

# IFT 6113

## MESH DEFORMATION AND SKINNING

[tiny.cc/ift6113](http://tiny.cc/ift6113)

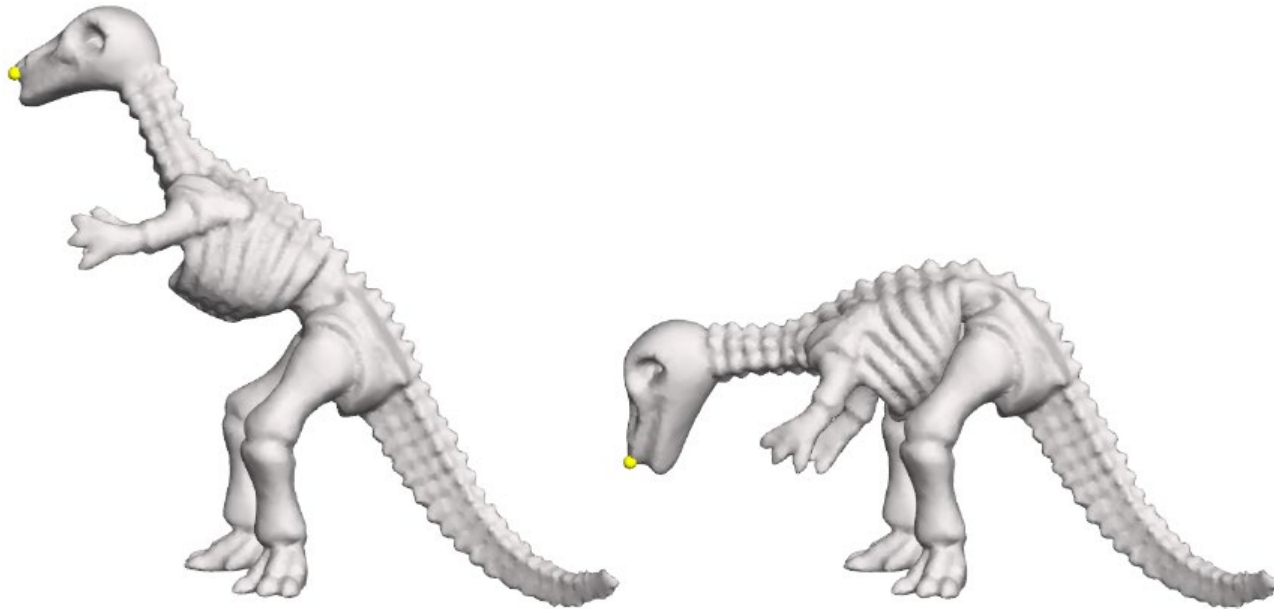


Image from 'As-Rigid-As-Possible Shape Modeling' by Sorkine & Alexa, 2007

# Mikhail Bessmeltsev

# What for?

- Animation!
- Mesh editing
- Image warping (2D)



This, and many other images in this presentation are from 'Polygon Mesh Processing' textbook by Botsch et al. or their website

Warning:

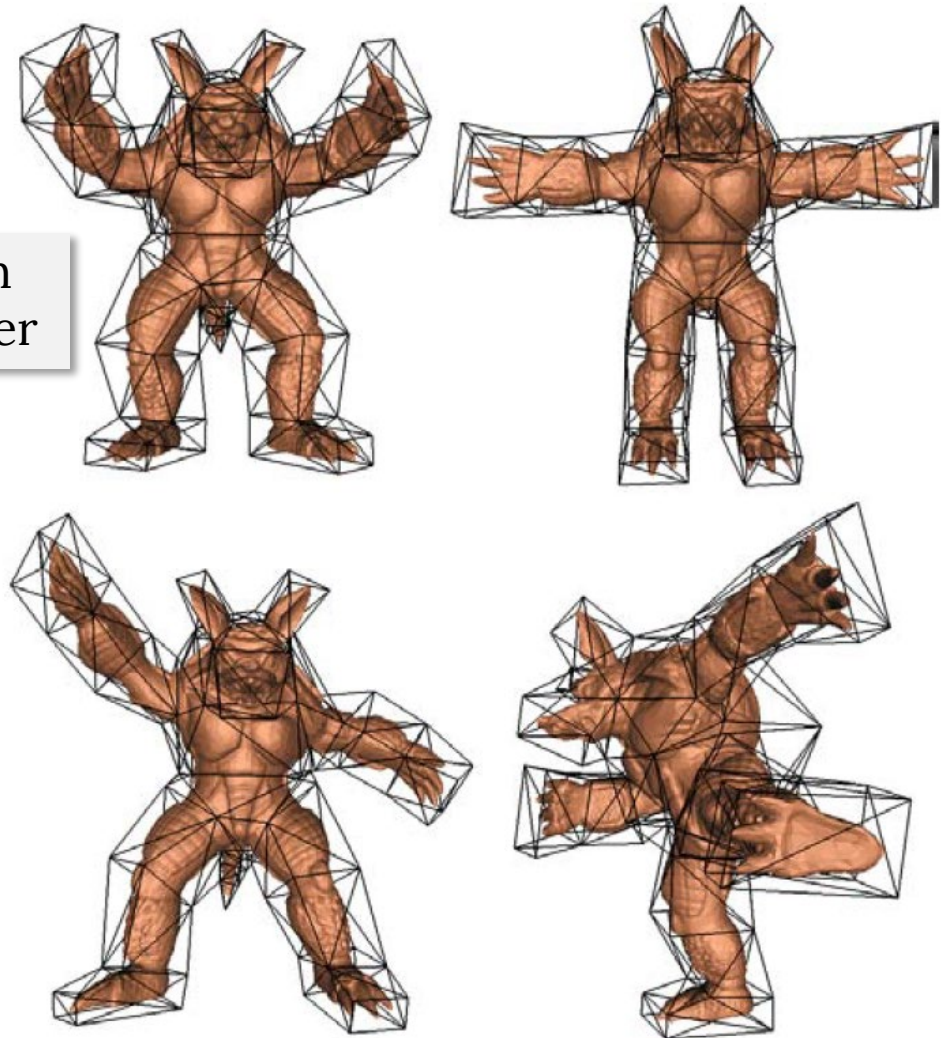
**TMI**

This topic is immense  
We'll only see a few samples

# Deformation: user interface

- Handles
- Cages
- Skeletons
- ...

More on  
those later





# Deformation models

Direct

$$v' = \left( \sum w_j T_j \right) v$$

- Linear Blend Skinning
- Dual Quaternion Skinning
- ...

Variational

$$v' = \operatorname{argmin}_x E(x)$$

- Multiresolution editing
- As-Rigid-As-Possible
- Laplacian Mesh Editing
- ...

# Deformation models

Direct

$$v' = \left( \sum w_j T_j \right) v$$

- Linear Blend Skinning
- Dual Quaternion Skinning
- ...

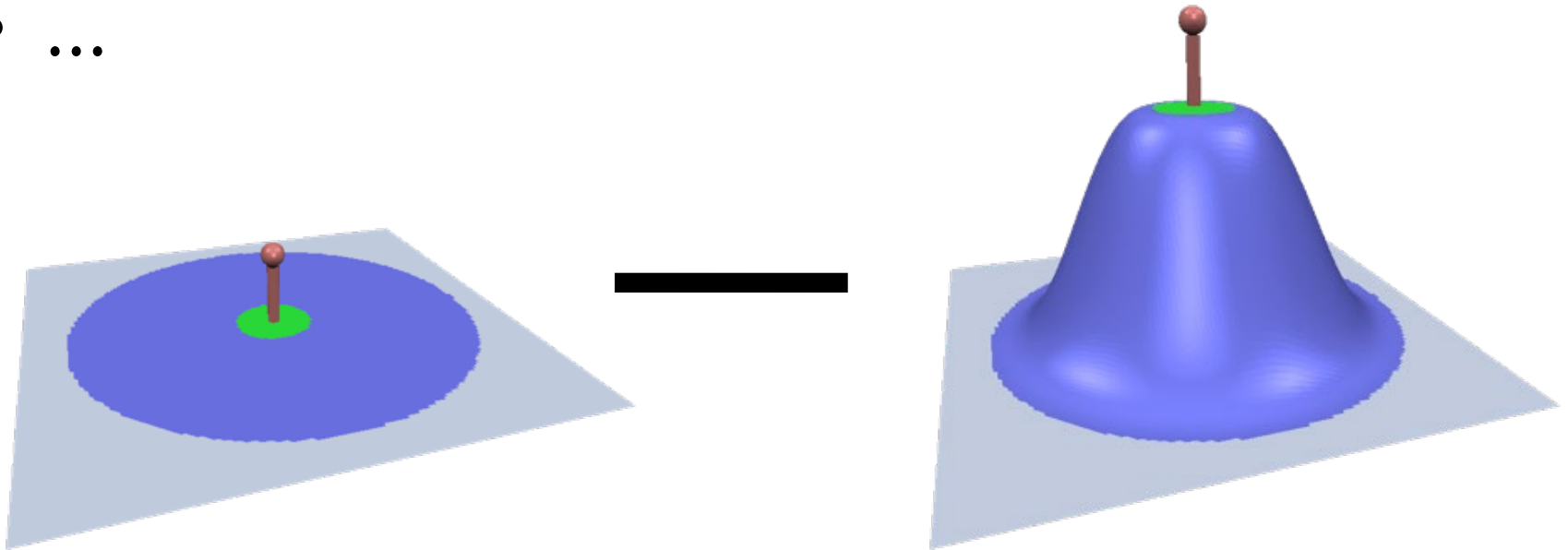
Variational

$$v' = \operatorname{argmin}_x E(x)$$

- Multiresolution editing
- As-Rigid-As-Possible
- Laplacian Mesh Editing
- ...

# Deformation: user interface

- **Handles**
- Cages
- Skeletons
- ...



# Modeling

Paint three surface areas:

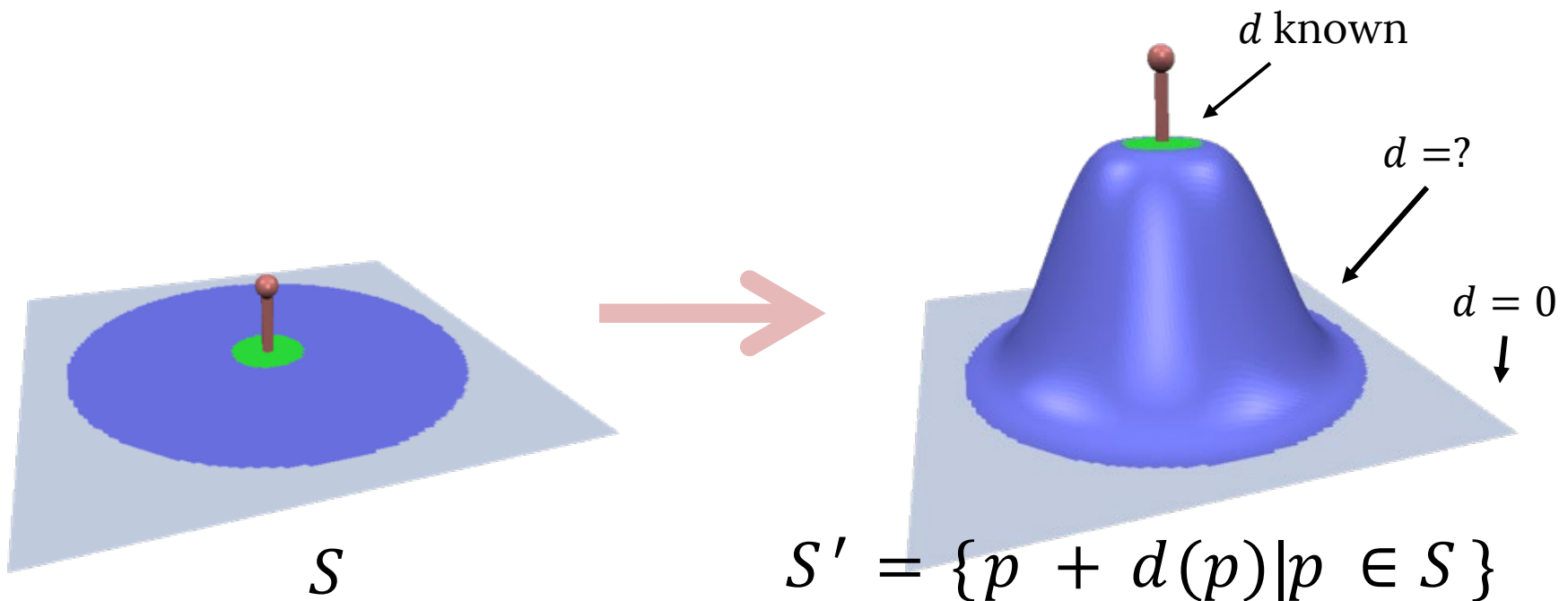
- Constrained
- Smooth falloff
- Fixed



# Formulation

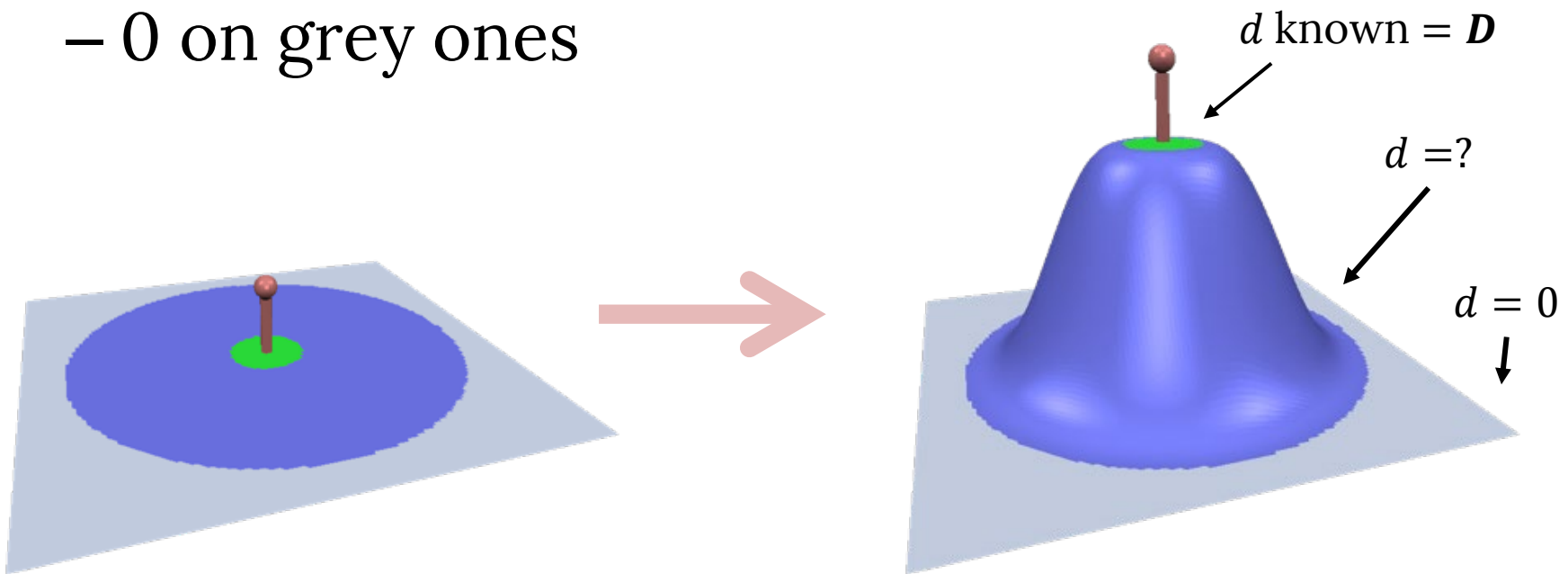
Find displacement vector field  $d$

- Smooth
- Satisfies constraints



# Simplest idea

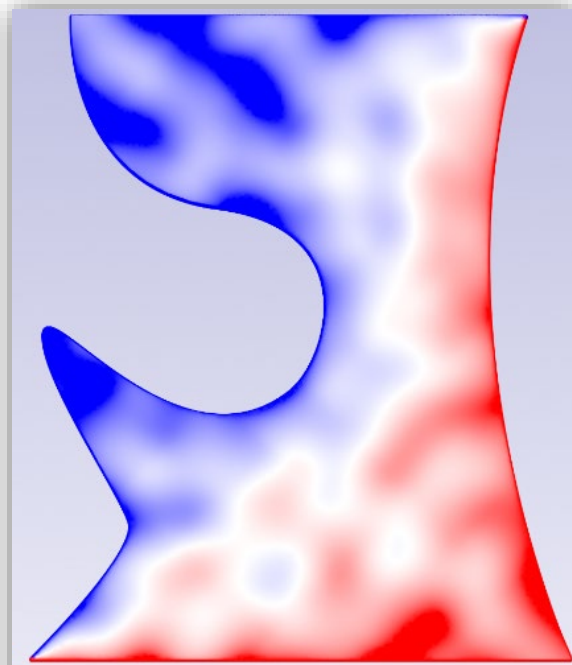
- $d = s(p) \cdot D$
- $s(p)$  is a smooth function:
  - 1 on green vertices
  - 0 on grey ones



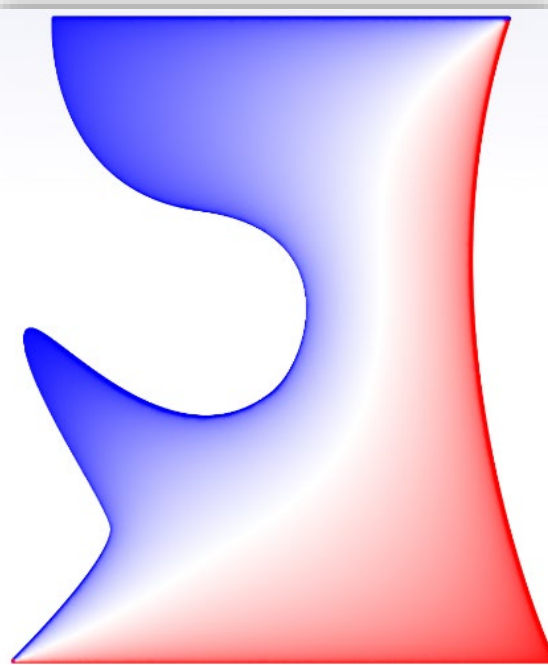
Long time ago:

# Dirichlet Energy

$$E[f] := \int_{\Omega} \langle \nabla f, \nabla f \rangle dA$$



non-smooth  $f(x)$



solution  $\Delta f = 0$

On board:

$$\begin{aligned} \min_f E[f] \\ \text{s.t. } f|_{\partial\Omega} = g \end{aligned}$$

$$\Delta f \equiv 0$$

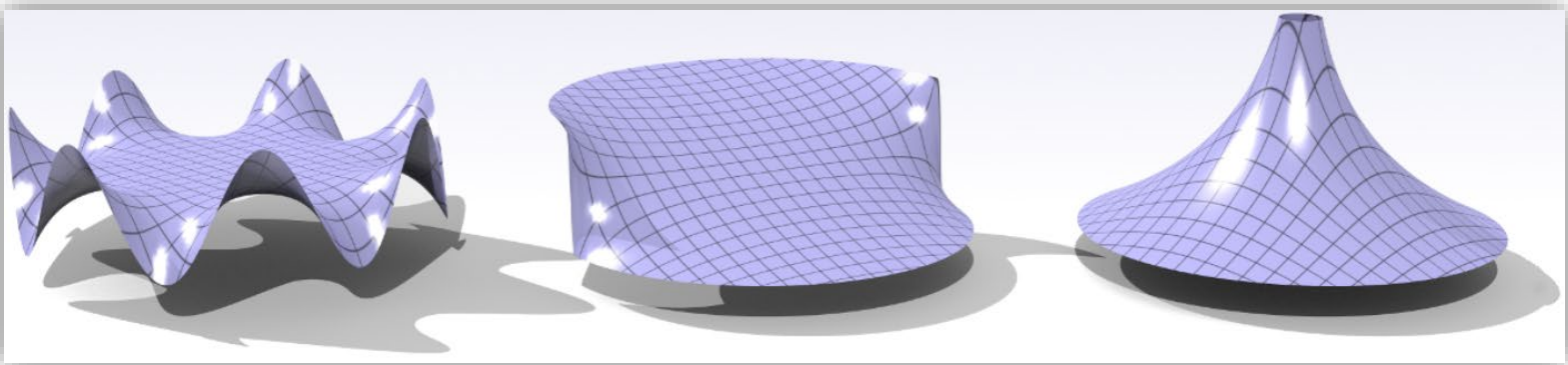
“Laplace equation”  
“Harmonic function”



Long time ago:

