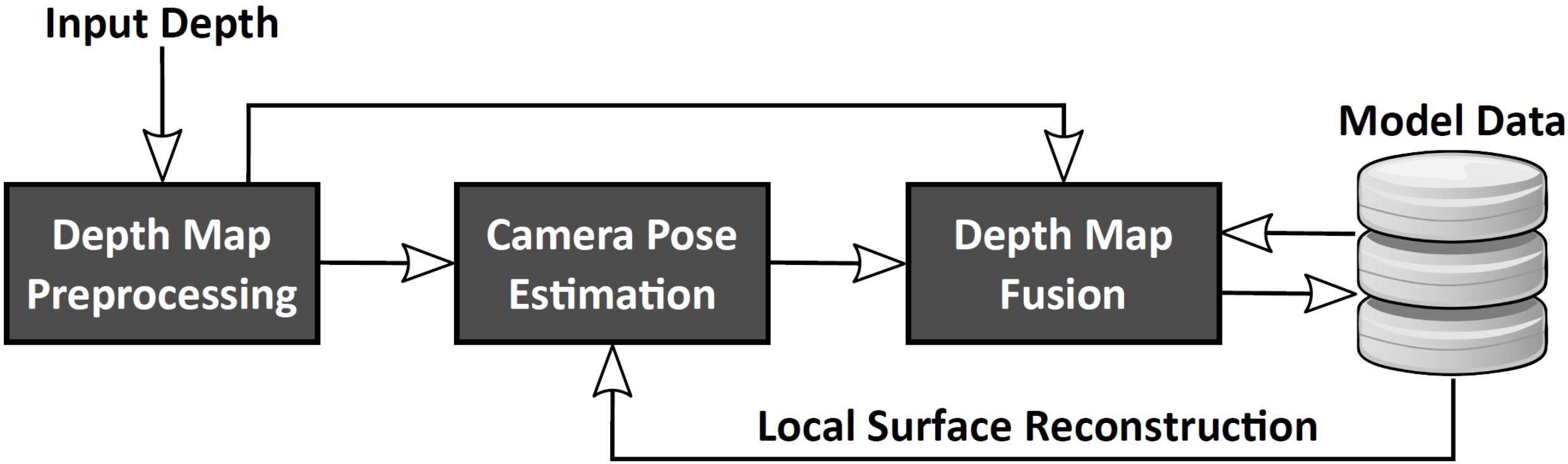
3D Scanning & Motion Capture

Deformation and Non-rigid Surface Tracking

