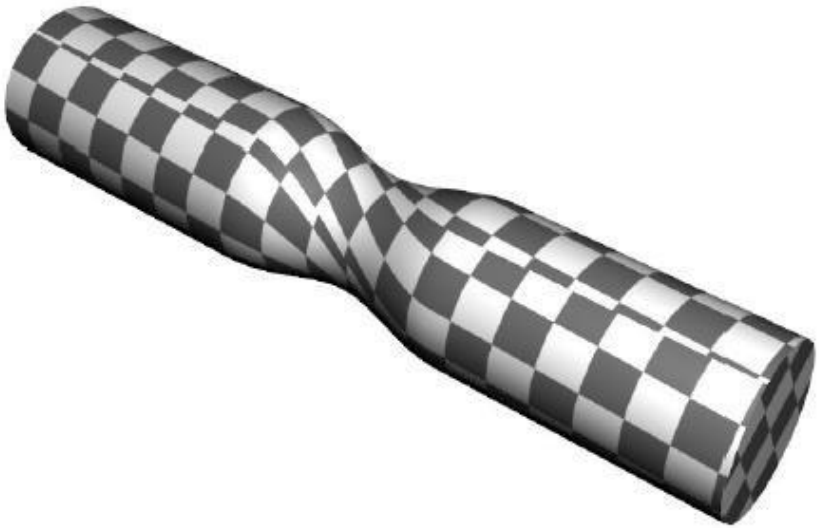
Rotation-invariant results in 4 weights per vertex/bone pair

$$X_{ij}^{LBS} = [y_{ij1}\mathbf{I}_3 \quad y_{ij2}\mathbf{I}_3 \quad y_{ij3}\mathbf{I}_3 \quad y_{ij4}\mathbf{I}_3]$$

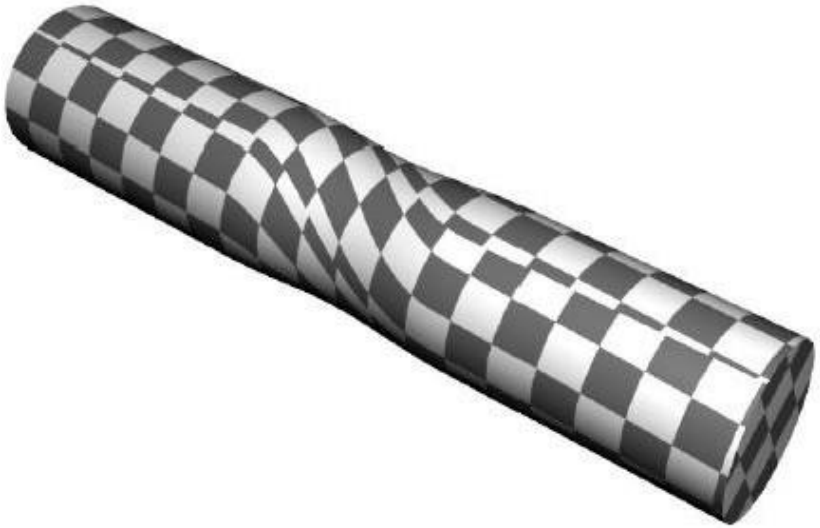
And translation invariance requires:

$$\sum_{j=1}^{m} y_{ij\mathbf{4}} = 1$$

Animation Space [Merry et al. 2006]



LBS



Animation space

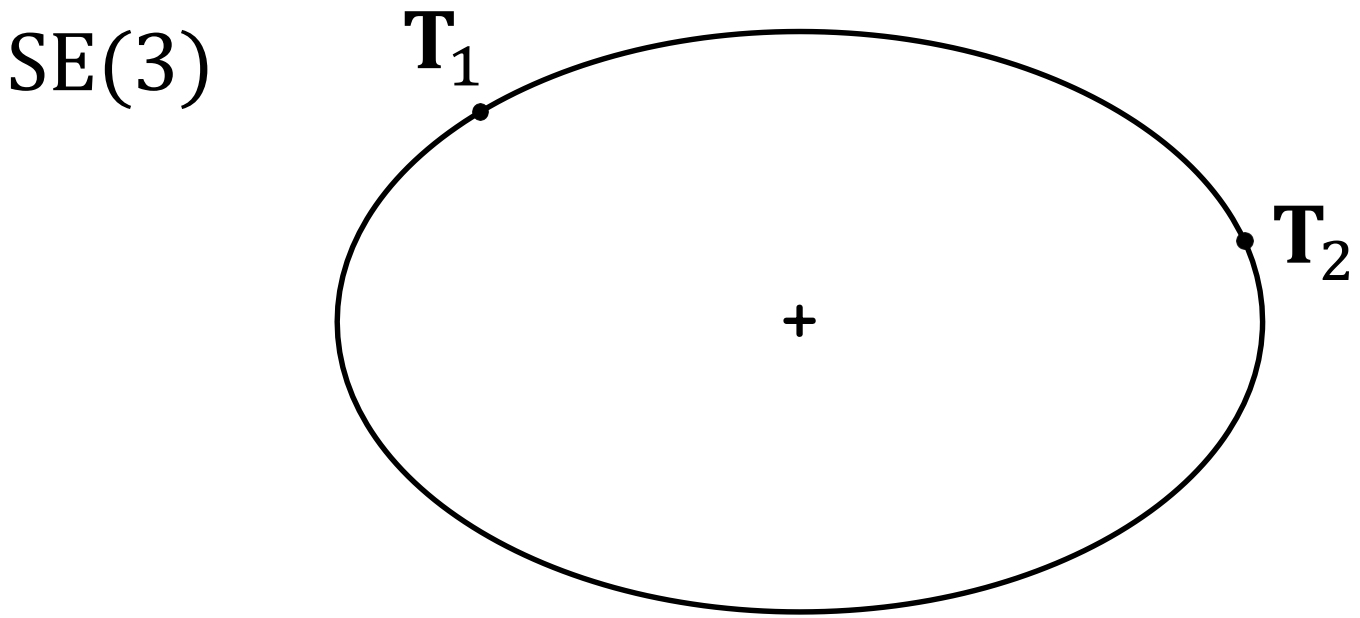
Non-linear Skinning

$$x_i' = \left(\sum_{j=1}^{m} \omega_{ij} T_j \right) x_i$$

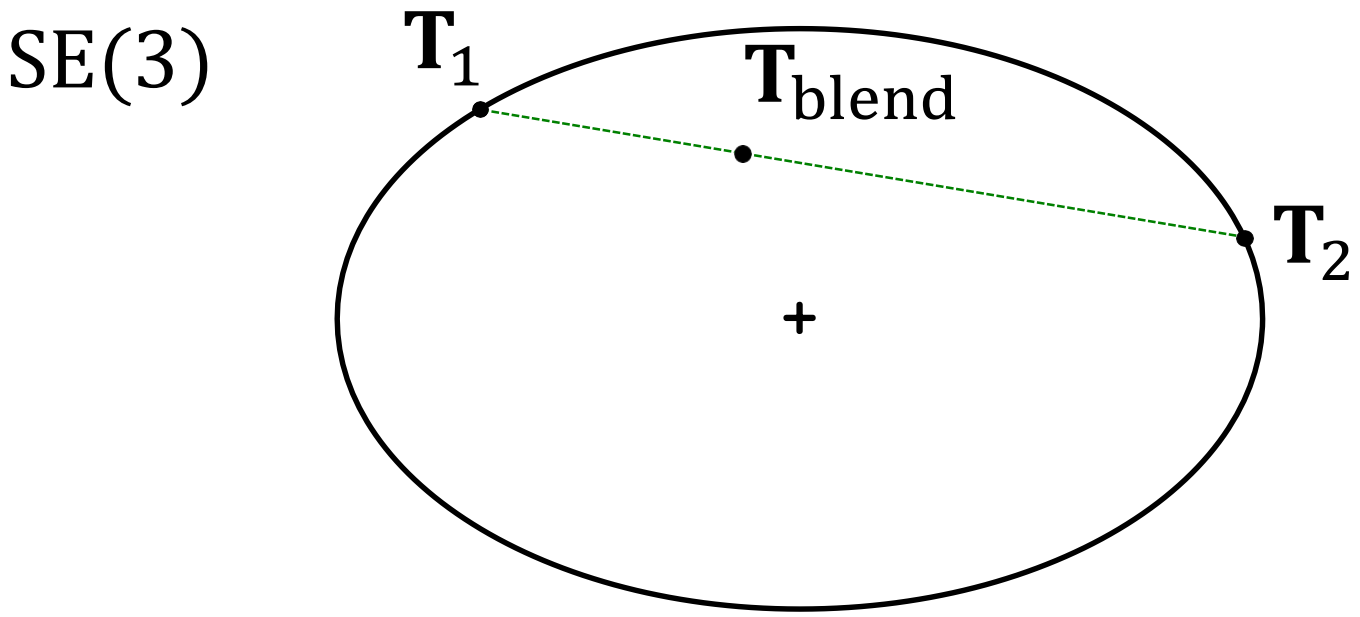
$T_j \in SE(3)$: 刚性变换群

$R \in SO(3)$: 旋转变换群

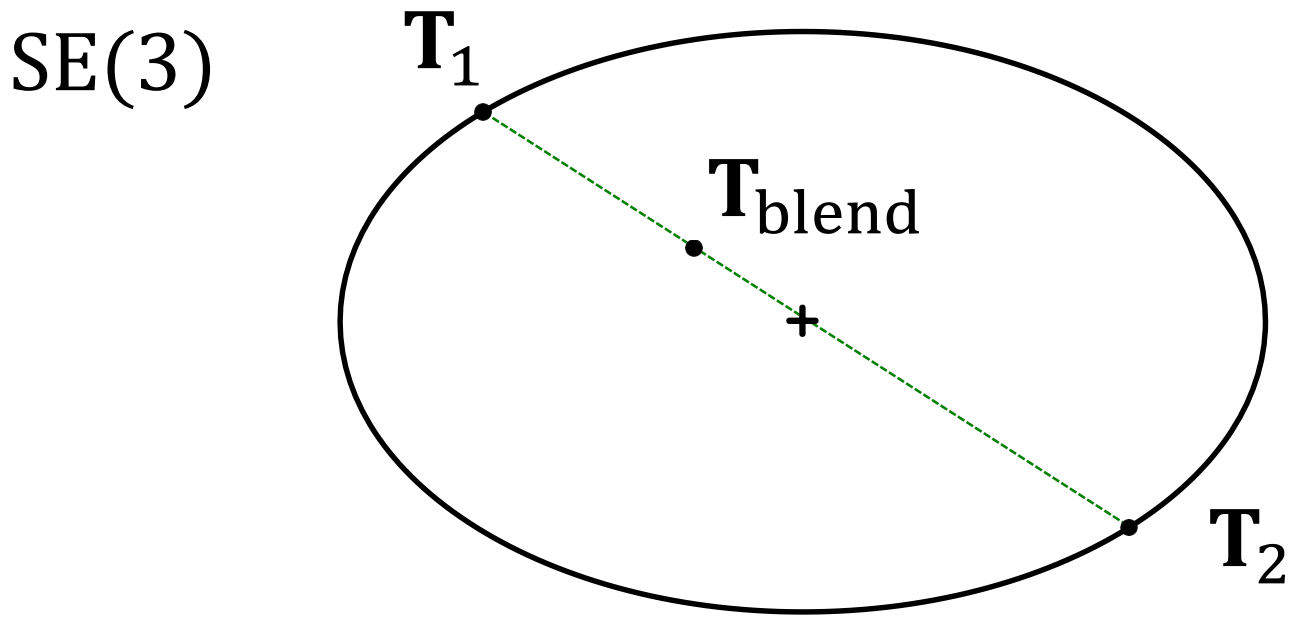
Interpolation in SE(3)



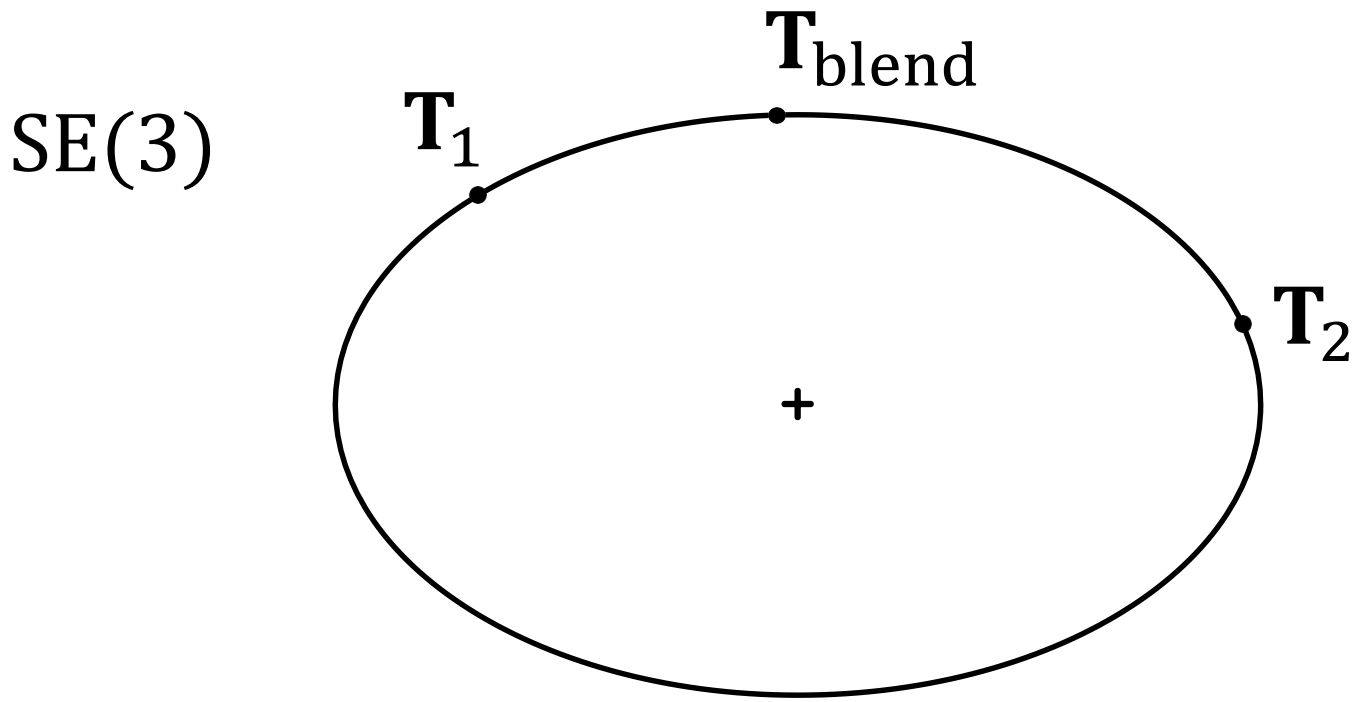
Interpolation in SE(3)



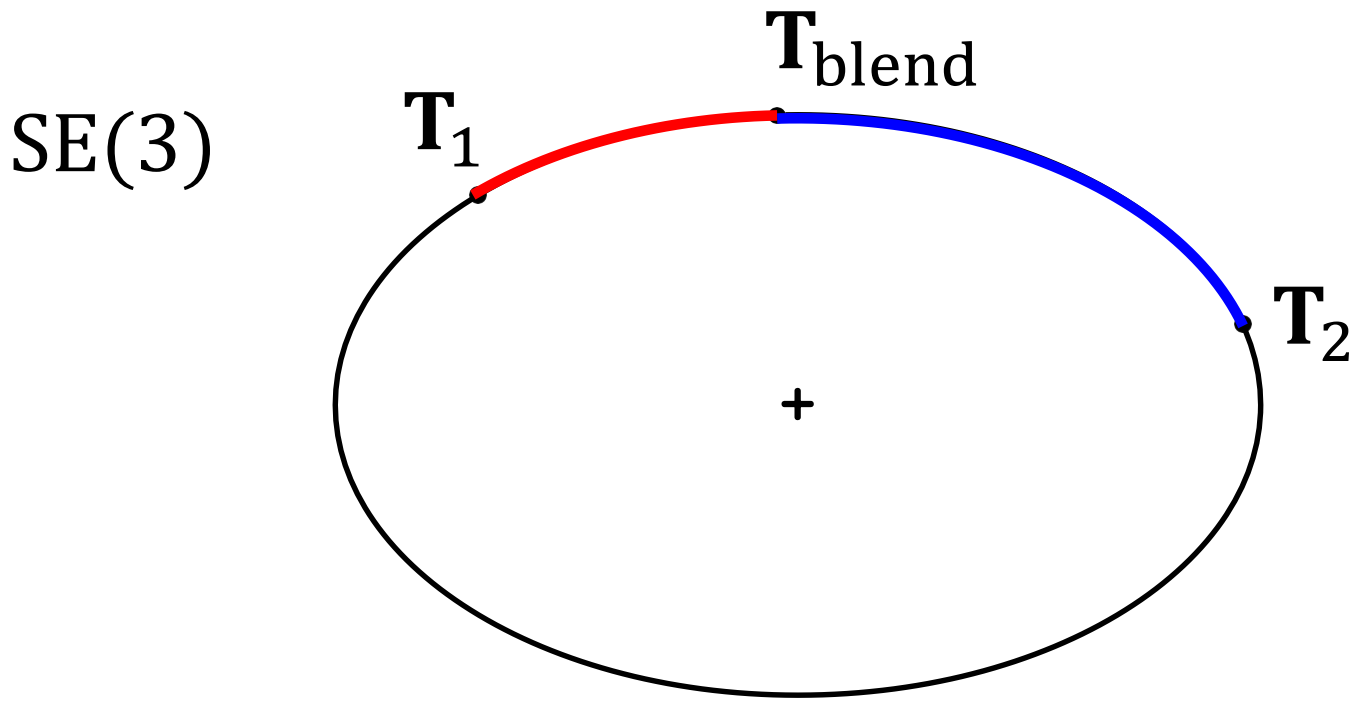
Interpolation in SE(3)



Intrinsic blending

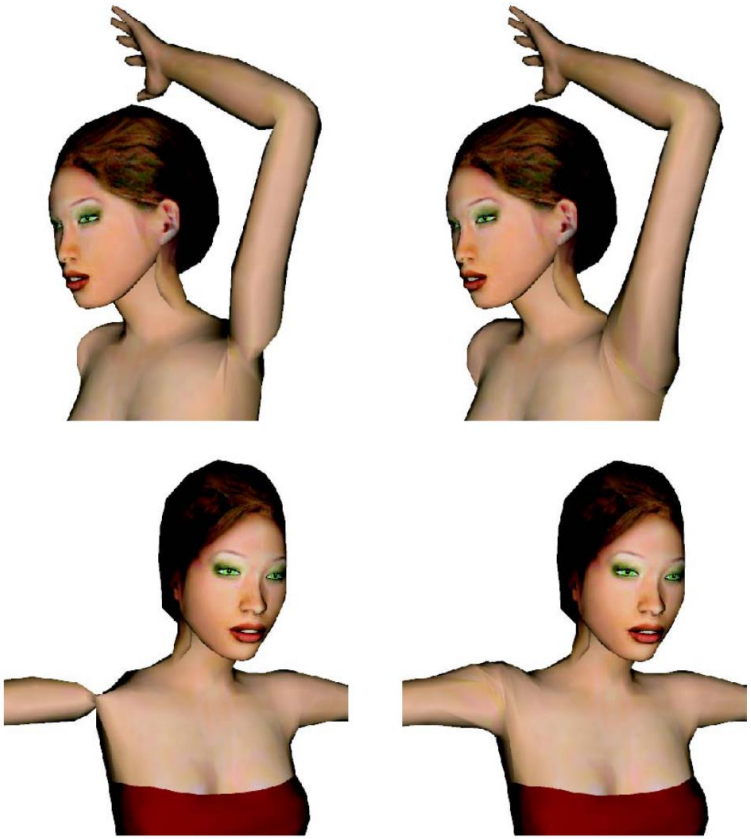


Intrinsic blending



Dual-Quaternion Skinning (DQS)

- Approximation of intrinsic averages in SE(3)



Ladislav Kavan, Steven Collins, Jiri Zara, Carol O'Sullivan. ***Geometric Skinning with Approximate Dual Quaternion Blending***, ACM Transaction on Graphics, 27(4), 2008.