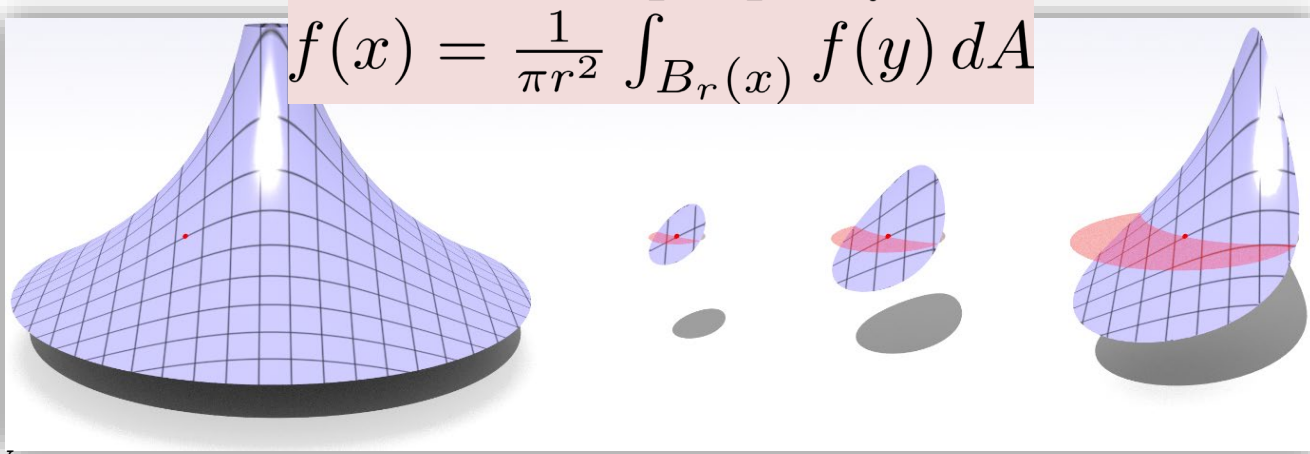
# Harmonic Functions

$$\Delta f \equiv 0$$



Mean value property:

$$f(x) = \frac{1}{\pi r^2} \int_{B_r(x)} f(y) dA$$



# How to find $s(p)$ ?

- Something inversely proportional to geodesic distance
- Or our favorite:

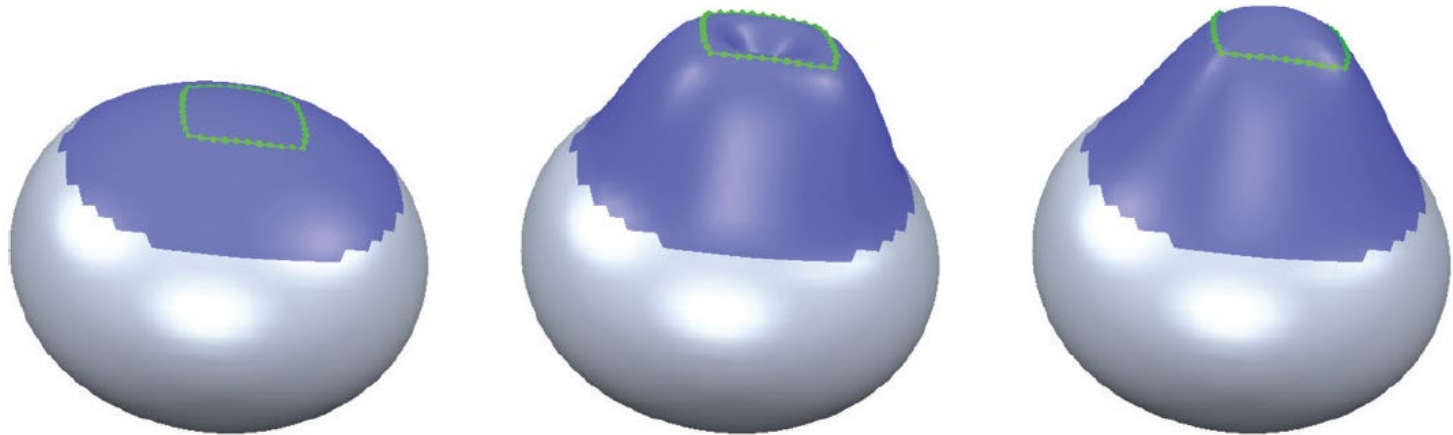
$$\Delta s(\mathbf{p}_i) = 0, \quad \mathbf{p}_i \in \mathcal{R},$$

$$s(\mathbf{p}_i) = 1, \quad \mathbf{p}_i \in \mathcal{H},$$

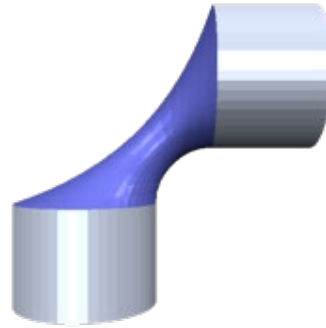
$$s(\mathbf{p}_i) = 0, \quad \mathbf{p}_i \in \mathcal{F}.$$

# Maximum principle

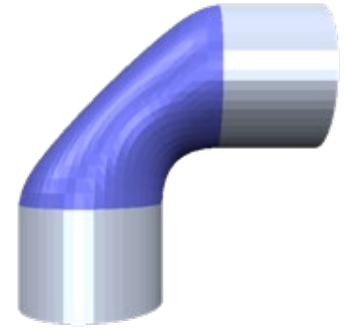
Reality vs Expectation



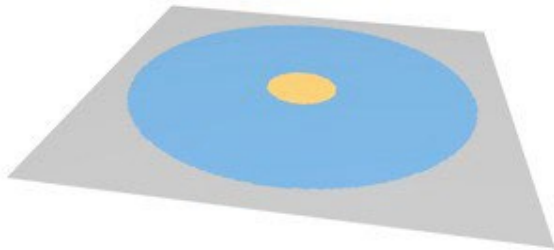
# Deformation Energies



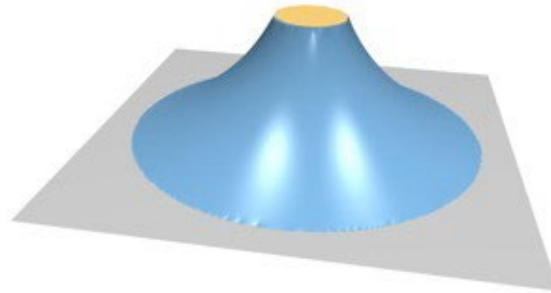
$$\Delta p = 0$$



$$\Delta^2 p = 0$$

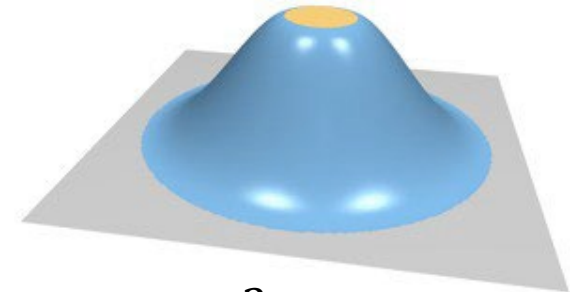


Initial state



$$\Delta d = 0$$

(Membrane)

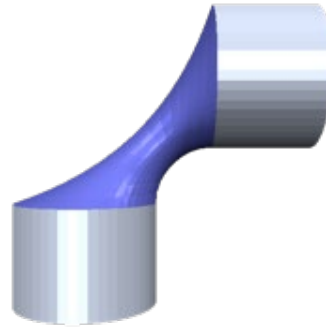


$$\Delta^2 d = 0$$

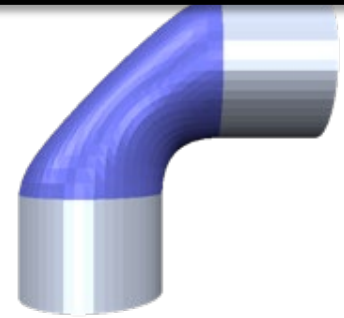
(Bilaplacian)

# Deformation Energy

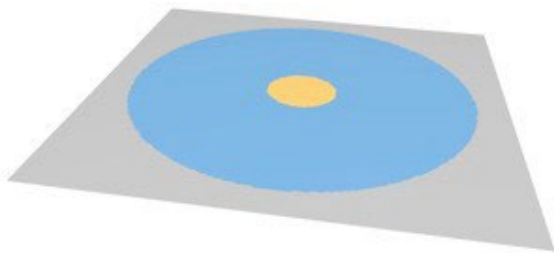
Higher order =>  
more boundary  
conditions



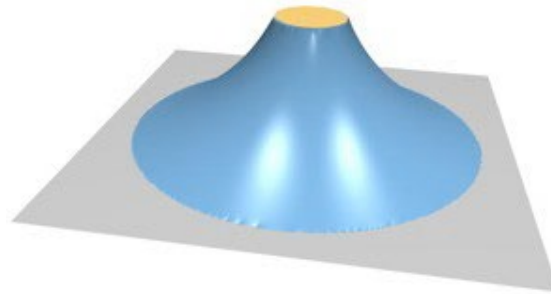
$$\Delta p = 0$$



$$\Delta^2 p = 0$$

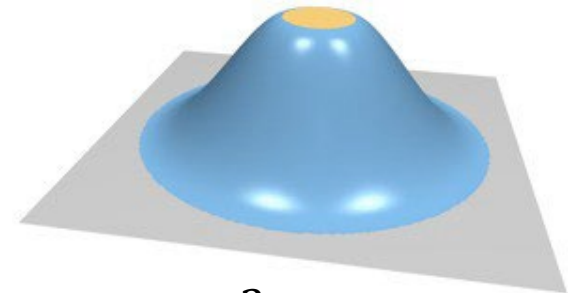


Initial state



$$\Delta d = 0$$

(Membrane)



$$\Delta^2 d = 0$$

(Bilaplacian)

# Physically-Based

Find a deformation that preserves both  
fundamental forms

Express the fundamental forms of  $S'$  via vector field  $d$

$$\int_{\Omega} k_s \underbrace{\|\mathbf{I} - \mathbf{I}'\|_F^2}_{\text{stretching}} + k_b \underbrace{\|\mathbf{II} - \mathbf{II}'\|_F^2}_{\text{bending}} du dv$$

**Expensive  
to optimize!**

# Shell-Based Deformation

Find a deformation that preserves both  
fundamental forms

**Linearize**—Express the fundamental forms of  $S'$  via vector field  $d$

$$\int_{\Omega} k_s \left( \|\mathbf{d}_u\|^2 + \|\mathbf{d}_v\|^2 \right) + k_b \left( \|\mathbf{d}_{uu}\|^2 + 2 \|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2 \right) \mathrm{d}u \mathrm{d}v$$



# Physically-Based

$$\int_{\Omega} k_s \underbrace{\left( \|\mathbf{d}_u\|^2 + \|\mathbf{d}_v\|^2 \right)}_{\text{stretching}} + k_b \underbrace{\left( \|\mathbf{d}_{uu}\|^2 + 2 \|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2 \right)}_{\text{bending}} du dv$$

Gateau derivative =>

$$-k_s \Delta d + k_b \Delta^2 d = 0$$

# Physically-Based

$$\int_{\Omega} k_s \left( \|\mathbf{d}_u\|^2 + \|\mathbf{d}_v\|^2 \right) + k_b \left( \|\mathbf{d}_{uu}\|^2 + 2 \|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2 \right) dudv$$

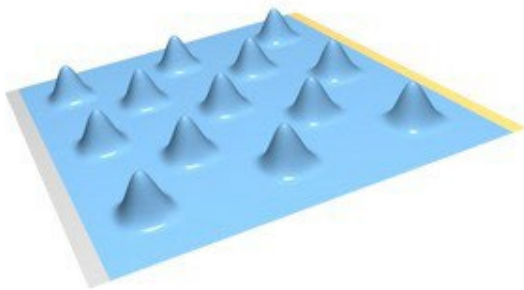
## Gateau derivative =>

$$-k_s \Delta d + k_b \Delta^2 d = 0$$

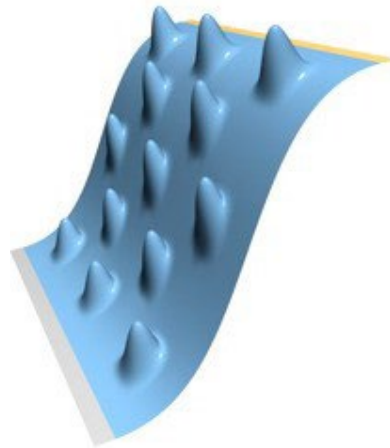
$\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ 
 Bi-Laplacian

# Solved?

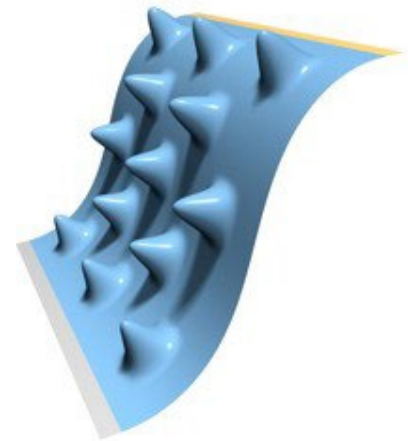
- Very fast
  - One linear solve!
- Physically-based
- Linearization => **lose details**



Original



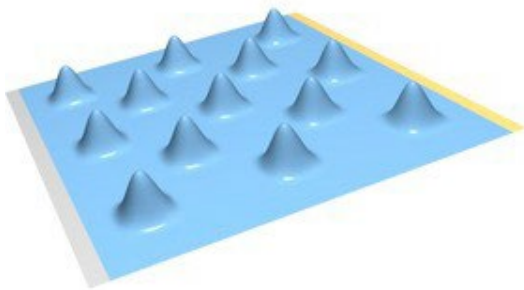
Linear  
deformation



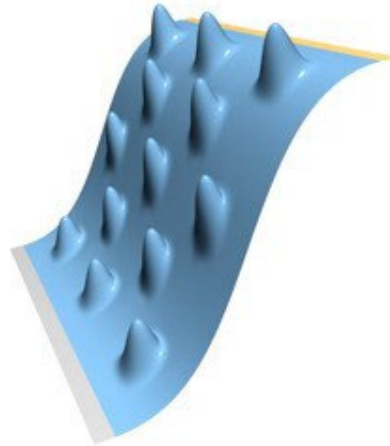
Non-linear  
deformation

# Issue

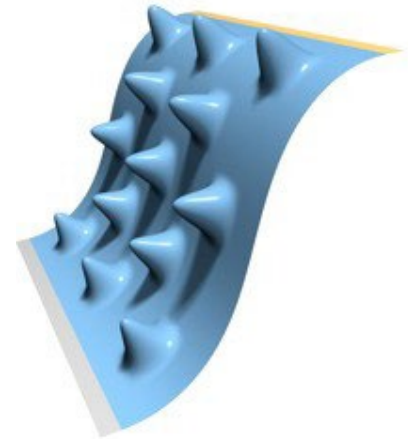
- We need to rotate details
  - Local rotation is nonlinear!
- Can we still survive with linear solves?



Original

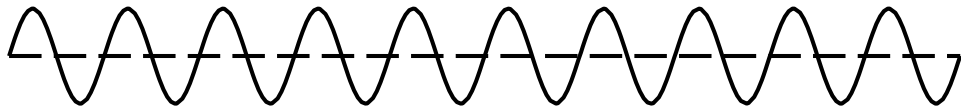


Linear  
deformation



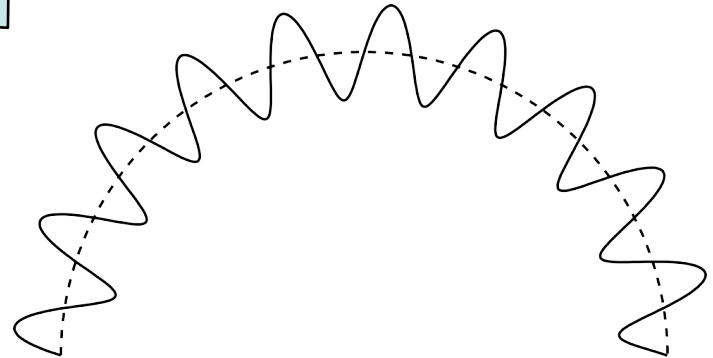
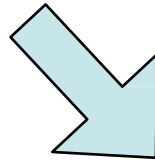
Non-linear  
deformation

# Multiresolution Editing



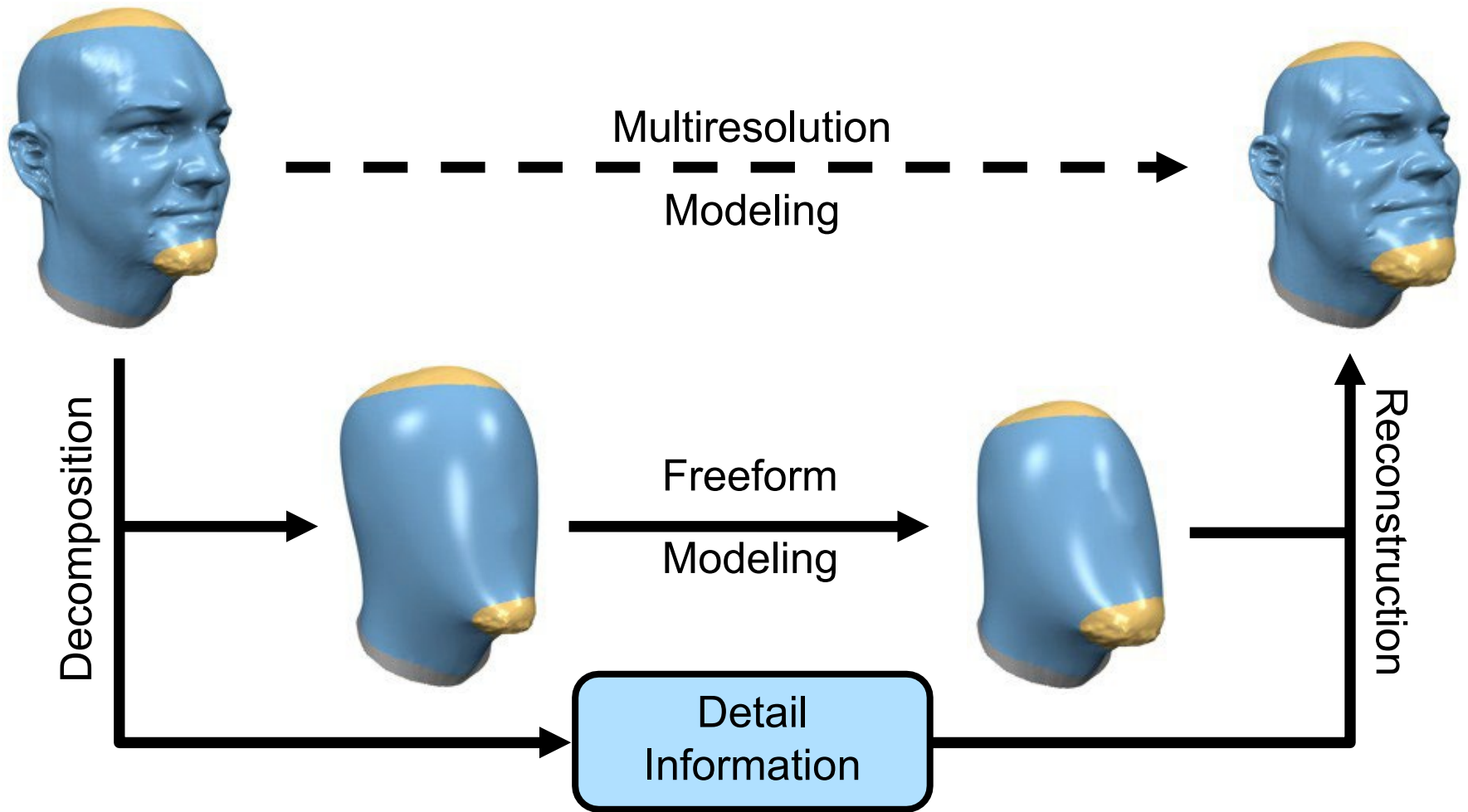
Frequency decomposition

Change low  
frequencies



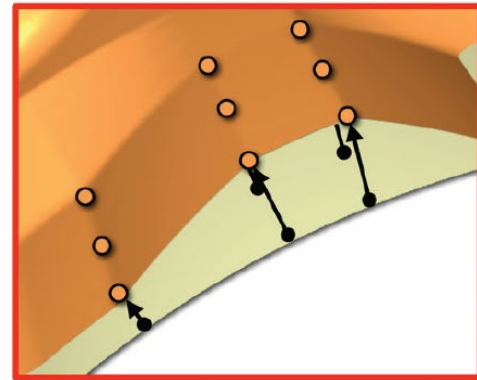
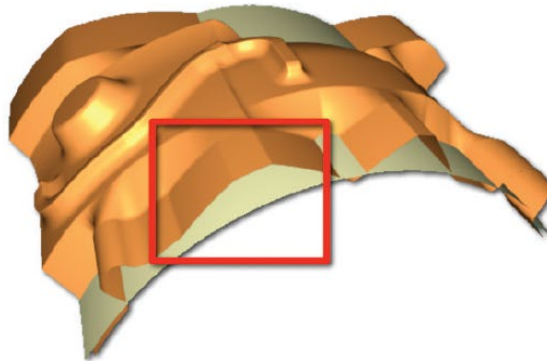
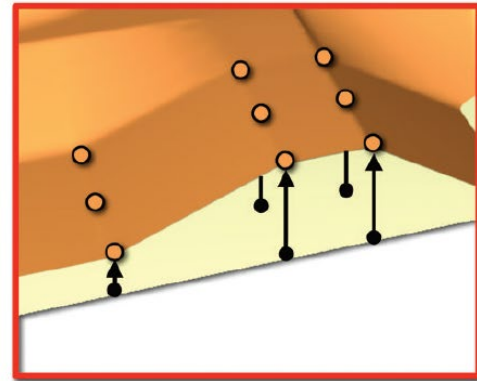
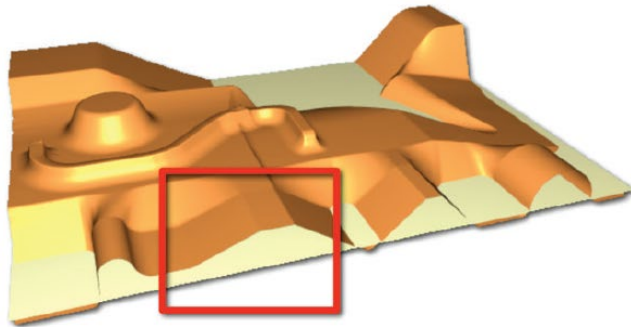
Add high frequency details,  
stored in local frames

# Multiresolution Editing



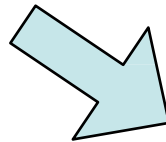
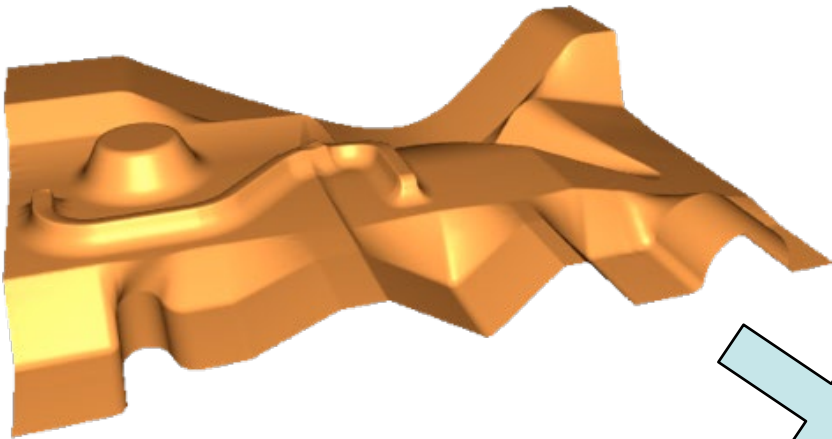
# How to represent details?