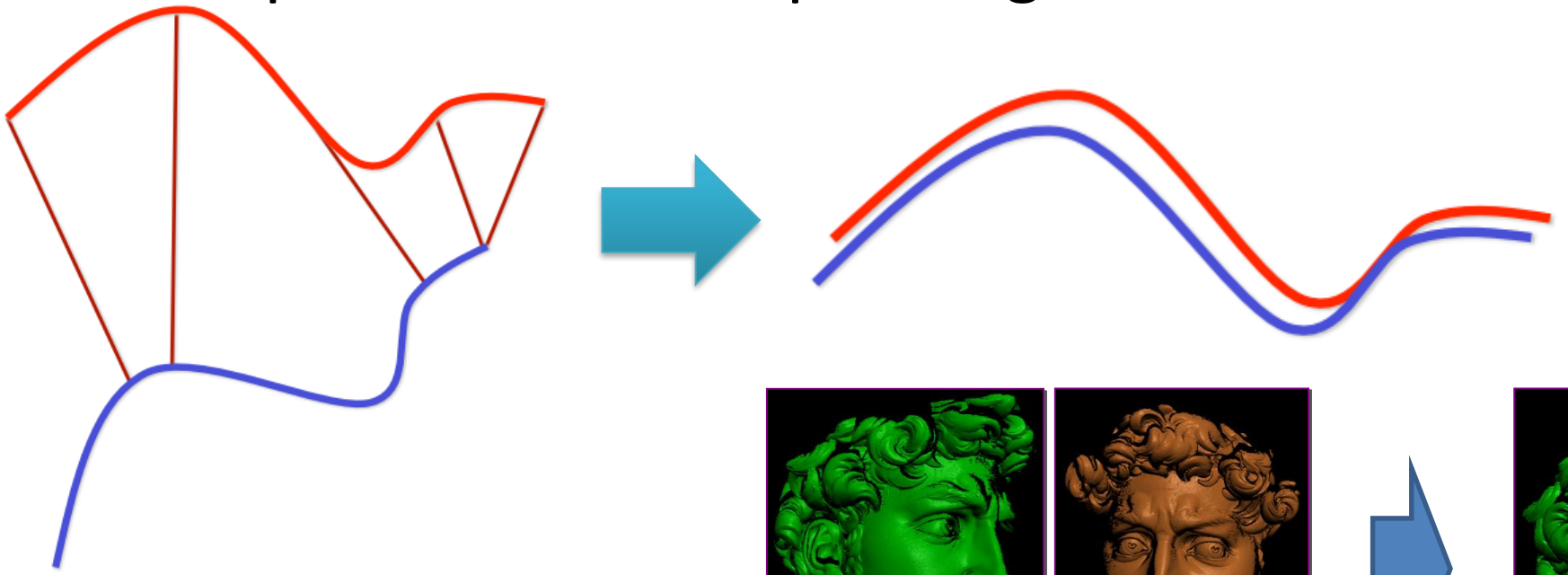
Prof. Matthias Nießner

Online 3D Reconstruction

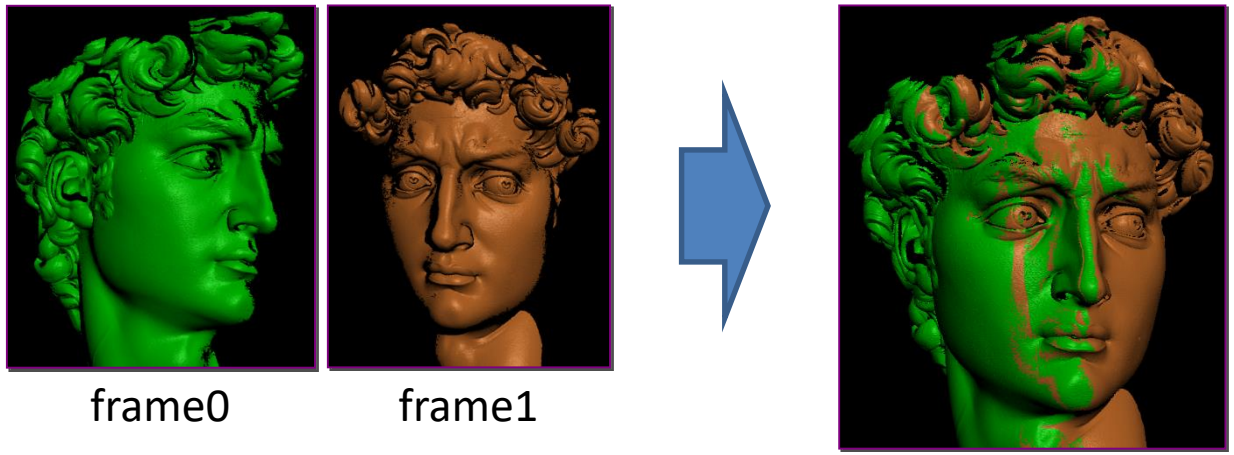