Dual-Quaternion

- Dual number

$$x = a + b\varepsilon$$

where $\varepsilon^2 = 0$

- Conjugate

$$\bar{x} = \overline{a + b\varepsilon} = a - b\varepsilon$$

- Multiplication

$$(a + b\varepsilon)(c + d\varepsilon) = ac + (ad + bc)\varepsilon$$

Dual-Quaternion

- Dual quaternion

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

where $\varepsilon^2 = 0$

A good note of dual-quaternion:
https://faculty.sites.iastate.edu/jia/files/inline-files/dual-quaternion.pdf

Dual-Quaternion

- Scalar Multiplication

$$s\hat{\boldsymbol{q}} = s\boldsymbol{q}_r + s\boldsymbol{q}_\varepsilon\varepsilon$$

- Addition

$$\hat{\boldsymbol{q}}_1 + \hat{\boldsymbol{q}}_2 = \boldsymbol{q}_{r1} + \boldsymbol{q}_{r2} + \varepsilon(\boldsymbol{q}_{\varepsilon1} + \boldsymbol{q}_{\varepsilon2})$$

- Multiplication

$$\hat{\boldsymbol{q}}_1\hat{\boldsymbol{q}}_2 = \boldsymbol{q}_{r1}\boldsymbol{q}_{r2} + \varepsilon(\boldsymbol{q}_{r1}\boldsymbol{q}_{\varepsilon2} + \boldsymbol{q}_{r2}\boldsymbol{q}_{\varepsilon1})$$

Dual-Quaternion

- Dual quaternion

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

- Conjugation

$$\text{I:} \quad \hat{\boldsymbol{q}}^* = \boldsymbol{q}_0^* + \varepsilon \boldsymbol{q}_\varepsilon^*$$

$$\text{II:} \quad \hat{\boldsymbol{q}}^\circ = \boldsymbol{q}_0 - \varepsilon \boldsymbol{q}_\varepsilon$$

$$\text{III:} \quad \hat{\boldsymbol{q}}^\star = \boldsymbol{q}_0^* - \varepsilon \boldsymbol{q}_\varepsilon^*$$

$$= (\hat{\boldsymbol{q}}^*)^\circ = (\hat{\boldsymbol{q}}^\circ)^*$$

$$(\hat{\boldsymbol{q}}_1 \hat{\boldsymbol{q}}_2)^\times = \hat{\boldsymbol{q}}_2^\times \hat{\boldsymbol{q}}_1^\times$$

- Norm

$$\|\hat{\boldsymbol{q}}\| = \sqrt{\hat{\boldsymbol{q}}^* \hat{\boldsymbol{q}}} = \|\boldsymbol{q}_0\| + \frac{\varepsilon(\boldsymbol{q}_0 \cdot \boldsymbol{q}_\varepsilon)}{\|\boldsymbol{q}_0\|}$$

Dual-Quaternion

- Norm

$$\|\hat{\boldsymbol{q}}\| = \sqrt{\hat{\boldsymbol{q}}^{*}\hat{\boldsymbol{q}}} = \|\boldsymbol{q}_0\| + \frac{\varepsilon(\boldsymbol{q}_0 \cdot \boldsymbol{q}_\varepsilon)}{\|\boldsymbol{q}_0\|}$$

- Unit dual quaternion: $\|\hat{\boldsymbol{q}}\| = 1$

which requires:

- $\|\boldsymbol{q}_0\| = 1$
- $\boldsymbol{q}_0 \cdot \boldsymbol{q}_\varepsilon = 0$

Dual-Quaternion

- Like quaternion, any rigid transformation $T \in SE(3)$ can be converted into a dual-quaternion

$$T\boldsymbol{x} = R\boldsymbol{x} + t$$

$$T = [R \mid t] \rightarrow \hat{\boldsymbol{q}} = \boldsymbol{q}_0 + \varepsilon \boldsymbol{q}_\varepsilon$$

$\boldsymbol{q}_0 = \boldsymbol{r}$ quaternion of R

$\boldsymbol{q}_\varepsilon = \frac{1}{2}\boldsymbol{t}\boldsymbol{r}$ pure quaternion $\boldsymbol{t} = (0, t)$

Dual-Quaternion

- Transform a vector using unit dual quaternion

$$\hat{\boldsymbol{v}}' = \hat{\boldsymbol{q}}\hat{\boldsymbol{v}}\hat{\boldsymbol{q}}^{\star}$$

Where

$$\begin{aligned}\hat{\boldsymbol{v}} &= 1 + \varepsilon(0, \boldsymbol{v}) \\ &= (1,0,0,0) + \varepsilon\left(0, v_x, v_y, v_z\right)\end{aligned}$$

$$\text{III:} \quad \begin{aligned}\hat{\boldsymbol{q}}^{\star} &= \boldsymbol{q}_0^* - \varepsilon \boldsymbol{q}_\varepsilon^* \\ &= (\hat{\boldsymbol{q}}^*)^\circ = (\hat{\boldsymbol{q}}^\circ)^*\end{aligned}$$

Dual-Quaternion

- Like quaternion, any rigid transformation $T \in SE(3)$ can be converted into a dual-quaternion

$$T = [R \mid t] \rightarrow \hat{\boldsymbol{q}} = \boldsymbol{q}_0 + \varepsilon \boldsymbol{q}_\varepsilon$$

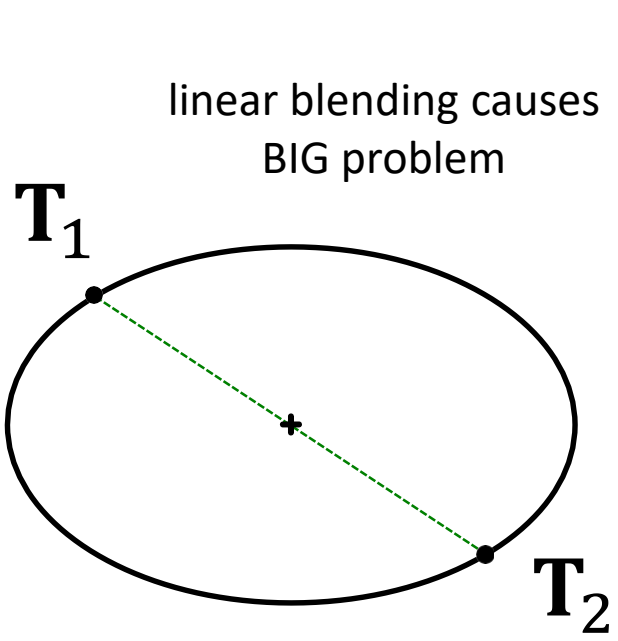
$$\boldsymbol{q}_0 = \boldsymbol{r}$$

$$\boldsymbol{q}_\varepsilon = \frac{1}{2}\boldsymbol{t}\boldsymbol{r}$$

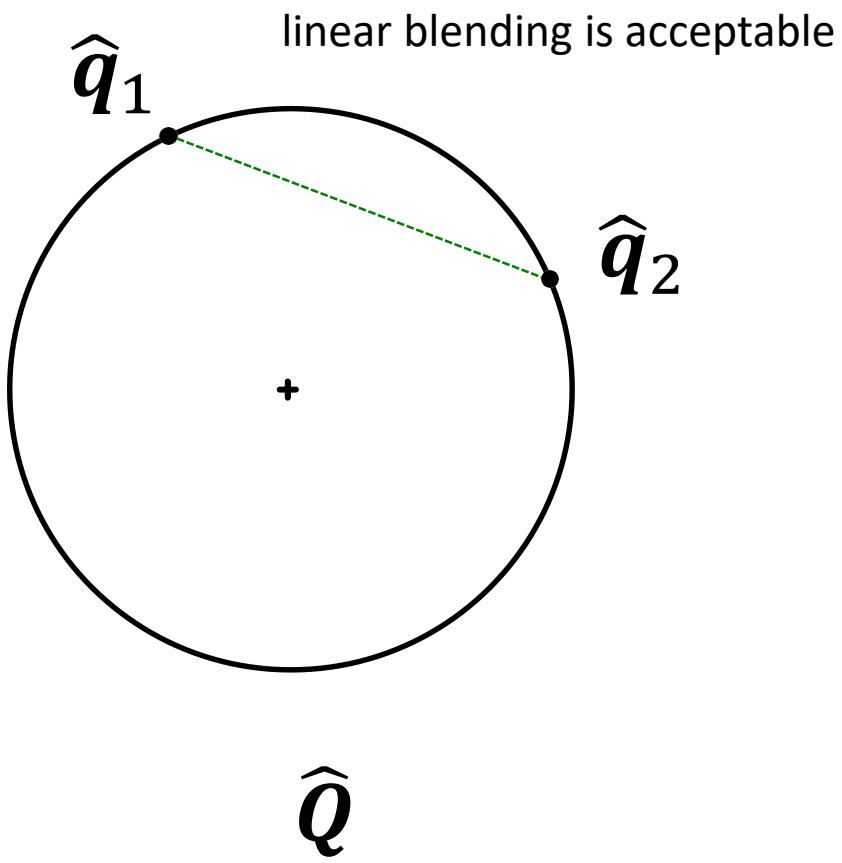
$\hat{\boldsymbol{q}}$ and $-\hat{\boldsymbol{q}}$ represent the same transformation

$\hat{\boldsymbol{Q}}$ is a double cover of SE(3)

Double Cover Visualized



$\mathrm{SE}(3)$

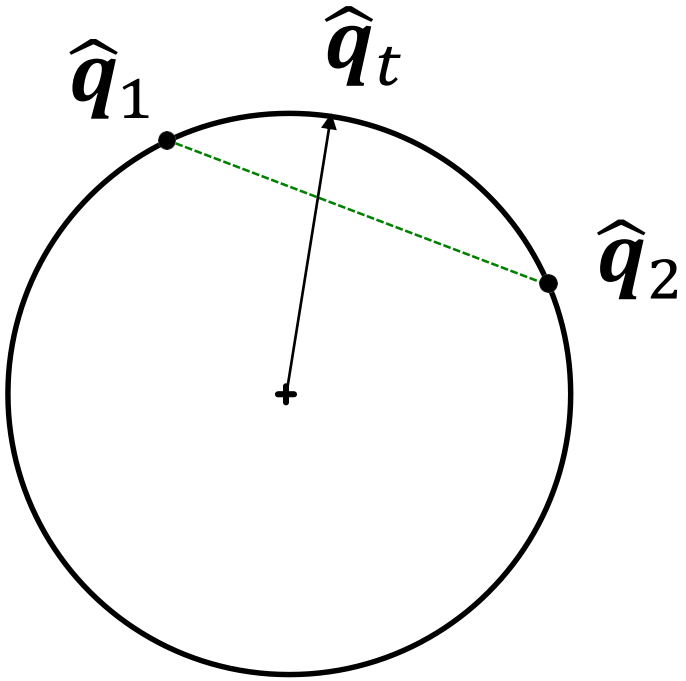


$\hat{\boldsymbol{Q}}$

Interpolating Dual-Quaternion

$$\hat{\boldsymbol{q}}_t = (1-t)\hat{\boldsymbol{q}}_{\boldsymbol{0}} + t\hat{\boldsymbol{q}}_{\boldsymbol{1}}$$

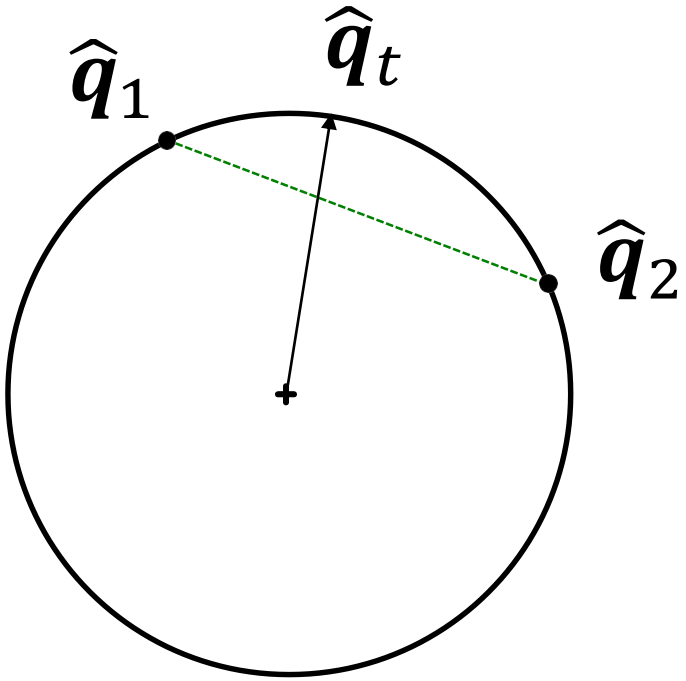
$$\hat{\boldsymbol{q}}_t = \frac{(1-t)\hat{\boldsymbol{q}}_{\boldsymbol{0}} + t\hat{\boldsymbol{q}}_{\boldsymbol{1}}}{\|(1-t)\hat{\boldsymbol{q}}_{\boldsymbol{0}} + t\hat{\boldsymbol{q}}_{\boldsymbol{1}}\|}$$



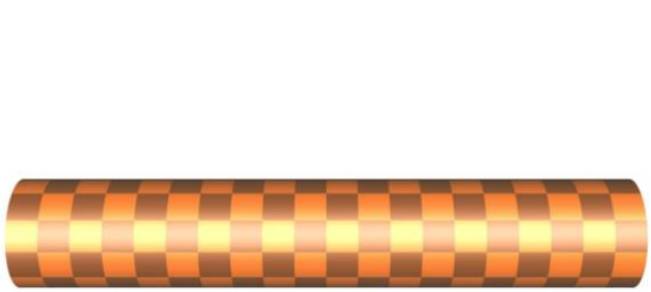
Dual-Quaternion Linear Blending

$$\hat{\boldsymbol{q}}_t = (1-t)\hat{\boldsymbol{q}}_{\boldsymbol{0}} + t\hat{\boldsymbol{q}}_{\boldsymbol{1}}$$

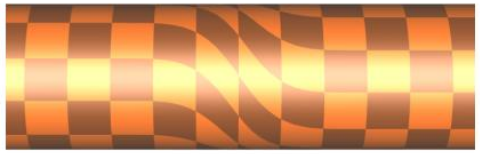
$$\hat{\boldsymbol{q}}_t = \frac{(1-t)\hat{\boldsymbol{q}}_{\boldsymbol{0}} + t\hat{\boldsymbol{q}}_{\boldsymbol{1}}}{\|(1-t)\hat{\boldsymbol{q}}_{\boldsymbol{0}} + t\hat{\boldsymbol{q}}_{\boldsymbol{1}}\|}$$



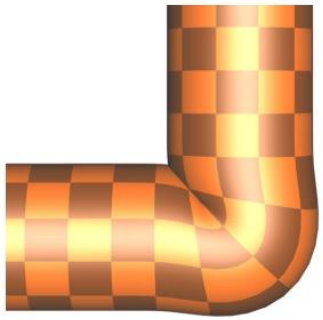
Dual-Quaternion Skinning (DQS)