- For example, normal displacements

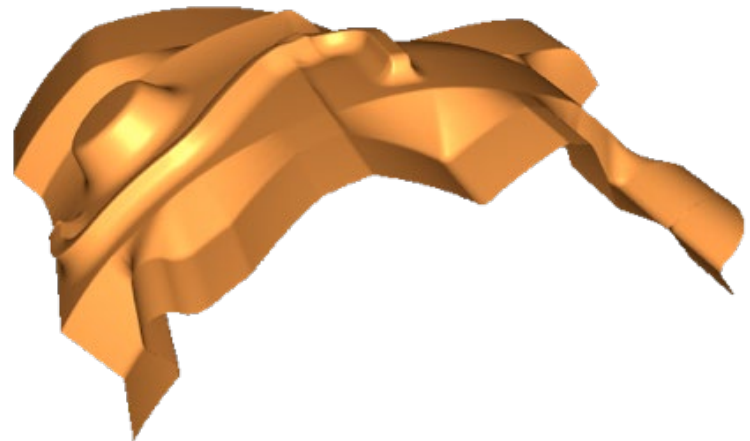




# Result



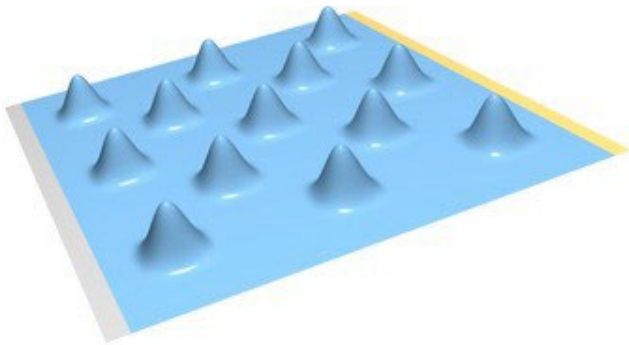
Global deformation  
with intuitive detail  
preservation



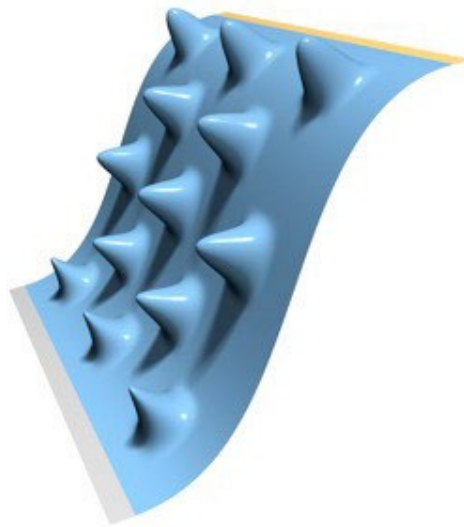
# Limitations

Neighboring displacements are not coupled

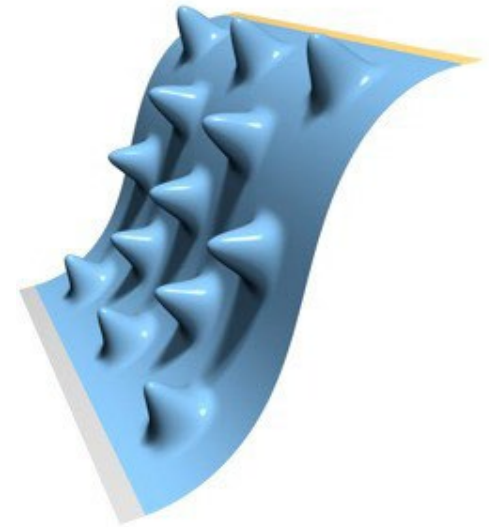
- Surface bending changes their angle
- Leads to volume changes or self-intersections



Original



Normal Displ.

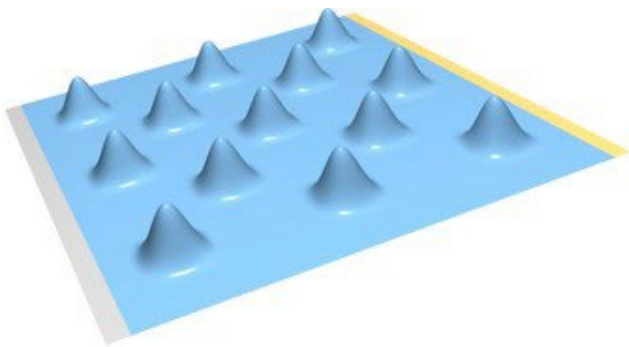


Nonlinear

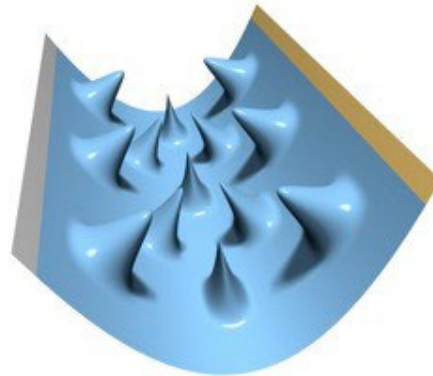
# Limitations

Neighboring displacements are not coupled

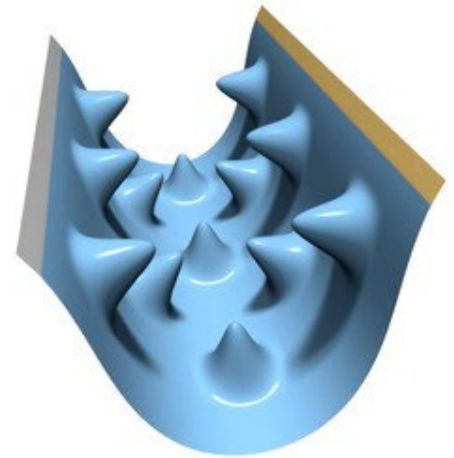
- Surface bending changes their angle
- Leads to volume changes or self-intersections



Original



Normal Displ.



Nonlinear

# New coordinates?

Express shape in *differential coordinates*

Transform those,  
then reconstruct the new shape

Long time ago:

# Mean Value Property

$$L_{vw} = A - D = \begin{cases} 1 & \text{if } v \sim w \\ -\text{degree}(v) & \text{if } v = w \\ 0 & \text{otherwise} \end{cases}$$

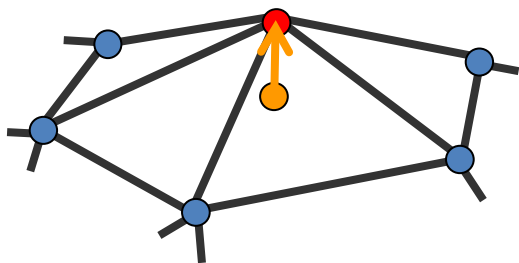
$$(Lx)_v = 0$$



Value at  $v$  is average of neighboring values


# Laplacian Mesh Editing

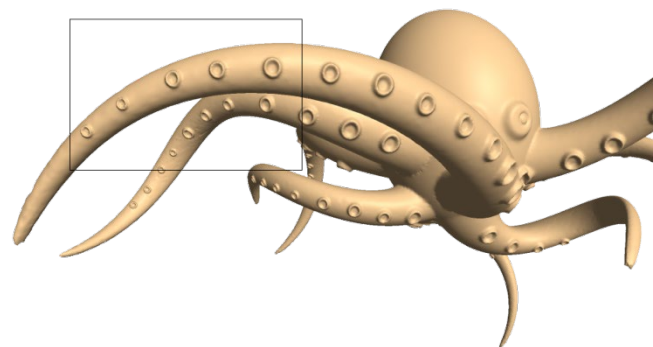
Graph Laplacian:



$$\delta_i = \mathbf{v}_i - \frac{1}{d_i} \sum_{j \in N(i)} \mathbf{v}_j$$

Local  
coordinates!


$$\delta = L\mathbf{v}$$



# Laplacian Mesh Editing

- Represent mesh using only  $\delta$
- Find a surface whose Laplacian coordinates are as close as possible to  $\delta$

$$\int_{\mathcal{S}} \|\Delta \mathbf{p}' - \delta'\|^2 d\mathcal{S} \rightarrow \min$$

$$\text{s.t. } p'_i = p_i, i \in \{\text{point constraints}\}$$



# Laplacian Mesh Editing

Find a surface whose Laplacian coordinates are as close as possible to  $\delta$

$$\min \sum \|\delta_i - L(p'_i)\|^2 + \sum_{i \in c} \|p'_i - p_i\|^2$$

# Laplacian Mesh Editing

Find a surface whose Laplacian coordinates are as close as possible to  $\delta$

$$\int_{\mathcal{S}} \|\Delta \mathbf{p}' - \delta'\|^2 d\mathcal{S} \rightarrow \min$$

$$\text{s.t. } p'_i = p_i, i \in \{\text{point constraints}\}$$

$$\text{Gateau derivative} \Rightarrow \Delta^2 \mathbf{p}' = \Delta \delta'$$

Before:

# Physically-Based

$$\int_{\Omega} k_s \left( \|\mathbf{d}_u\|^2 + \|\mathbf{d}_v\|^2 \right) + k_b \left( \|\mathbf{d}_{uu}\|^2 + 2 \|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2 \right) \mathrm{d}u \mathrm{d}v$$

# Gateua derivative =>

$$-k_s \Delta d + k_b \Delta^2 d = 0$$

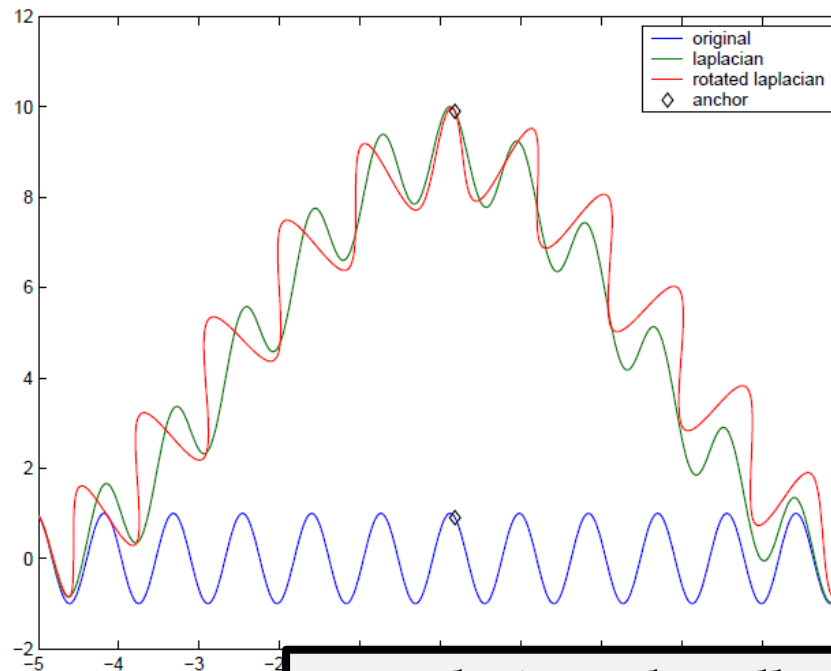
(almost) the same equation?

# Bi-Laplacian

# Issue

Reconstructing from differential coordinates  
makes sense only if they are  
**rotation and translation invariant**

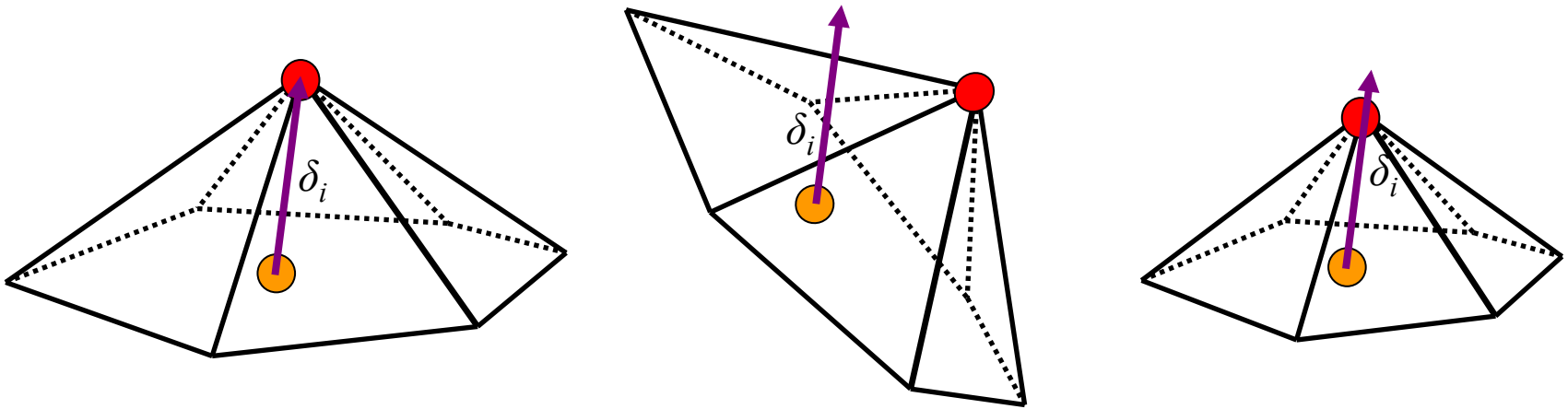
Otherwise, you get this



Translating a handle induces local rotations!

# Laplacian Coordinates

- Translation invariant
- Not rotation/scale invariant



$$\delta_i = L(\mathbf{v}_i) = L(\mathbf{v}_i + \mathbf{t}); \forall \mathbf{t} \in \mathbb{R}^3$$

# Solutions

1. Transform, ignoring rotations or details
2. **while** (not converged)
  - Estimate rotations (*from normals*)
  - Rotate differential coordinates and solve

$$E(\mathbf{V}') = \sum_{i=1}^n \|\mathbf{R}_i \delta_i - L(p'_i)\|^2 + \sum_{i \in c} \|p'_i - p_i\|^2$$

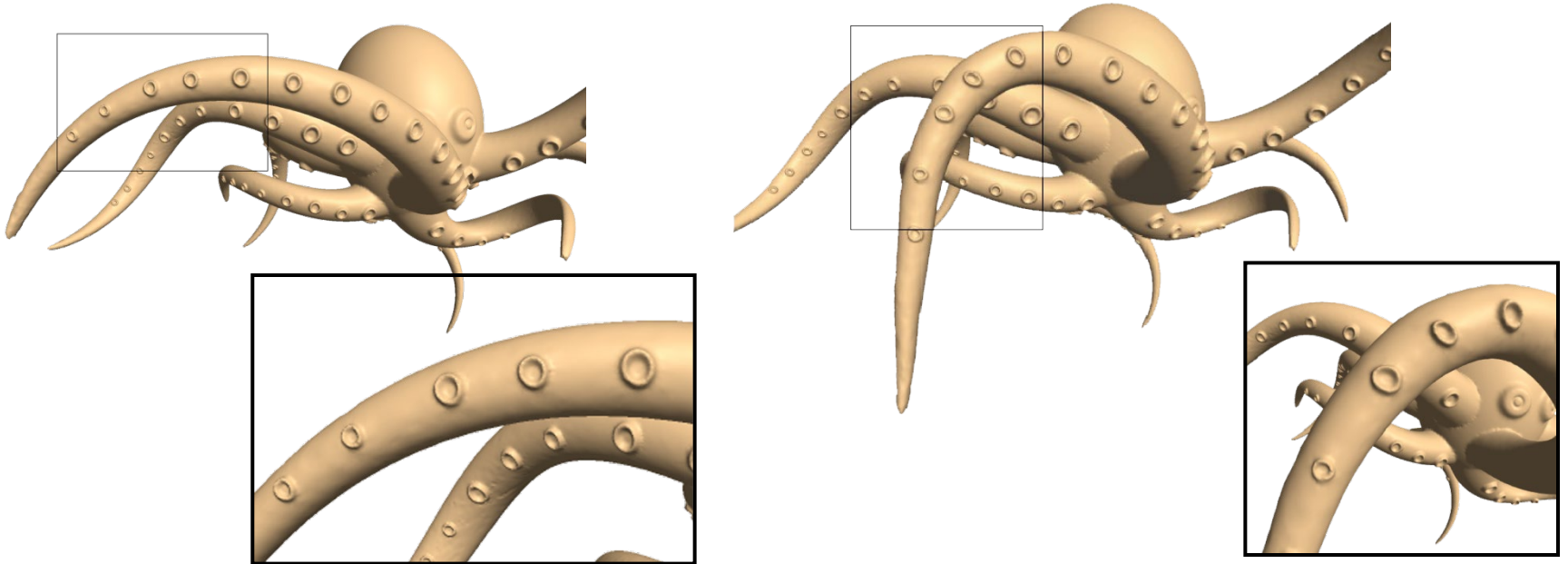
[Y. Lipman, O. Sorkine, D. Cohen-Or, D. Levin, C. Rossi and H. P. Seidel, "Differential coordinates for interactive mesh editing," *Proceedings Shape Modeling Applications*, 2004]

# Solutions

1. Transform, ignoring rotations or details

2. **while** (not converged)

- Estimate rotations (*from normals*)
- Rotate differential coordinates and solve



[Y. Lipman, O. Sorkine, D. Cohen-Or, D. Levin, C. Rossi and H. P. Seidel, "Differential coordinates for interactive mesh editing," *Proceedings Shape Modeling Applications*, 2004]

# Rotations + **scaling** – invariant?

Add local transformations  $T_i$  as variables

$$E(\mathbf{V}') = \sum_{i=1}^n \|T_i \delta_i - L(p'_i)\|^2 + \sum_{i \in c} \|p'_i - p_i\|^2$$

[O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Ross, H.-P. Seidel, Laplacian Surface Editing, EUROGRAPHICS/Symposium on Geometry Processing, 2004]