Iterative Closest Points (ICP) *[Besl and McKay 92]*

- Assume closest points are corresponding

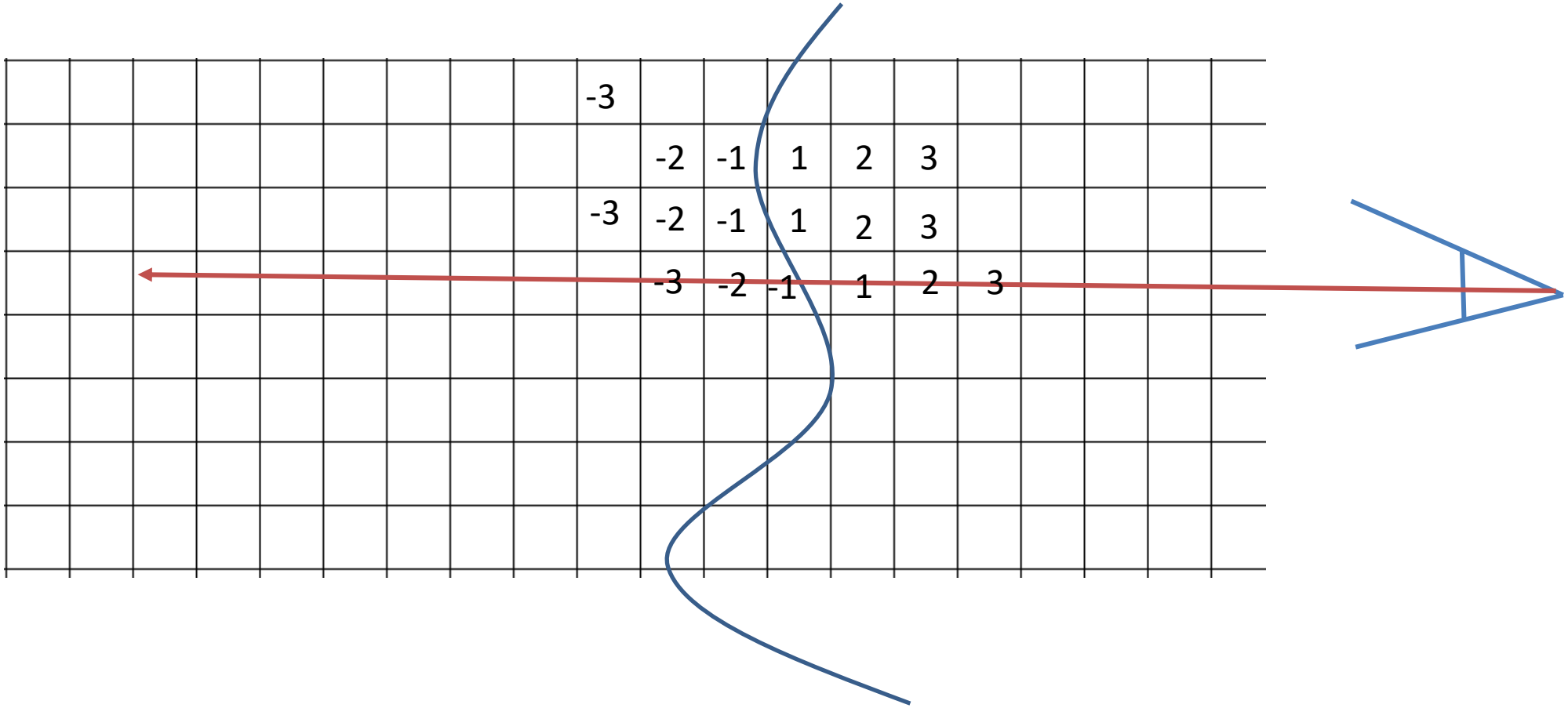


$$E_{frame-to-frame}(T) = \sum_k \|p_k - Tq_k\|_2^2$$

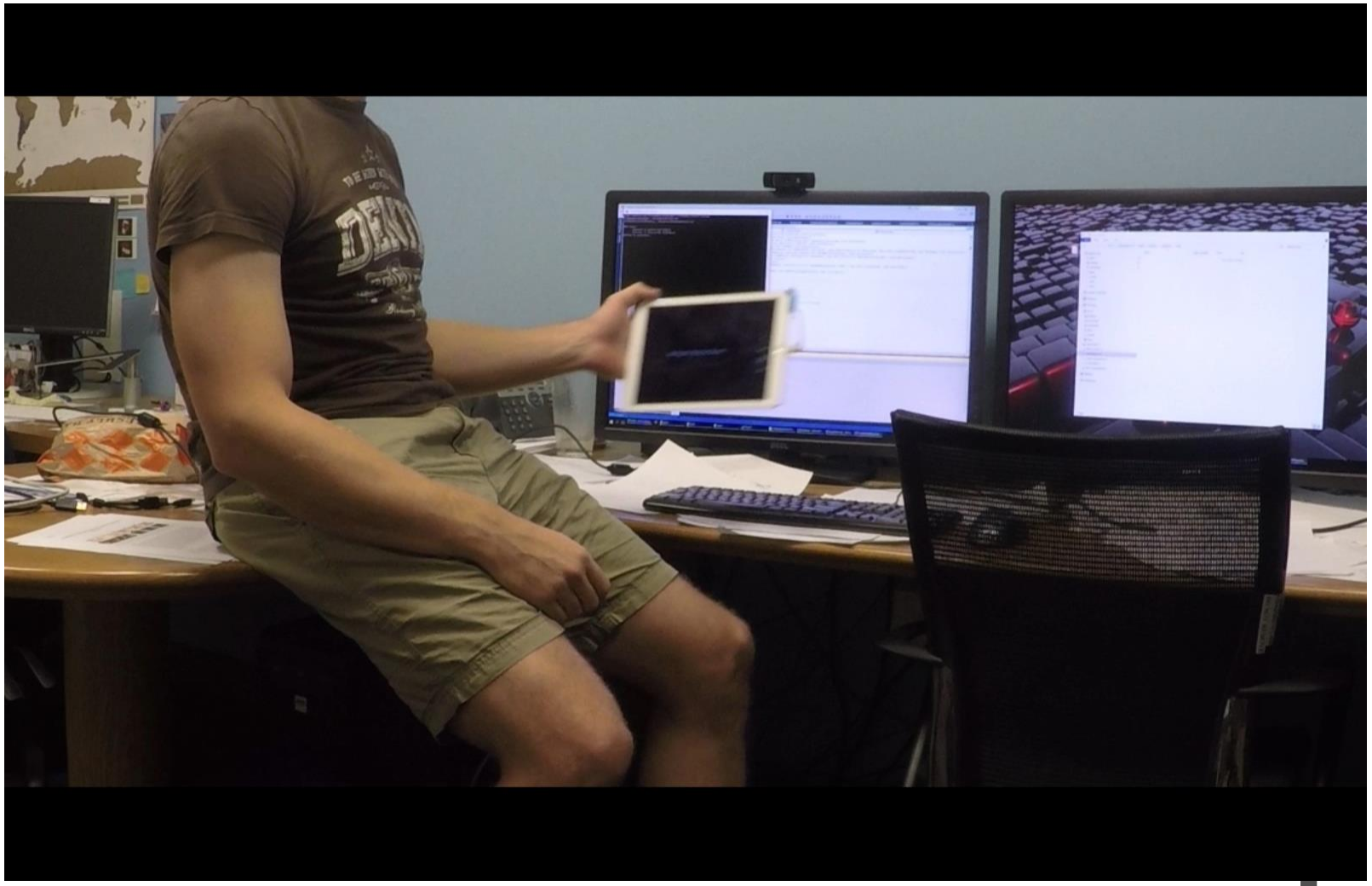


