L16: Deformation

Hao Su

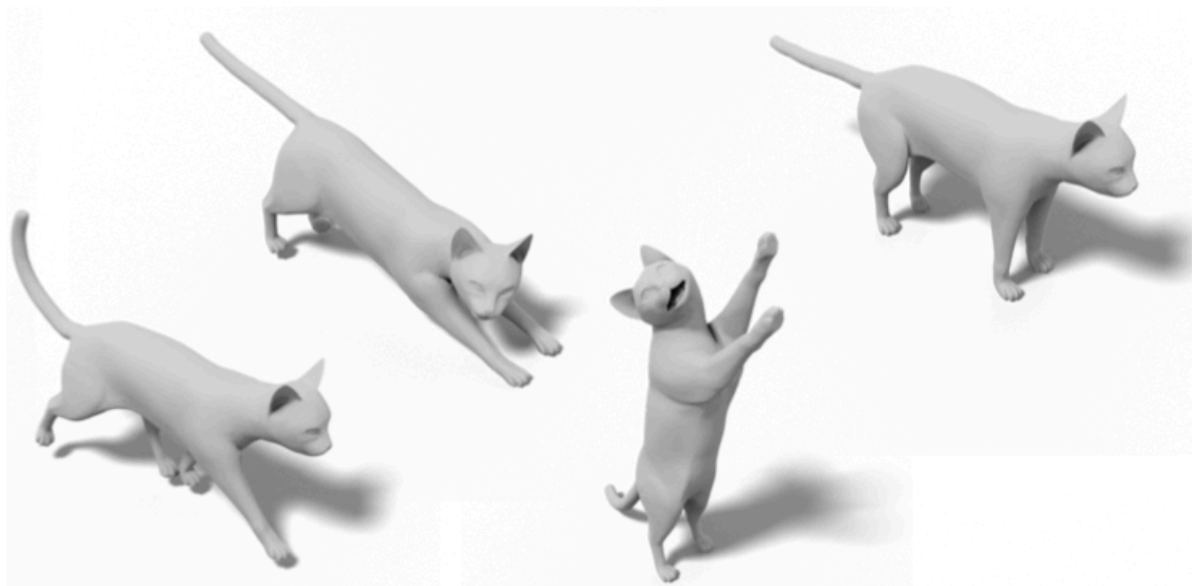
Ack: Yuzhe Qin and Fanbo Xiang for helping to prepare slides

Agenda

- Introduction
- Surface Deformation
- Space Deformation
- Skeleton Skinning

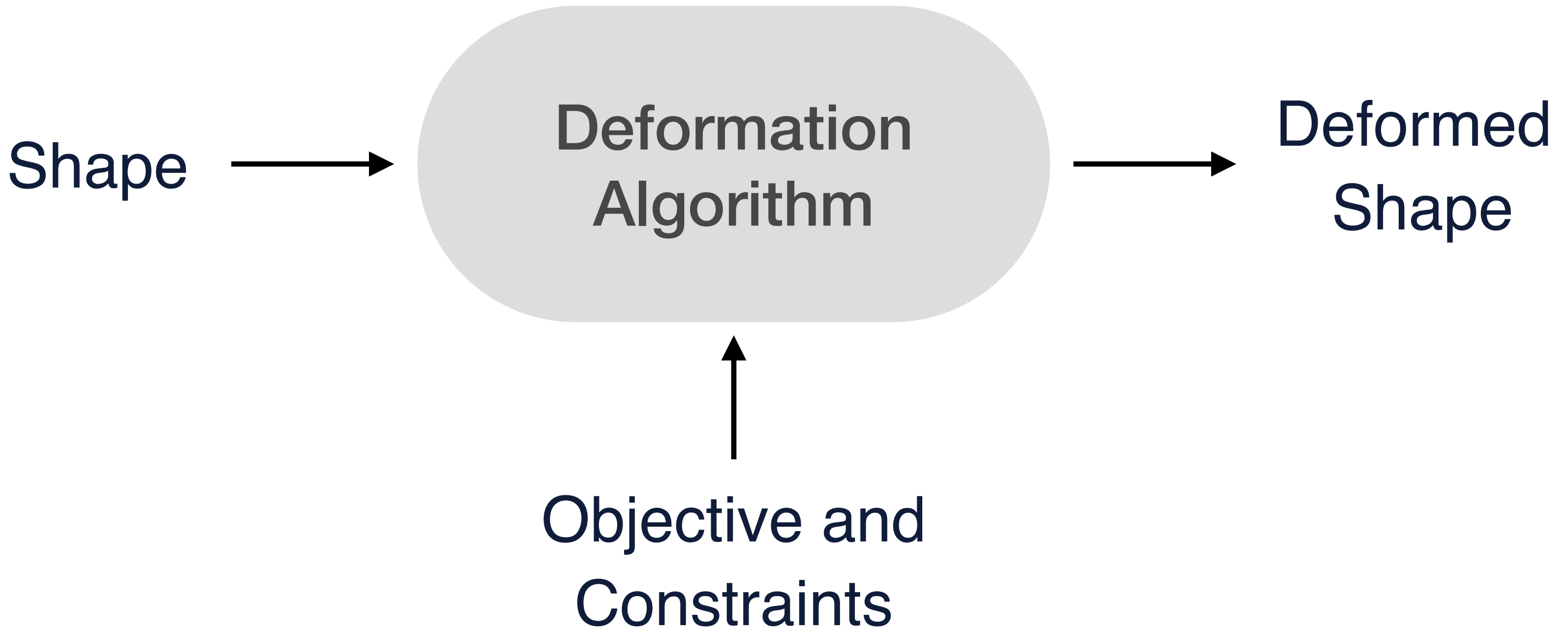
Shape Deformation

- Generate new shape by deforming an existing one
  - e.g., to create animate character motion



Gao, et al. Limit Shapes – A Tool for Understanding Shape Differences and Variability in 3D Model Collections

Shape Deformation



Shape Deformation

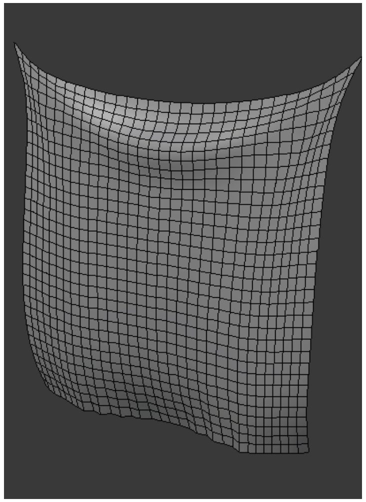
Shape → Deformation Algorithm → Deformed Shape

↑

Objective and Constraints

Objective:
Modeler Input
e.g., drag

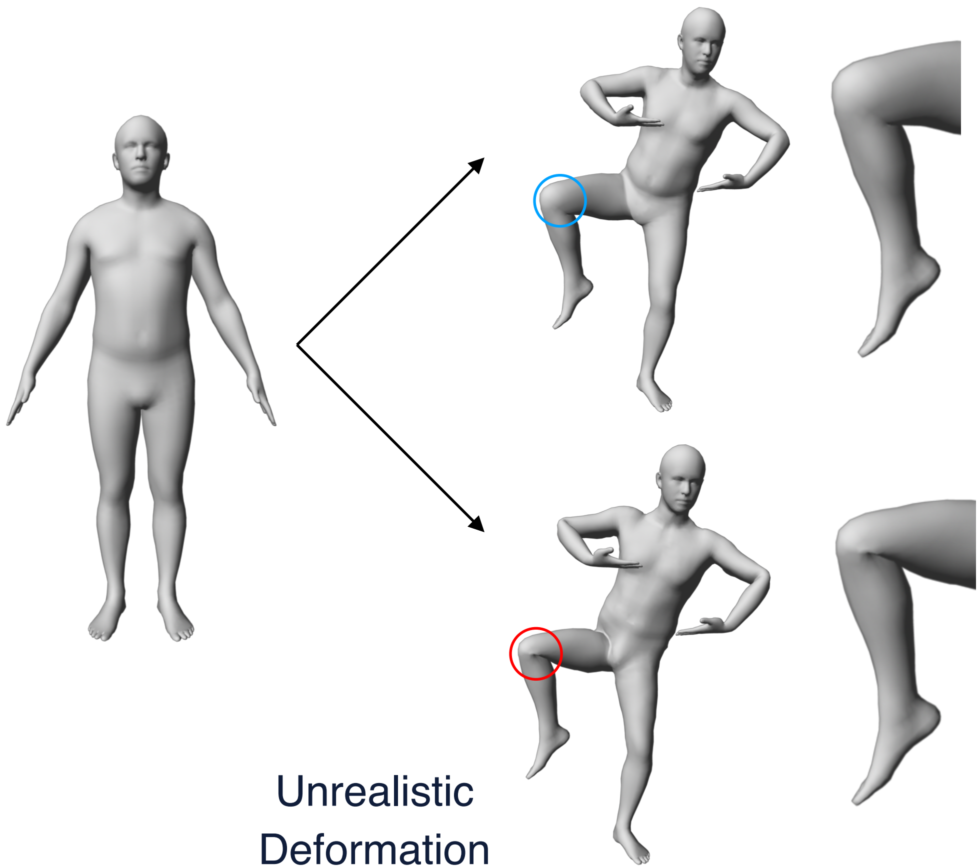
Constraints:
Geometry and Physics
e.g., gravity