# Rotations + **scaling** – invariant?

Add local transformations  $T_i$  as variables

$$E(\mathbf{V}') = \sum_{i=1}^n \|\mathbf{T}_i \delta_i - L(p'_i)\|^2 + \sum_{i \in \mathcal{C}} \|p'_i - p_i\|^2$$

$$\min_{T_i} \left( \|T_i \mathbf{v}_i - \mathbf{v}'_i\|^2 + \sum_{j \in \mathcal{N}_i} \|T_i \mathbf{v}_j - \mathbf{v}'_j\|^2 \right).$$

[O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Ross, H.-P. Seidel, Laplacian Surface Editing, EUROGRAPHICS/Symposium on Geometry Processing, 2004]

# Rotations + **scaling** – invariant?

Add local transformations  $T_i$  as variables

$$\min_{T_i} \left( \|T_i \mathbf{v}_i - \mathbf{v}'_i\|^2 + \sum_{j \in \mathcal{N}_i} \|T_i \mathbf{v}_j - \mathbf{v}'_j\|^2 \right).$$

$T_i$  = translation + rotation + scaling

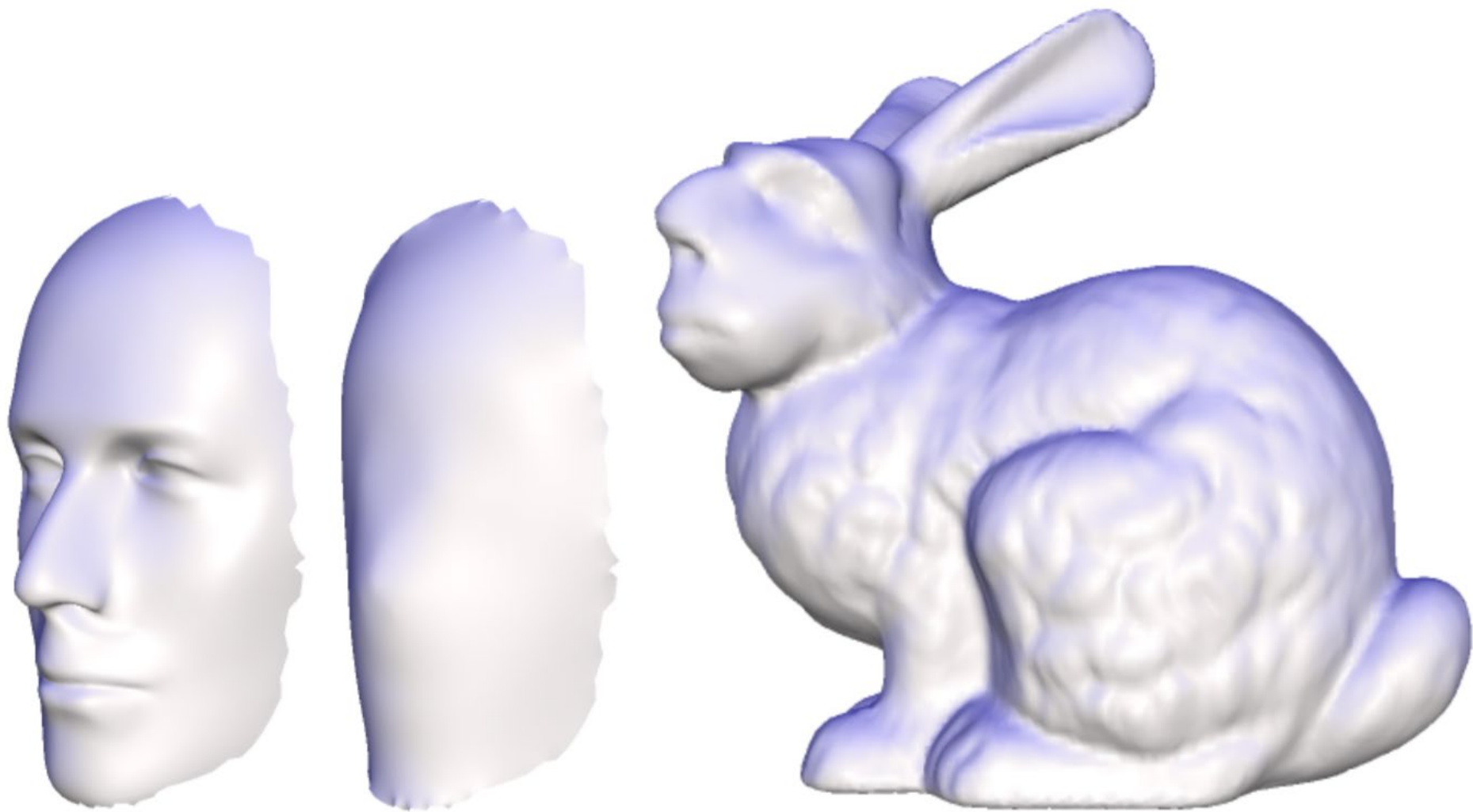
Represent (a linearization of)  $T_i$  using translation/rotation/scaling parameters

# Rotations + **scaling** – invariant?

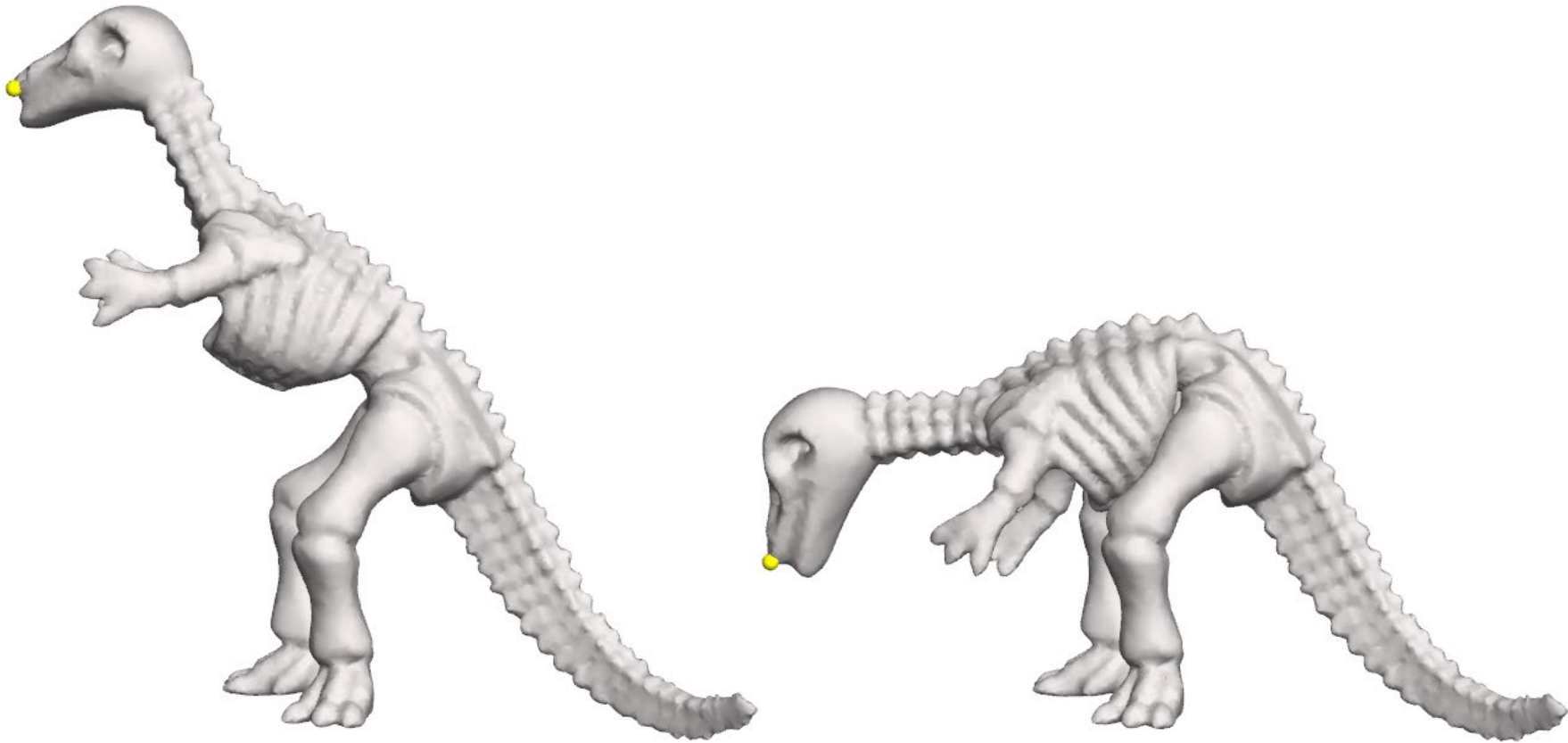
Add local transformations  $T_i$  as variables

$$E(\mathbf{V}') = \sum_{i=1}^n \|\mathbf{T}_i \delta_i - L(p'_i)\|^2 + \sum_{i \in \mathcal{C}} \|p'_i - p_i\|^2$$

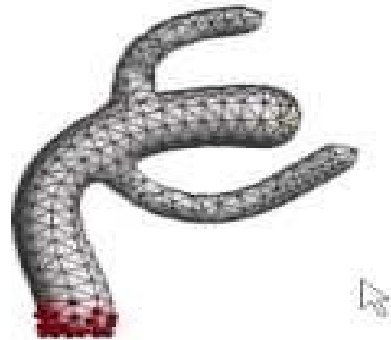
$\Rightarrow T_i$  is a linear function of  $V'$   
 $\Rightarrow$  Quadratic optimization  
 $\Rightarrow$  Linear solve!



# As-Rigid-As-Possible Surface Modelling

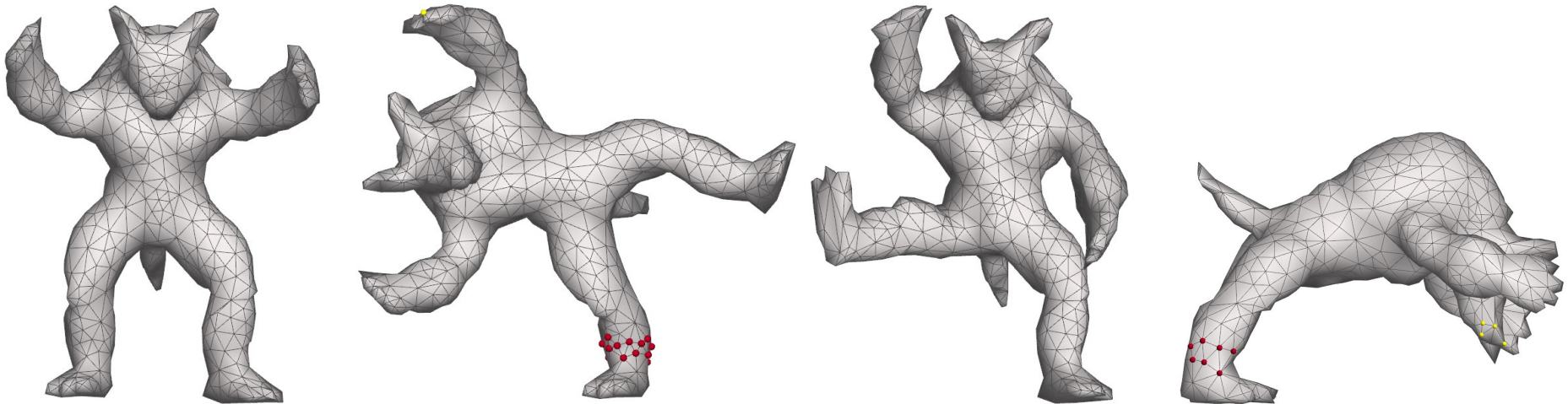


# As-rigid-as-possible (ARAP)



# As-rigid-as-possible (ARAP)

- “Intuitive” deformations
  - Smooth deformations at large scale
- Preserve local features
- Fast, for interactive mesh editing



# ARAP in a nutshell...

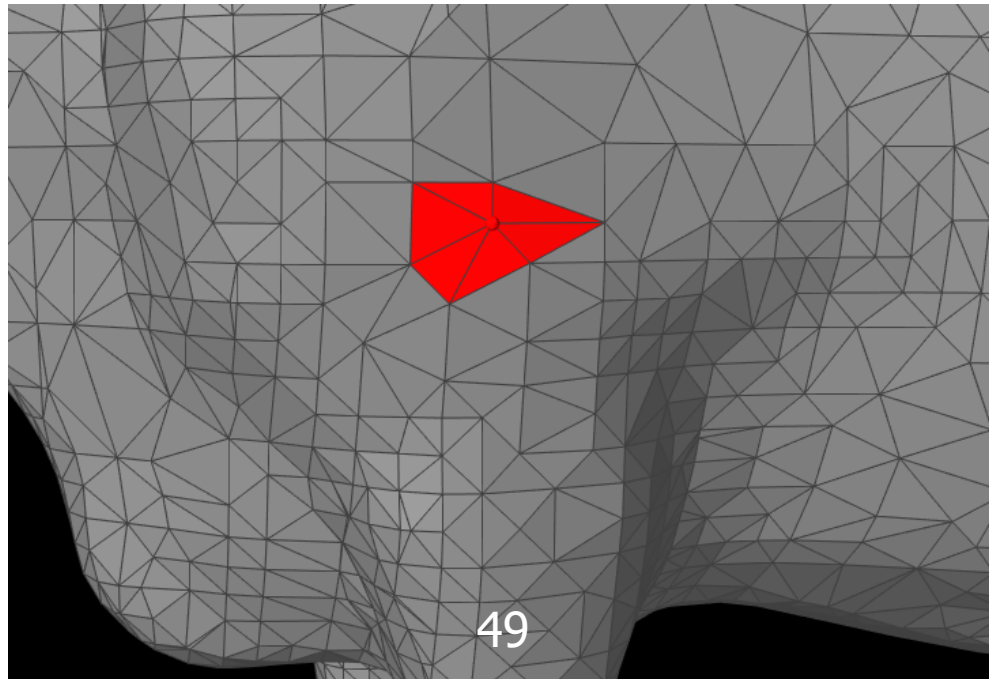
1. Break mesh into overlapping pieces
2. Try to move each piece rigidly
3. Combine all local transformations into a smooth one



# Pieces

## Vertex Umbrella

- Covers entire surface
- One cell per vertex
- All triangles exist in 3 cells

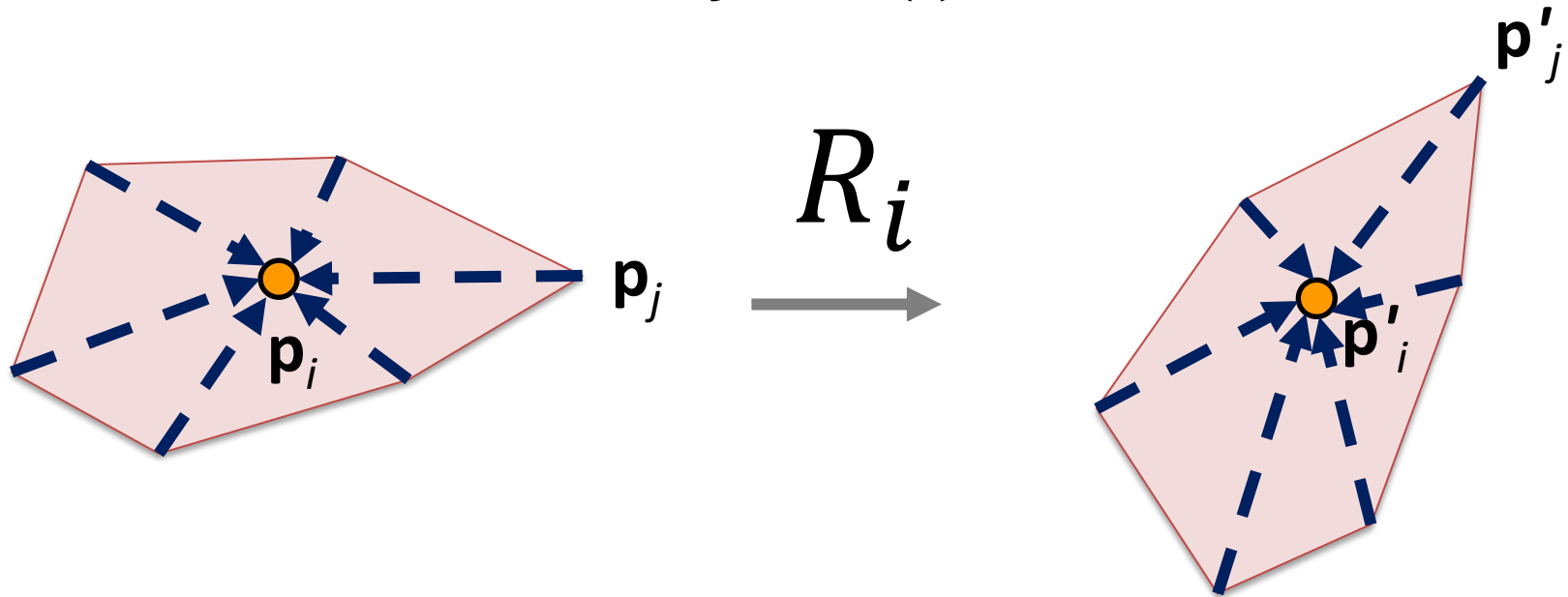


# Rigid motion

If cell  $i$  moved rigidly:

$$p'_j - p'_i = R_i(p_j - p_i)$$

$$\forall j \in N(i)$$



# Deviation from rigid motion

If cell  $i$  moved rigidly:

$$p'_j - p'_i = R_i(p_j - p_i)$$

$$\forall j \in N(i)$$

$$E = \sum_{j \in N(i)} \|p'_j - p'_i - R_i(p_j - p_i)\|^2$$

For the whole mesh

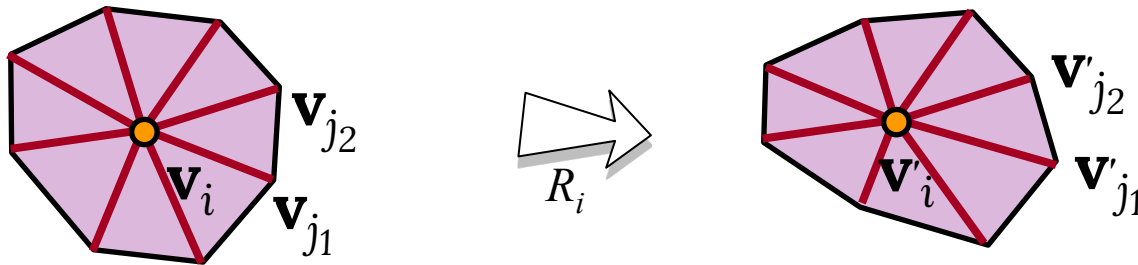
$$E = \sum_i \sum_{j \in N(i)} \|p'_j - p'_i - R_i(p_j - p_i)\|^2$$

For the whole mesh

$$E = \sum_i \sum_{j \in N(i)} \mathbf{w}_{ij} \|p'_j - p'_i - R_i(p_j - p_i)\|^2$$

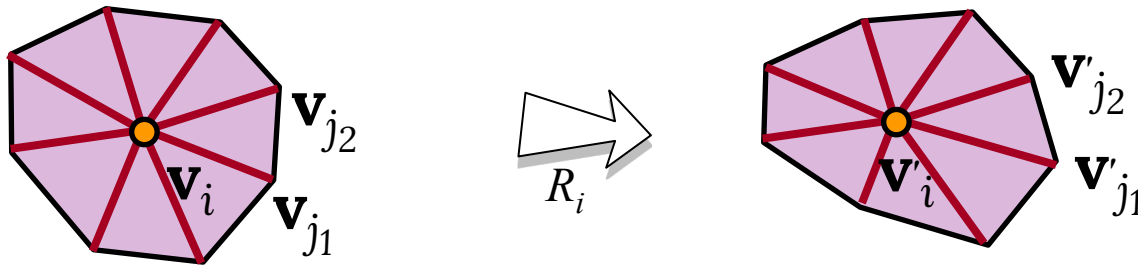
# Orthogonal Procrustes problem

How to find the best rotation matrix aligning  $V$  with  $V'$ ?