frame0 frame1

Volumetric Fusion: Surface Integration

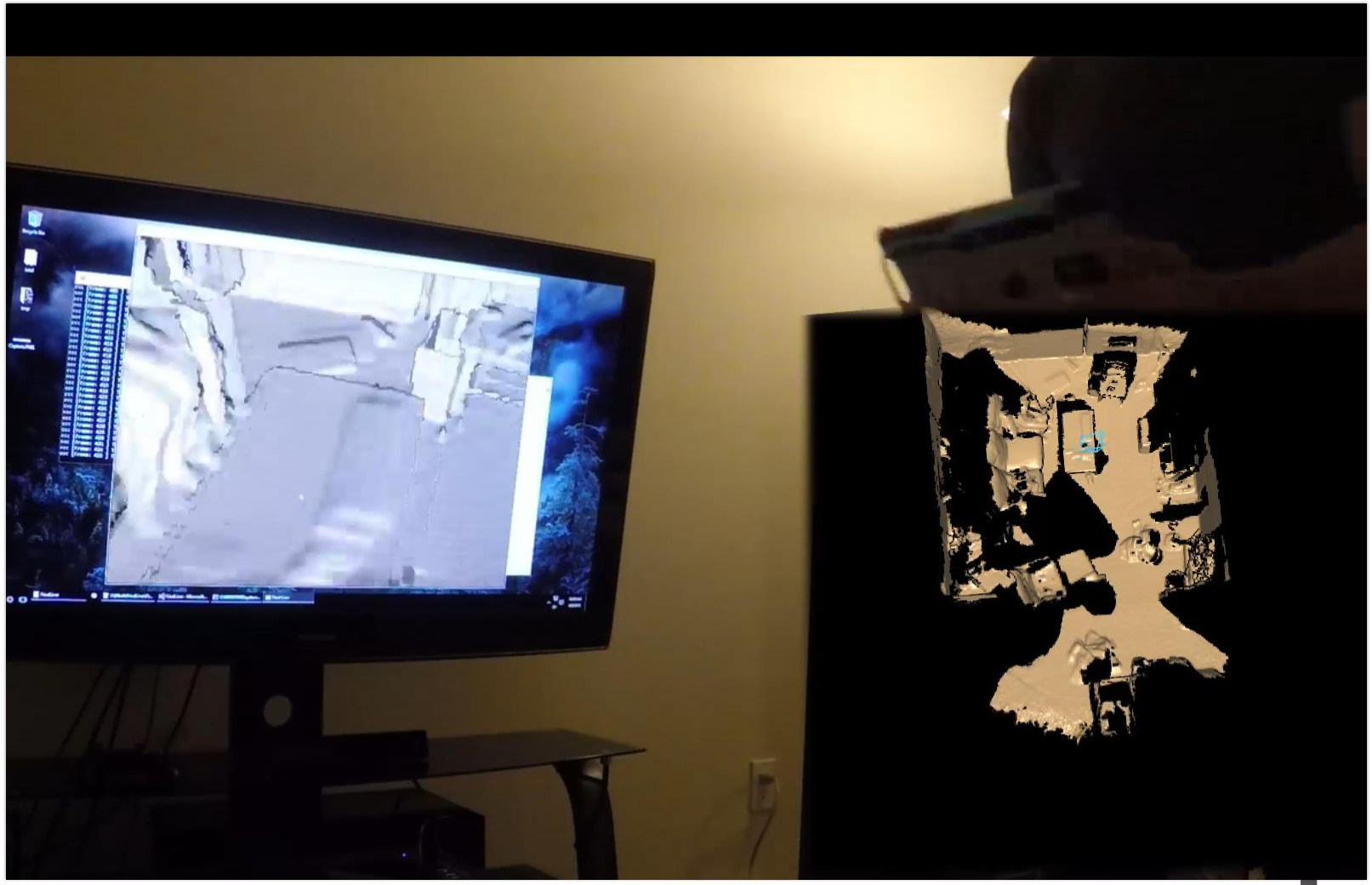