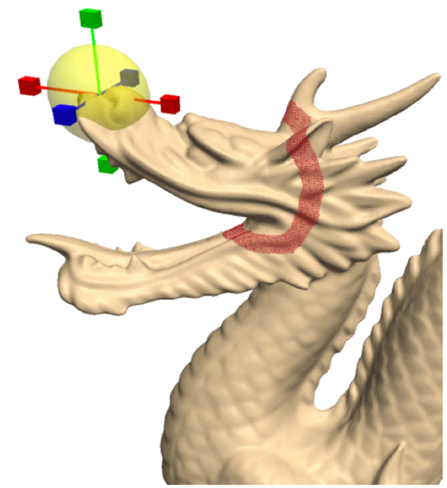


Gao, et al. ACAP: Sparse Data Driven Mesh Deformation

Preferred Deformation Algorithm

- Deformation should be natural
- Modeler works less, algorithm does more





Simply Drag
the Handle

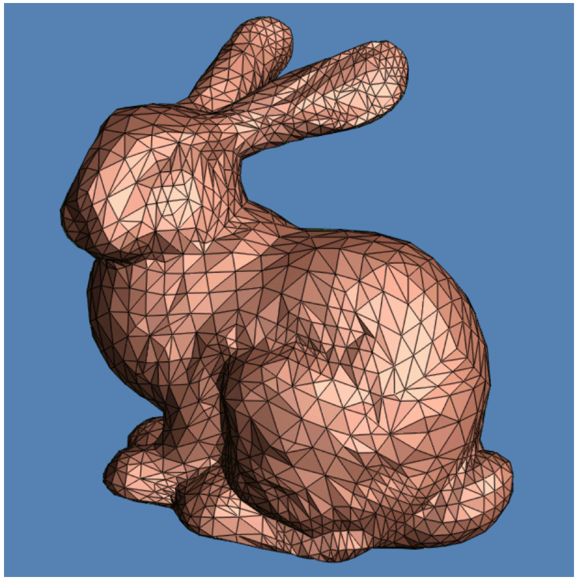
Surface Deformation

Laplacian Surface Editing

As-Rigid-As-Possible Deformation

Shape Surface Representation

- Recall Lecture 4:
  - Piece-wise Linear Surface Representation
  - E.g., triangular mesh



$$V = v_1, v_2, \ldots, v_n \subset \mathbb{R}^3$$

$$E = e_1, e_2, \ldots, e_n \subseteq V \times V$$

$$F = f_1, f_2, \ldots, f_n \subseteq V \times V \times V$$

Surface Deformation

- Deformation is only defined on the surface
- Surface deformation: $d : V \to \mathbb{R}^3$
- V is vertices of mesh

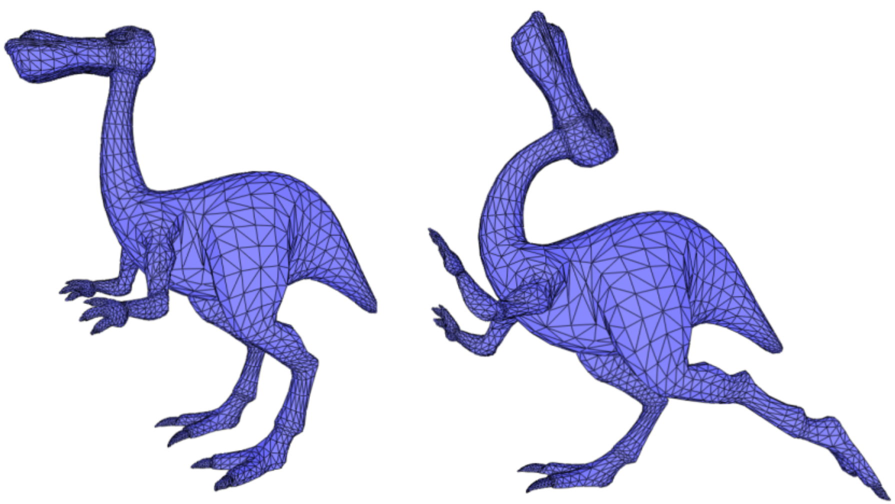
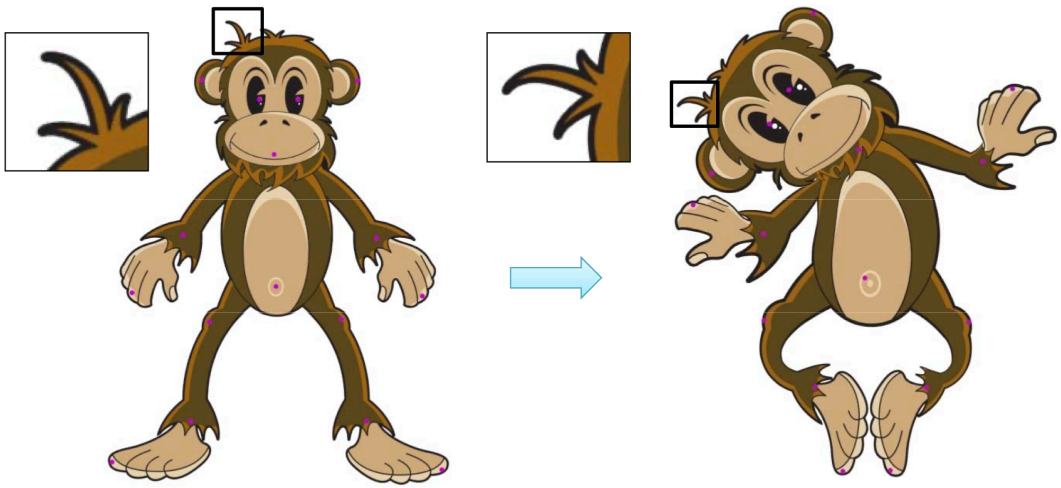


image: https://doc.cgal.org/latest/Surface_mesh_deformation/

Desired Surface Deformation

- Deformation is "natural"
  - It tries to preserve local geometry.



- Modelers do less, algorithms do more
  - E.g., given **vertex position objective** (new location of a few vertices), other points follow "naturally".

image: http://graphics.stanford.edu/courses/cs468-10-fall/LectureSlides/18_Deformation_1.pdf

How to Preserve Local Geometry?

- Recall: Curvature completely determines local surface geometry

- We want to find a “natural” deformation that preserves curvature

How to Preserve Curvature?

- Let us start with preserving mean curvature

- Recall: in HW2, Laplacian can be used to approximate mean curvatures

(b) The difference between a vertex x and the average position of its 1-ring neighborhood is a quantity that provides interesting geometric insight of the shape (see Figure 1). It can be shown that,

$$x - \frac{1}{|N(x)|} \sum_{y_i \in N(x)} y_i \approx H\vec{n}\Delta A \tag{2}$$

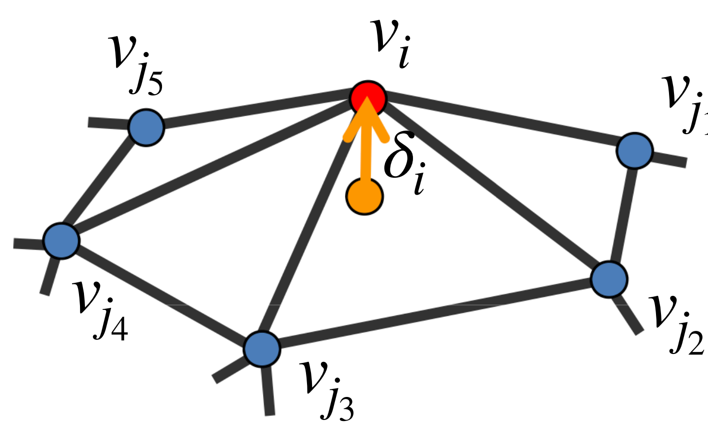
for a good mesh, where $N(x)$ is the 1-ring neighorhood vertices of x by the mesh topology, $H = \frac{1}{2}(\kappa_{min} + \kappa_{\max})$ is the mean curvature at x (in the sense of the underlying continuous surface being approximated), $\vec{n}$ is the surface normal vector at x, and ΔA is a quantity proportional to the total area of the 1-ring fan (triangles formed by x and vertices along the 1-ring).

Laplacian Coordinates

- Different from common mesh representation in global coordinates, we can represent a point relative to its neighbors

- $\delta_i = v_i - \sum_{j \in N(v_i)} \frac{1}{d_i} v_j$

- d_i: degree of vertex i



Sorkine, et al. Laplacian Surface Editing

Laplacian Coordinates

- Recall:
  Laplacian matrix: $L = D - A$
  D is degree matrix and A is adjacency matrix

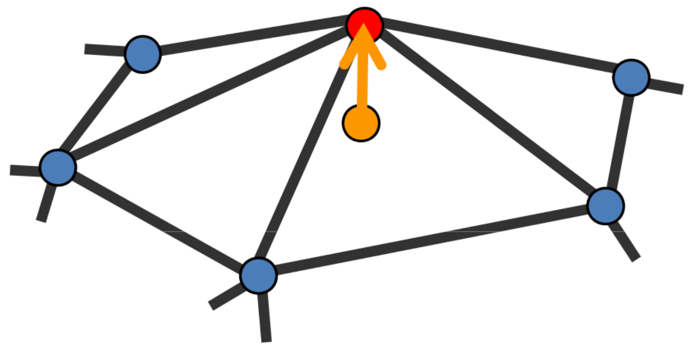
- Differential coordinates can be computed by a **normalized laplacian matrix**

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

$$\delta = (I - D^{-1}A)V = (D^{-1}L)V$$

- V is a nx3 matrix denotes vertices position

Laplacian Coordinates Property



- Direction of δ_i approximates the **normal direction**

- Size of the δ_i approximates the **mean curvature**

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

- Note: mean curvature cannot fully determine local geometry. 2 numbers are needed

Deform by Laplacian Coordinates

- Input: vertex (control point) position objective

- Consider a simple objective of moving several vertices towards the new location: $v_i' = u_i$, where v' is vertex after deformation.
- Energy function:

$\mathscr{L}(v_i')$ is laplacian coordinates of v_i'