# Orthogonal Procrustes problem

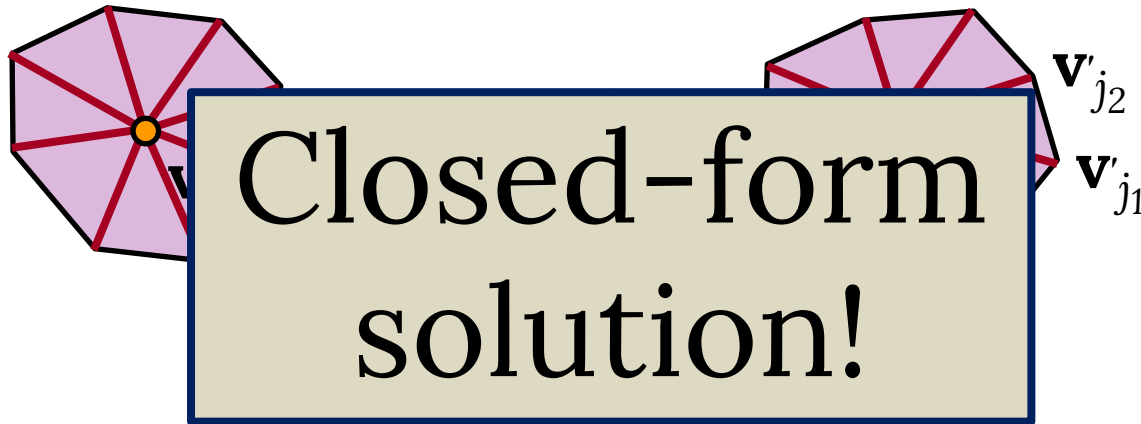
How to find the best rotation matrix aligning  $V$  with  $V'$ ?



$$\begin{aligned} \operatorname{argmin}_R & \|RA - B\|_F \\ \text{s. t. } & R^T R = I \end{aligned}$$

???

# Procrustes problem



1. Build covariance matrix  $S = AB^T$
2. SVD:  $S = U\Sigma W^T$
3.  $R_i = UW^T$



# Mesh Deformation

$$\min \sum_i \sum_{j \in N(i)} \mathbf{w}_{ij} \|p'_j - p'_i - R_i(p_j - p_i)\|^2$$
$$\text{s.t. } p'_i = \tilde{p}_i$$



point constraints

Caveats:

- $\{\mathbf{p}'_i\}$  and  $\{R_i\}$  are unknown
- Non-linear optimization problem

# As-Rigid-As-Possible

1. Initialize  $R_i = I$ , for all  $i$
2. **Global Step.** Given  $\{R_i\}$ , minimize energy to find  $\{\mathbf{p}'_i\}$
3. **Local Step.** Fix  $\{\mathbf{p}'_i\}$ , find optimal rotations  $\{R_i\}$  via SVD.
4. Repeat steps 2 and 3 until convergence.

$$\sum_{j \in N(i)} w_{ij} (\mathbf{p}'_i - \mathbf{p}'_j) = \sum_{j \in N(i)} \frac{w_{ij}}{2} (R_i + R_j) (\mathbf{p}_i - \mathbf{p}_j)$$

$$L\mathbf{p}' = \mathbf{b}$$

# Advantages

## Laplacian

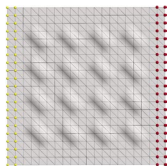
- Depends only on original mesh
- Only needs to be factored once!

## Rotations can be computed in parallel

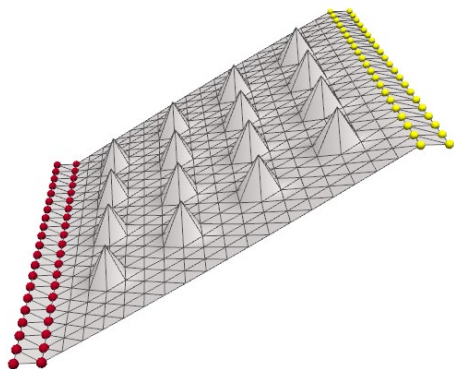
- Each iteration reduces energy
  - Updating rotations guaranteed to reduce cell-error
  - Updating positions guaranteed to reduce global error

*Guaranteed Convergence*

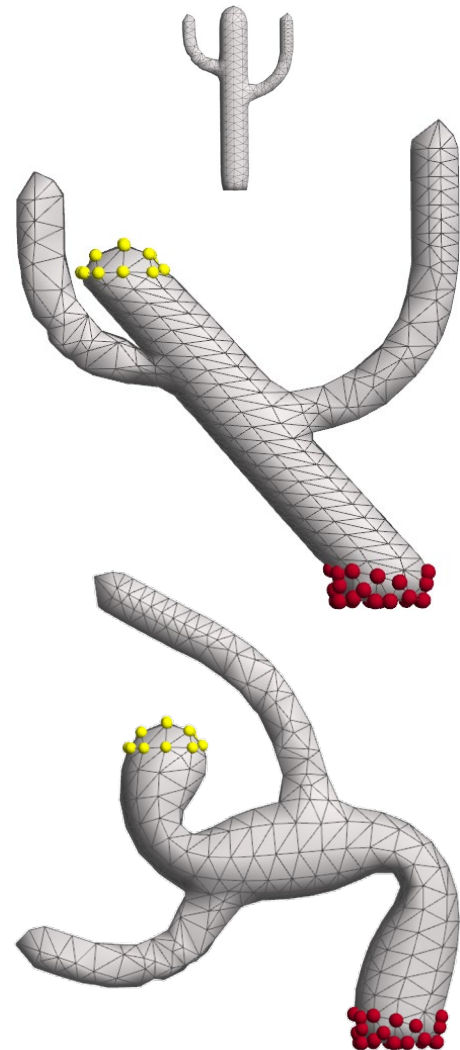
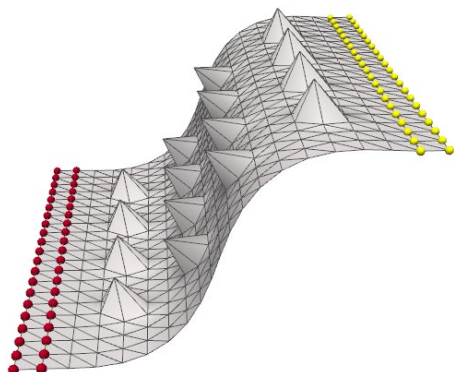
# Results (vs Poisson)



Poisson:



ARAP:



# Deformation models

Direct

$$v' = \left( \sum w_j T_j \right) v$$

- Linear Blend Skinning
- Dual Quaternion Skinning
- ...

Variational

$$v' = \operatorname{argmin}_x E(x)$$

- Multiresolution editing
- As-Rigid-As-Possible
- Laplacian Mesh Editing
- ...

# Deformation models

Direct

$$v' = \left( \sum w_j T_j \right) v$$

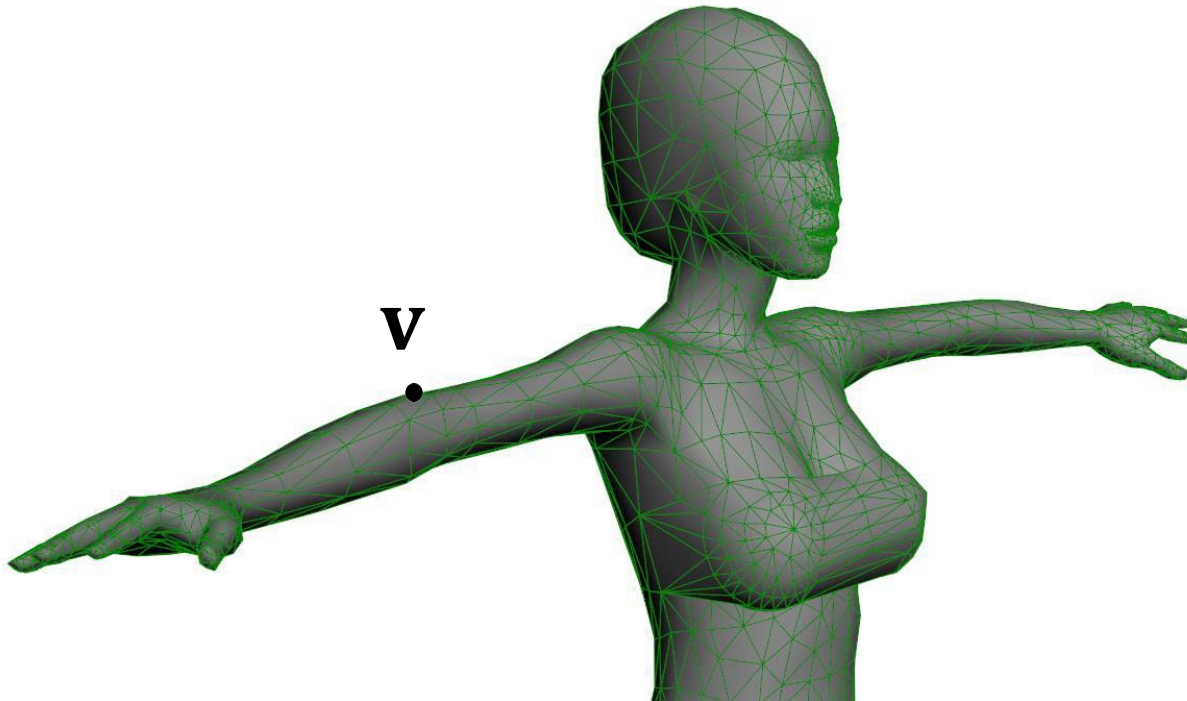
- Linear Blend Skinning
- Dual Quaternion Skinning
- ...

Variational

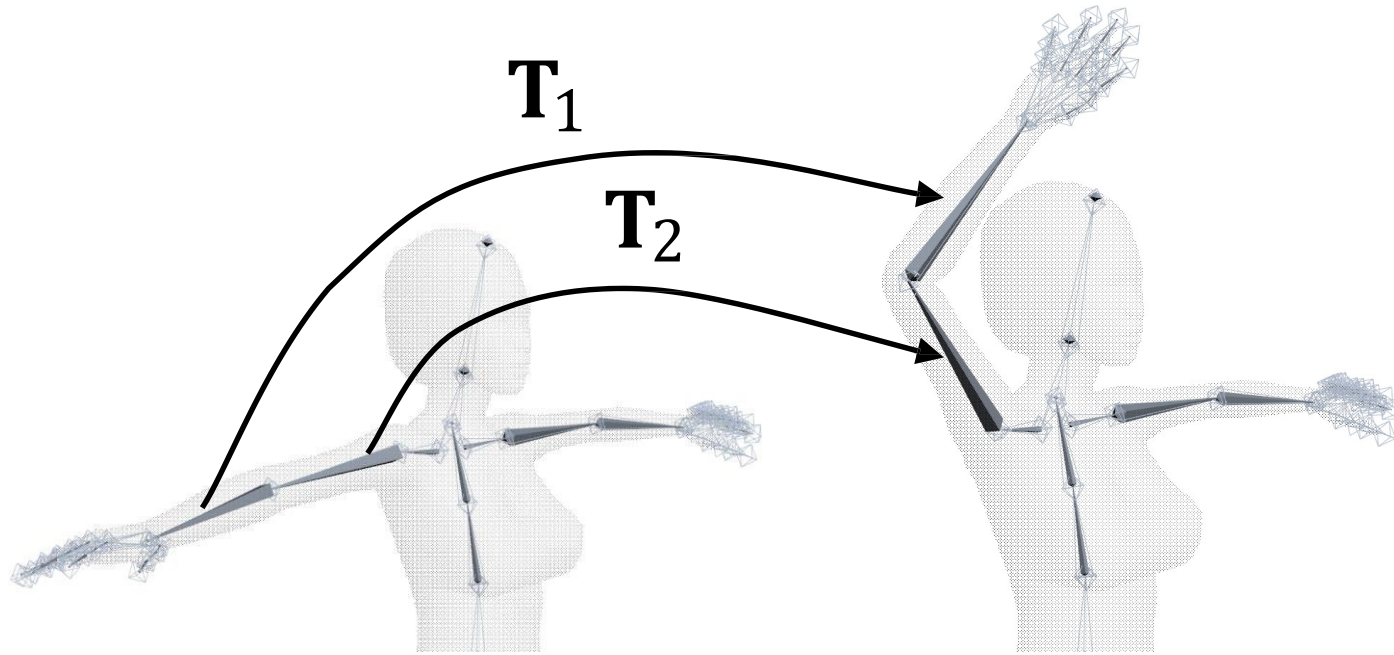
$$v' = \operatorname{argmin}_x E(x)$$

- Multiresolution editing
- As-Rigid-As-Possible
- Laplacian Mesh Editing
- ...

# 1) Rest pose

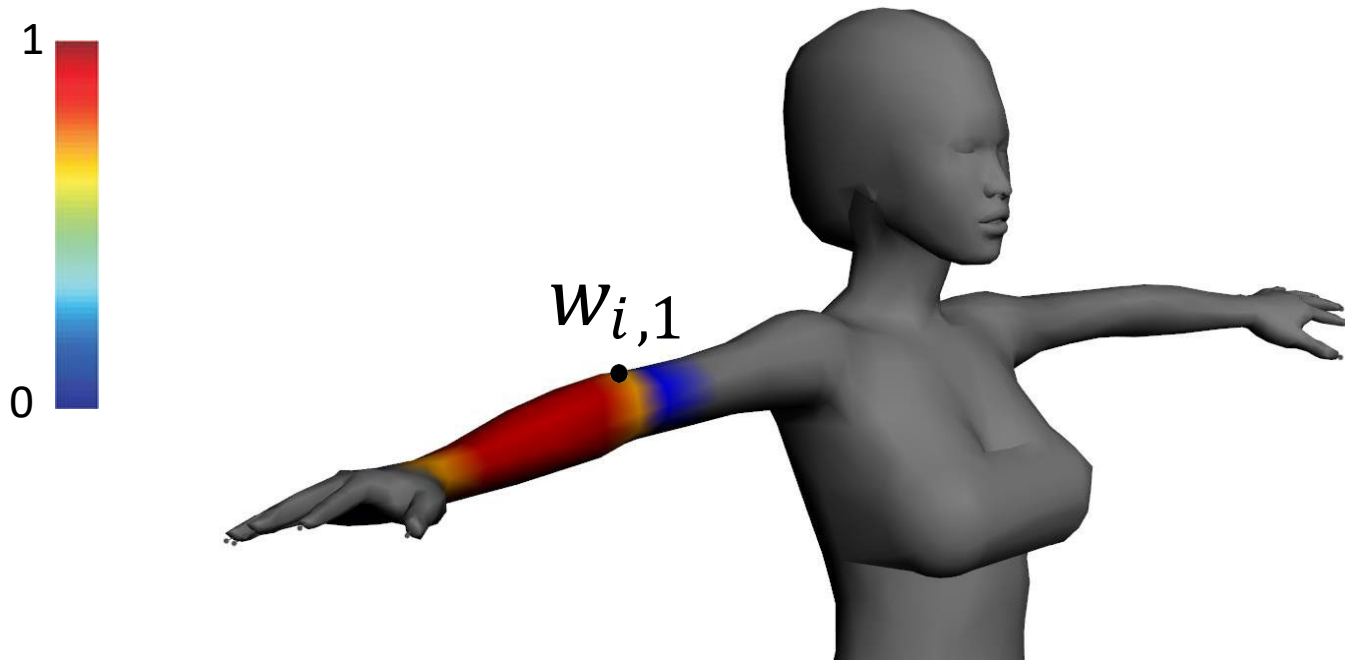


## 2) Skinning transformations

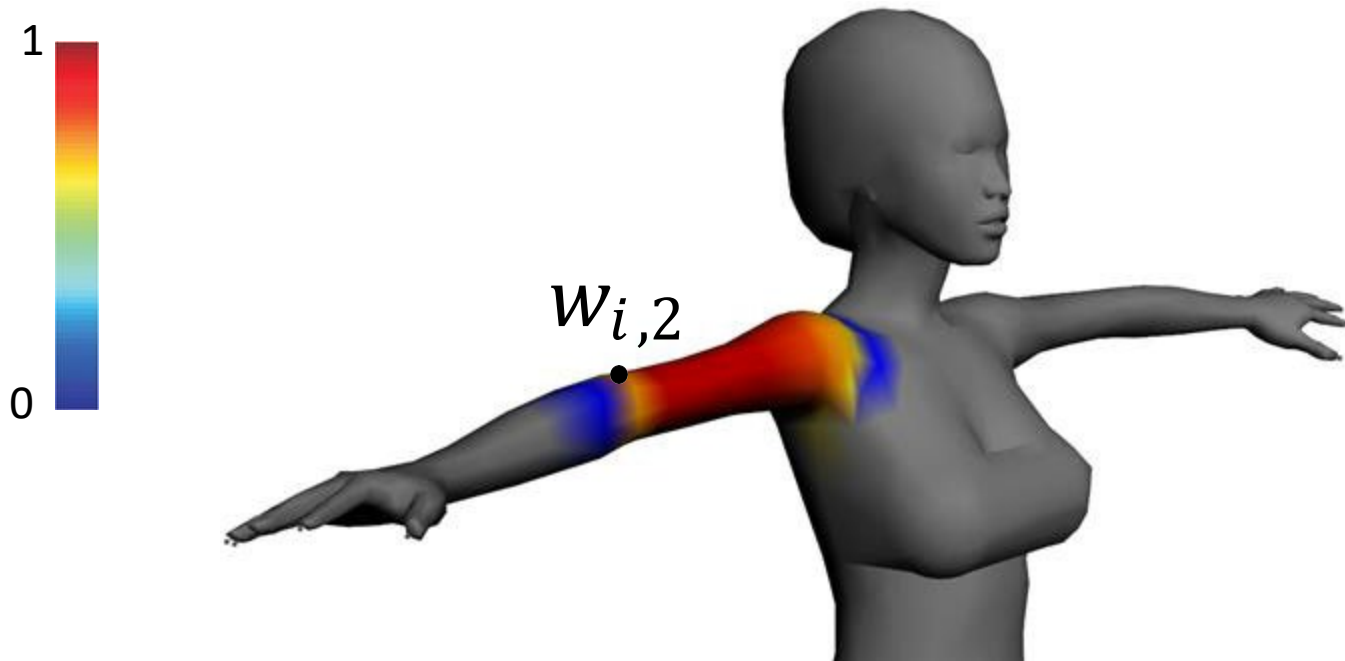




### 3) Skinning weights



### 3) Skinning weights



# Linear blend skinning (LBS)

$$v' = (\sum w_j T_j) v$$

# LBS is used widely in the industry

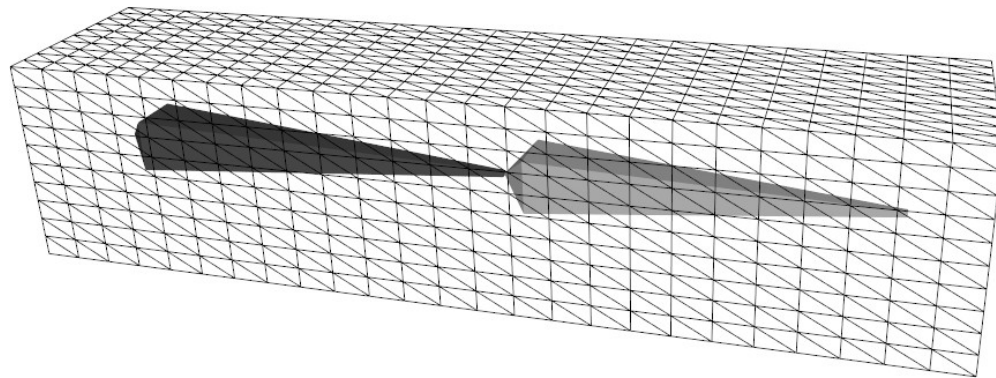


Halo 3



Bolt

# LBS: candy-wrapper artifact



# LBS: candy-wrapper artifact

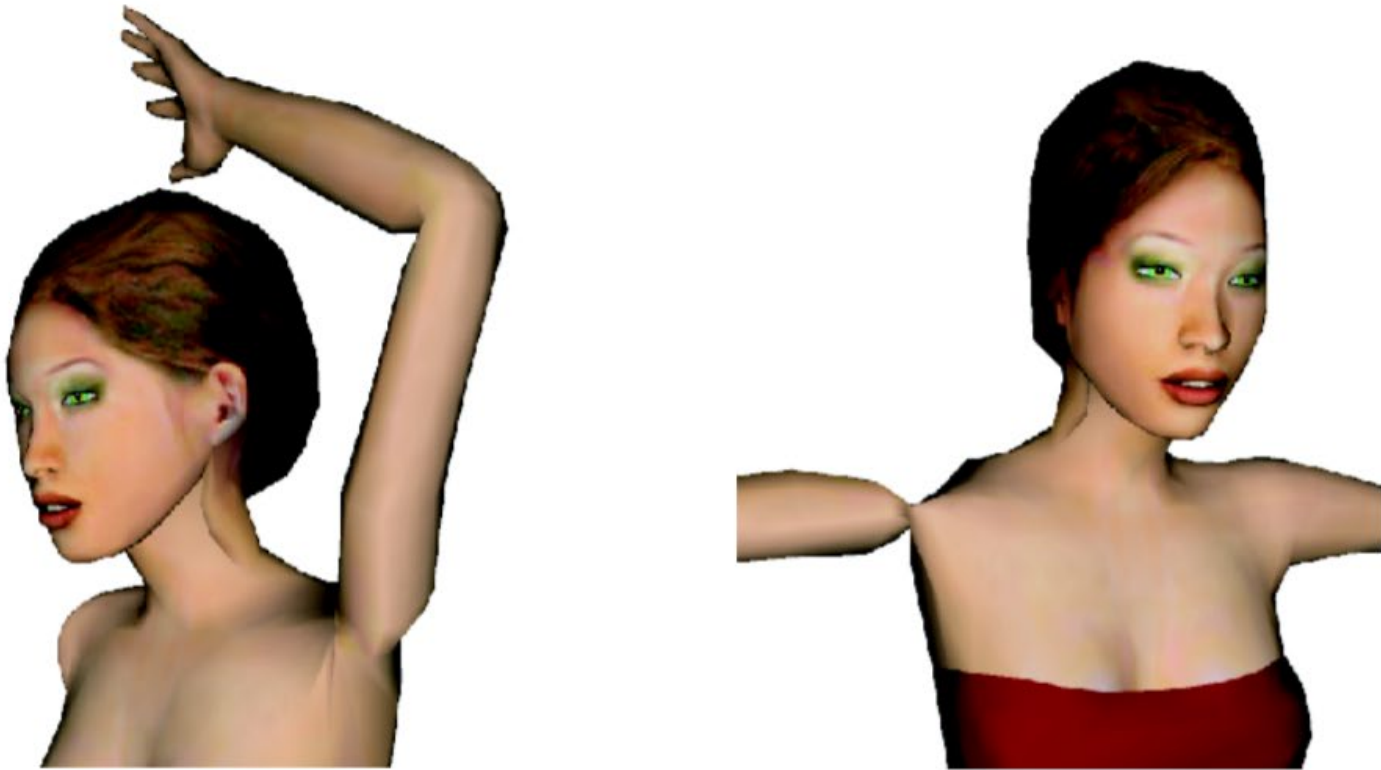


Figure 2: Typical “candy-wrapper” artifacts of linear blend skinning.