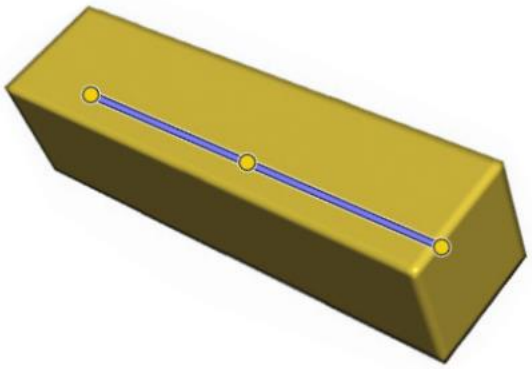
Rest pose



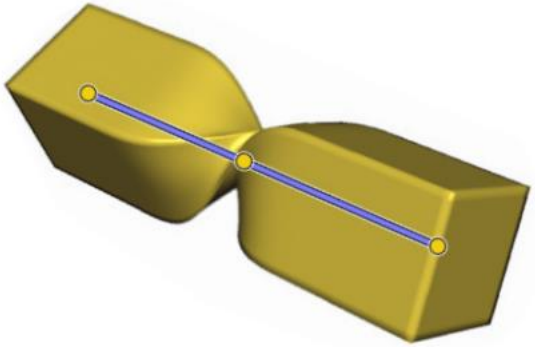
Dual quaternions: twist



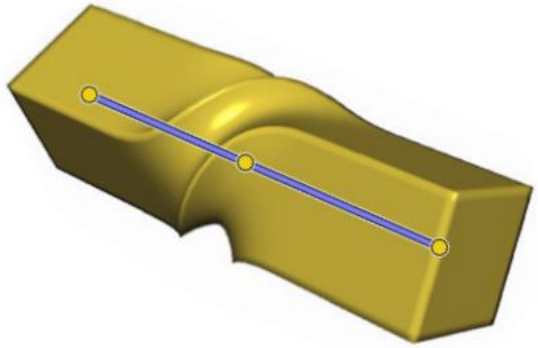
Dual quaternions: bend



Rest pose

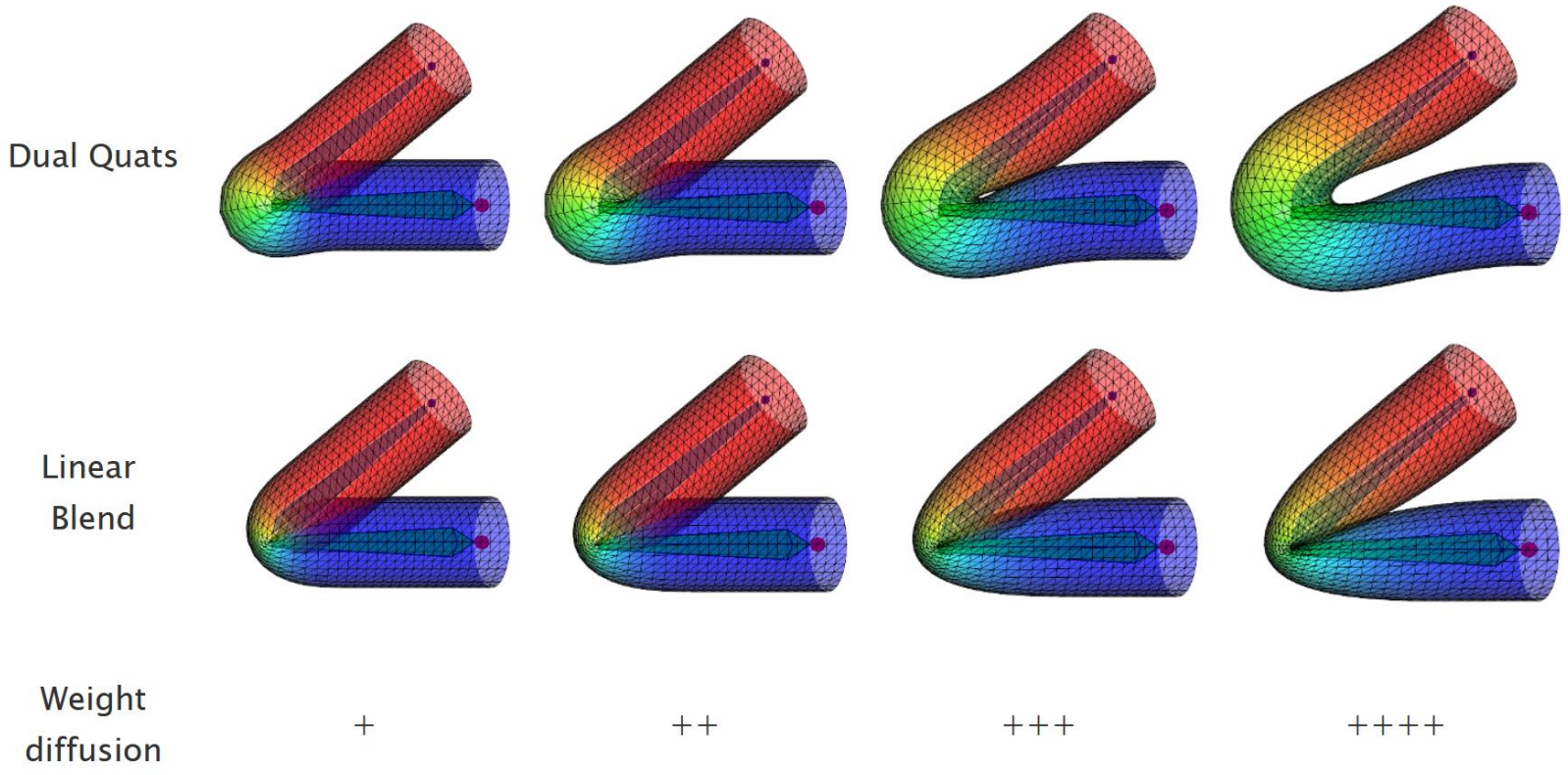


Linear blend skinning



Dual quaternion skinning

Budging Artifact of DQS



http://rodolphe-vaillant.fr/entry/29/dual-quaternions-skinning-tutorial-and-c-codes

Non-rigid transformations

1. Scaling

2. Rotations

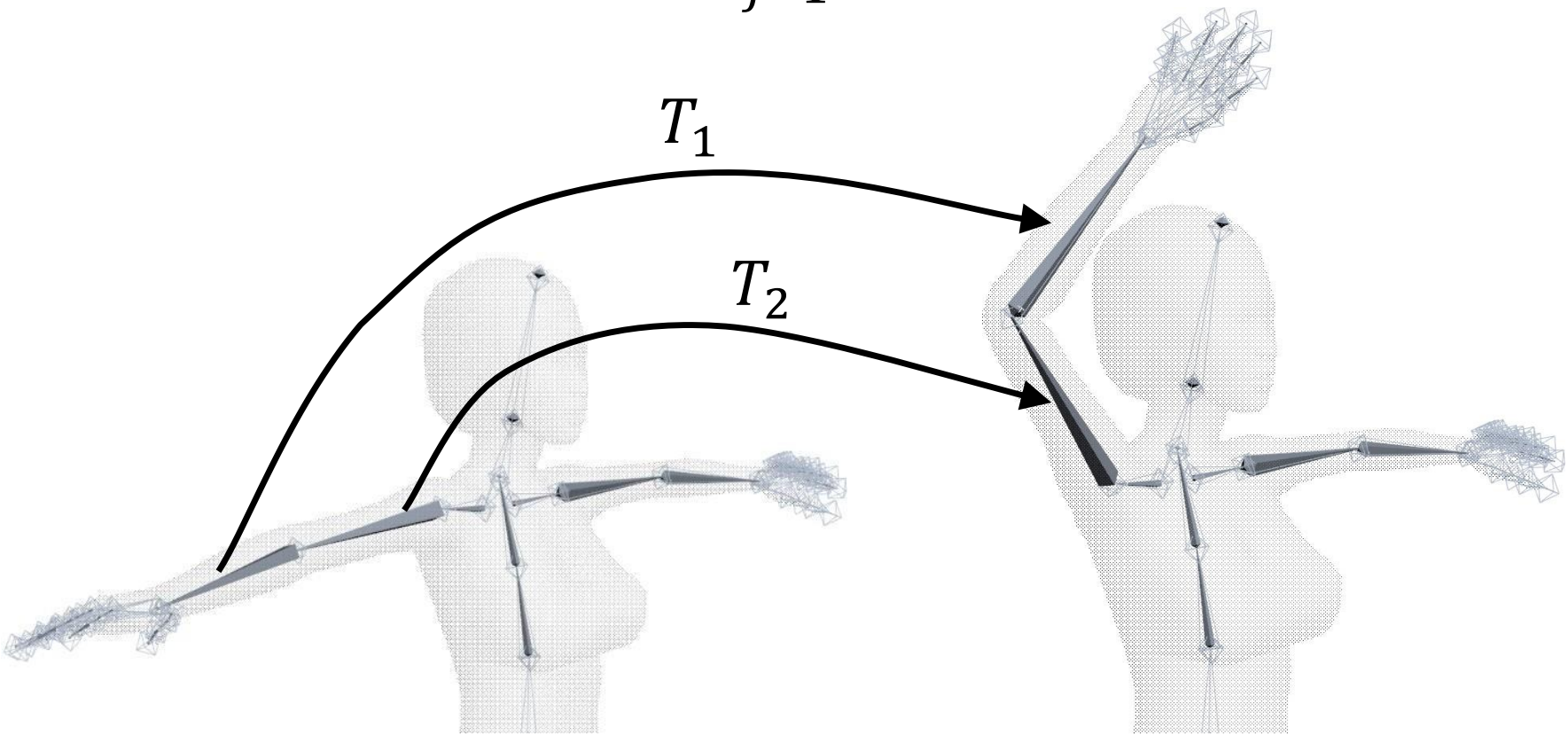
Two-phase skinning

Enhanced for production use by Disney

[Lee et al. 2013]

Linear Blend Skinning (LBS)

$$x_i' = \sum_{j=1}^{m} \omega_{ij} T_j x_i$$



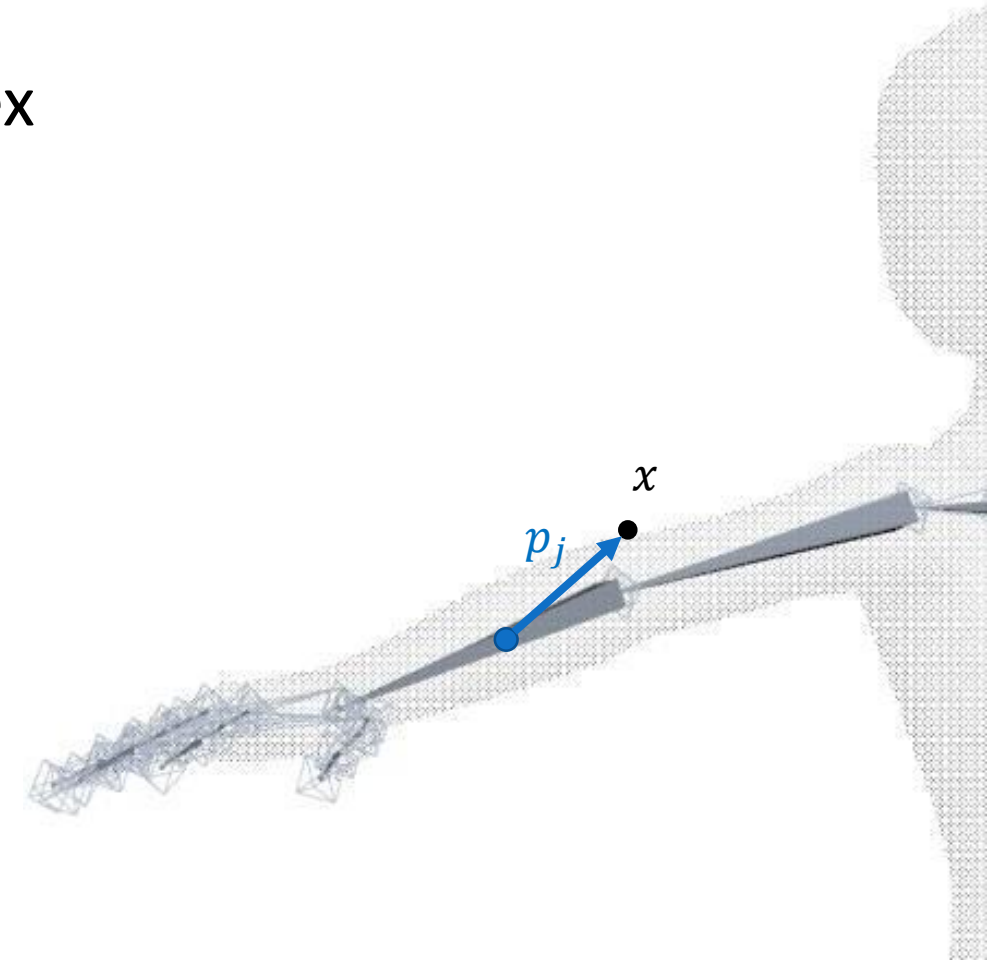
Linear Blend Skinning (LBS)

- Consider a single vertex

$$x' = \sum_{j=1}^{m} \omega_j T_j x$$

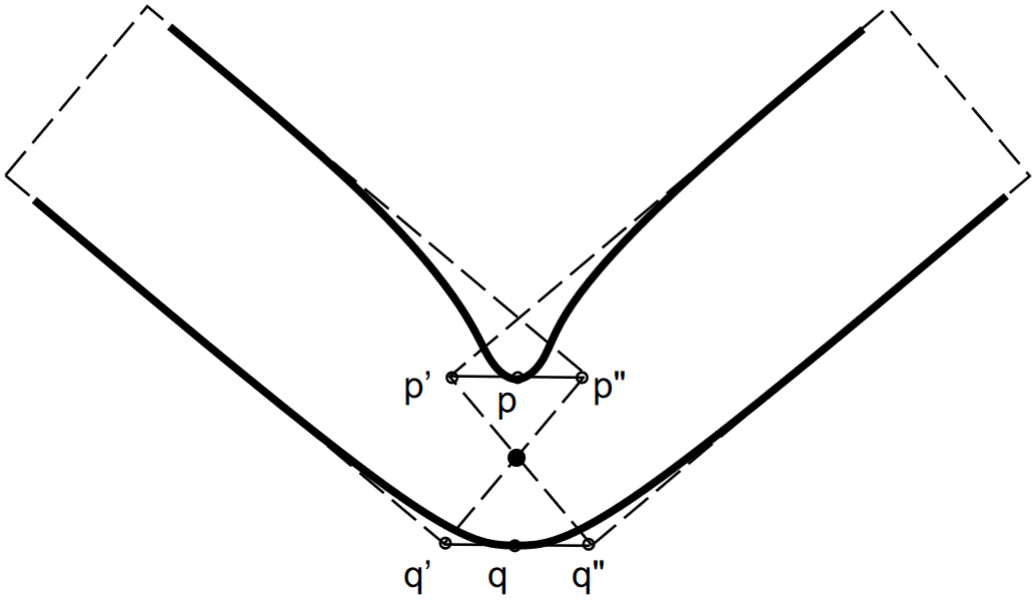
$$= \sum_{j=1}^{m} \omega_j T_j^{\delta} \hat{\boldsymbol{T}}_{\boldsymbol{j}}^{-\mathbf{1}} \boldsymbol{x}$$

$$= \sum_{j=1}^{m} \omega_j T_j^{\delta} \boldsymbol{p_j}$$



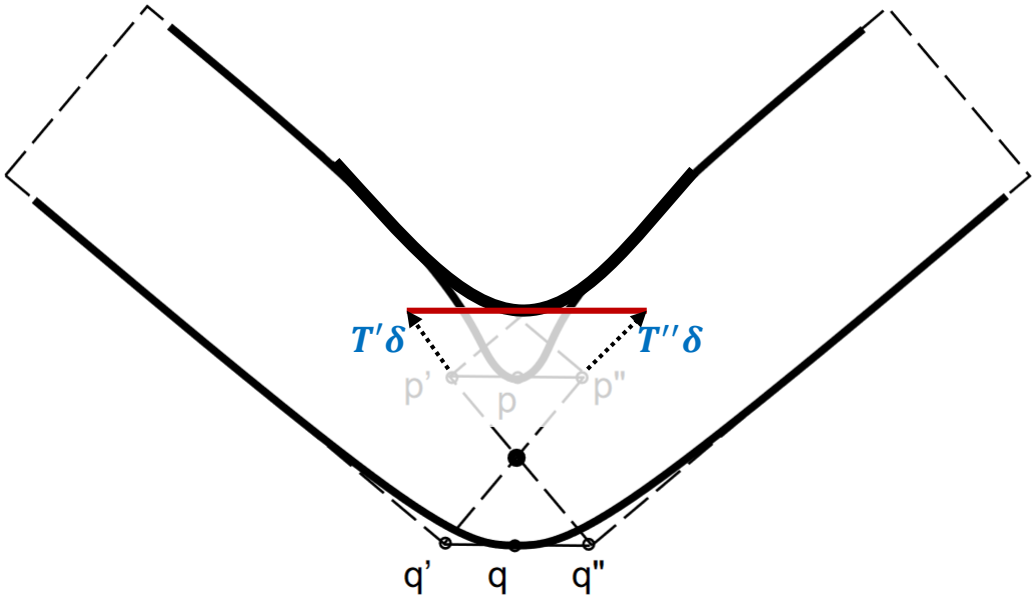
How to Correct LBS?

$$\boldsymbol{x}' = \sum_{j=1}^{m} \omega_j T_j \boldsymbol{x}$$



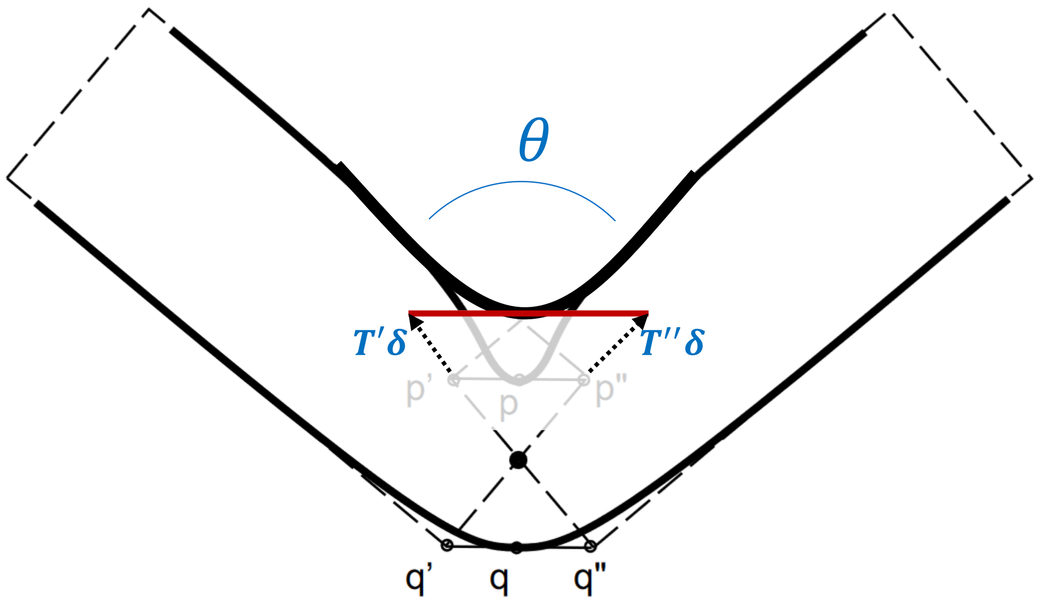
How to Correct LBS?