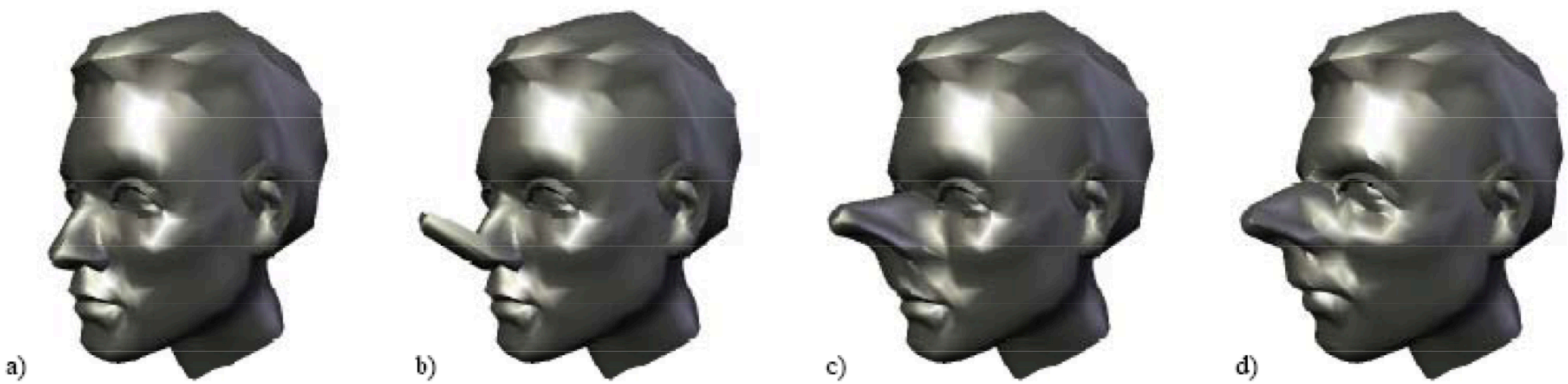
$$E(V') = \sum_{i=1}^{n} \|\delta_i - \mathscr{L}(v_i')\|^2 + \sum_{i=m}^{n} \|v_i' - u_i\|^2$$

- δ_i: Laplacian Coordinate of Original Mesh
- $\mathscr{L}(v_i')$: Laplacian Coordinate of Deformed Mesh
- $\|v_i' - u_i\|^2$: Objective: Control Point

Deform by Laplacian Coordinates

- Deformed shape can be solved by minimizing $E(V')$

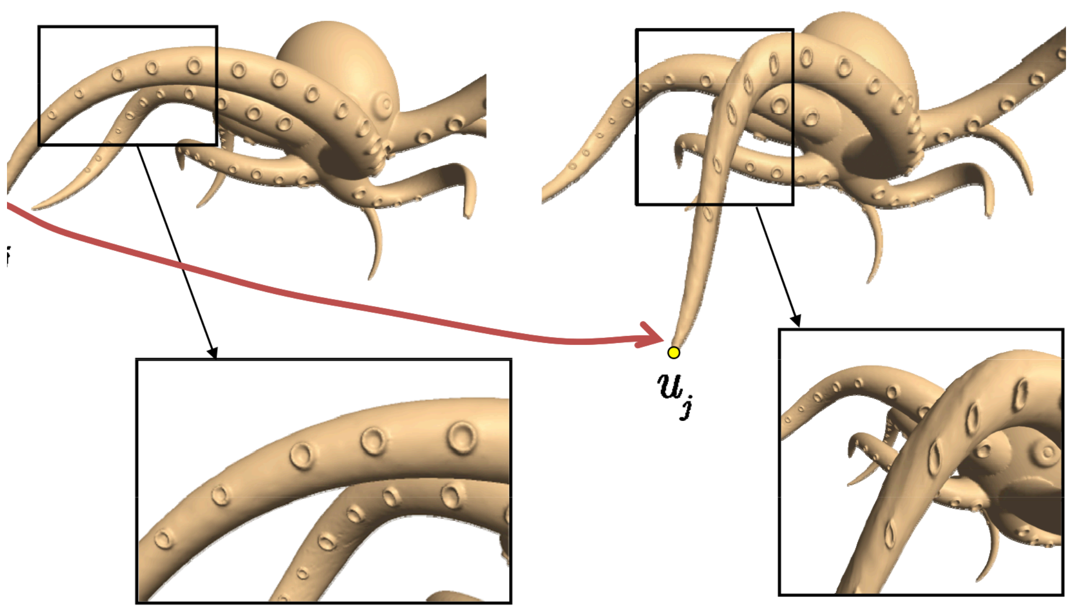
$$E(V') = \sum_{i=1}^{n} \|\delta_i - \mathscr{L}(v_i')\|^2 + \sum_{i=m}^{n} \|v_i' - u_i\|^2$$



$E(V')$ decreases by iterations

Issues?

- Other than preserving the mean curvature, $\sum_{i=1}^{n} \|\delta_i - \mathcal{L}(v_i')\|^2$ also have tried to preserve normal.
- However, normal preservation is undesired
- How can we cancel the effect of normal preservation?



Laplacian Coordinates Under Transformation

- Normals are invariant under translation, so

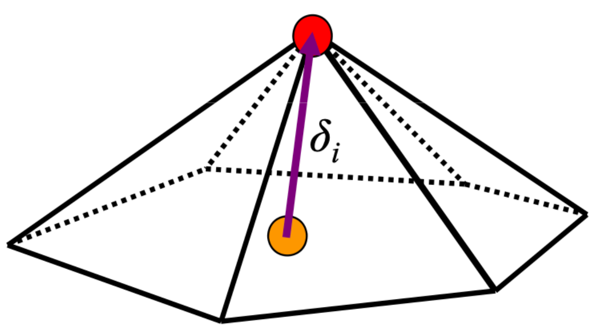
$$\mathscr{L}(v_i) = \mathscr{L}(v_i + t)$$

Laplacian Coordinates Under Transformation

- Normals are invariant under translation, so

$$\mathscr{L}(v_i) = \mathscr{L}(v_i + t)$$

- However, normal changes under rotation, so Laplacian coordinates change under rotation

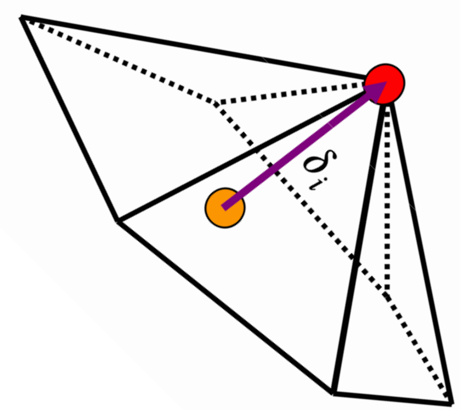


Laplacian Coordinates Under Transformation

- Normals are invariant under translation, so

$$\mathscr{L}(v_i) = \mathscr{L}(v_i + t)$$

- However, normal changes under rotation, so Laplacian coordinates change under rotation



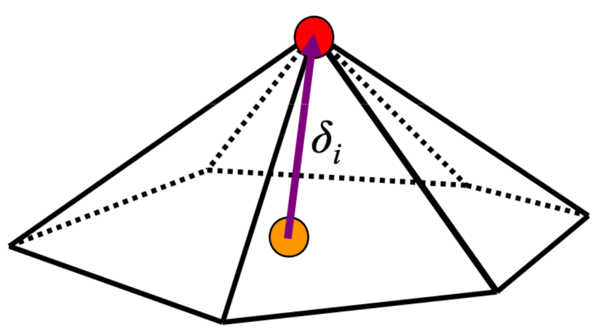
Laplacian Coordinates Under Transformation

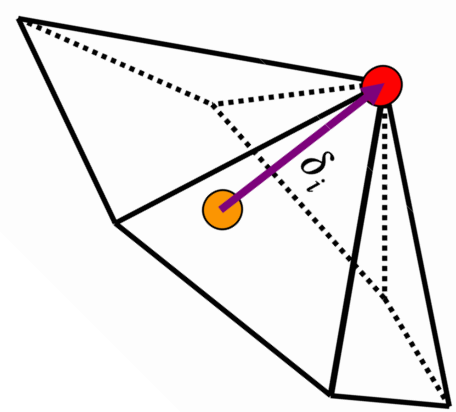
- Normals are invariant under translation, so

$$\mathscr{L}(v_i) = \mathscr{L}(v_i + t)$$

- However, normal changes under rotation, so Laplacian coordinates change under rotation

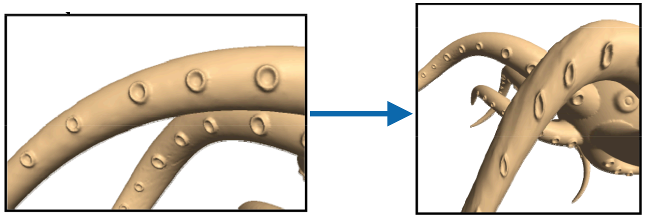
$$R\mathscr{L}(v_i) = \mathscr{L}(Rv_i)$$





Solution

- After deformation, assuming that the local region of the surface will rotate



- Original optimization target:

$$E(V') = \sum_{i=1}^{n} \|\delta_i - \mathscr{L}(v_i')\|^2 + \sum_{i=m}^{n} \|v_i' - u_i\|^2$$

- New optimization target by introducing a variable to cancel the rotation:

$$E(V') = \min_{\{R_i\}} \sum_{i=1}^{n} \|{\color{red}R_i}\delta_i - \mathscr{L}(v_i')\|^2 + \sum_{i=m}^{n} \|v_i' - u_i\|^2$$

Alternating Optimization

We optimize vertex V and rotation R iteratively

1. Estimate rotation R from the deformed shape

$$\min_{V'} E(V') = \sum_{i=1}^{n} \|R_i \delta_i - \mathscr{L}(v_i')\|^2 + \sum_{i=m}^{n} \|v_i' - u_i\|^2$$

2. Estimate shape V' given rotation

$$\min_{R_i}(\|R_i \delta_i - \mathscr{L}(v_i')\|^2 + \sum_{j \in N(v_i)} \|R_i v_j - v_j'\|^2)$$

Numerical Method

- Known $\{R_i\}$ to get V': Quadratic optimization with a closed-form solution

$$\min_{V'} E(V') = \sum_{i=1}^{n} \|R_i\delta_i - \mathscr{L}(v_i')\|^2 + \sum_{i=m}^{n} \|v_i' - u_i\|^2$$

- Known V' to get $\{R_i\}$: Quadratic optimization with a constraint on R in $SO(3)$

$$\min_{R_i} \; \|R_i v_i - v_i'\|^2 + \sum_{j \in N(v_i)} \|R_i v_j - v_j'\|^2$$

$$R_i R_i^T = I, \det(R) = 1$$

- Recall: Orthogonal Procrustes Problem in Lecture 11!