# What went wrong?

$$v' = (\sum w_j T_j) v$$

# What went wrong?

$$v' = (\sum w_j T_j) v$$

$$\mathbf{R}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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



# What went wrong?

$$v' = (\sum w_j T_j) v$$

$$\mathbf{R}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

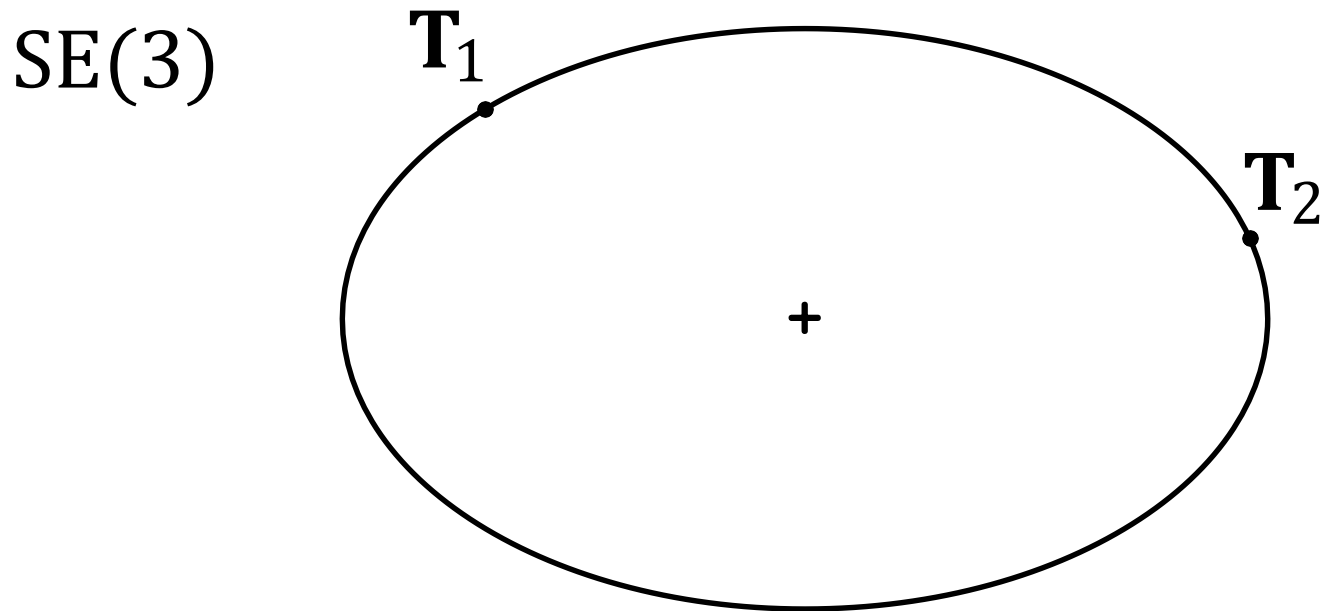
$$\mathbf{R}_2 = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Why can't we just sum up rotation matrices?

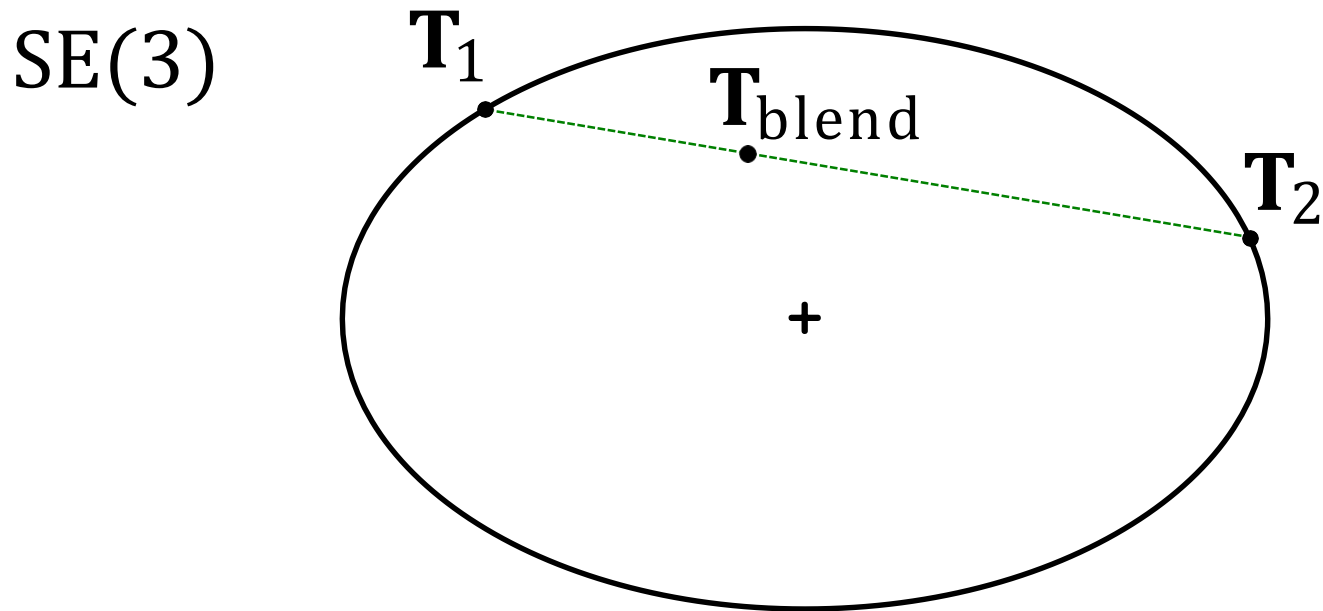
# Geometry of linear blending

$SE(3)$

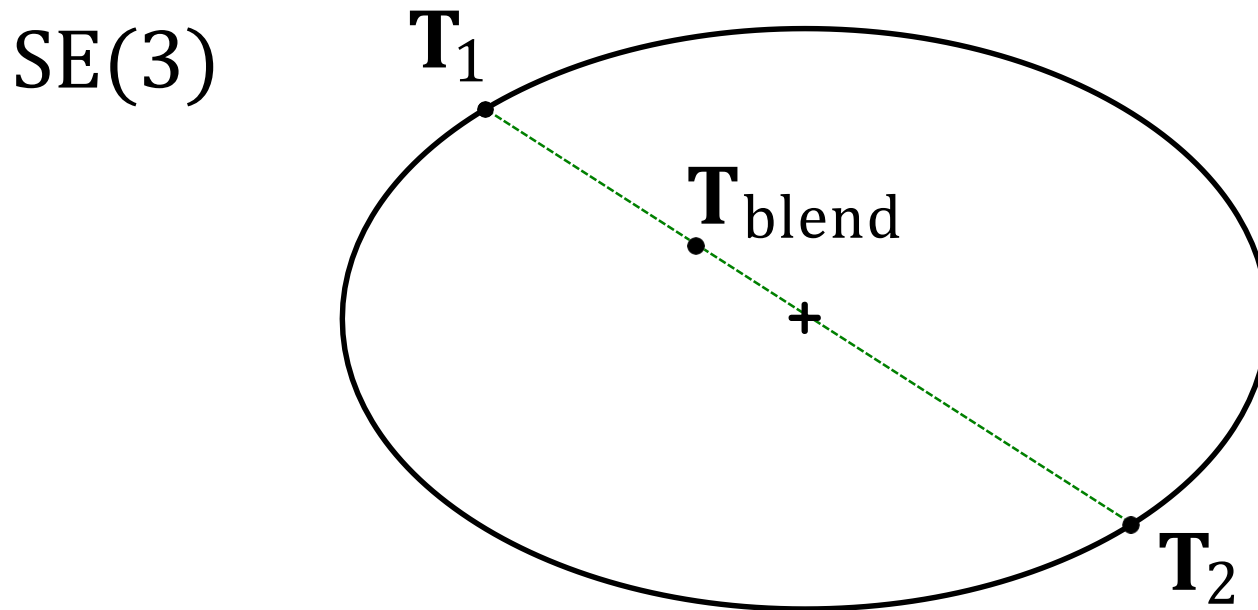
# Geometry of linear blending



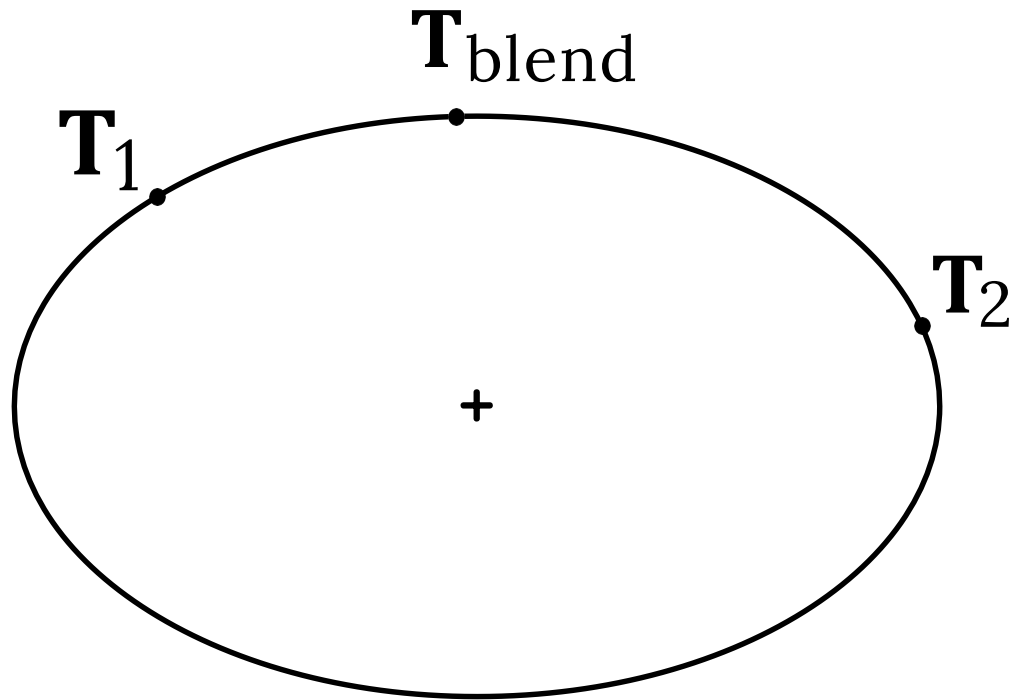
# Geometry of linear blending

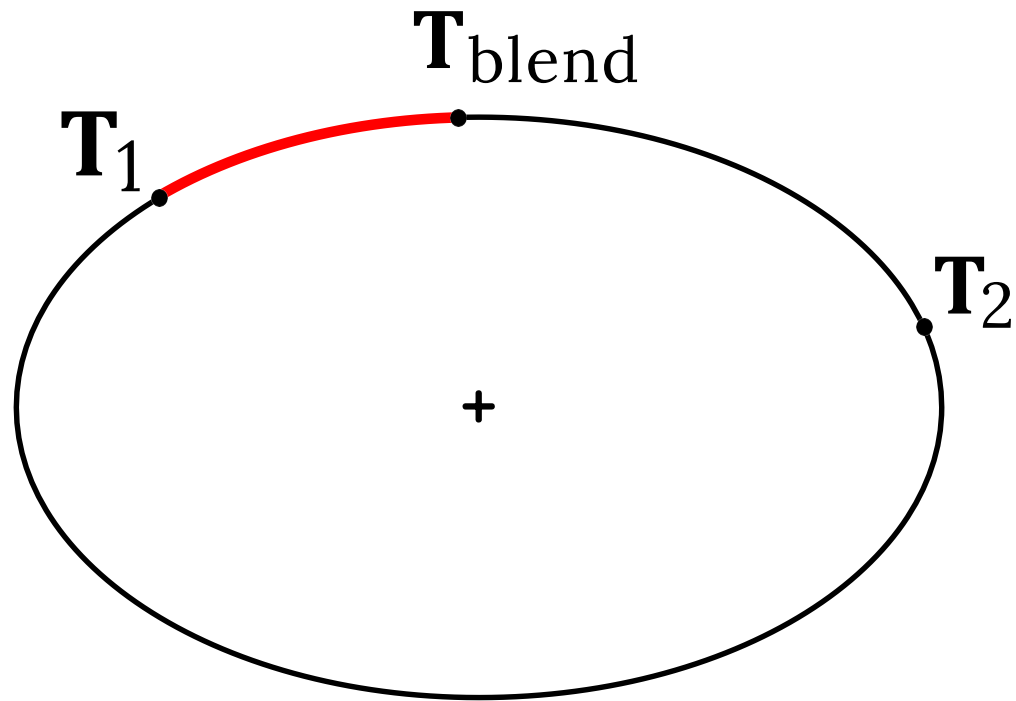


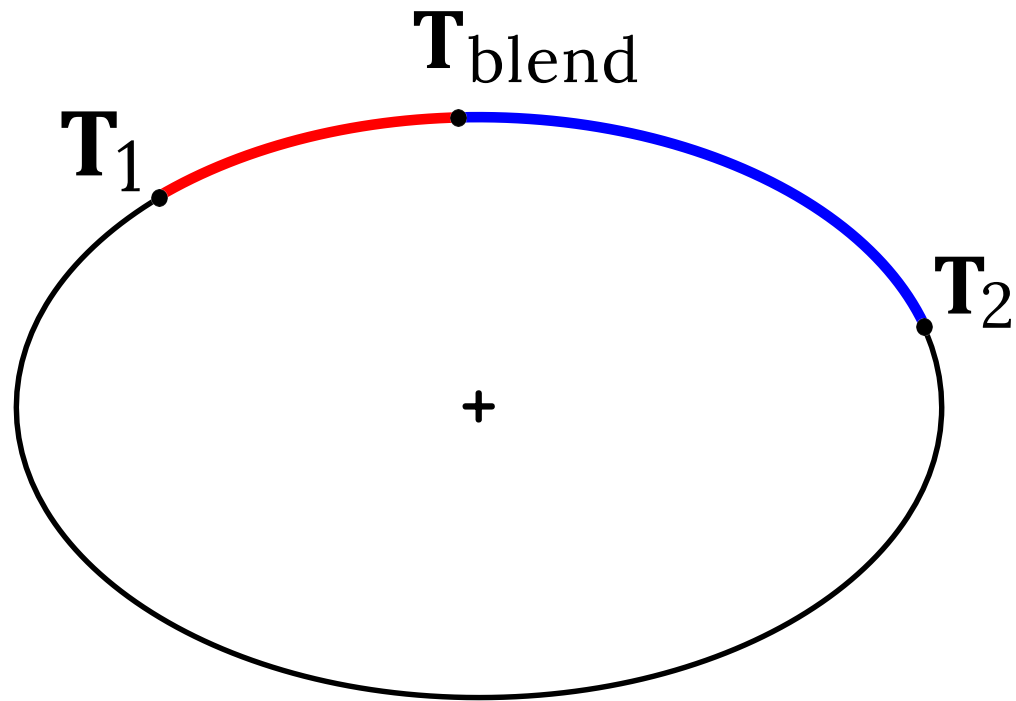
# Geometry of linear blending



# Intrinsic blending









# Intrinsic blending using Lie algebras

[Buss and Fillmore 2001, Alexa 2002, Govindu 2004, Rossignac and Vinacua 2011]

$$\operatorname{argmin}_X \sum_j w_j d(X, T_j)$$

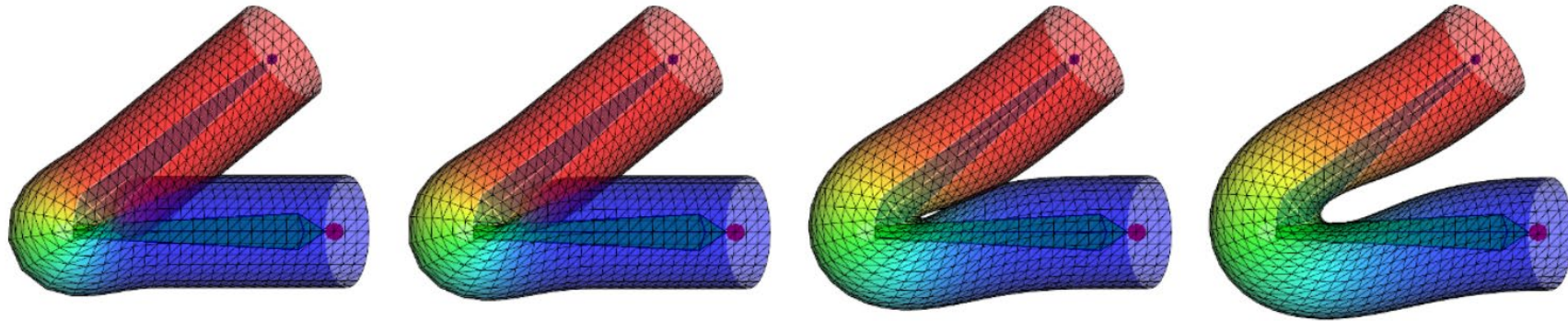
$$d(\mathbf{X}, \mathbf{Y}) = \|\log(\mathbf{Y}\mathbf{X}^{-1})\|^2$$