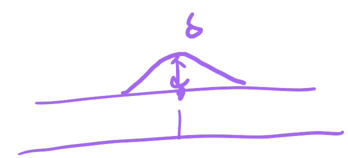
$$\boldsymbol{x}' = \sum_{j=1}^{m} \omega_j T_j\big(\boldsymbol{x} + \boldsymbol{\delta}(\boldsymbol{x})\big)$$



How to Correct LBS?

$$x' = \sum_{j=1}^{m} \omega_j T_j\left(x + \delta(x, \theta)\right)$$

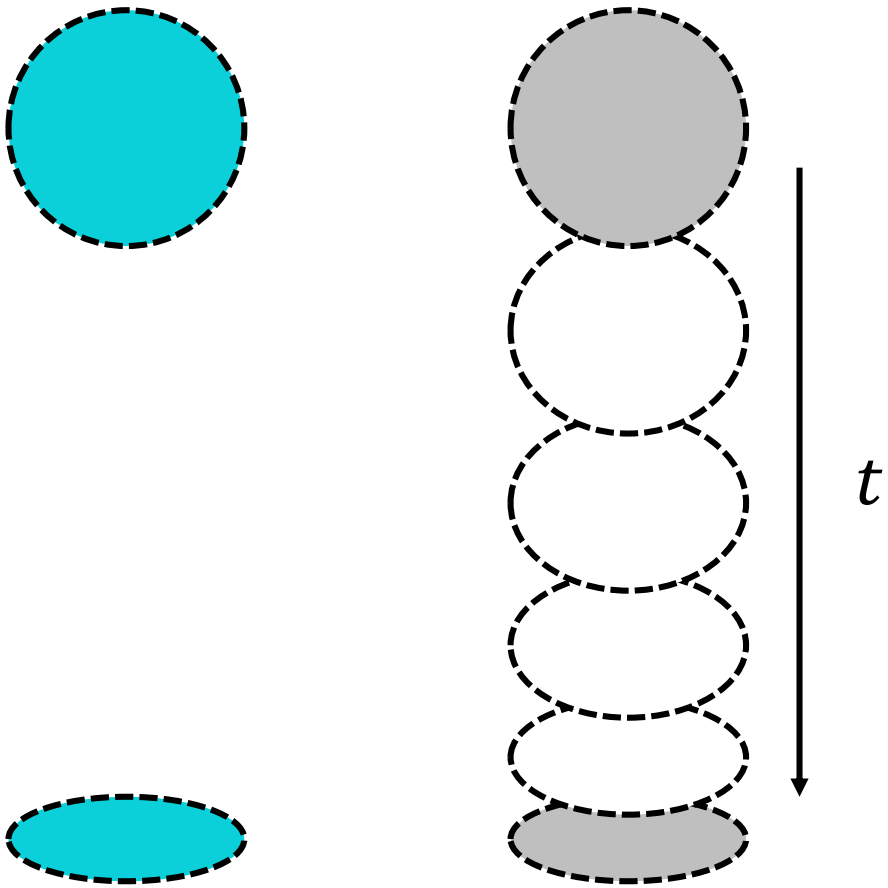




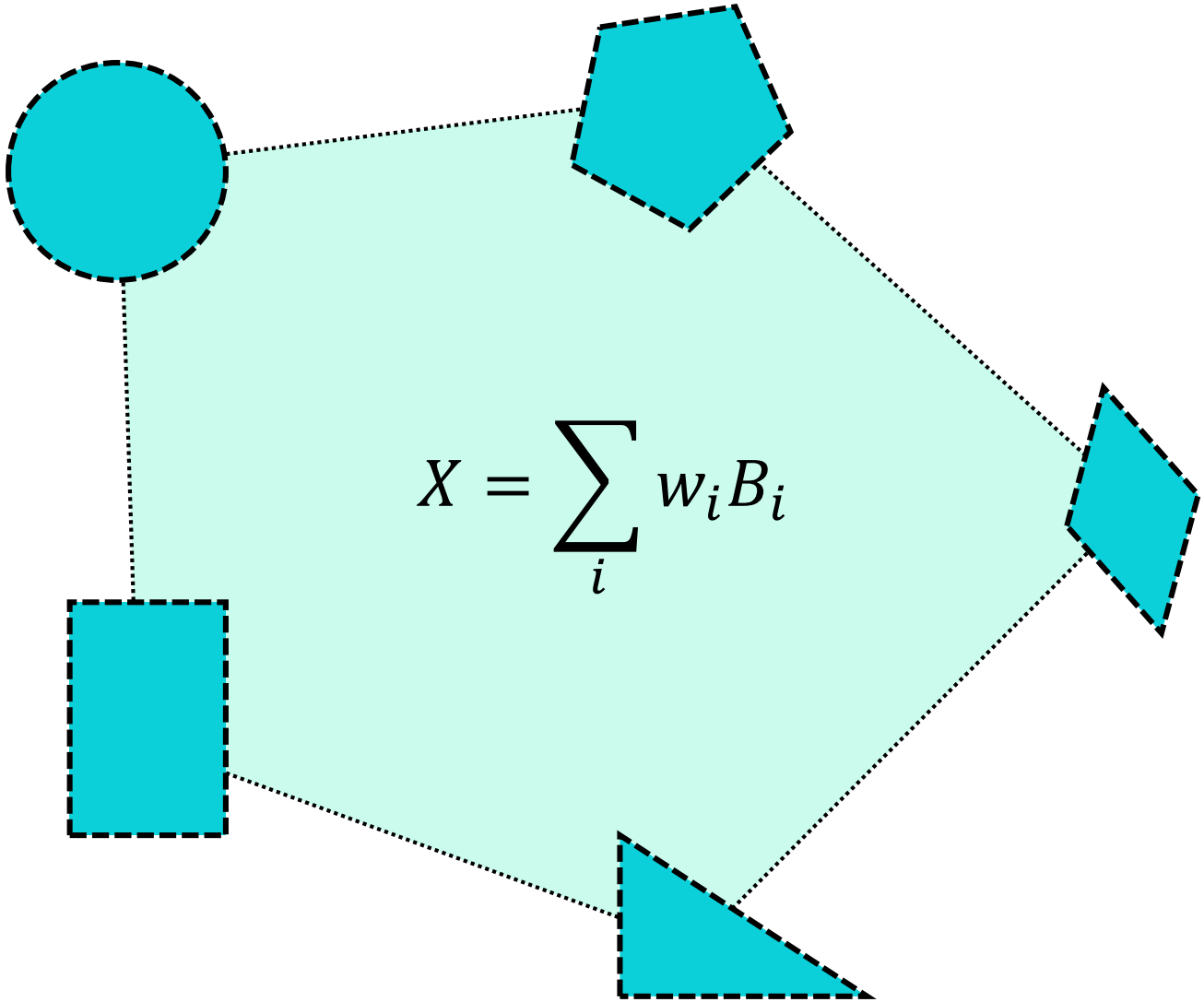
这样插值之后不会下凹，实际改变了原本的skinny的模型

也可不改多，改δ(x,θ)

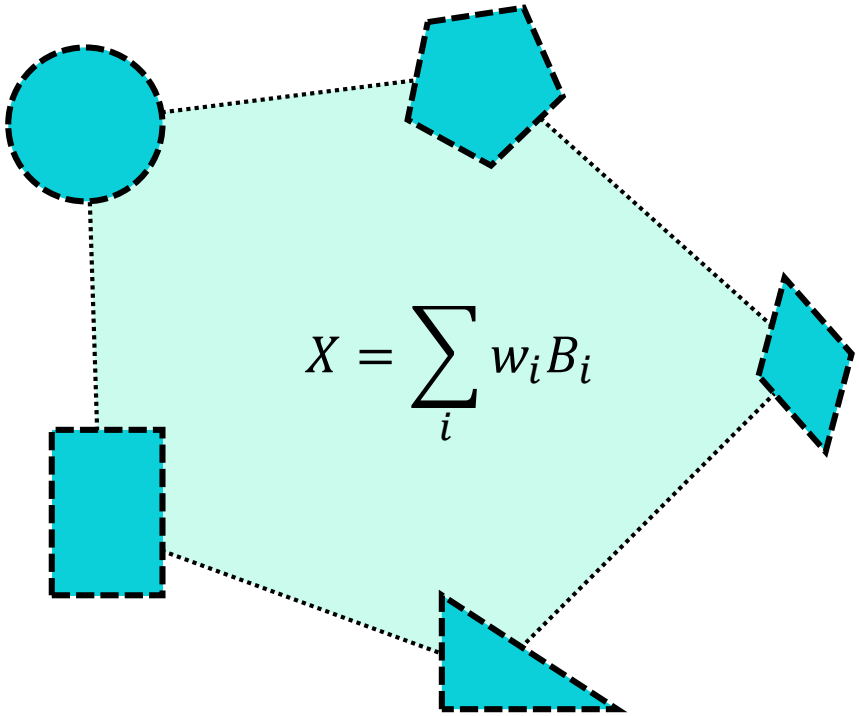
Example-based Shape Deformation



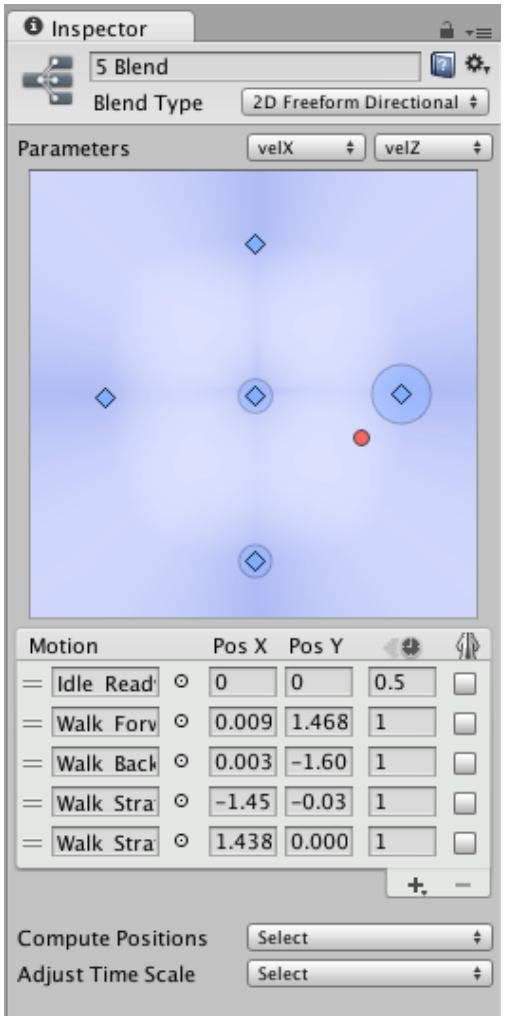
Blendshapes / Blend Space



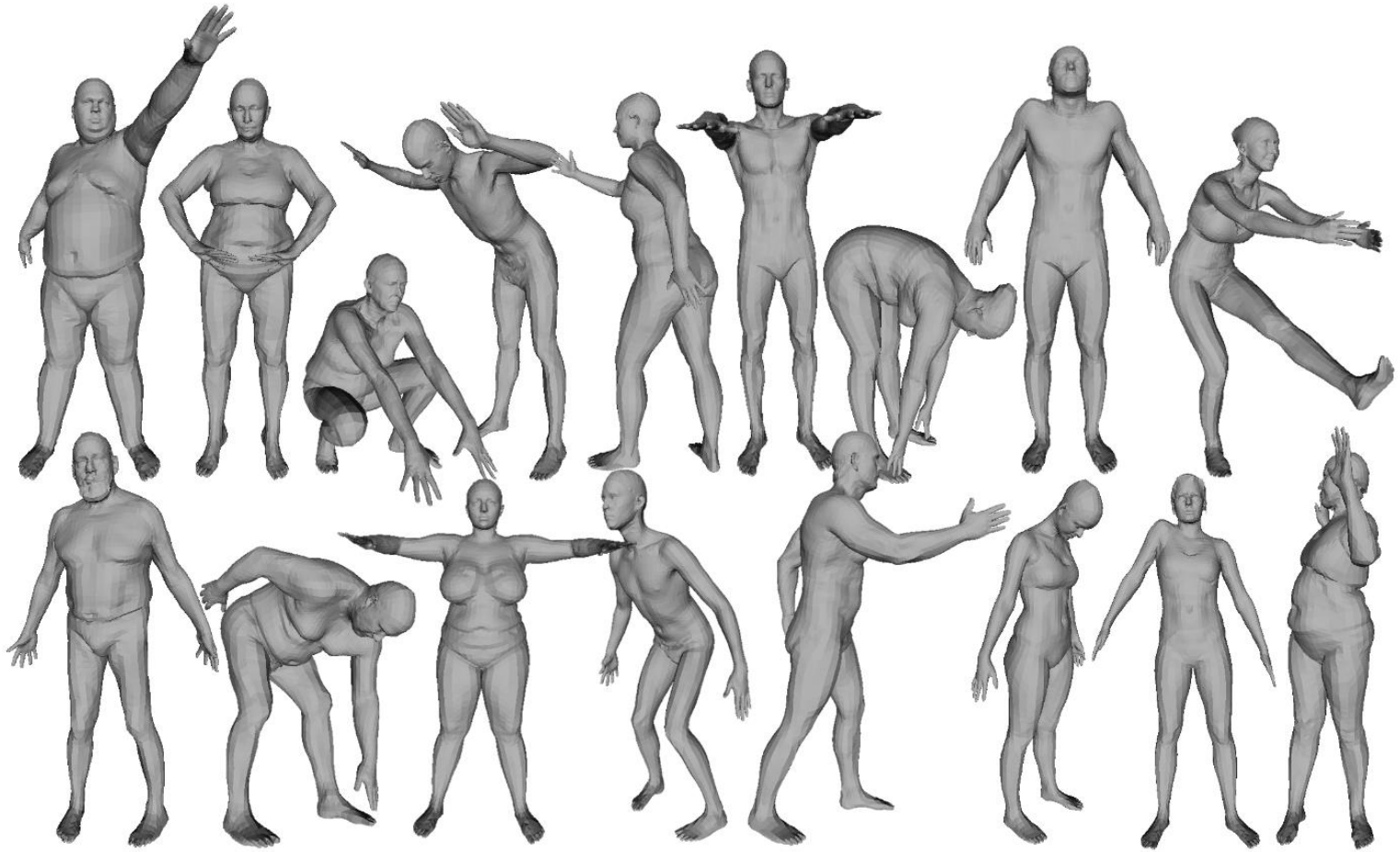
Blendshapes / Blend Space



由一组基构成的linear system
同时也是对animation有用，可以把多个姿势生成一个新的姿态



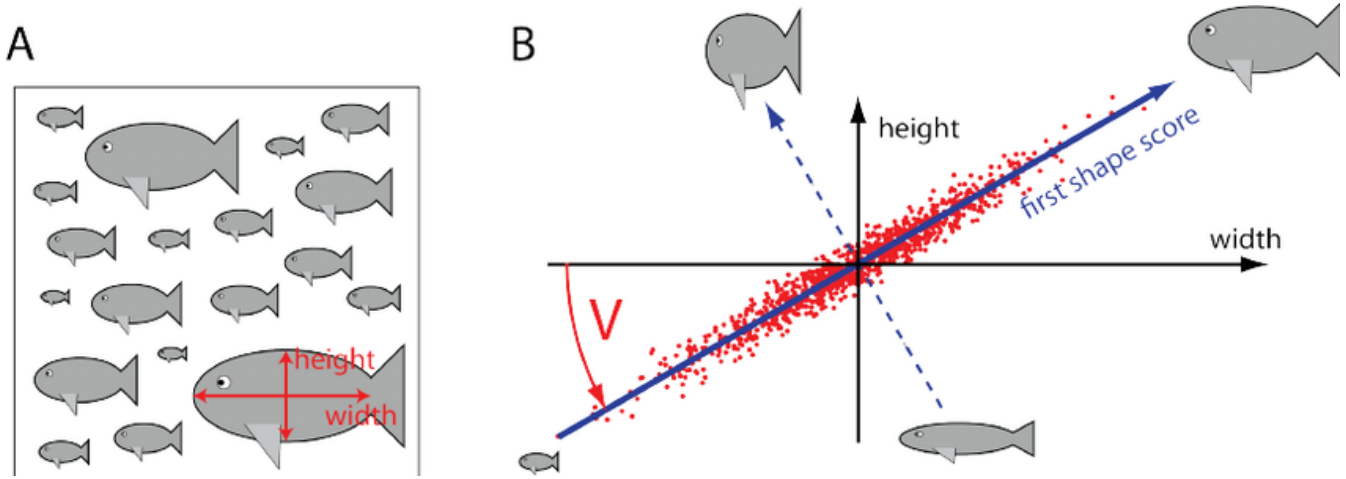
How to deal with massive examples?



[SMPL: A Skinned Multi-Person Linear Model]

Principal Component Analysis (PCA)

- A technique for
  - finding out the correlations among dimensions
  - dimensionality reduction



Werner, Steffen & Rink, Jochen & Friedrich, Benjamin. (2014). Shape Mode Analysis Exposes Movement Patterns in Biology: Flagella and Flatworms as Case Studies. PloS one. 9. 10.1371/journal.pone.0113083.