Other Issues?

- Only mean curvature is considered
  - Full curvature (2 numbers) is required to fully determine the local geometry.
- Solution to the issue
  - Deformation energy should consider both mean curvature and Gaussian curvature (geodesic distance preservation)

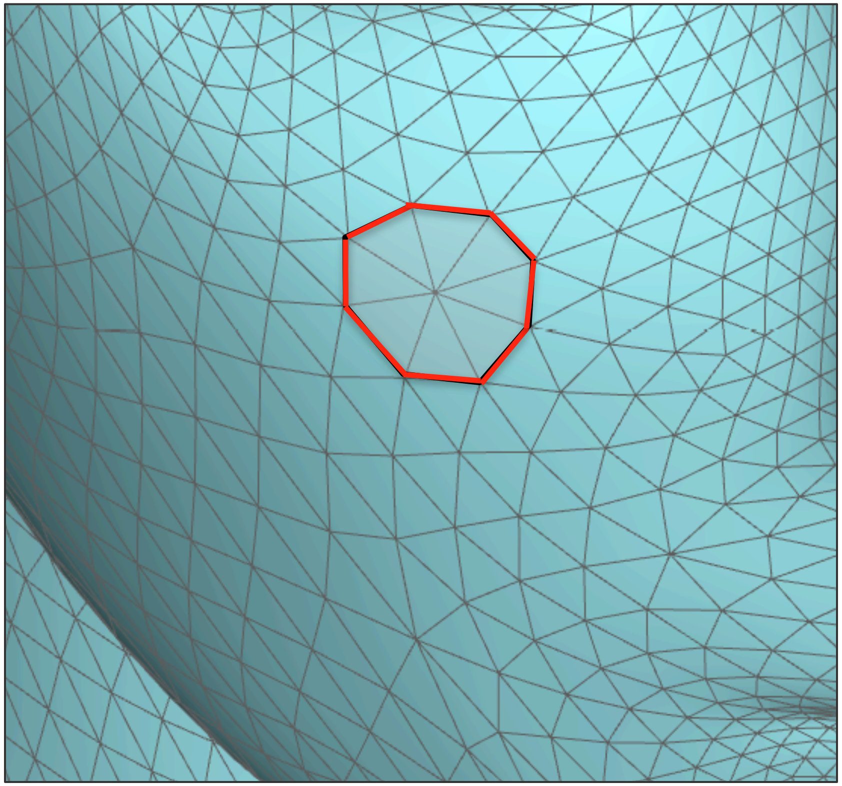
Surface Deformation

Laplacian Surface Editing

As-Rigid-As-Possible Deformation

Local Deformation

- Let's look at a local region centered at a vertex (called a cell).

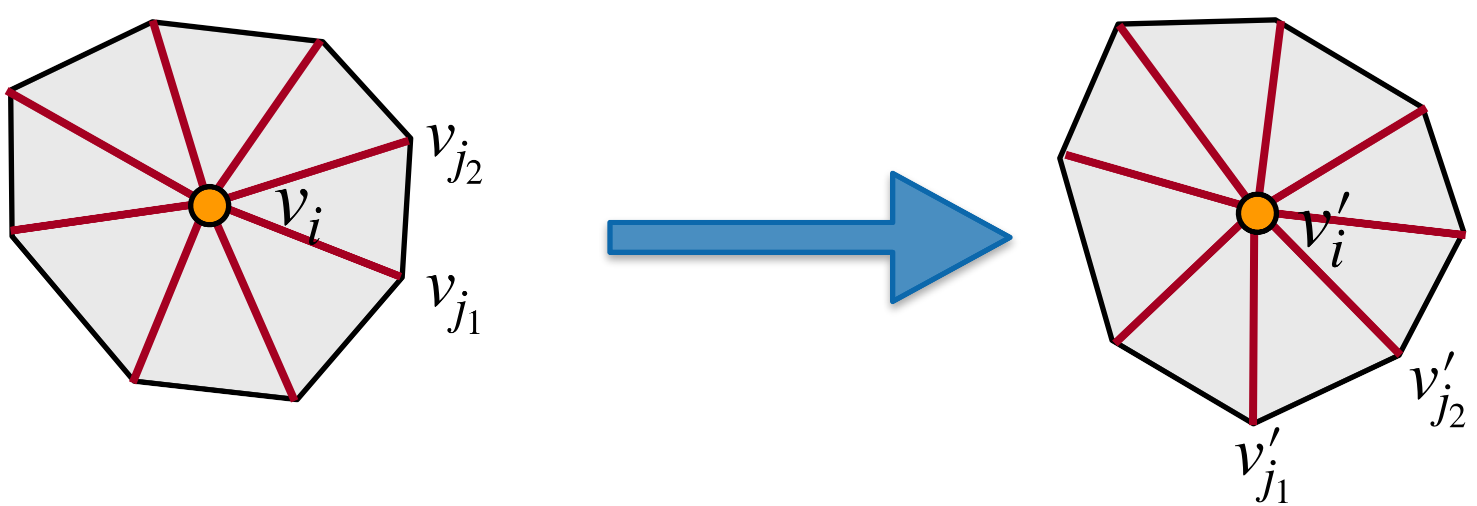


How do we define a better deformation energy of this cell?

Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Desired Property for Deformation Energy

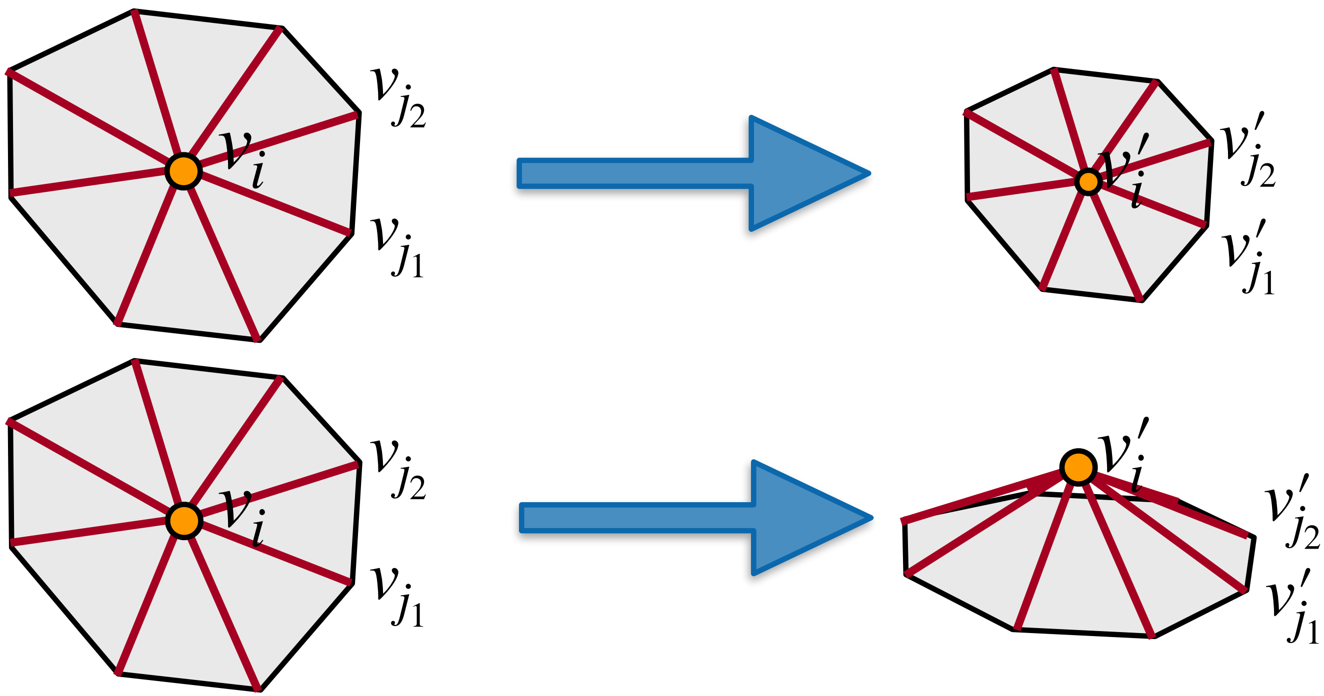
- Translation and rotation should not change the deformation energy.



Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Desired Property for Deformation Energy

- Translation and rotation should not change the deformation energy.
- **Stretching** (length change) and **bending** (angle change) increase deformation energy.



Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Local Deformation Energy

- Translation and rotation should not change the deformation energy.
- **Stretching** (length change) and **bending** (angle change) increase deformation energy.

$$E(v_i) = \min_{R_i} \sum_{j \in N(i)} \|(v_i' - v_j') - R_i(v_i - v_j)\|^2$$

Local Deformation Energy

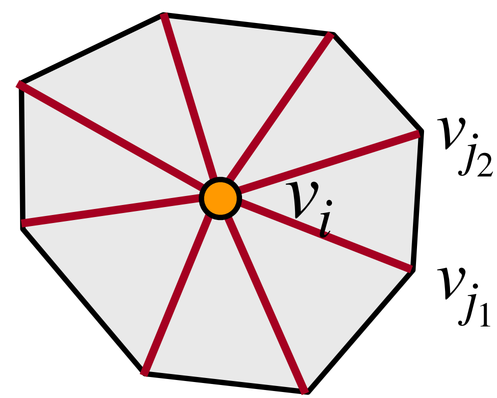
Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Local Deformation Energy

- Translation and rotation should not change the deformation energy.
- **Stretching** (length change) and **bending** (angle change) increase deformation energy.