Karcher / Frechet mean

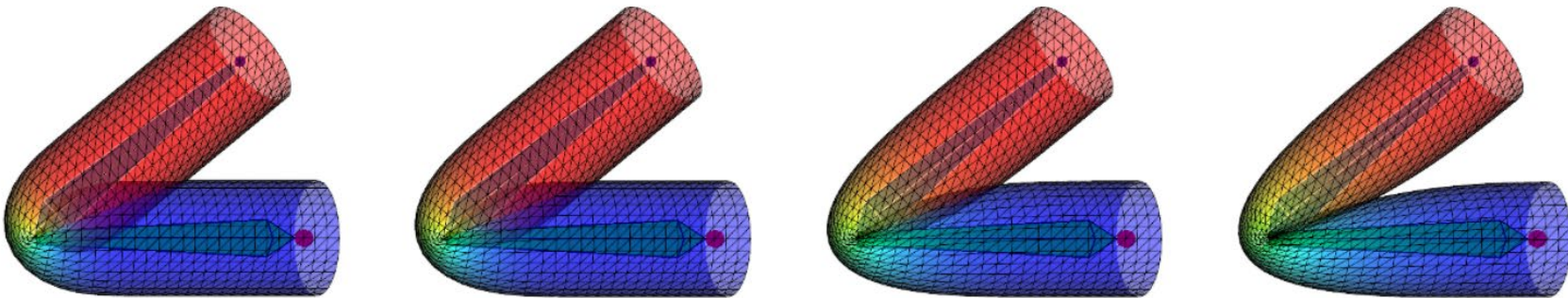
81

# Dual Quaternion Skinning

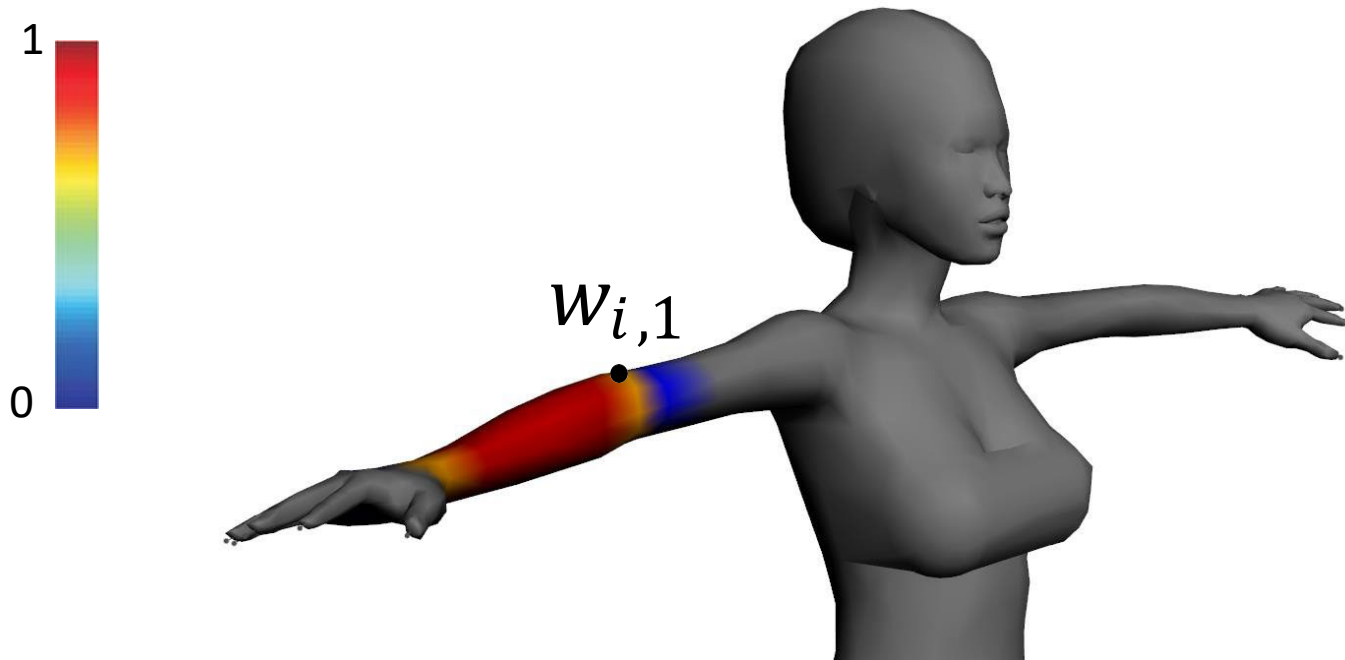
Dual  
Quats



Linear  
Blend



# Where do the weights come from?



# Manual?



Automatic skinning  
weight computation

# Weights should obtain a few basic qualities



Inverse Euclidean distance weights are too crude,  
show obvious artifacts



$$w_j(\mathbf{v}) = \frac{1}{\|\mathbf{c}_i - \mathbf{v}\|^2}$$

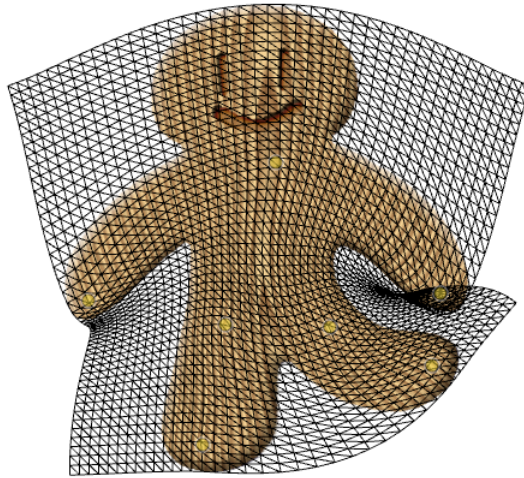
[Shepard 1968],  
[Schaefer et al. 2006], etc.



weights optimized *inside* shape



# Inverse Euclidean distance weights are too crude



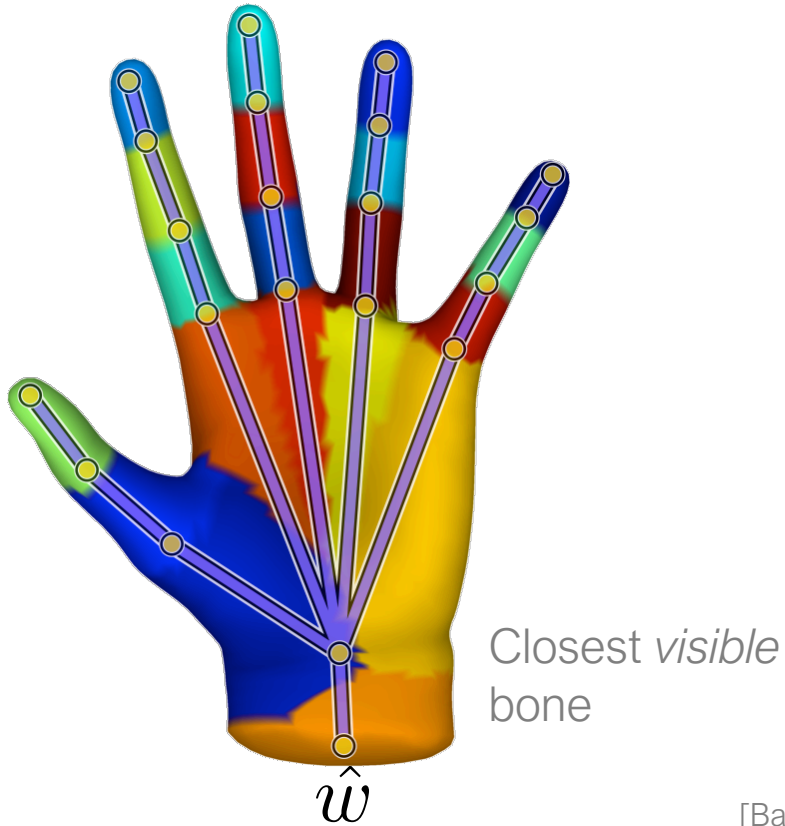
$$w_j(\mathbf{v}) = \frac{1}{\|\mathbf{c}_i - \mathbf{v}\|^2}$$

weights optimized *inside* shape

[Shepard 1968],  
[Schaefer et al. 2006], etc.

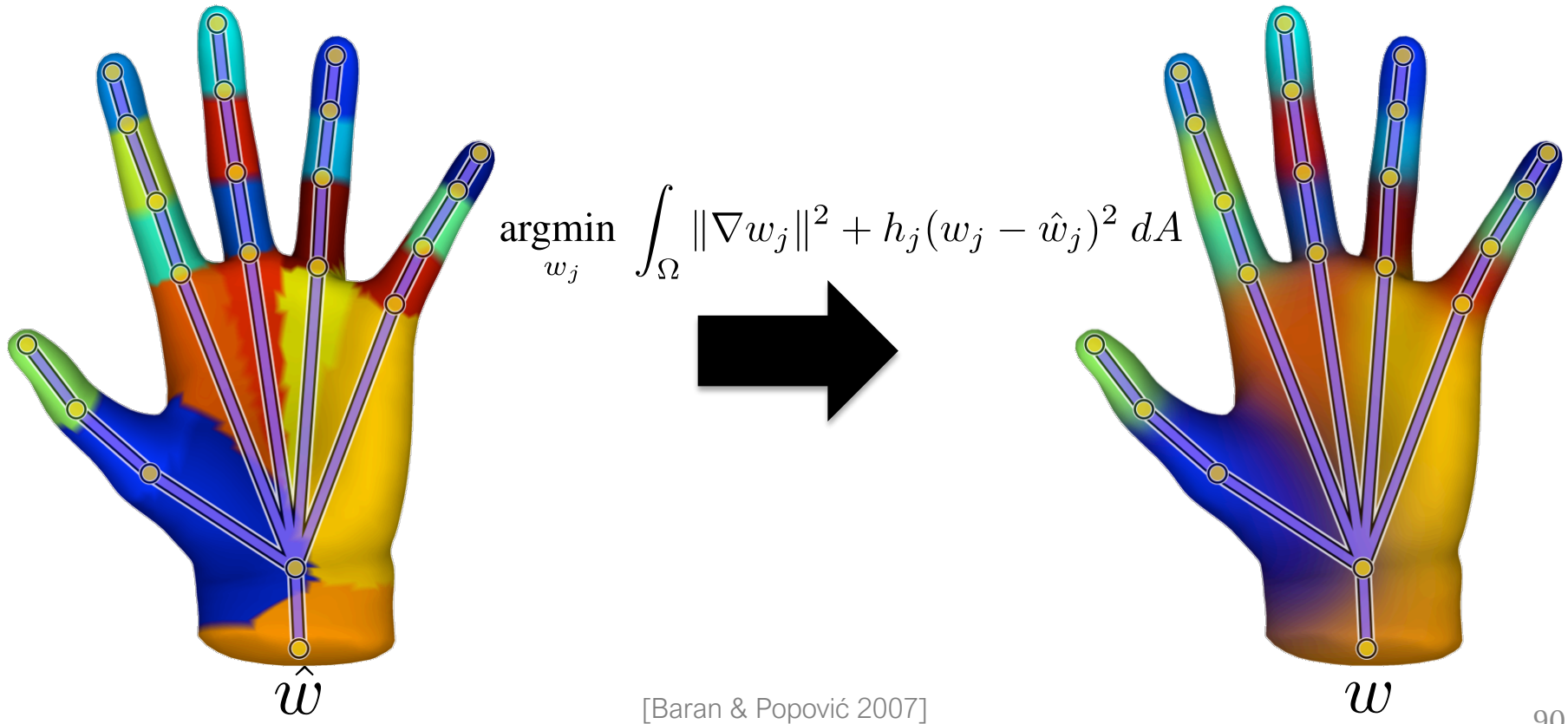


*Discontinuous* projection onto surface  
can be smoothed out

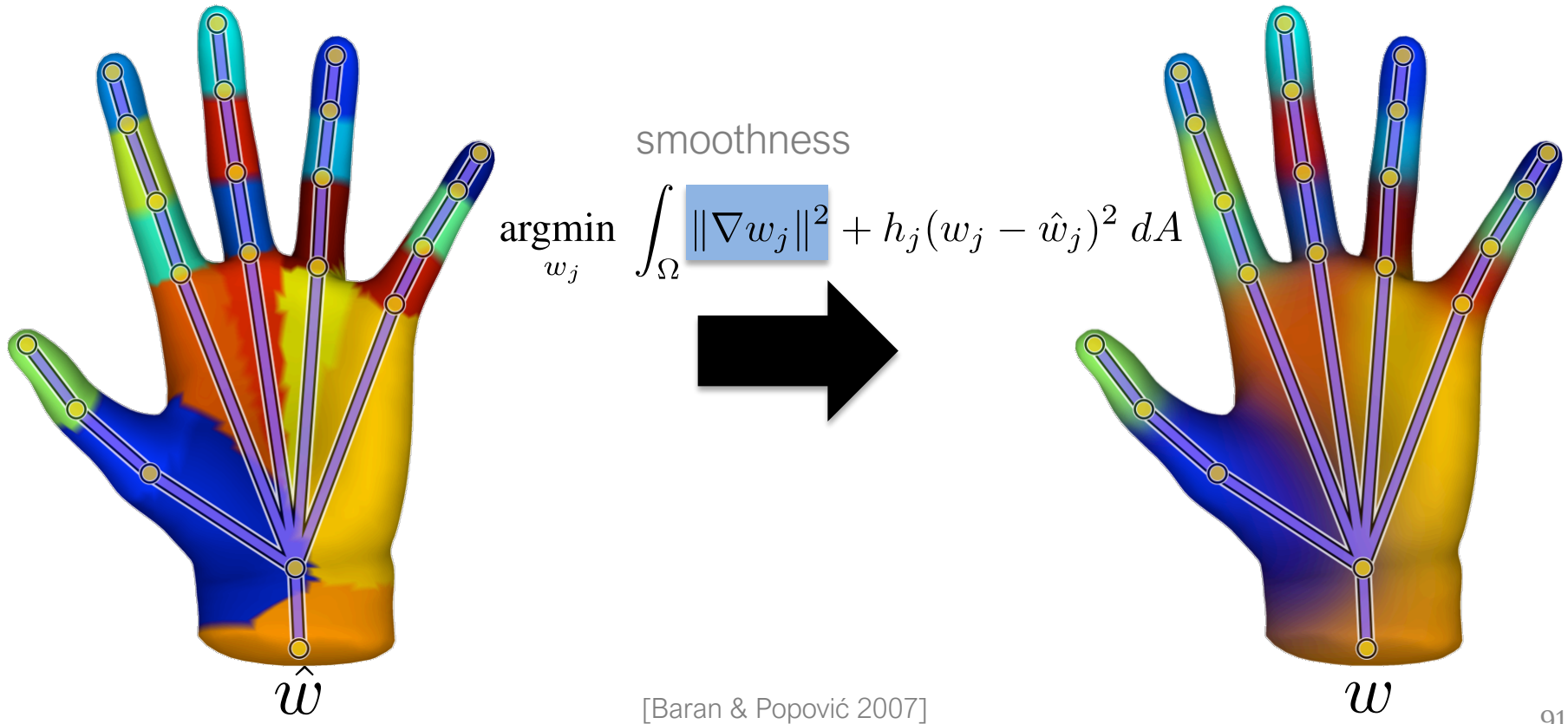


[Baran & Popović 2007]

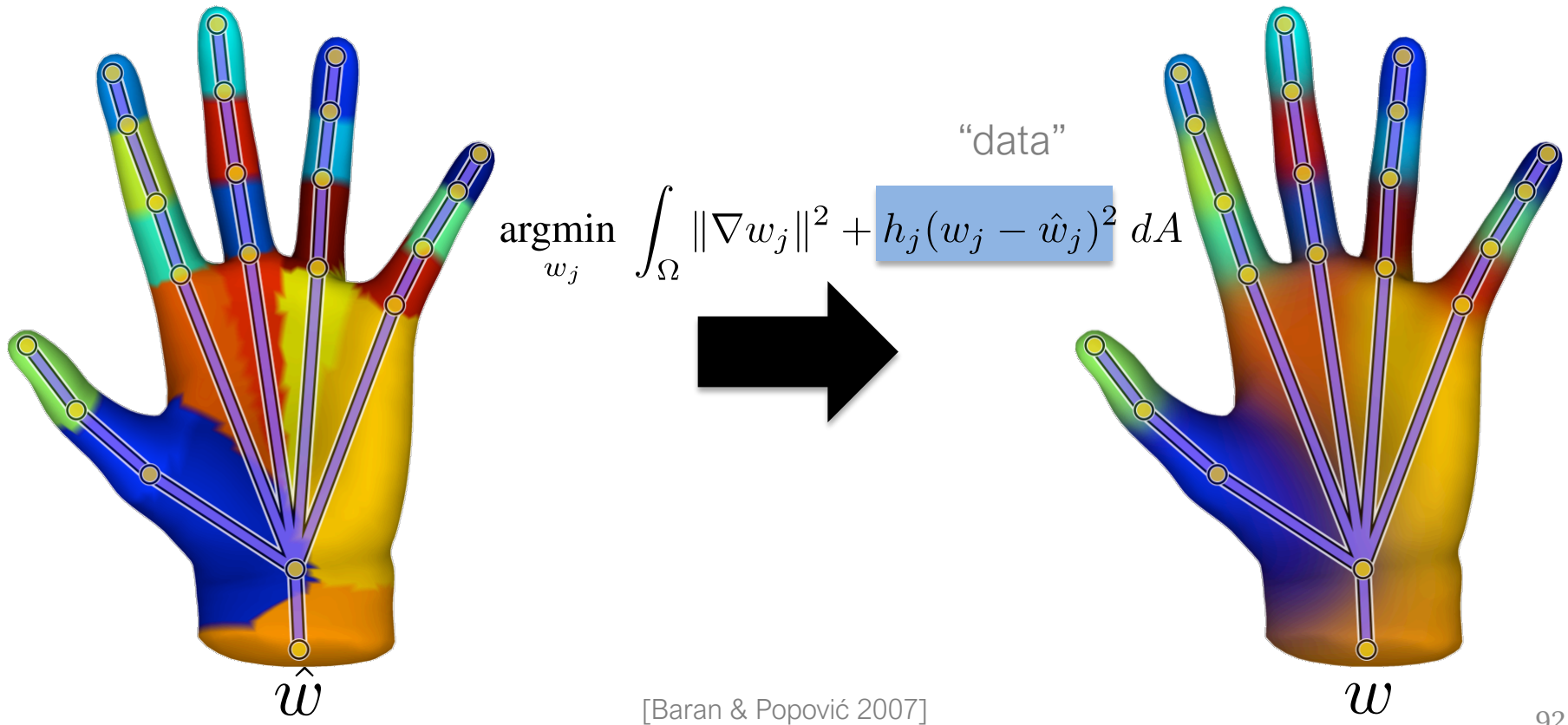
*Discontinuous* projection onto surface  
can be smoothed out



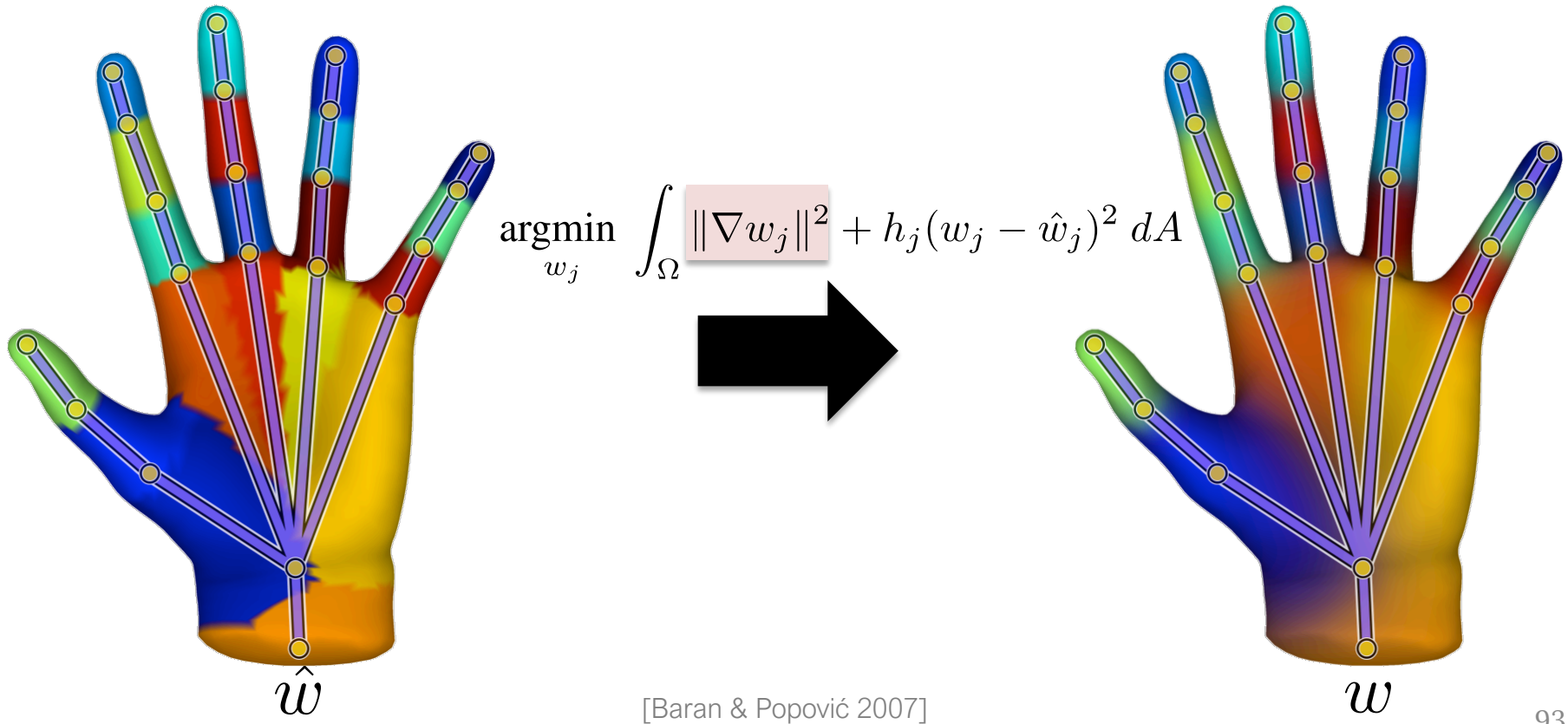
*Discontinuous* projection onto surface  
can be smoothed out