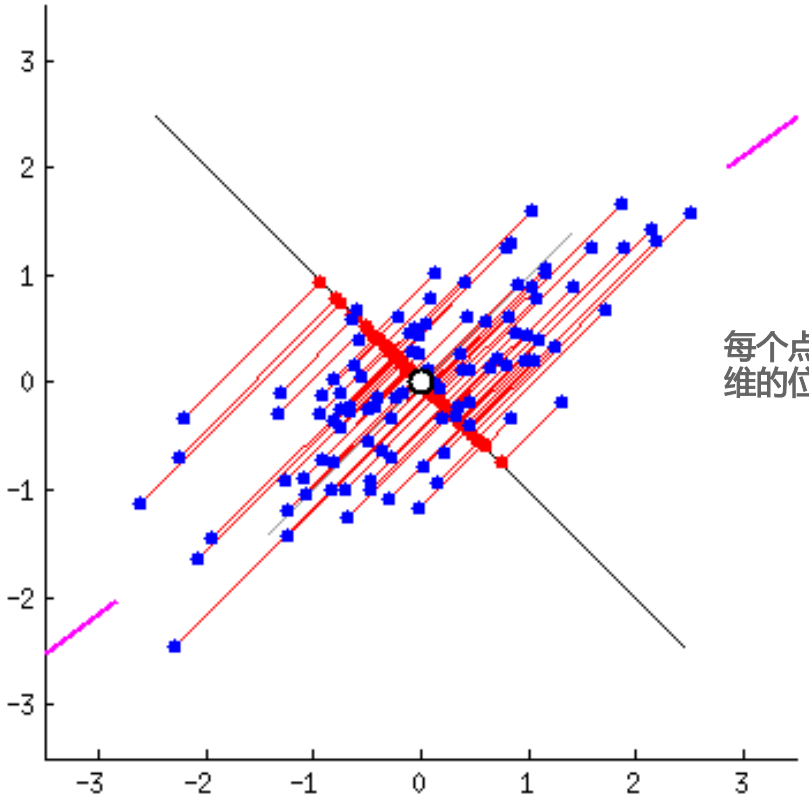
PCA in a Nutshell

- Given a dataset $\{\boldsymbol{x}_i\}, \boldsymbol{x}_i \in \mathbb{R}^N$, then PCA gives

$$\boldsymbol{x}_i = \overline{\boldsymbol{x}} + \sum_{k=1}^{n} w_{i,k}\boldsymbol{u}_k$$

- $\boldsymbol{u}_k$ is the k-th principal component
  - A direction in $\mathbb{R}^N$ along which the projection of $\{\boldsymbol{x}_i\}$ has the k-th maximal **variance**
- $w_{i,k} = (\boldsymbol{x}_i - \overline{\boldsymbol{x}}) \cdot \boldsymbol{u}_k$ is the score of $\boldsymbol{x}_i$ on $\boldsymbol{u}_k$

PCA in a Nutshell



每个点都在直线上对应一个一维的位置

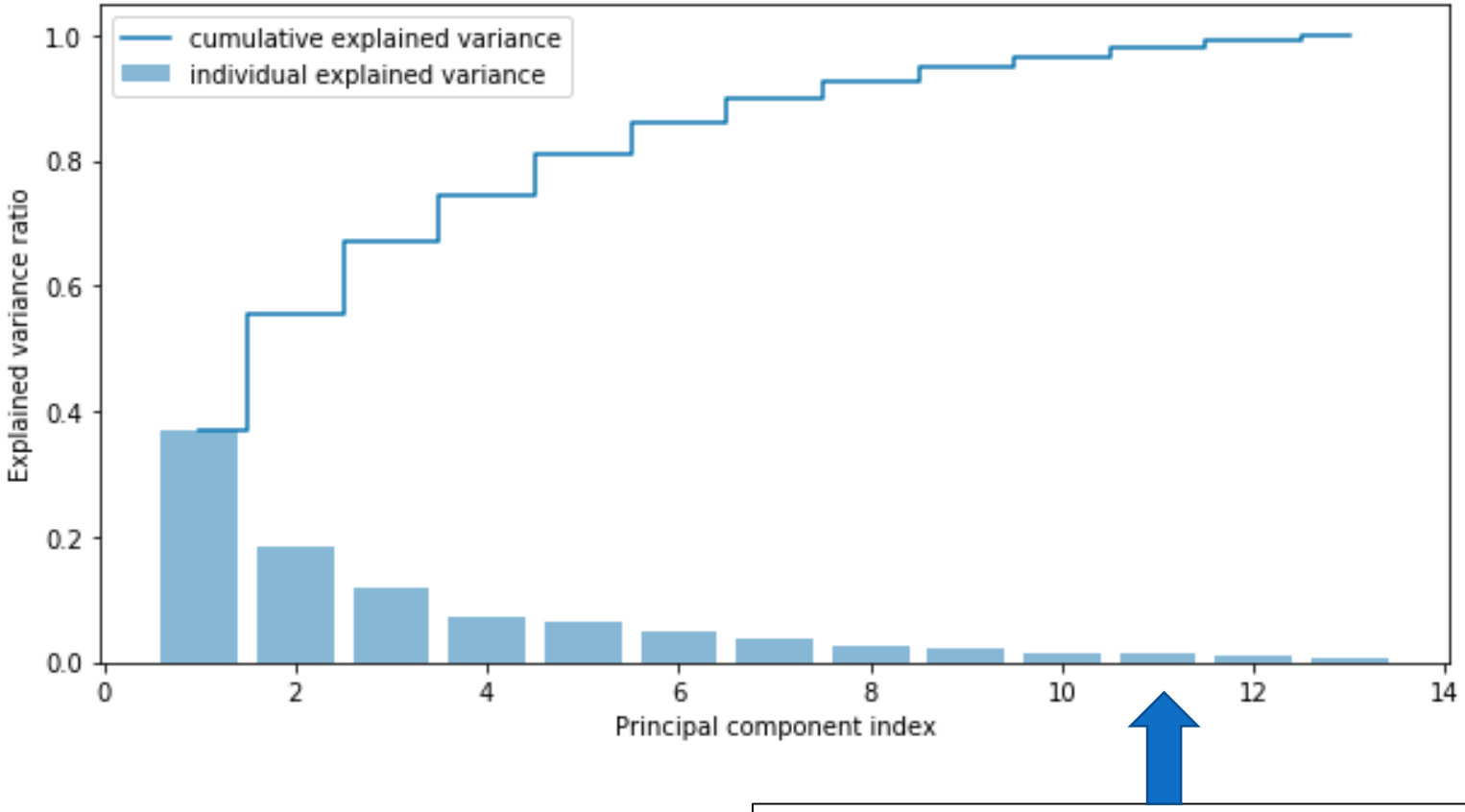
PCA in a Nutshell

- Given a dataset $\{\boldsymbol{x}_i\}, \boldsymbol{x}_i \in \mathbb{R}^N$, the PCA can be computed by apply eigen decomposition on covariance matrix

$$X^T X = U \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_N \end{bmatrix} U^T$$

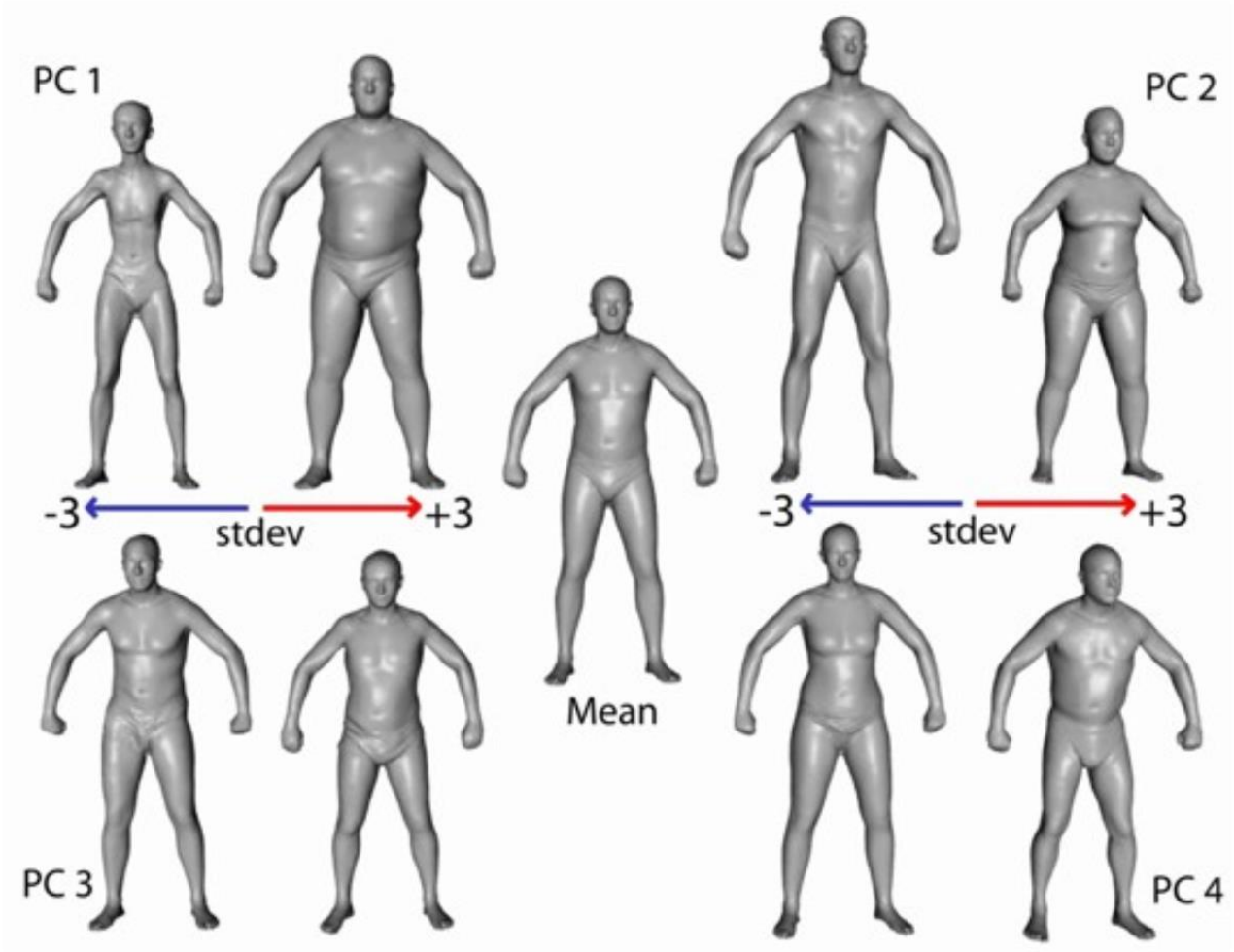
- $X = [x_0 - \bar{x}, x_1 - \bar{x}, \dots, x_N - \bar{x}]^T$
- $\lambda_i \geq \lambda_j$ when $i < j$, corresponds to the Explained Variance
- $U = [\boldsymbol{u_1}, \boldsymbol{u_2}, \dots, \boldsymbol{u_N}]$

PCA in a Nutshell



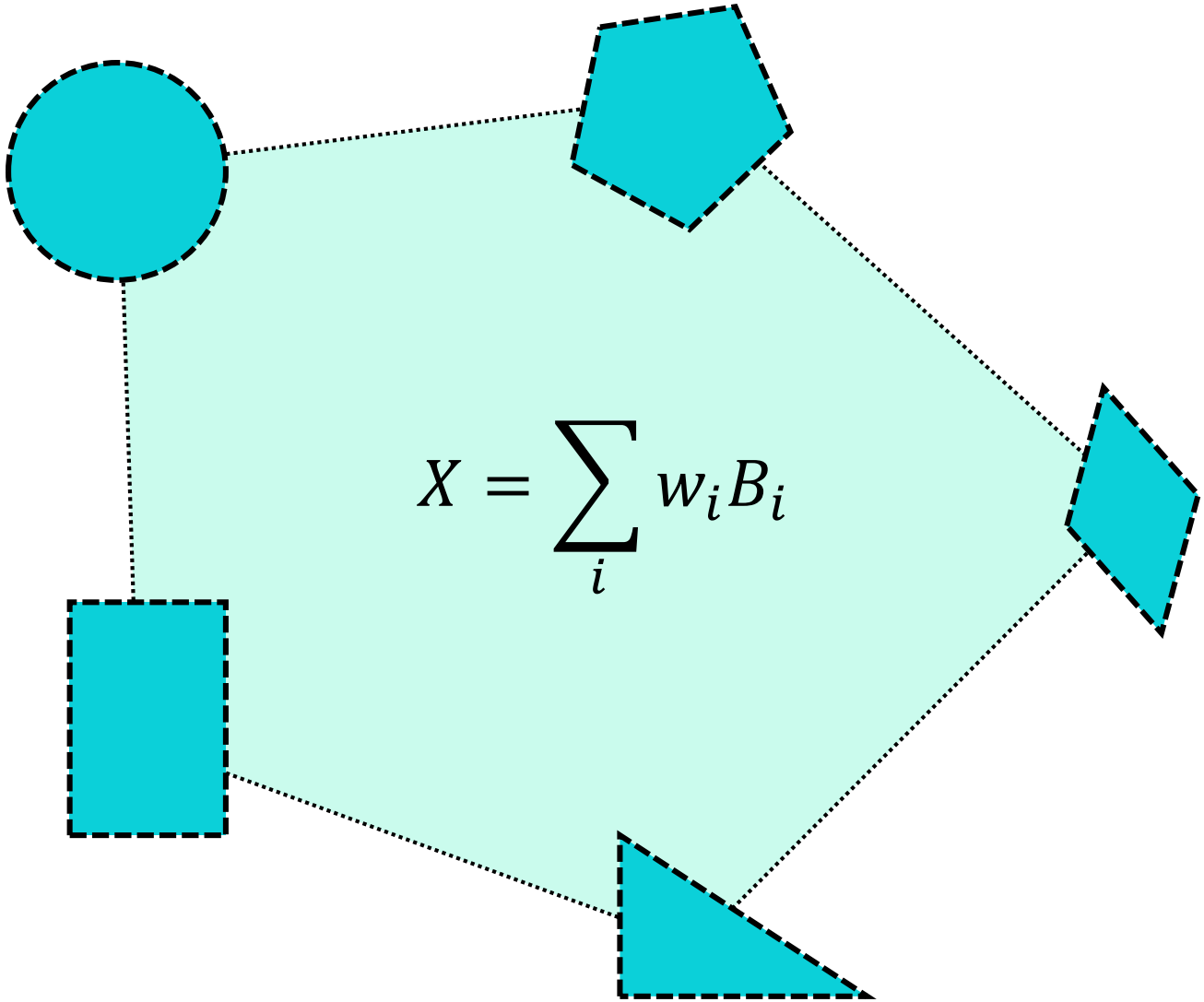
Choose n principal components that explains enough (e.g. 95%) of the variance

PCA over Body Shapes

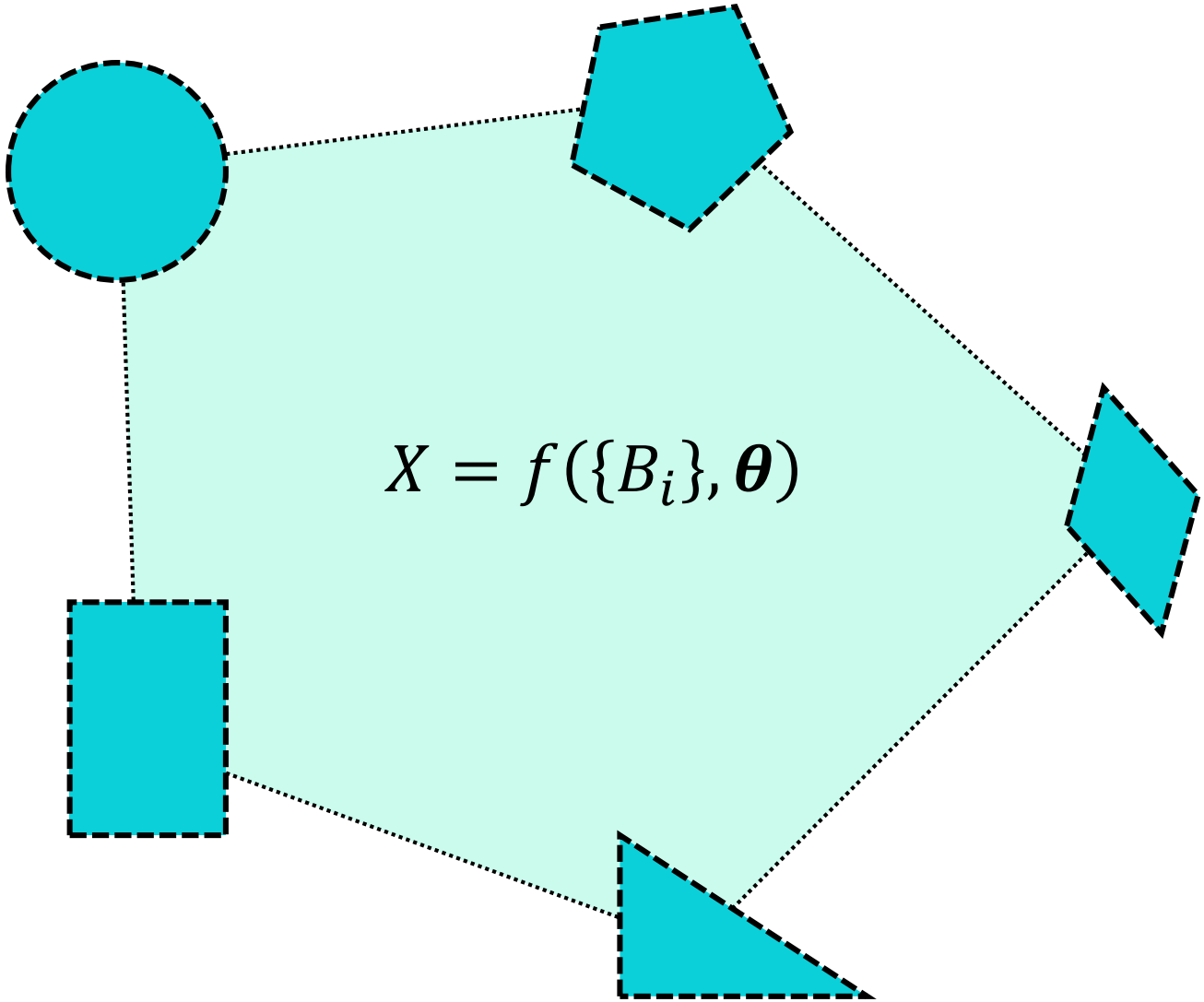


Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. 2005. ***SCAPE: shape completion and animation of people***. *ACM Trans. Graph.* 24, 3 (July 2005), 408–416.