Minimum over all rotations, rotation-invariant

$$E(v_i) = \min_{R_i} \sum_{j \in N(i)} \|(v_i' - v_j') - R_i(v_i - v_j)\|^2$$



Relative to the cell center (v_i or v_i'), translation-invariant

Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Local Deformation Energy

- Translation and rotation should not change the deformation energy.
- **Stretching** (length change) and **bending** (angle change) increase deformation energy.

Penalize change of length

$$E(v_i) = \min_{R_i} \sum_{j \in N(i)} \|(v_i' - v_j') - R_i(v_i - v_j)\|^2$$

R_i is shared by the cell, penalize change of angle

Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Local Deformation Energy

$$E(v_i) = \min_{R_i} \sum_{j \in N(i)} \|(v_i' - v_j') - R_i(v_i - v_j)\|^2$$

- It is (again) an Orthogonal Procrustes Problem!

Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Total Deformation Energy

- Sum up the local deformation energy over all vertices

$$E(V') = \min_{v'} \sum_{i=1}^{n} \min_{R_i} \sum_{j \in N(i)} \|(v_i' - v_j') - R_i(v_i - v_j)\|^2$$

$$s.t. \quad v_j' = c_j, j \in C$$

C: the set of control point indices

- Minimizing total deformation energy
  - As-Rigid-As-Possible deformation (ARAP deformation)

Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Total Deformation Energy

- Alternating optimization
  - Given initial guess v'_0, find optimal rotations R_i.
    - This is a per-cell task! We already showed how to estimate R_i when v, v' are known
  - Given the R_i (fixed), minimize the energy by finding new v'

$$E(V') = \min_{v'} \sum_{i=1}^{n} \sum_{j \in N(i)} \|(v'_i - v'_j) - R_i(v_i - v_j)\|^2$$

Linear Least Square

Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Total Deformation Energy

- Alternating optimization
  - Given initial guess v'_0, find optimal rotations R_i.
    - This is a per-cell task! We already showed how to estimate R_i when v, v' are known
  - Given the R_i (fixed), minimize the energy by finding new v'

$$E(V') = \min_{v'} \sum_{i=1}^{n} \sum_{j \in N(i)} \|(v'_i - v'_j) - R_i(v_i - v_j)\|^2$$

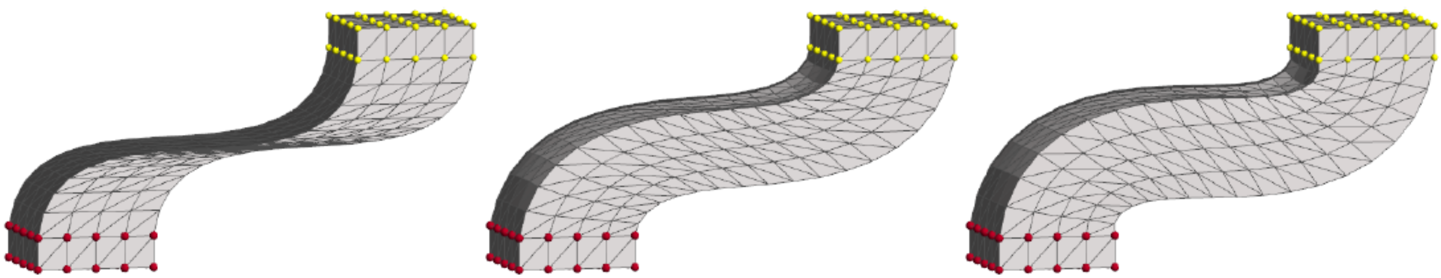
⇩ use Laplacian

$$\min_{V'} \|\mathscr{L} V' - b\|^2$$

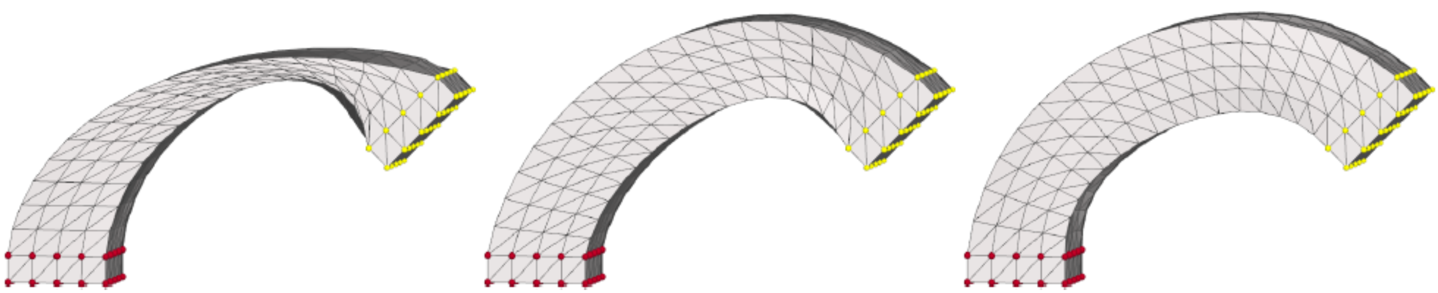
Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Initialization

Start from naïve Laplacian editing as initial guess and iterate



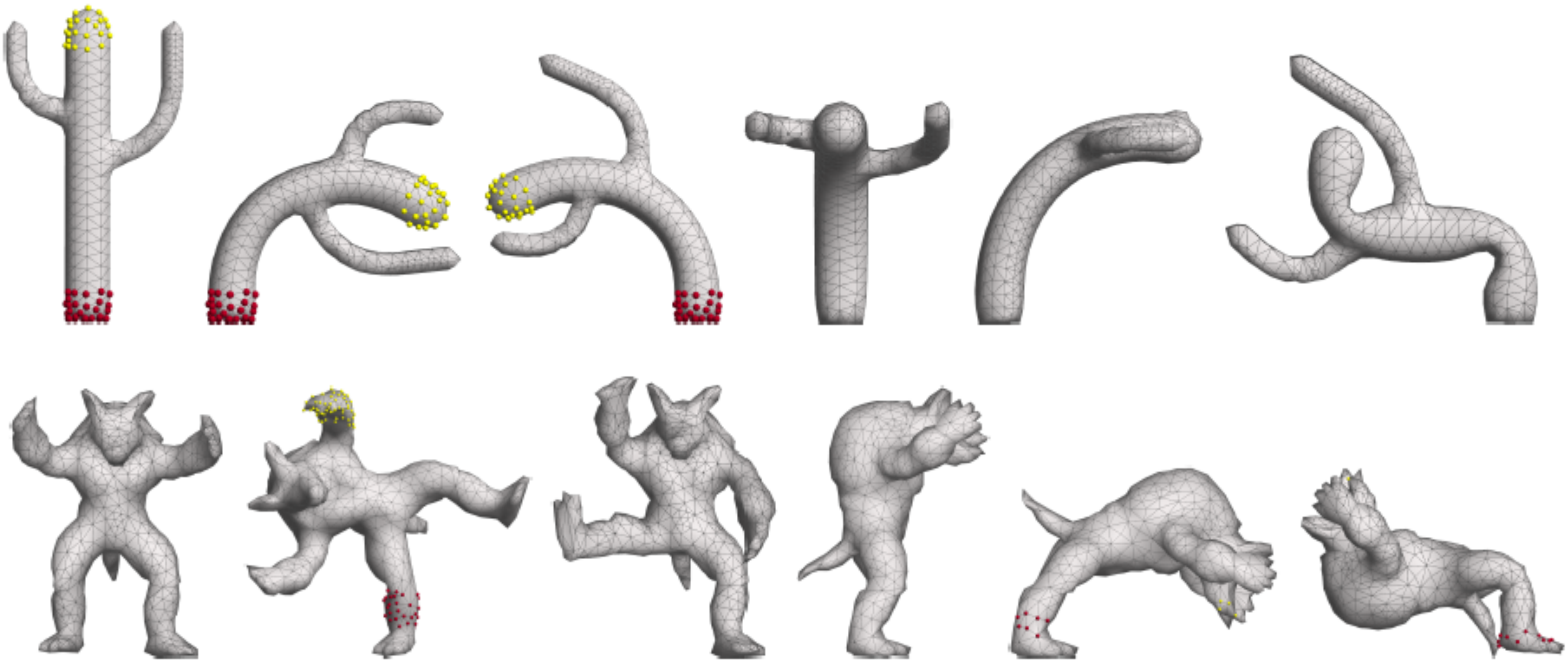
initial guess 1 iteration 2 iterations



initial guess 1 iterations 4 iterations

Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Examples



Sorkine, et al. As-Rigid-As-Possible Surface Modeling

Summary

- As-rigid-as-possible deformation iteratively minimize the deformation energy.
- The deformation energy penalizes both mean curvature change and length change.