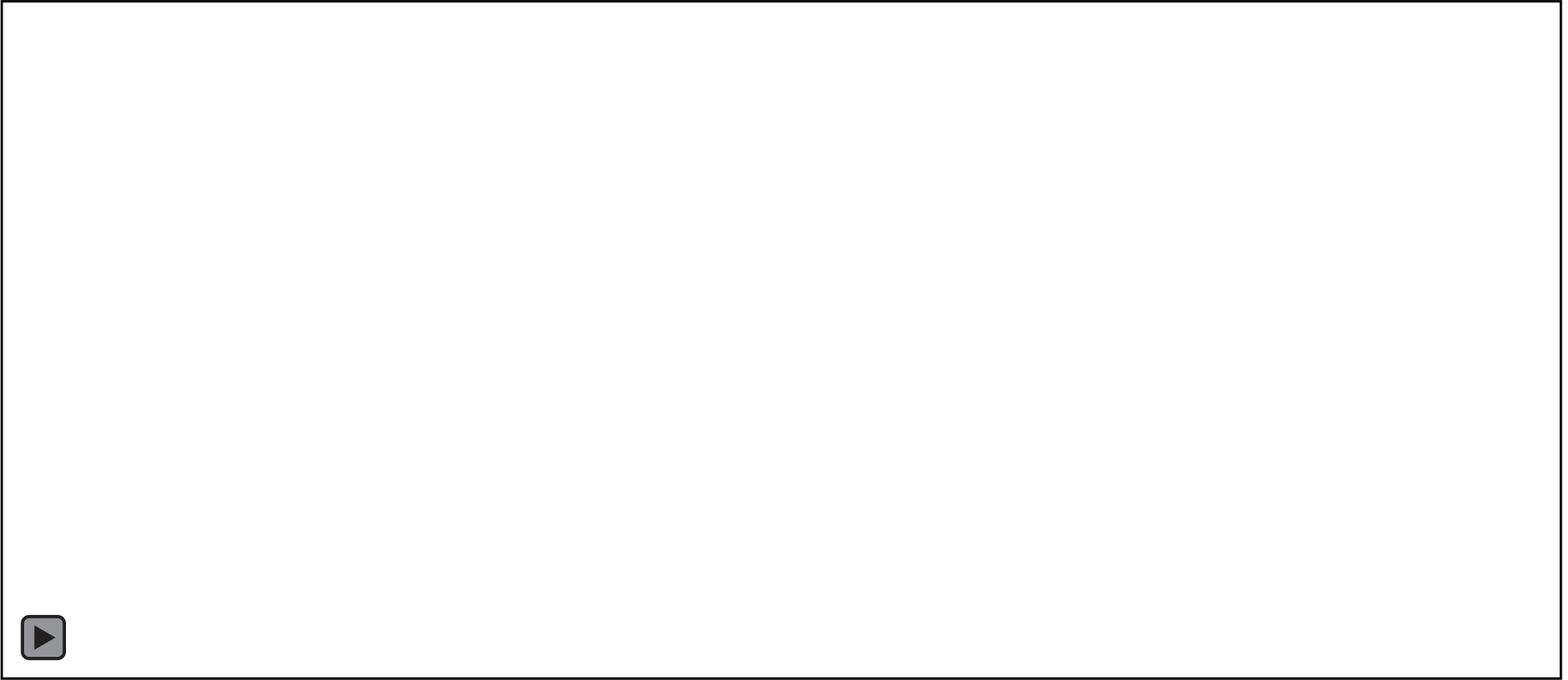
*Discontinuous* projection onto surface  
can be smoothed out



*Discontinuous* projection onto surface  
can be smoothed out



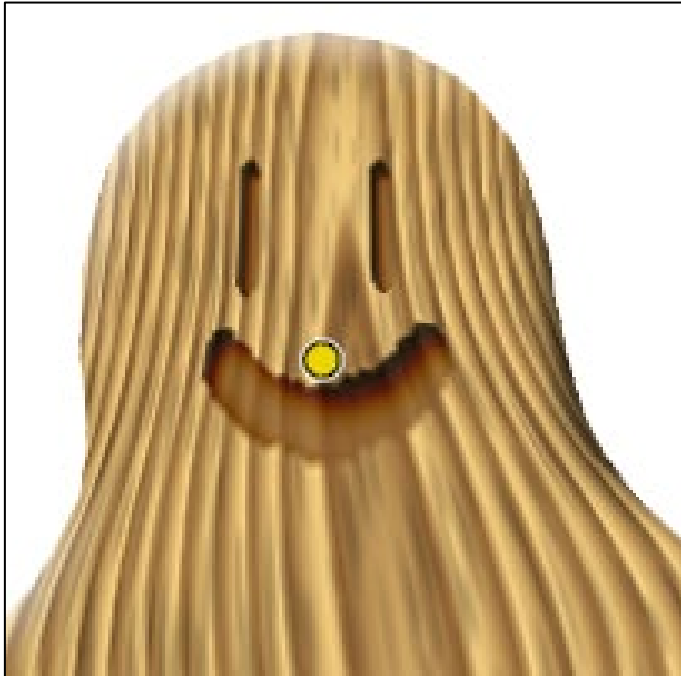
# Gradient energy weights not smooth at handles



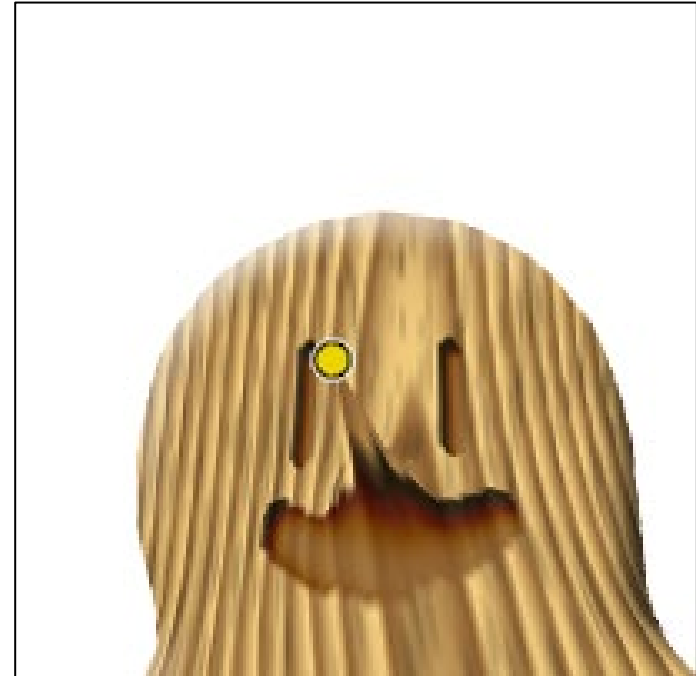
$$\operatorname{argmin}_{w_j} \int_{\Omega} (\Delta w_j)^2 dA$$

$$\operatorname{argmin}_{w_j} \int_{\Omega} \|\nabla w_j\|^2 dA$$

# Gradient energy weights not smooth at handles

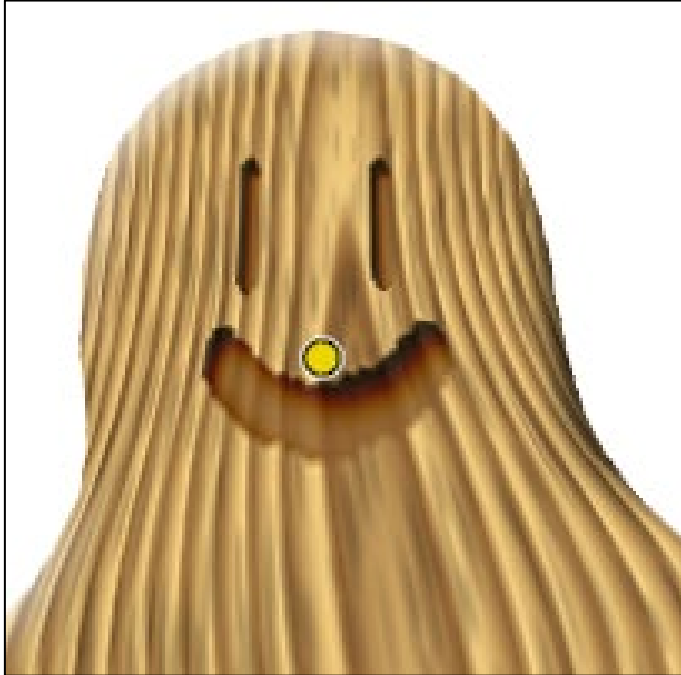


$$\operatorname{argmin}_{w_j} \int_{\Omega} (\Delta w_j)^2 dA$$

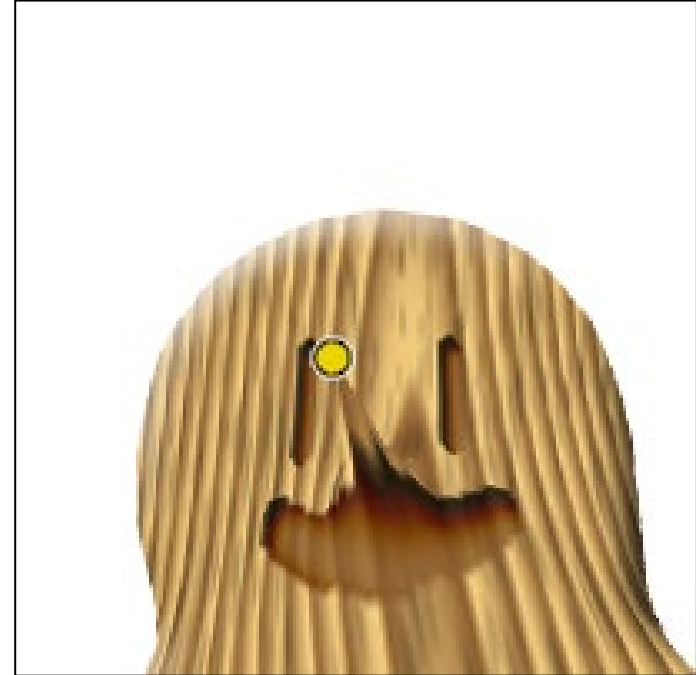


$$\operatorname{argmin}_{w_j} \int_{\Omega} \|\nabla w_j\|^2 dA$$

# Gradient energy weights not smooth at handles



$$\Delta^2 w_j = 0$$



$$\Delta w_j = 0$$



# Point constraints for Laplace equation



<https://www.facebook.com/521399544544480/photos/a.523048724379562/800968259920939/?type=1&theater> , Facebook group “Circus tents and circus equipment”

# Non-negative, local weights are mandatory

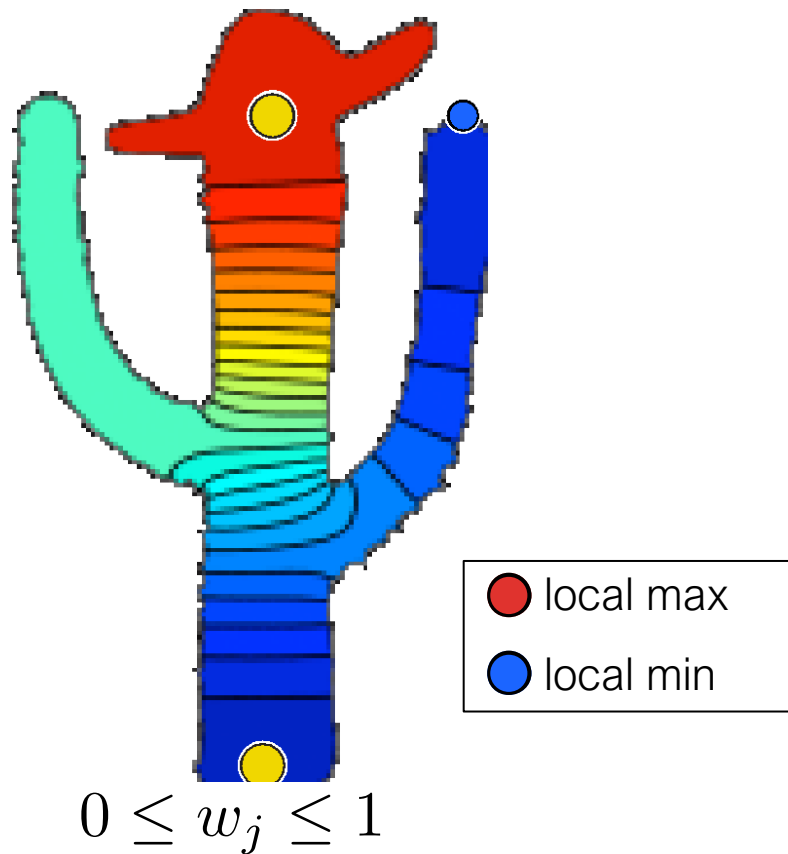
$$0 \leq w_j \leq 1$$

$$\operatorname{argmin}_{w_j} \int_{\Omega} (\Delta w_j)^2 dA$$

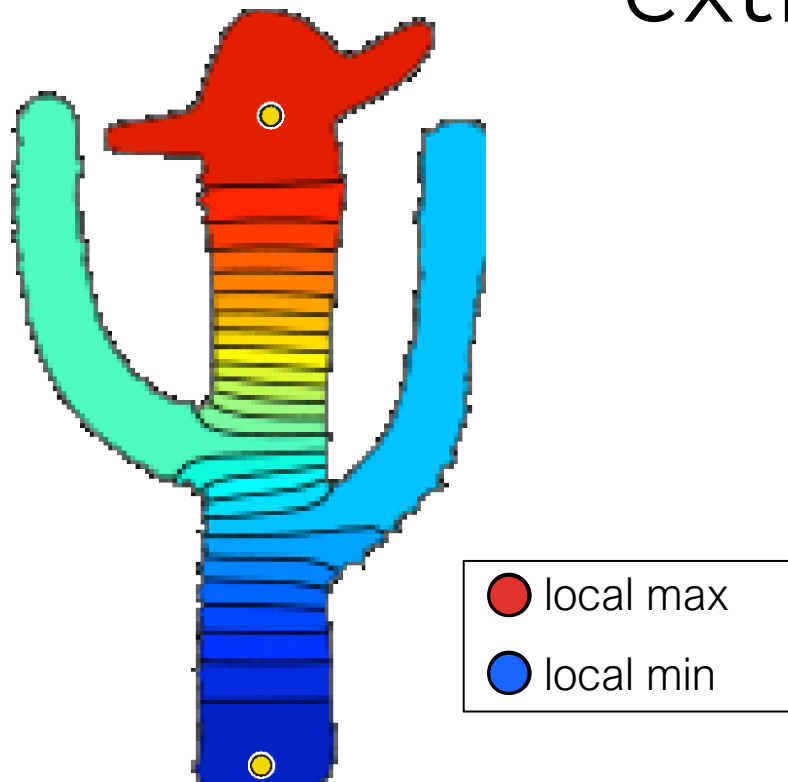
[Botsch & Kobbelt 2004]



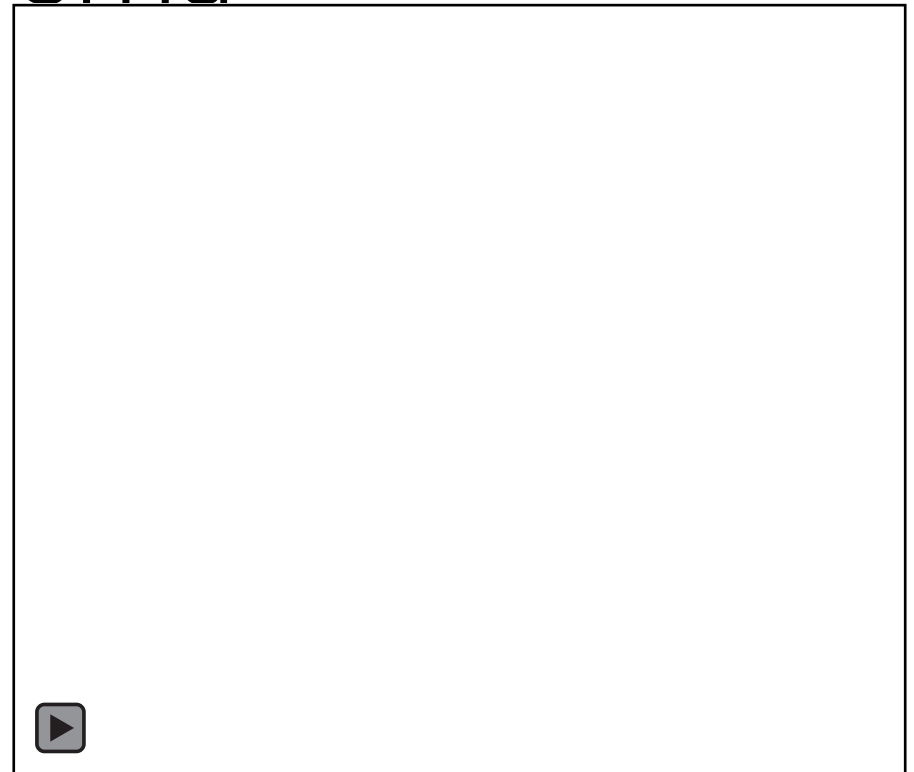
# Spurious extrema cause distracting artifacts



# Must explicitly prohibit spurious extrema



$w_j$  is “monotonic”



# Previous methods fail in one way or another

	Euclidean	$\Delta w_j$	$\Delta^2 w_j$
smooth	✓	—	✓
non-negative	✓	✓	-
shape-aware	—	✓	✓
local	-/✓	—	-
monotonic	-	✓	-
arbitrary handles	-	✓	✓
	[Shepard 1968, Sibson 1980, Schaefer et al. 2006]	[Baran & Popovic 2007, Joshi et al. 2007]	[Botsch & Kobbelt 2004, Sorkine et al. 2004, Finch et al. 2011]

# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

- + shape-aware
- + smoothness

[Botsch & Kobbelt 2004, Sorkine et al. 2004, Joshi & Carr 2008, Jacobson et al. 2010, Finch et al. 2011, Andrews et al. 2011]

# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

- + shape-aware
- + smoothness
- + arbitrary handles

$$w_j(\mathbf{v}) = \begin{cases} 1 & \mathbf{v} \in h_j, \\ 0 & \mathbf{v} \in h_k \\ \text{linear on cage facets} \end{cases}$$

[Botsch & Kobbelt 2004, Sorkine et al. 2004, Joshi & Carr 2008, Jacobson et al. 2010, Finch et al. 2011, Andrews et al. 2011]

# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

$$0 \leq w_j \leq 1,$$

$$\sum_{j=1}^m w_j = 1$$

- + shape-aware
- + smoothness
- + arbitrary handles
- + non-negativity

[Jacobson et al. 2011]



# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

$$0 \leq w_j \leq 1,$$

$$\sum_{j=1}^m w_j = 1$$

- + shape-aware
- + smoothness
- + arbitrary handles
- + non-negativity
- + locality

[Jacobson et al. 2011]

# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

$$\|w\|_1 = 1$$

- + shape-aware
- + smoothness
- + arbitrary handles
- + non-negativity
- + locality

[Rustamov 2011]

# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

$$\|w\|_1 = 1 \rightarrow \sum_{j=1}^m |w_j| = 1$$

- + shape-aware
- + smoothness
- + arbitrary handles
- + non-negativity
- + locality

[Rustamov 2011]

# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

$$\|w\|_1 = 1 \rightarrow \sum_{j=1}^m |w_j| = 1 \rightarrow \sum_{j=1}^m w_j = 1,$$

$$0 \leq w_j \leq 1$$

- + shape-aware
- + smoothness
- + arbitrary handles
- + non-negativity
- + locality

[Rustamov 2011]

# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

$$\nabla w_j \cdot \nabla u_j > 0$$

- + shape-aware
- + smoothness
- + arbitrary handles
- + non-negativity
- + locality
- + monotonicity

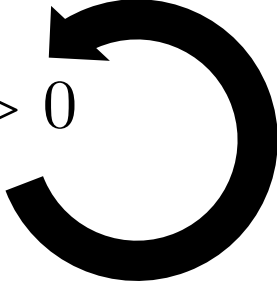
[Weinkauff et al. 2011, Jacobson et al. 2012, Günther et al. 2014]

# Previous methods fail in one way or another

	Euclidean	$\Delta w_j = u$	$\Delta^2 w_j$
smooth	✓	—	✓
non-negative	✓	✓	—
shape-aware	—	✓	✓
local	-/✓	—	—
monotonic	—	✓	—
arbitrary handles	—	✓	✓
	[Shepard 1968, Sibson 1980, Schaefer et al. 2006]	[Baran & Popovic 2007, Joshi et al. 2007]	[Botsch & Kobbelt 2004, Sorkine et al. 2004, Finch et al. 2011]

# Constrained optimization ensures satisfaction of all properties

$$\operatorname{argmin}_{w_j, j=1, \dots, m} \sum_{j=1}^m \int_{\Omega} (\Delta w_j)^2 dV$$

$$\nabla w_j \cdot \nabla u_j > 0$$


- + shape-aware
- + smoothness
- + arbitrary handles
- + non-negativity
- + locality
- + monotonicity

[Weinkauff et al. 2011, Jacobson et al. 2012, Günther et al. 2014]