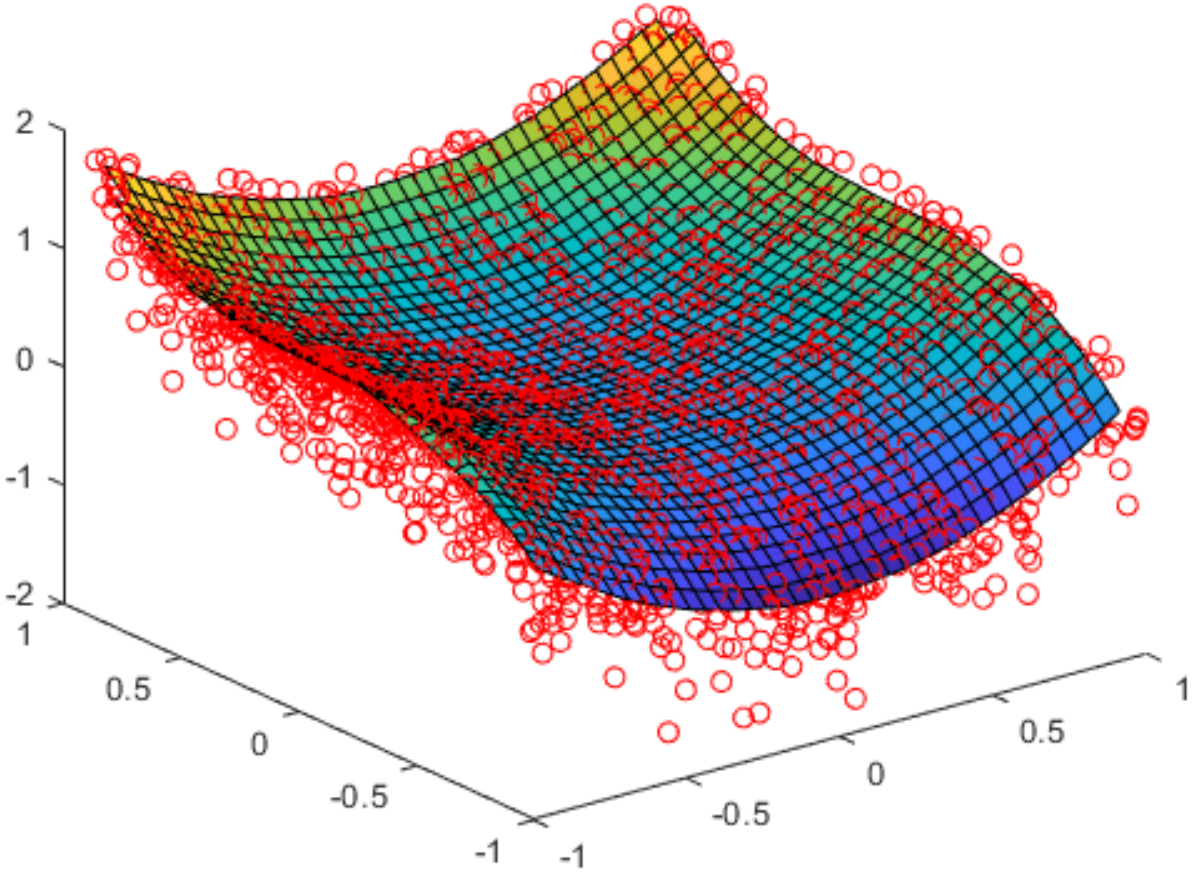
Blendshapes / Blend Space



Pose Space Deformation



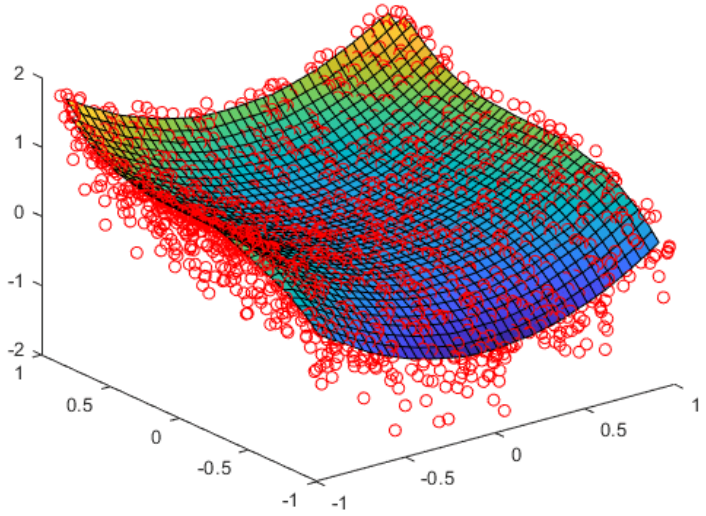
Scattered Data Interpolation



https://www.mathworks.com/help/matlab/ref/griddata.html

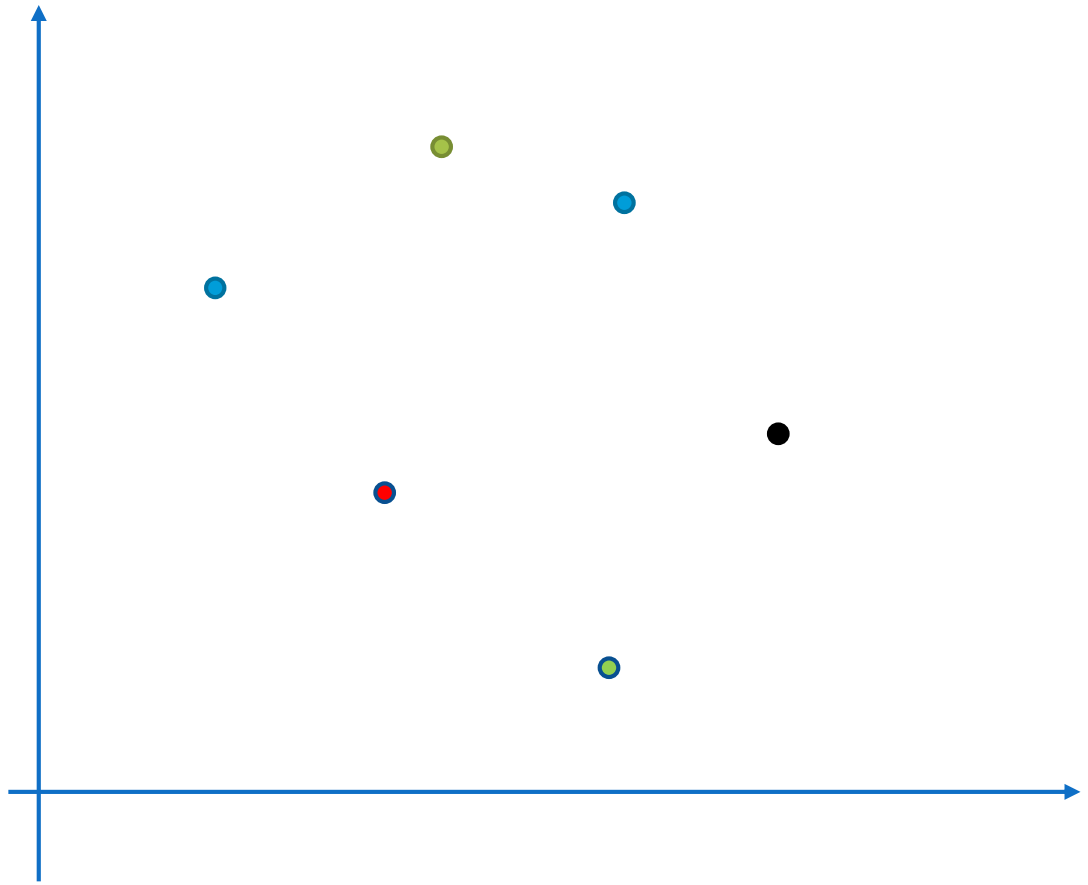
Scattered Data Interpolation

- Linear
  - Least squares
- Splines
- Inverse distance weighting
- Gaussian process
- Radial Basis Function
- ……

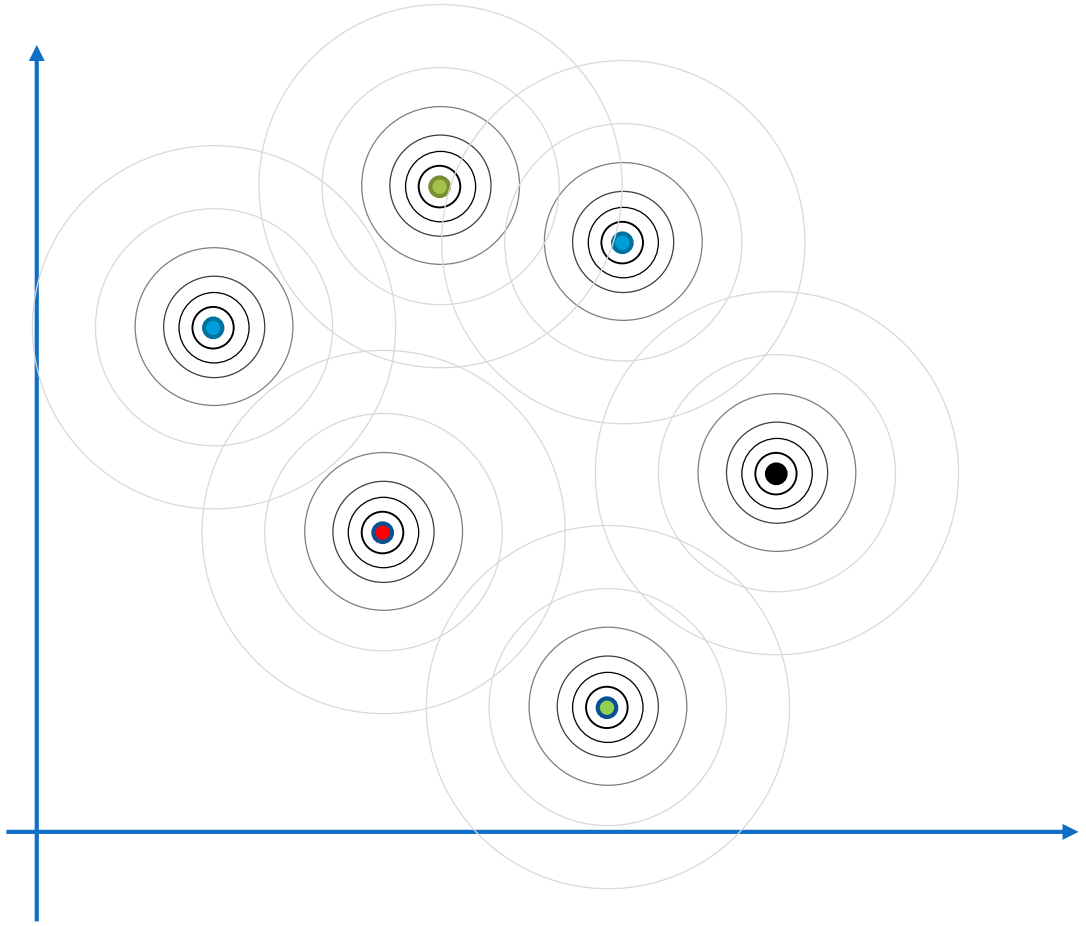


https://www.mathworks.com/help/matlab/ref/griddata.html

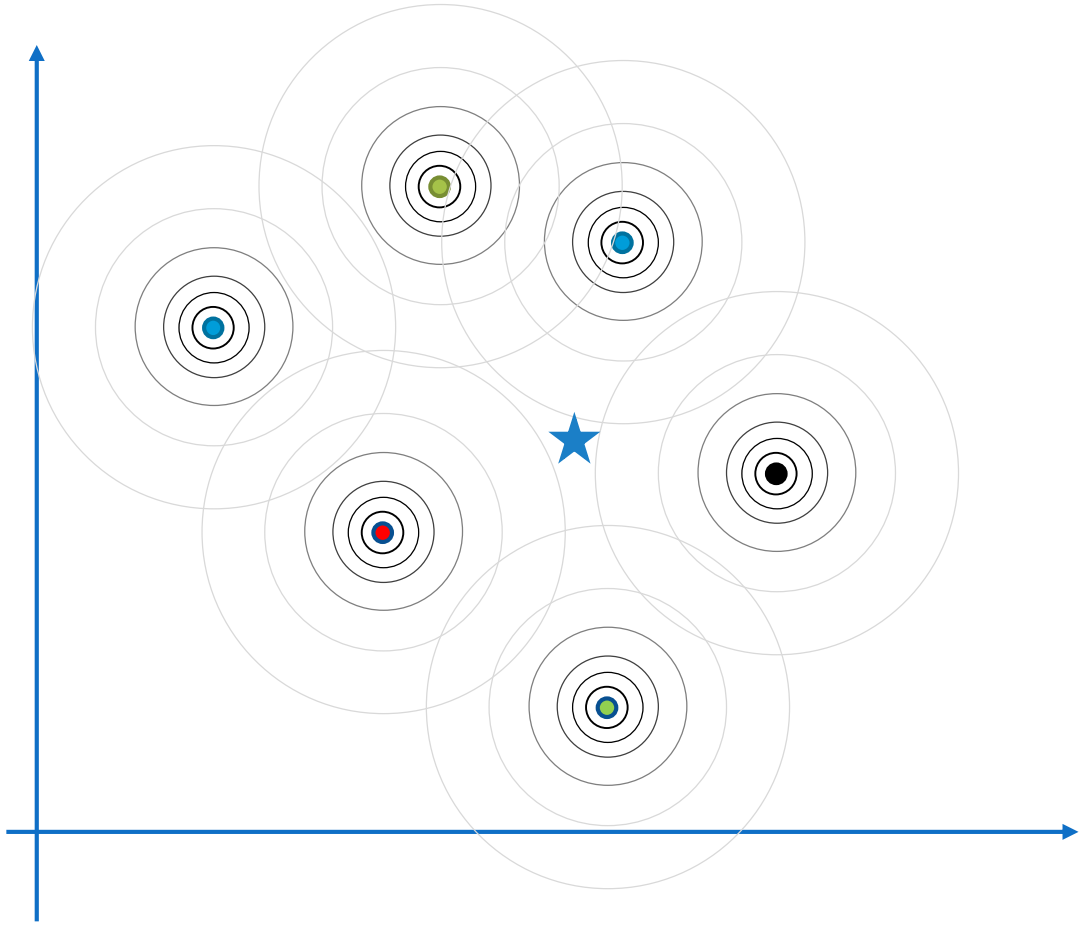
Radial Basis Function (RBF) Interpolation



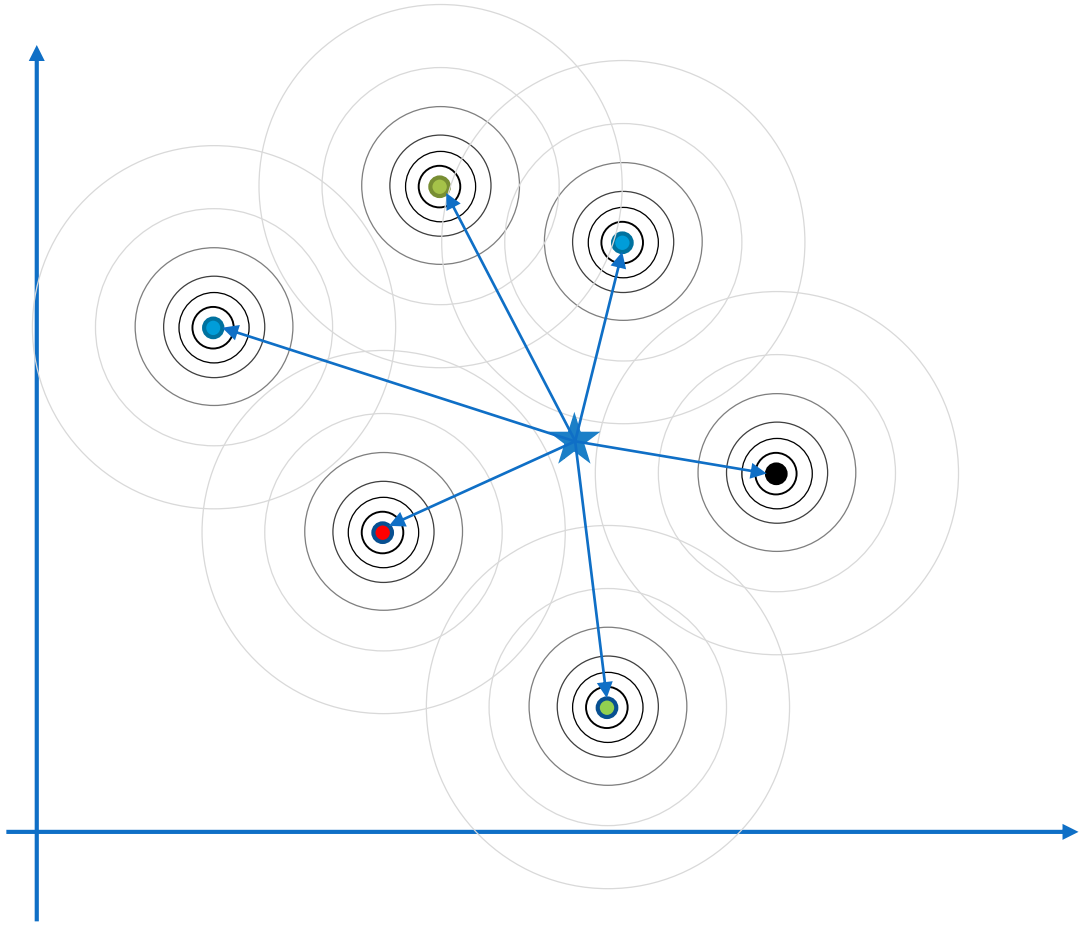
Radial Basis Function (RBF) Interpolation