Further Comments

- Iterative algorithm, slow on large meshes.
- Guaranteed to converge (energy is bounded and monotonically decreasing for each iteration)
- The idea can generalize to other energy definition or 3D volume deformation (real physical deformation)

Space Deformation

a.k.a. Free-Form Deformation

Surface vs Space Deformation

- Previously: surface deformation
  - Move vertices of the mesh

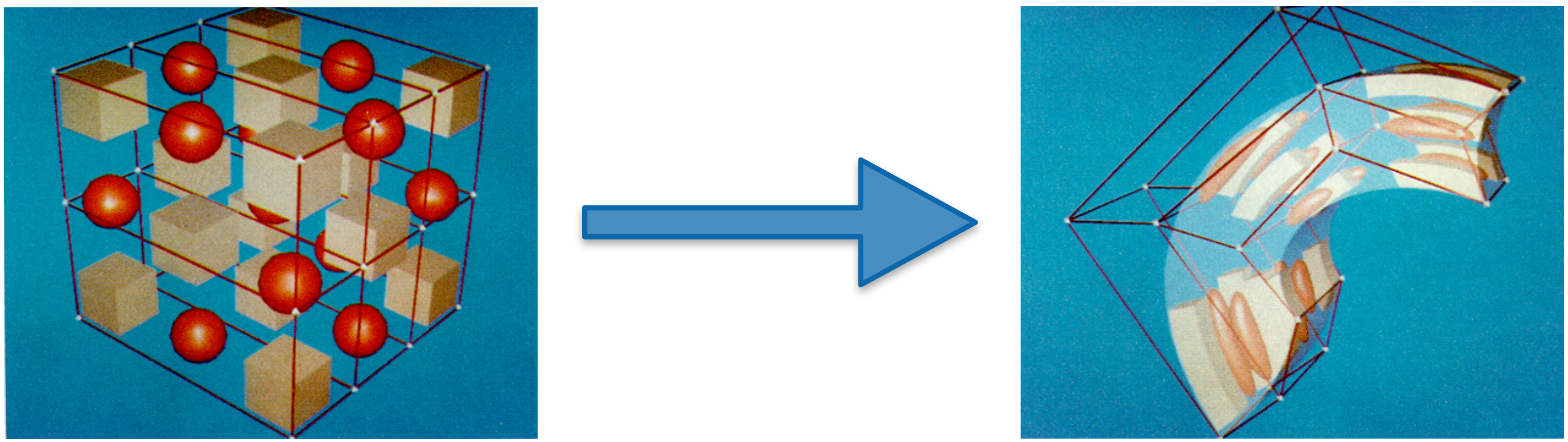
- Space deformation
  - Define a function that warps the $\mathbb{R}^3$ space.

$$f : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

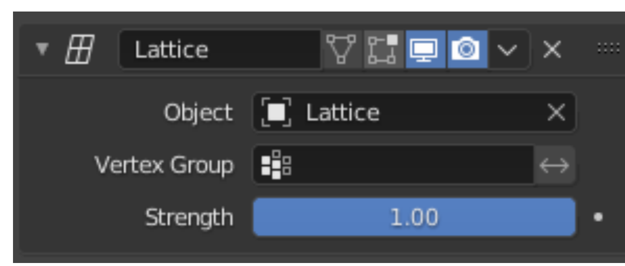
  - Evaluate the space deformation on mesh vertices to deform the mesh.

Free-Form Deformation

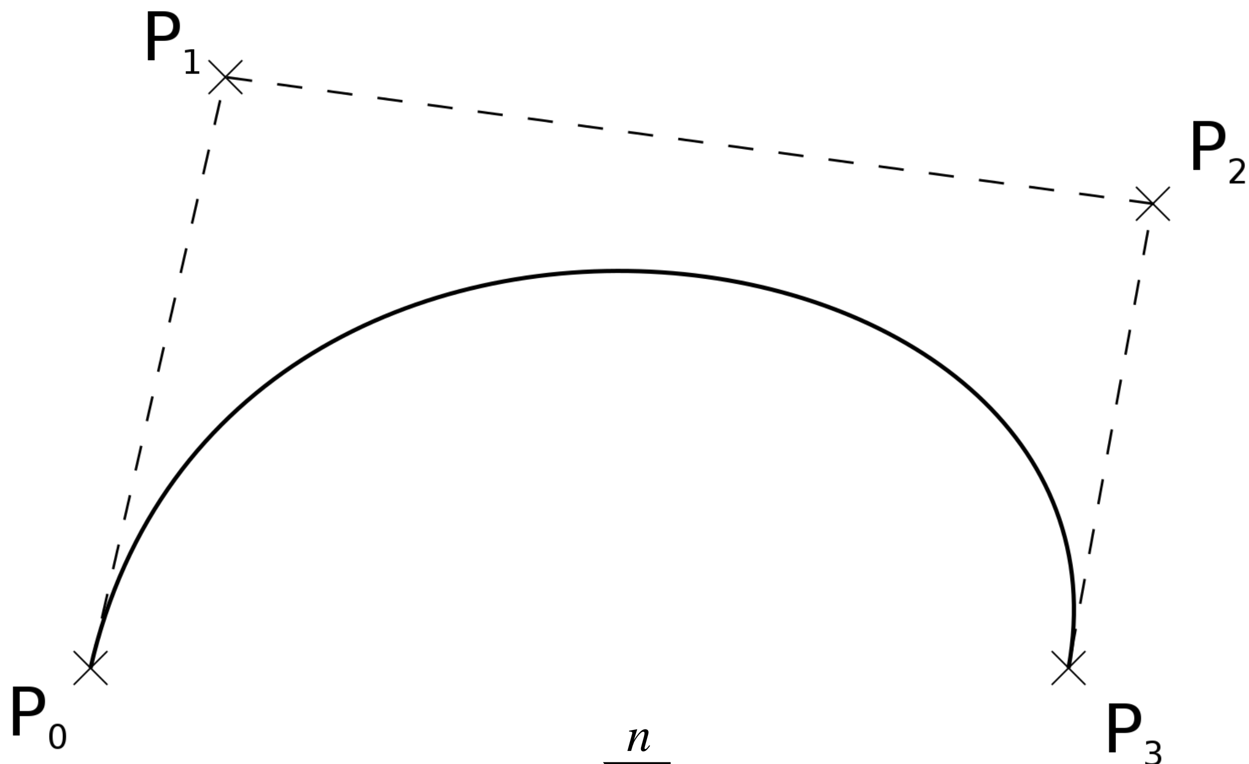
- Free-Form Deformation (Sederberg & Parry, 1986)



- Still widely used today
  - e.g. Blender Lattice modifier



Recall: Bezier Curve from Lecture 1



$$s(t) = \sum_{i=0}^{n} p_i B_i^n(t)$$

3D Free-Form Deformation

- Control points: 3D lattice

- Modelers drag the vertices of the lattice to define displacements d_i.

- Displacements of points in space are computed by interpolating d_i with interpolating weights B_i

$$d(x) = \sum_i B_i(x) d_i$$

3D Free-Form Deformation

$$d(x) = \sum_i B_i(x) d_i$$

- Compute the Bezier parameters in each dimension and apply **tricubic** interpolation.

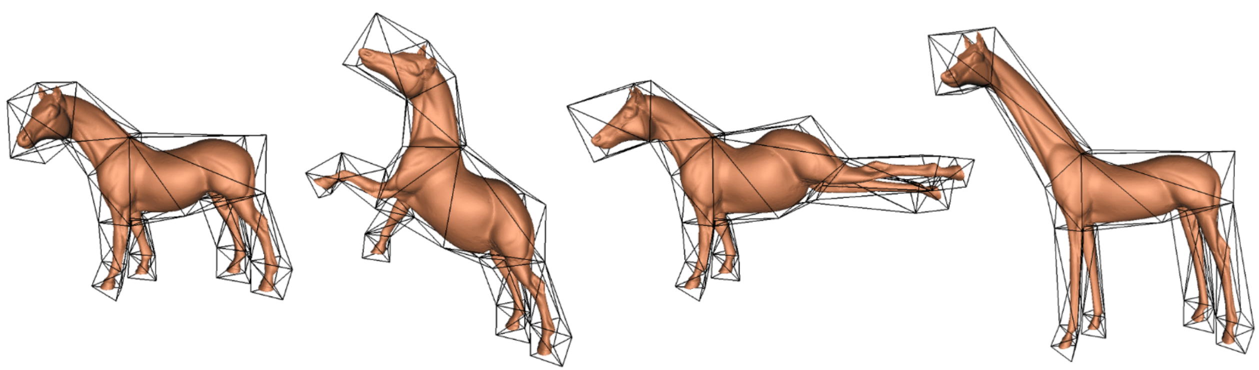
$$d(x, y, z) = \sum_i \sum_j \sum_k B_i(x) B_j(y) B_k(z) \mathbf{d}_{ijk}$$

Issues

- Lattice can be large. Modelers do too much: move control points one by one by hand.

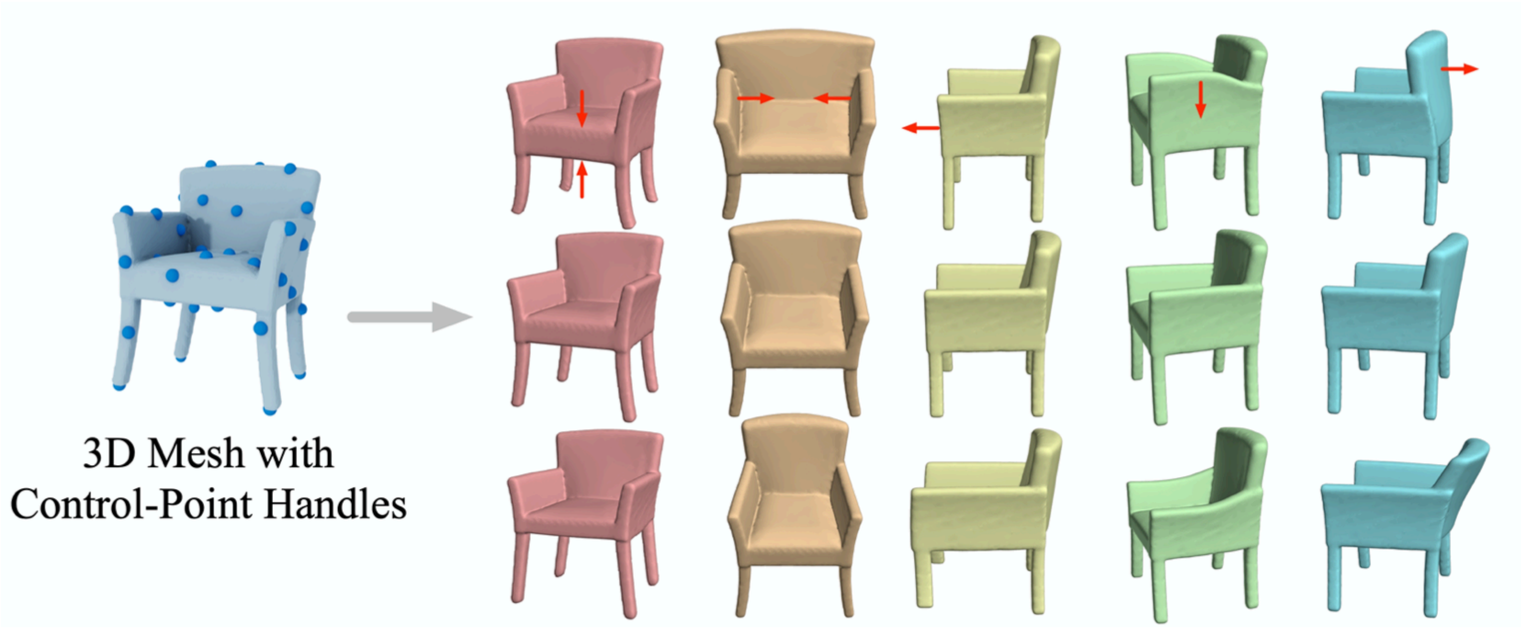
- Like Bezier curves, not easy to intuitively relate position of control points with the geometry.

- There are approaches using fewer point points, e.g., cage deformation, key-point based deformation



Learning-based Deformation Field by Keypoints

- Use keypoints as control points
- Use network to learn a basis function from data!



Liu, et al. DeepMetaHandles: Learning Deformation Meta-Handles of 3D Meshes with Biharmonic Coordinates

Summary

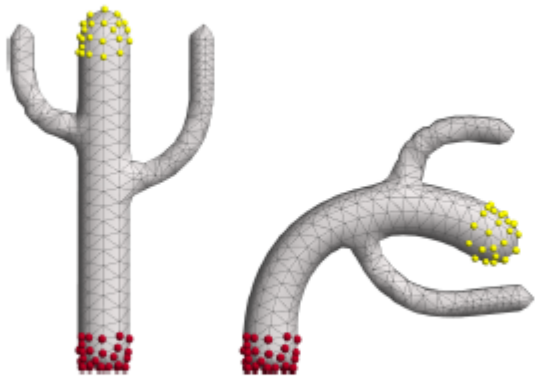
- Space deformation are typically very fast
- Run in real time
- Widely used in real-time animation

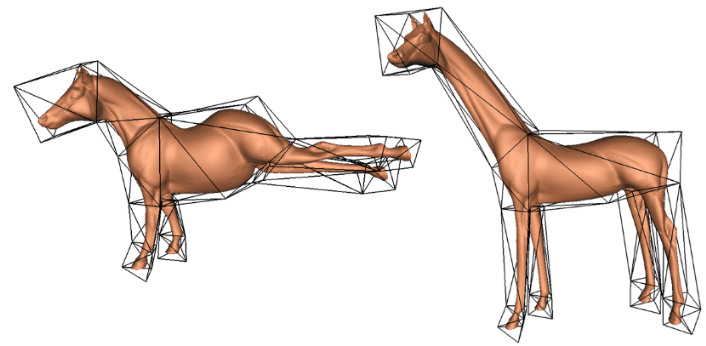
Skeleton Skinning

Linear Blend Skinning

Read yourself

Boneless Shape Editing





Surface Deformation

Space Deformation

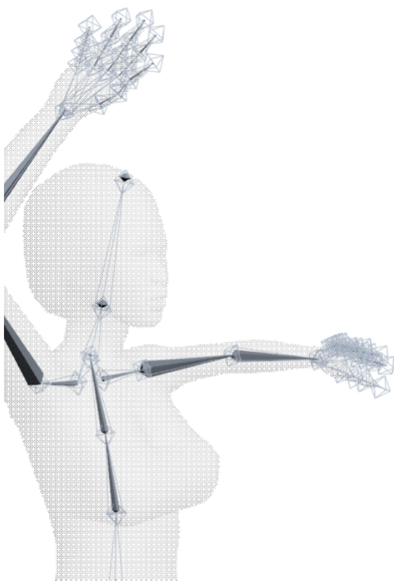
- Pro: Automatically preserve curvatures
- Con: Slow

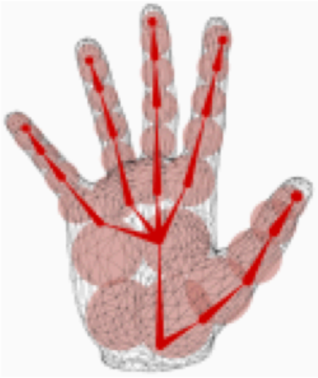
- Pro: Fast
- Con: Need artists to tune control point movements to achieve curvature preservation

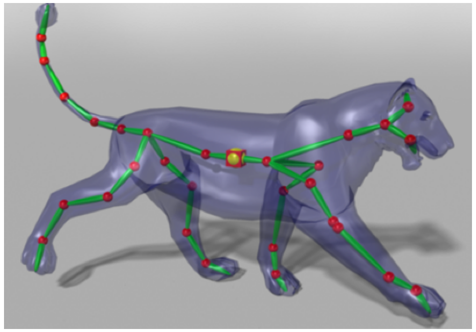
Read yourself

Deformation for Objects with Bones

- Many objects have “bones” — deformation may be interpreted as
  - coarse-level bone transformation; and
  - fine-level skin transformation







Kavan, et al. Direct Skinning Methods and Deformation Primitives
Romero, et al. Embodied Hands: Modeling and Capturing Hands and Bodies Together
Le, et al. Robust and Accurate Skeletal Rigging from Mesh Sequences

Read yourself

Skeleton

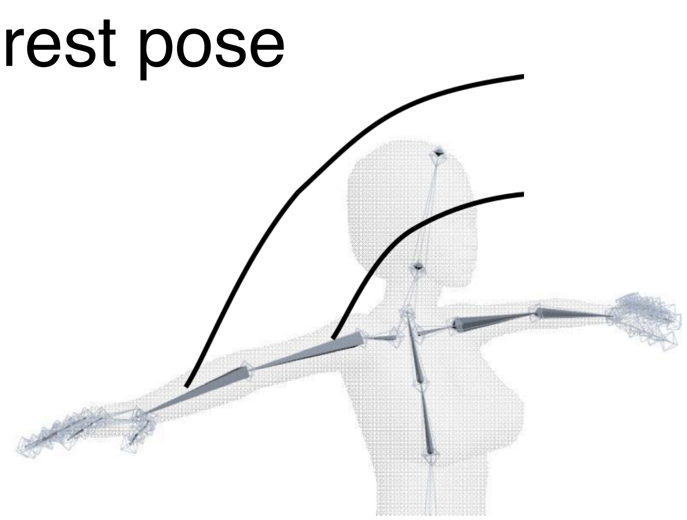
- Skeleton: bones of body linked together
- The pose of bones can be represented using a set of matrices $T_i \in SE(3)$ from current pose to rest pose

Read yourself

Kavan, et al. Direct Skinning Methods and Deformation Primitives

Skeleton

- Skeleton: bones of body linked together
- The pose of bones can be represented using a set of matrices $T_i \in SE(3)$ from current pose to rest pose

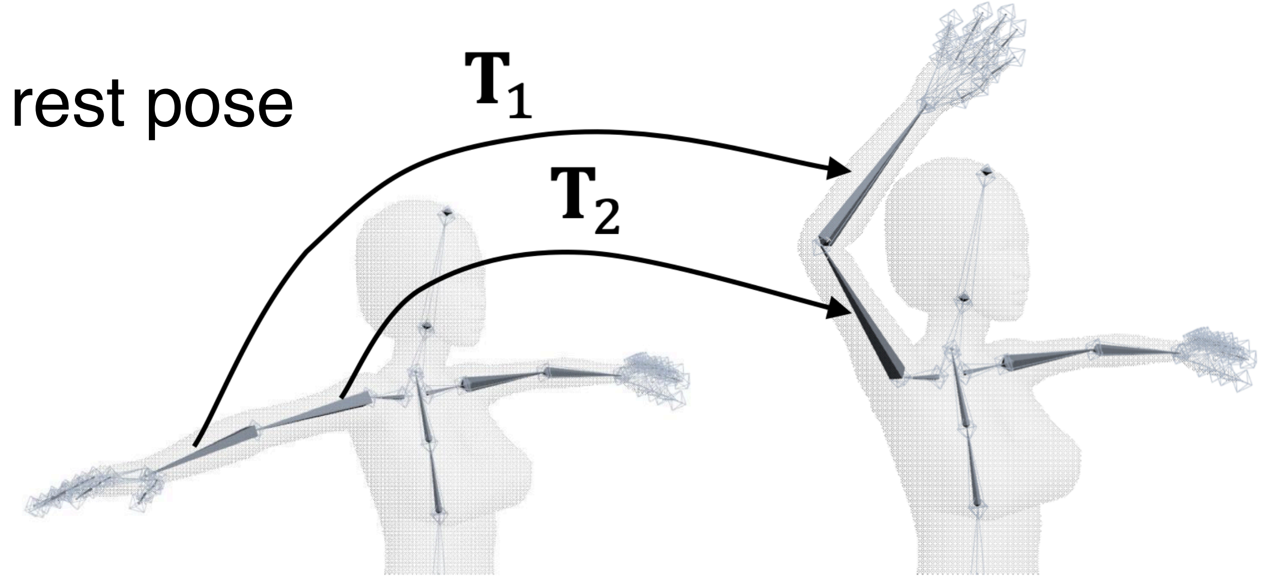


Read yourself

Kavan, et al. Direct Skinning Methods and Deformation Primitives

Skeleton

- Skeleton: bones of body linked together
- The pose of bones can be represented using a set of matrices $T_i \in SE(3)$ from current pose to rest pose