Radial Basis Function (RBF) Interpolation

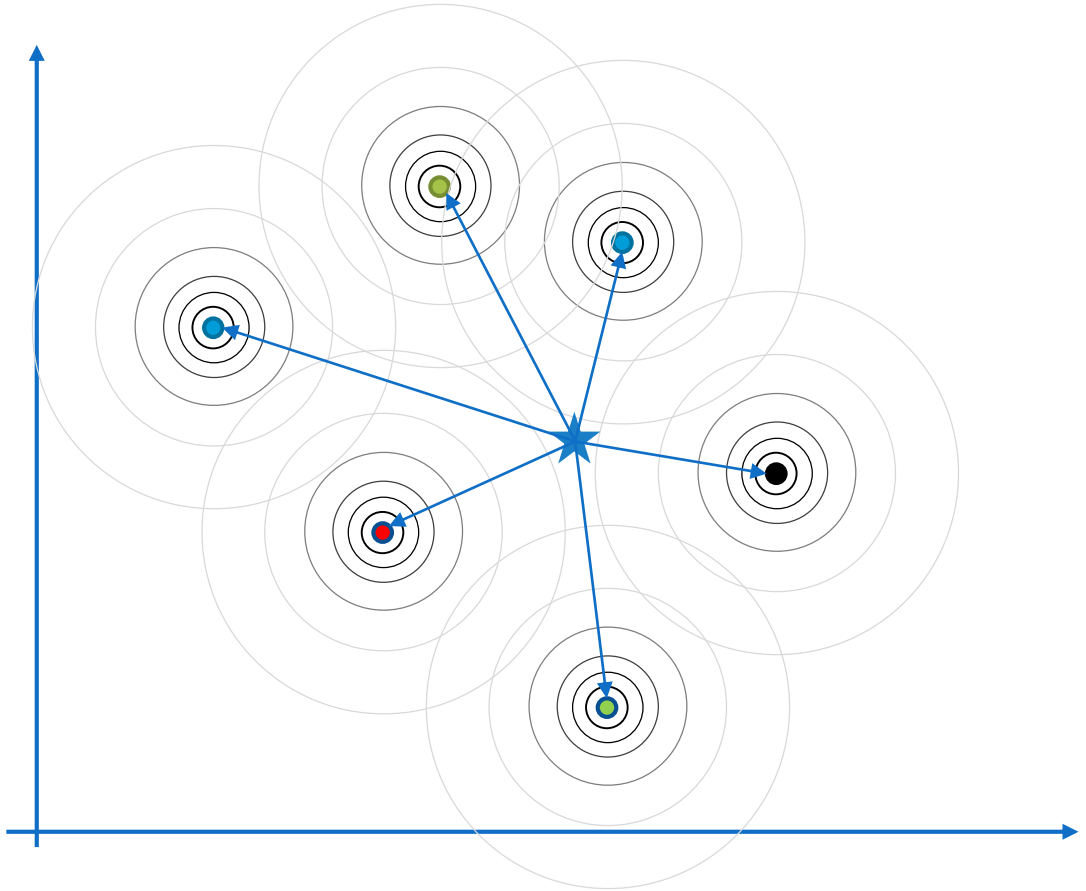


Radial Basis Function (RBF) Interpolation



Radial Basis Function (RBF) Interpolation

$$x = \sum_{i=1}^{K} w_i \varphi(\|x - x_i\|)$$



Radial Basis Function (RBF) Interpolation

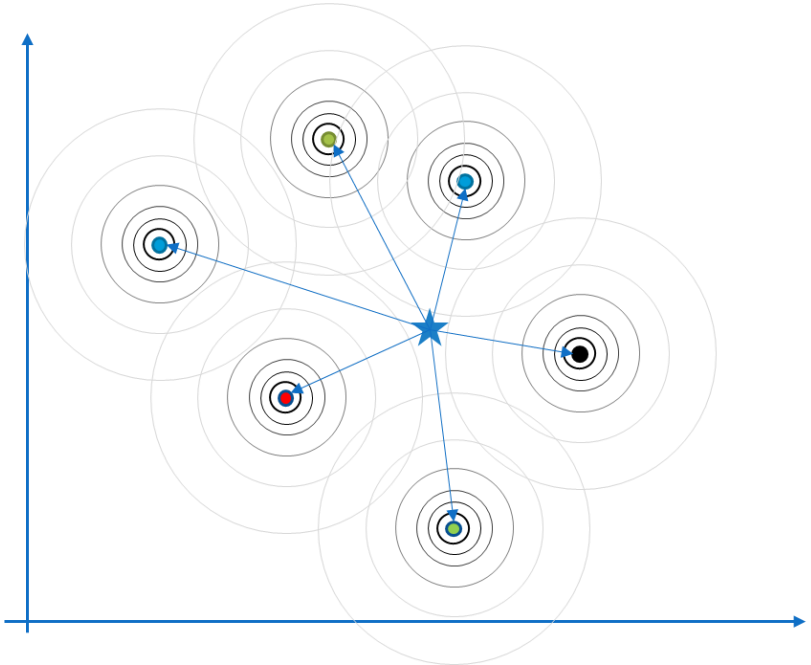
$$x = \sum_{i=1}^{K} w_i \varphi(\|x - x_i\|)$$

How to compute w_i ?

Radial Basis Function (RBF) Interpolation

$$x = \sum_{i=1}^{K} w_i \varphi(\|x - x_i\|)$$

How to compute w_i ?



$$\begin{bmatrix} R_{1,1} & R_{1,2} & \cdots & R_{1,K} \\ R_{2,1} & R_{2,2} & & \vdots \\ \vdots & & \ddots & \vdots \\ R_{K,1} & \cdots & \cdots & R_{K,K} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \end{bmatrix}$$

$$R_{i,j} = \varphi(\|x_i - x_j\|)$$

Radial Basis Function (RBF)

$$x = \sum_{i=1}^{K} w_i \varphi(\|x - x_i\|)$$

- Gaussian: $\varphi(r) = e^{-(r/c)^2}$
- Inverse multiquadric: $\varphi(r) = \frac{1}{\sqrt{r^2+c^2}}$
- Thin plate spline: $\varphi(r) = r^2 \log r$
- Polyharmonic splines: $\varphi(r) = \begin{cases} r^k, & k = 2n+1 \\ r^k \log r, & k = 2n \end{cases}$

Pose Space Deformation

$$\boldsymbol{x}' = \sum_{j=1}^{m} \omega_j T_j\big(\boldsymbol{x} + \boldsymbol{\delta}(\boldsymbol{x}, \theta)\big)$$

- $\boldsymbol{x}' = SKIN\big(PSD(\boldsymbol{x})\big)$
- PSD is implemented as RBF interpolation

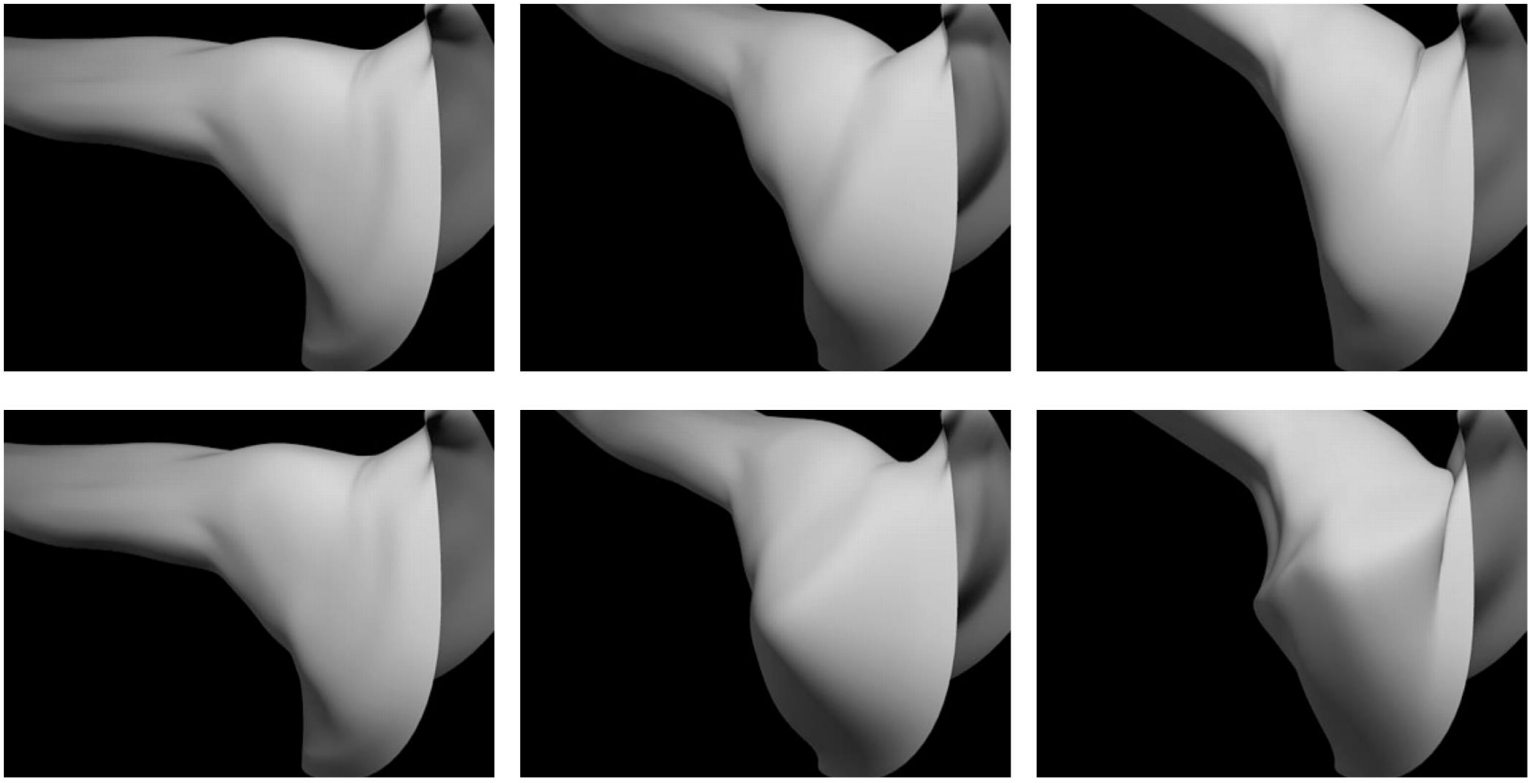
PSD: local

J. P. Lewis, Matt Cordner, and Nickson Fong. 2000. ***Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation***. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '00), ACM Press/Addison-Wesley Publishing Co., USA, 165–172.

Pose Space Deformation

PSD

LBS



J. P. Lewis, Matt Cordner, and Nickson Fong. 2000. ***Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation***. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '00), ACM Press/Addison-Wesley Publishing Co., USA, 165–172.

Issues

- Per-shape or per-vertex interpolation
  - Should we interpolate a shape as a whole?
- Local or global interpolation?
  - Should a vertex be affected by all joints?
- Interpolation algorithm?
  - Is RBF the only choice?

Example-based Skinning (EBS) vs. Skeleton Subspace Deformation (SSD)

*SSD (LBS, DQS, etc.)

- Good: Easy to control
- Good: Good quality
- Good: Pose-dependent details (e.g. bulging muscle and extruding veins)
- Bad: Creating examples can be cumbersome
- Bad: Extra storage for examples
- Bad: Interpolation needs careful tuning

Other Example-based Skinning

- Automatically compute weights using examples
- Example-based simulation
  - We will touch this part later…

SMPL Model

- A widely adopted human model in ML/CV
- Learned on real scan data
- Combines SSD and EBS techniques

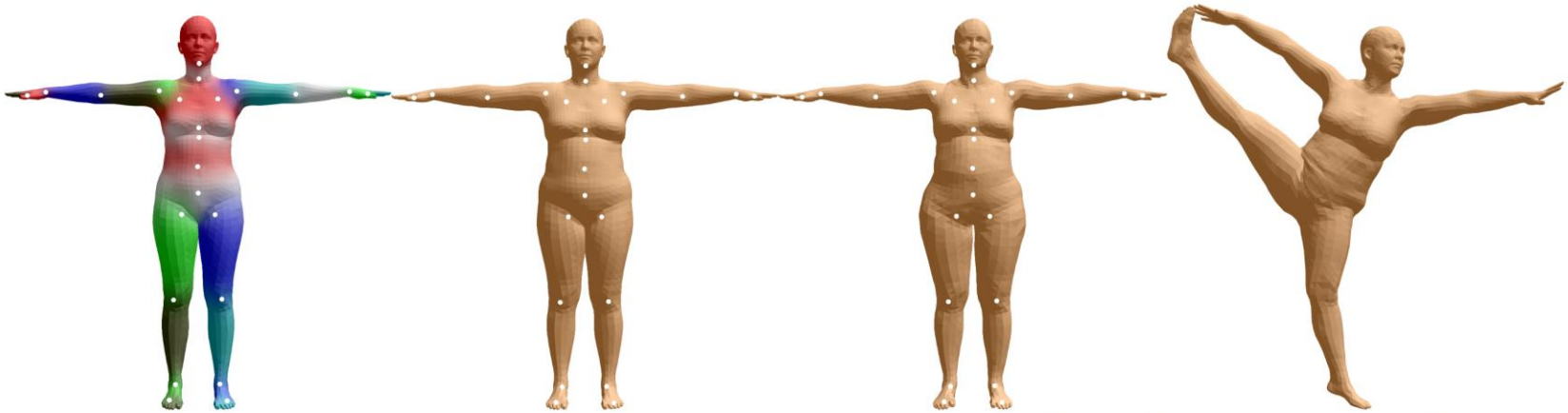


Figure 3: SMPL model. *(a) Template mesh with blend weights indicated by color and joints shown in white. (b) With identity-driven blendshape contribution only; vertex and joint locations are linear in shape vector $\vec{\beta}$. (c) With the addition of of pose blend shapes in preparation for the split pose; note the expansion of the hips. (d) Deformed vertices reposed by dual quaternion skinning for the split pose.*

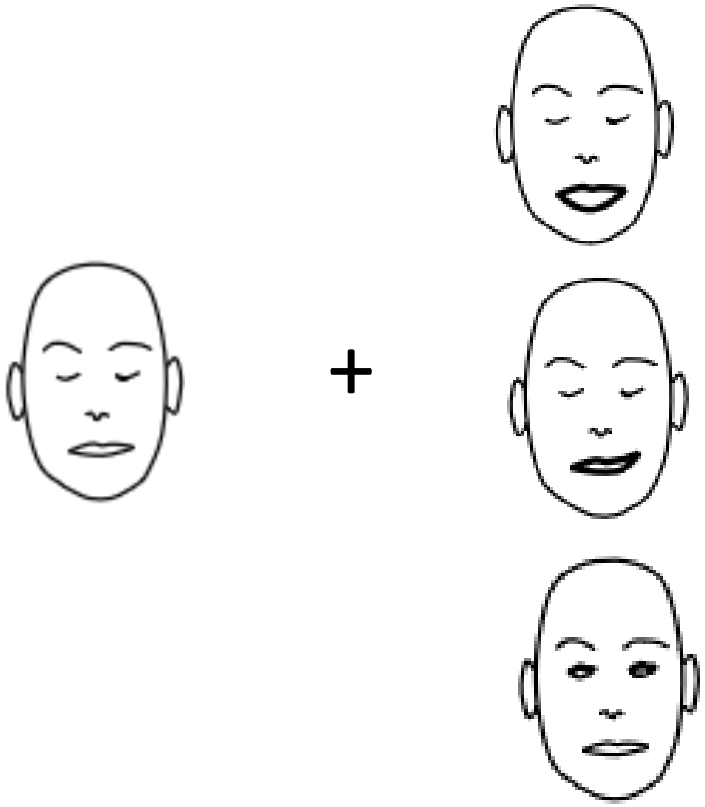
[SMPL: A Skinned Multi-Person Linear Model]

Facial Animation

[UnrealEngine]

Facial Animation

Facial Animation = Identity + Expression



Facial Animation

Facial Animation = Identity + Expression

$$X = X_0 + \sum_i \beta_i B_i^{\mathrm{ID}} + \sum_j \theta_j B_j^{\mathrm{Exp}}$$

X0是一个平均脸，identity是不做表情的基础脸部形状，如大脸之类的，实际上是base；
expression是一个基于example的参数，实际上混合了PCA和EXAMPLE

Facial Animation

Facial Animation = Identity + Expression

$$X = X_0 + \sum_i \beta_i B_i^{\mathrm{ID}} + \sum_j \theta_j B_j^{\mathrm{Exp}}$$

平均脸　　捏脸　　调表情

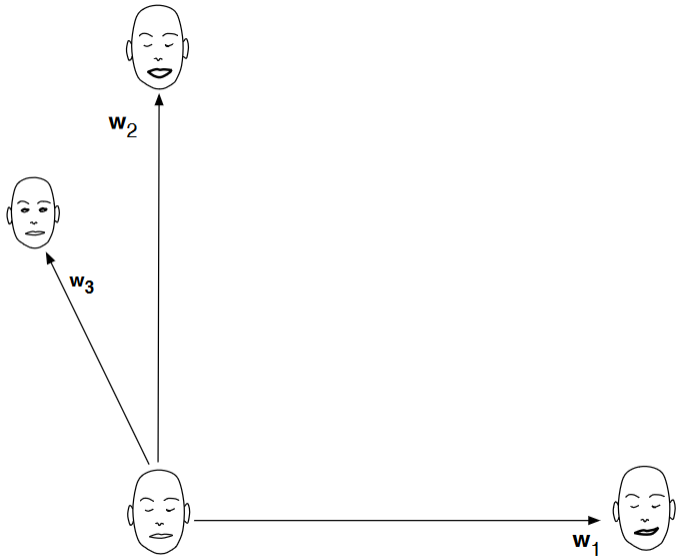
Facial Blendshapes

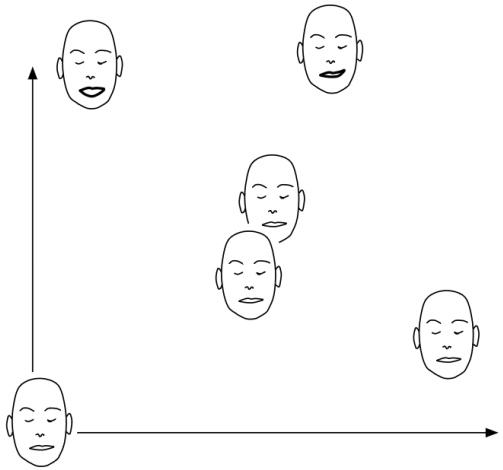
http://www.santhoshkoneru.com/facial-blendshapes

A Typical Set of Blendshapes (ARKit)

- eyeBlinkLeft
- eyeLookDownLeft
- eyeLookInLeft
- eyeLookOutLeft
- eyeLookUpLeft
- eyeSquintLeft
- eyeWideLeft
- eyeBlinkRight
- eyeLookDownRight
- eyeLookInRight
- eyeLookOutRight
- eyeLookUpRight
- eyeSquintRight
- eyeWideRight
- jawForward
- jawLeft
- jawRight
- jawOpen
- mouthClose
- mouthFunnel
- mouthPucker
- mouthRight
- mouthLeft
- mouthSmileLeft
- mouthSmileRight
- mouthFrownRight
- mouthFrownLeft
- mouthDimpleLeft
- mouthDimpleRight
- mouthStretchLeft
- mouthStretchRight
- mouthRollLower
- mouthRollUpper
- mouthShrugLower
- mouthShrugUpper
- mouthPressLeft
- mouthPressRight
- mouthLowerDownLeft
- mouthLowerDownRight
- mouthUpperUpLeft
- mouthUpperUpRight
- browDownLeft
- browDownRight
- browInnerUp
- browOuterUpLeft
- browOuterUpRight
- cheekPuff
- cheekSquintLeft
- cheekSquintRight
- noseSneerLeft
- noseSneerRight
- tongueOut
- gemfield

Blendshapes vs. Example-based Skinning





Blendshapes vs. Example-based Skinning

$$X = X_0 + \sum_i \beta_i B_i^{\mathrm{ID}} + \sum_j \theta_j \left(B_j^{\mathrm{Exp}} + +\delta(\{\beta_i, \theta_j\}) \right)$$

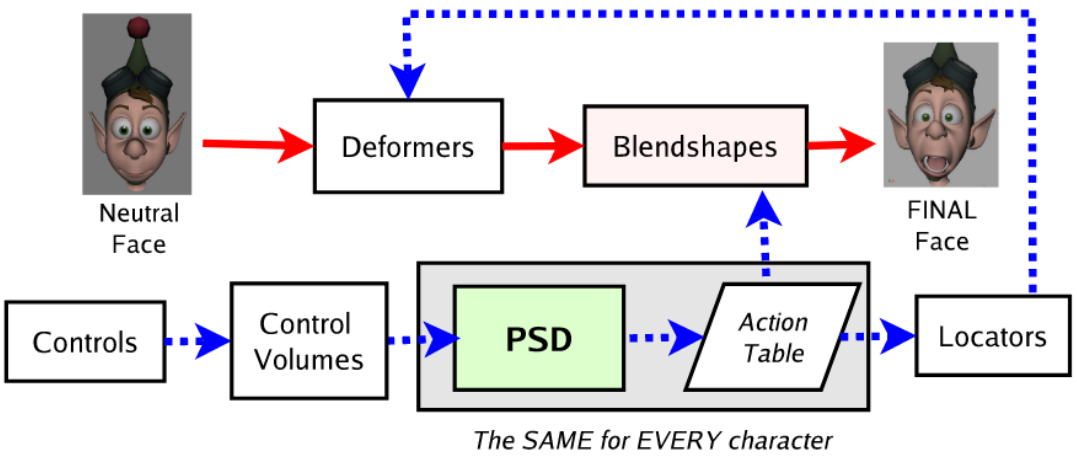
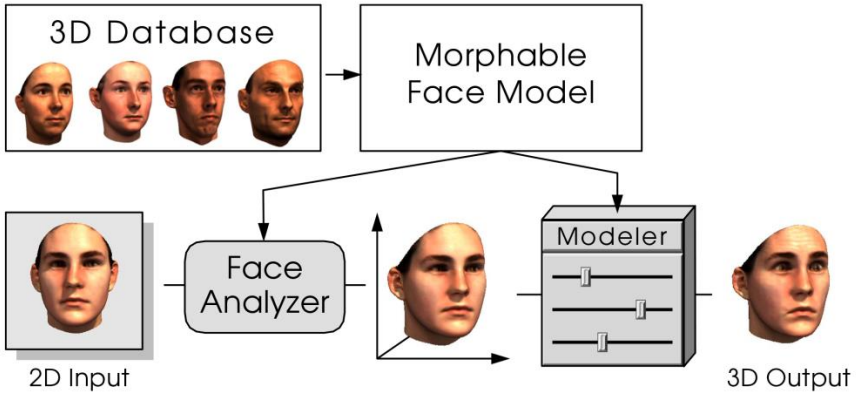


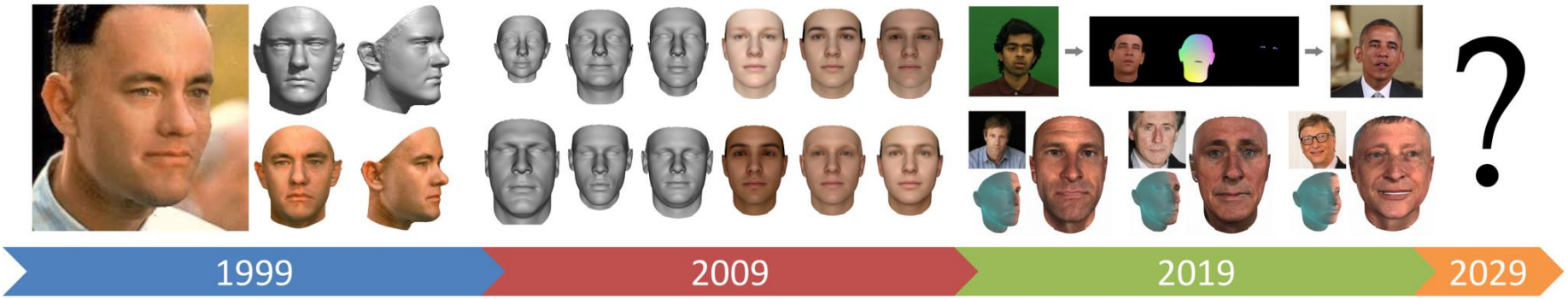
Figure 2: *Architecture of Hybrid Approach*

David Komorowski, Vinod Melapudi, Darren Mortillaro, and Gene S. Lee. 2010. ***A hybrid approach to facial rigging***. In *ACM SIGGRAPH ASIA 2010 Sketches* (SA '10), Association for Computing Machinery, New York, NY, USA, 1–2.

Morphable Face Models

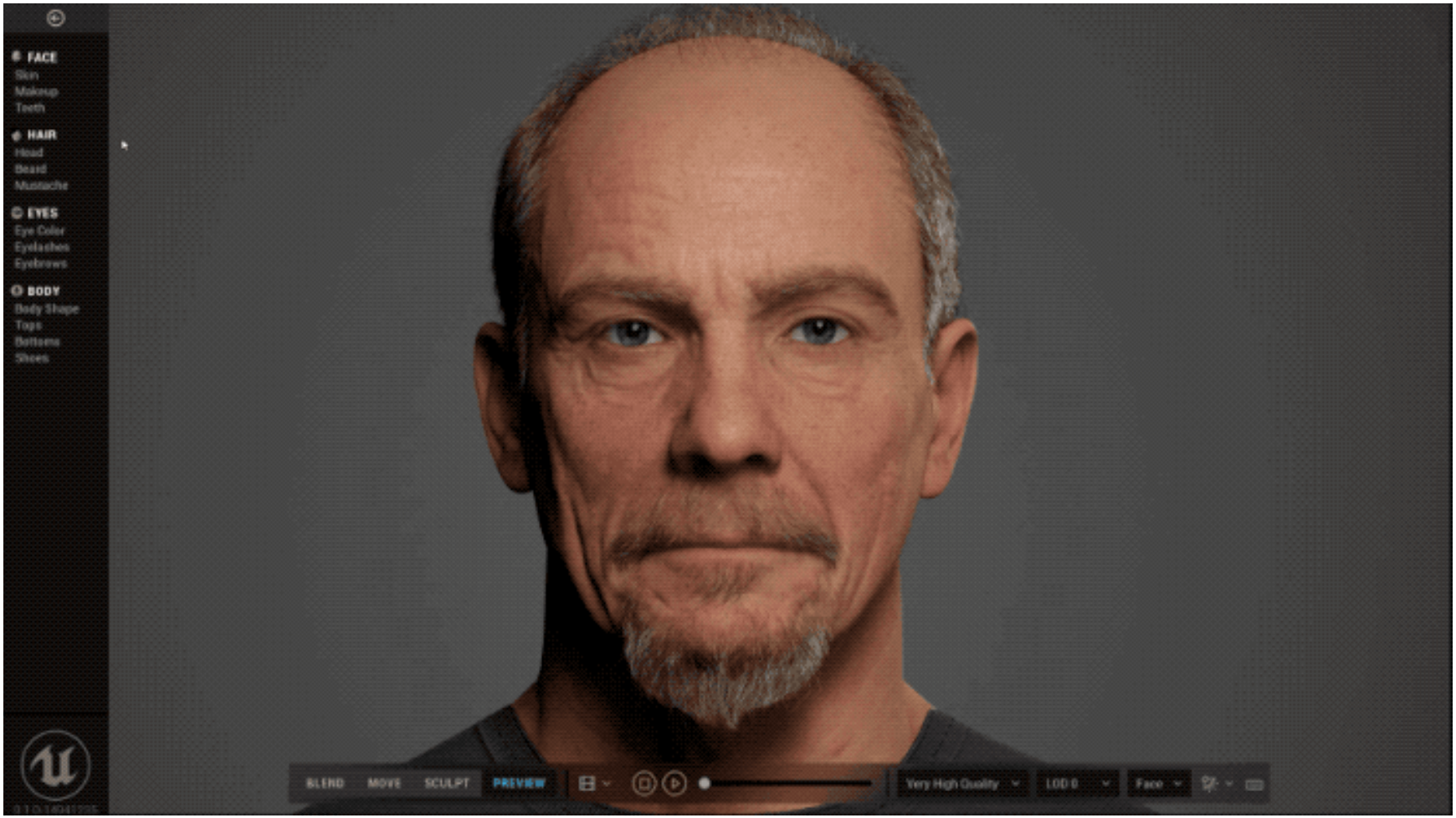
$$X = X_0 + \sum_i \beta_i B_i^{\mathrm{ID}} + \sum_j \theta_j B_j^{\mathrm{Exp}}$$





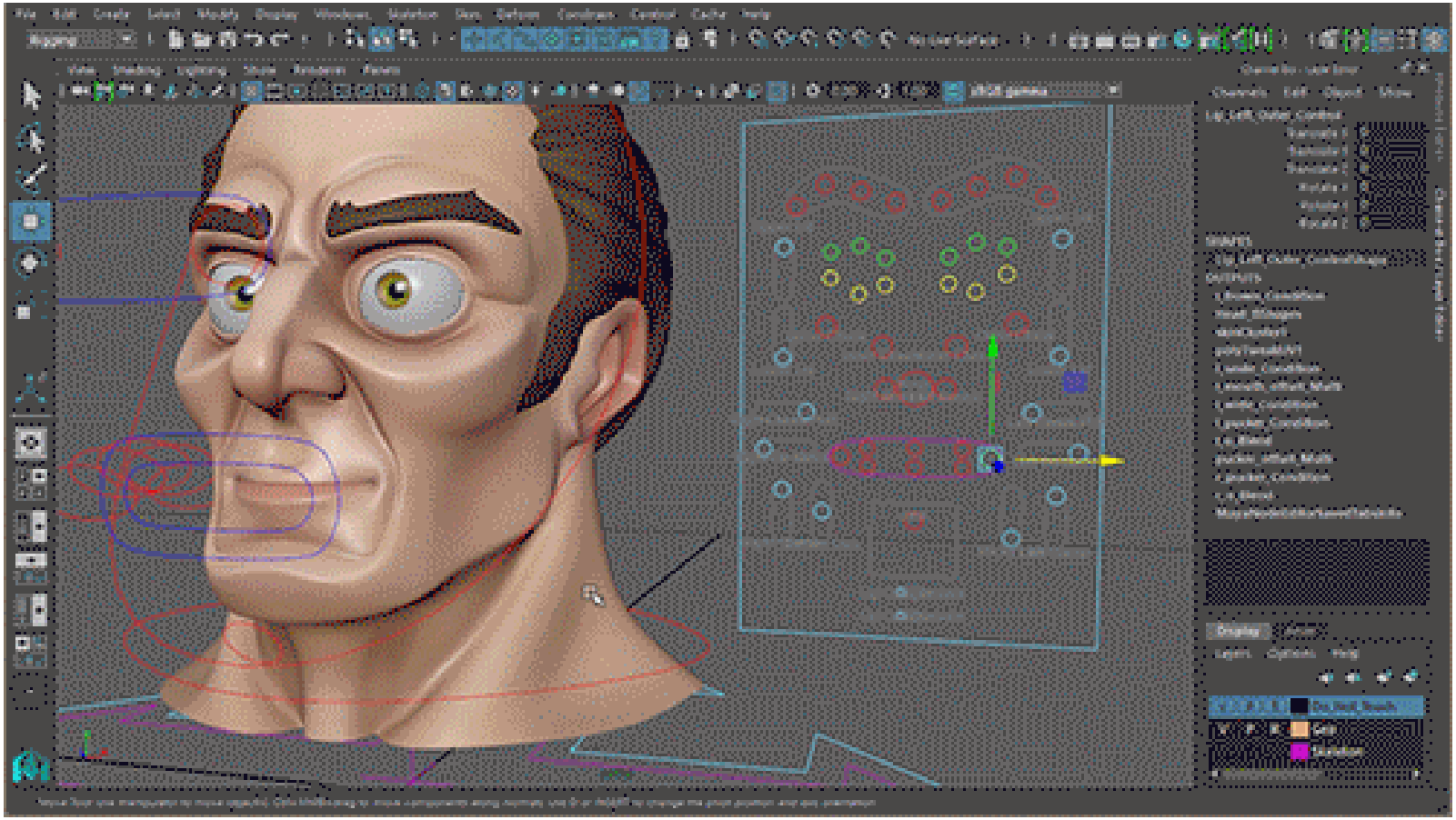
Egger et al. 2020. ***3D Morphable Face Models - Past, Present, and Future.*** *ACM Trans. Graph.* 39, 5 (June 2020), 157:1-157:38.

Morphable Face Models



Meta Human - UnrealEngine

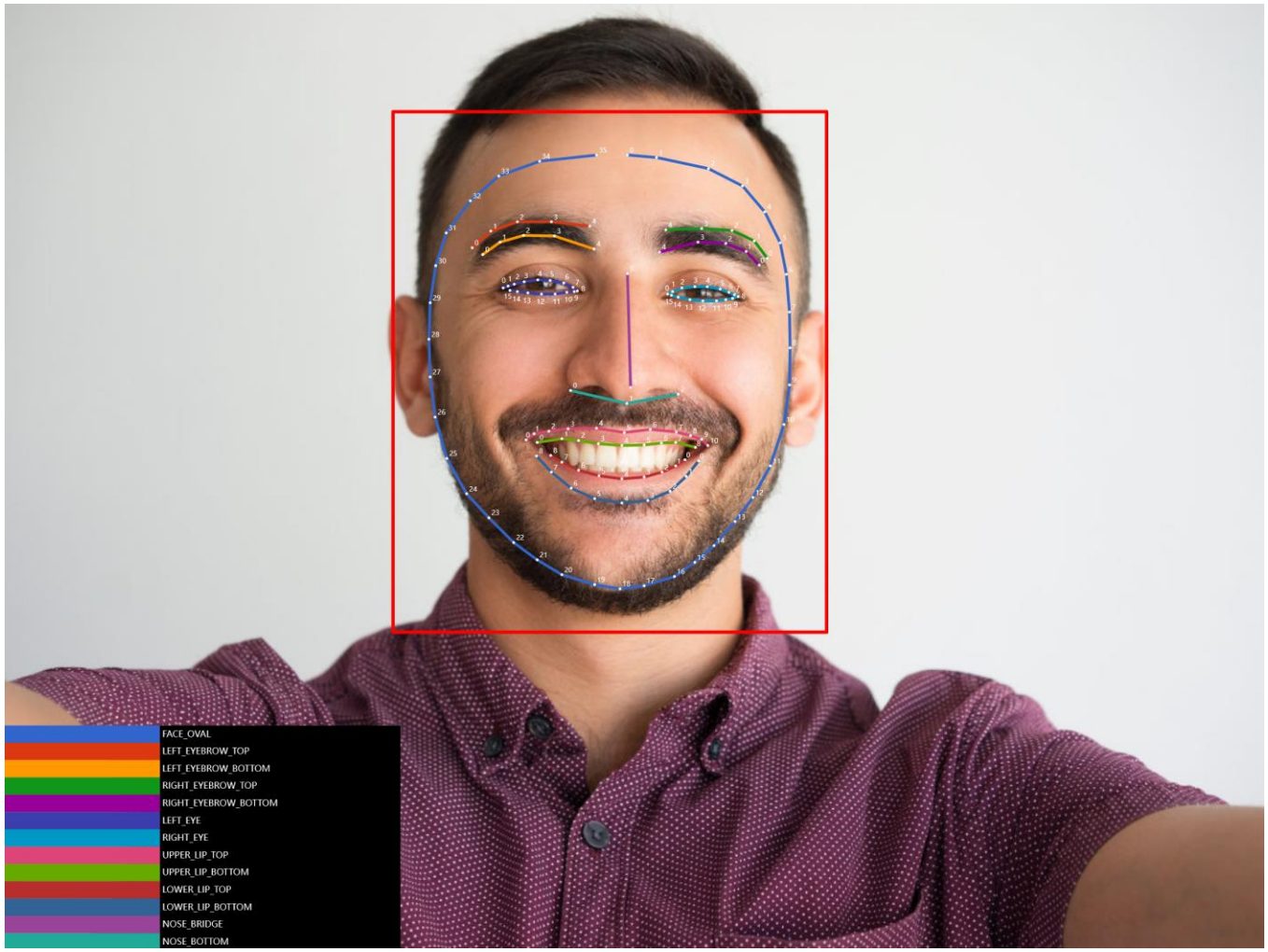
How to Animate a Face?



Face Tracking

https://developers.google.com/ml-kit/vision/face-detection

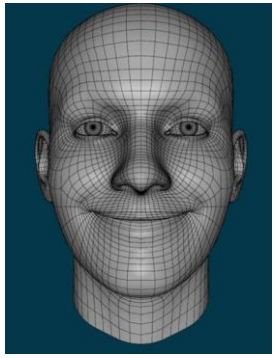
Face Tracking

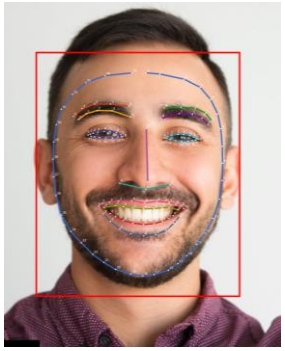


https://developers.google.com/ml-kit/vision/face-detection

Face Tracking

$$\min_{\beta_i,\theta_j} \left\| X_0 + \sum_i \beta_i B_i^{\mathrm{ID}} + \sum_j \theta_j B_j^{\mathrm{Exp}} - Y \right\| + E(\beta_i, \theta_j)$$





Speech-driven

Uncovered Topics

- Skinning weight reduction and compression
- Skeleton extraction and auto-rigging

本节内容

- 几何形状的编辑
  - 自由形变 (Free-form Deformation)
  - 基于优化的形变
- 蒙皮形变 (Skinning Deformation)
  - 线性蒙皮
    - Linear-Blend Skinning (LBS)
    - Multi-linear Skinning
  - 非线性蒙皮
    - Dual-quaternion Skinning (DQS)
  - 基于样例的蒙皮
    - Blendshapes
  - 人脸动画

有没有问题?

Any Questions