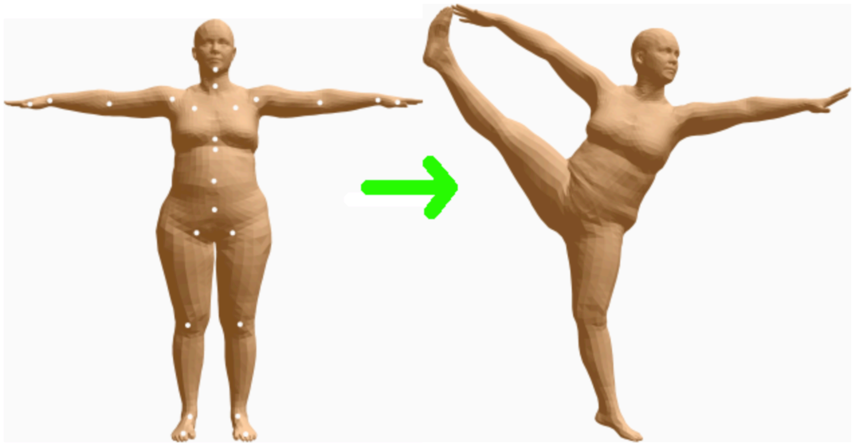


Read yourself

Kavan, et al. Direct Skinning Methods and Deformation Primitives

Skinning

- The surface of body **deforms** as the skeletons are transformed rigidly



Read yourself

Kavan, et al. Direct Skinning Methods and Deformation Primitives

Linear Blend Skinning

- Skin vertex move when pose of bone T_j change
  - If v_i on the j-th bone, then it will move to $T_j v_i$

- Around joints there will be cracks. In practice, each vertex is governed by multiple bones,
  - e.g., averaged by a **linear** model (SMPL):

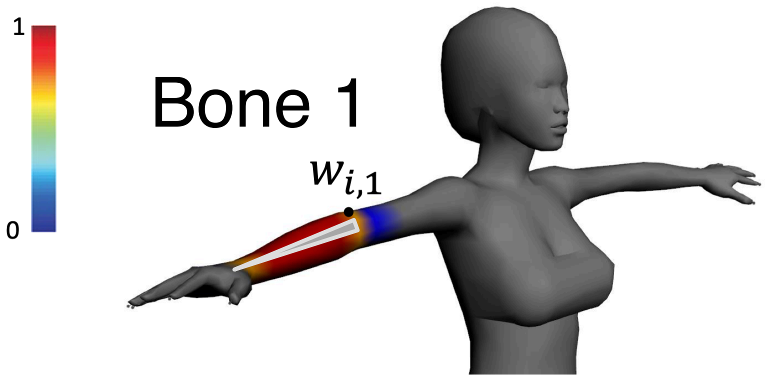
$$v_i' = \sum_{j=1}^{m} w_{i,j} T_j v_i = (\sum_{j=1}^{m} w_{i,j} T_j) v_i$$

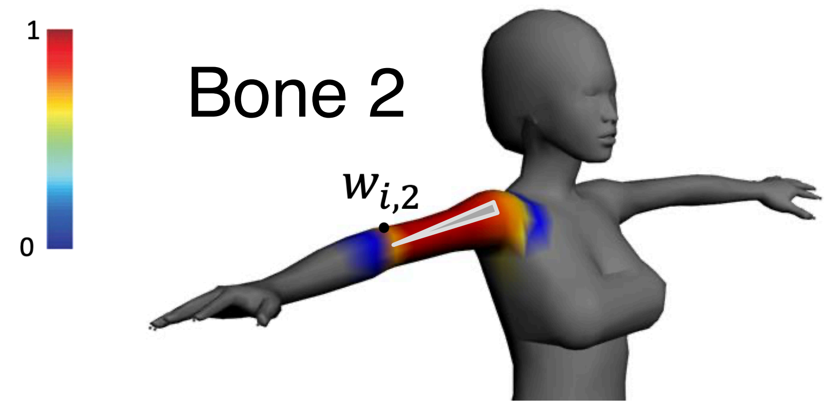
- $w_{i,j}$: skinning weights
  - Describes the amount of influence of bone j on vertex i

Read yourself

Skinning Weights

- We commonly require $w_{i,j} \geq 0, \sum_{j=0}^{m} w_{i,j} = 1$



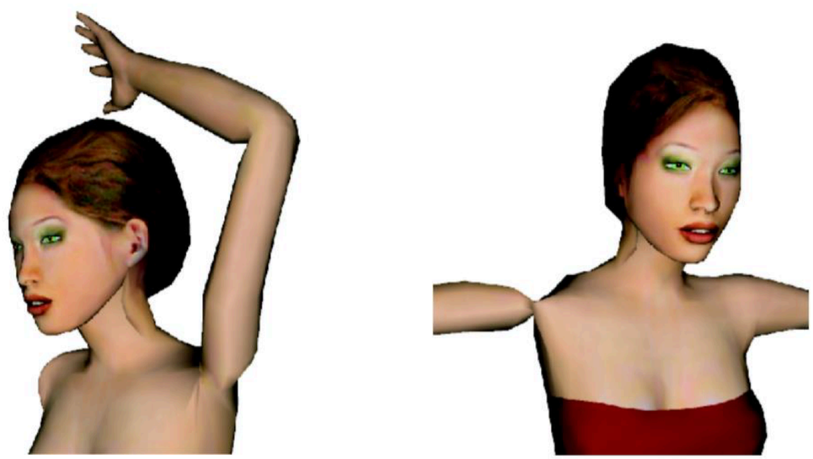


Kavan, et al. Direct Skinning Methods and Deformation Primitives

Read yourself

Limitations I

- Linear combination of transformations is simple
- However, note that rotation matrices are not in a linear space



Candy-wrapper artifacts

Image: Kavan, et al. Geometric Skinning with Approximate Dual Quaternion Blending

Read yourself

Limitations I

- Linear combination of transformations is simple
- However, note that rotation matrices are not in a linear space

How to address the issue?

- Use quaternion and some tricks to achieve linear interpolation of rotations
- SLERP: **S**pherical **L**inear Int**ERP**olation (https://en.wikipedia.org/wiki/Slerp)

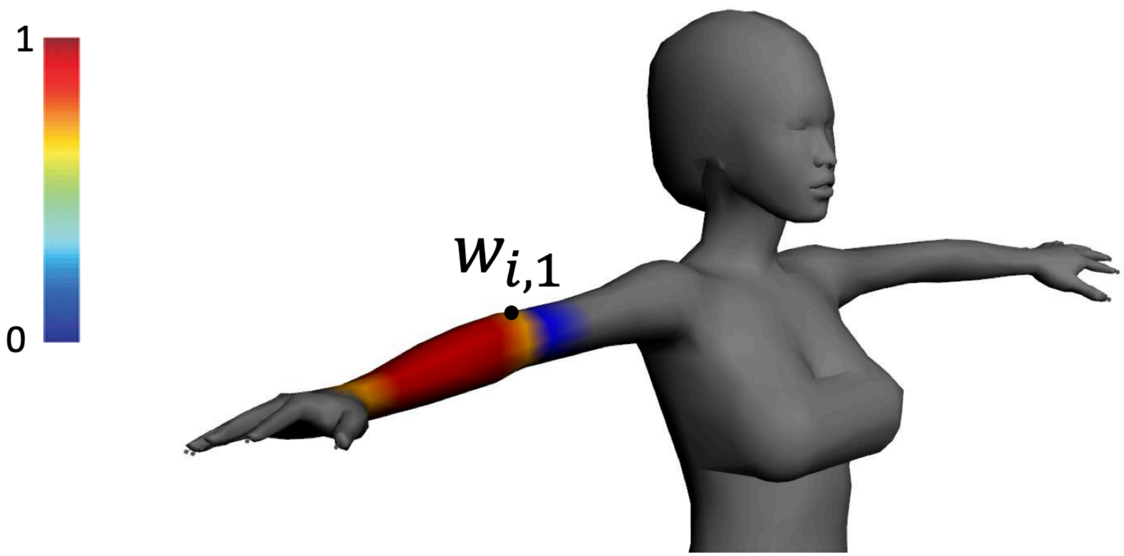
Read yourself

Image: Kavan, et al. Geometric Skinning with Approximate Dual Quaternion Blending

Limitations II

- Modelers do too much: Assigning skinning weights is cumbersome

- Can we learn weights from data? Next lecture!



Kavan, et al. Direct Skinning Methods and Deformation Primitives

Read yourself

Summary

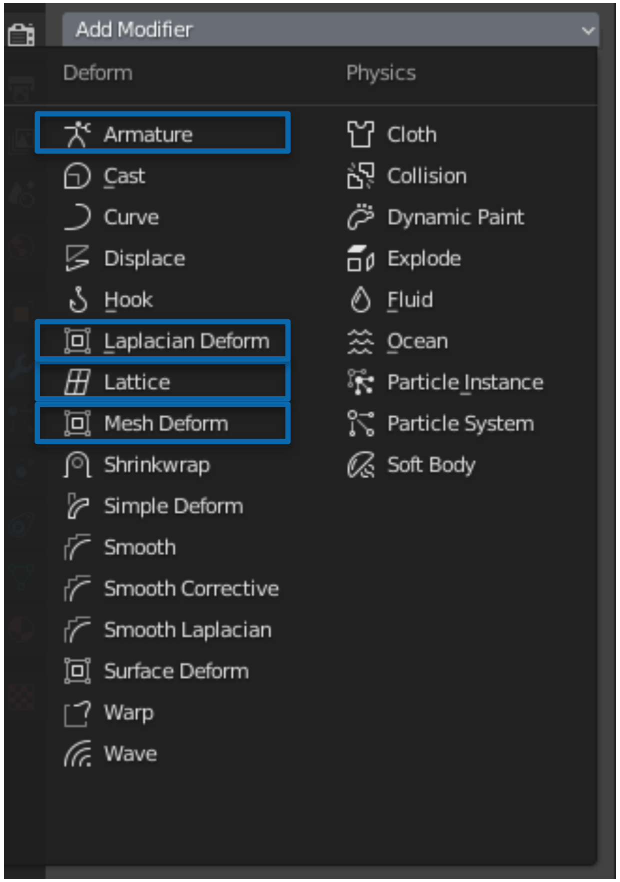
- Skeleton: linked bones
- Skinning: deform the surface along skeleton transformation
- Linear Blend Skinning:
  - Rest pose
  - Bone transformation
  - Skinning weight

Read yourself

Deformation in Blender

- You can find these deformation algorithms in blender. Try it yourself!

- There are a lot more to play with!



Read yourself