Bundle Fusion: Live Capture



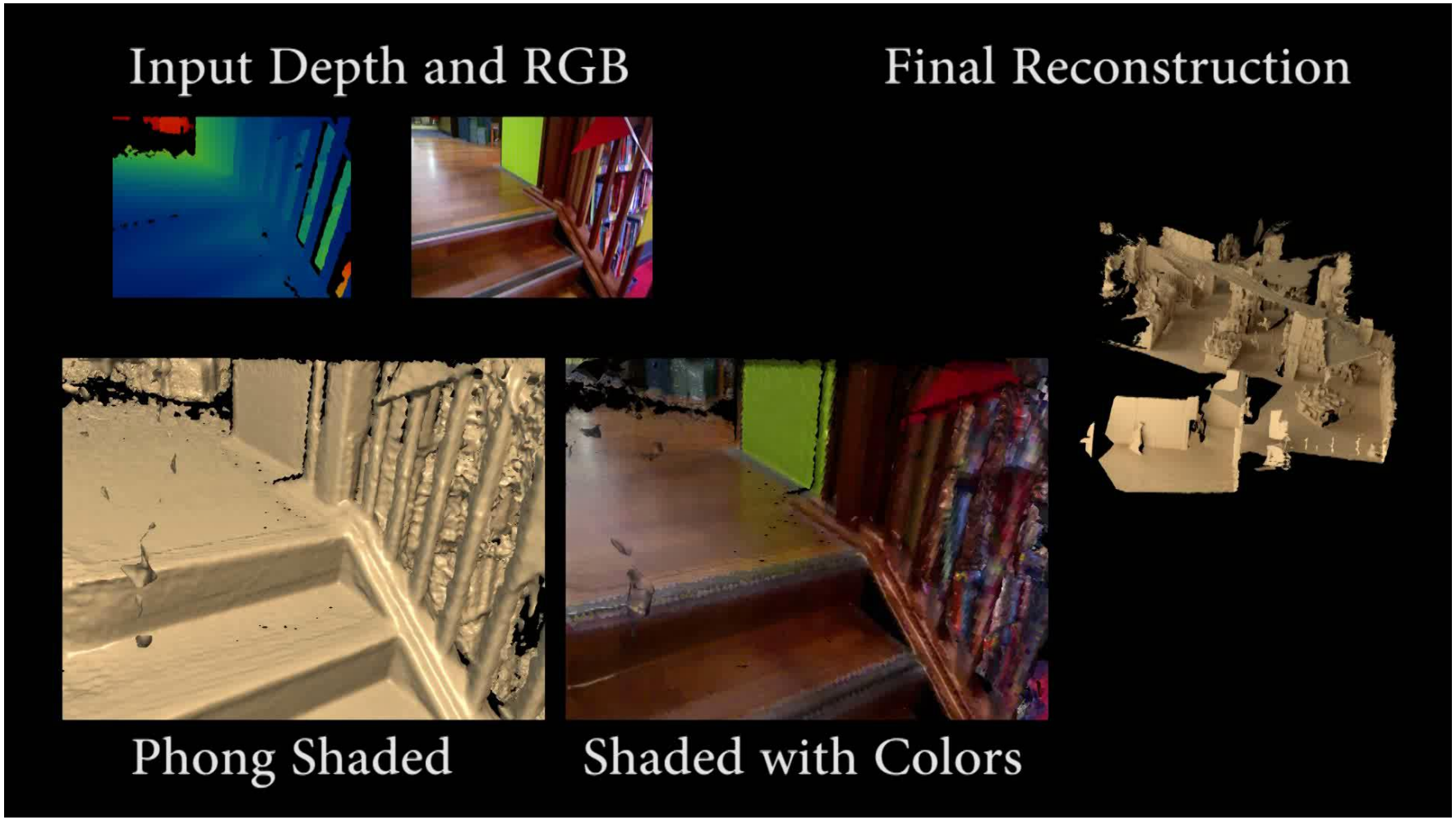
Bundle Fusion: Loop Closure

Bundle Fusion: Re-Localization



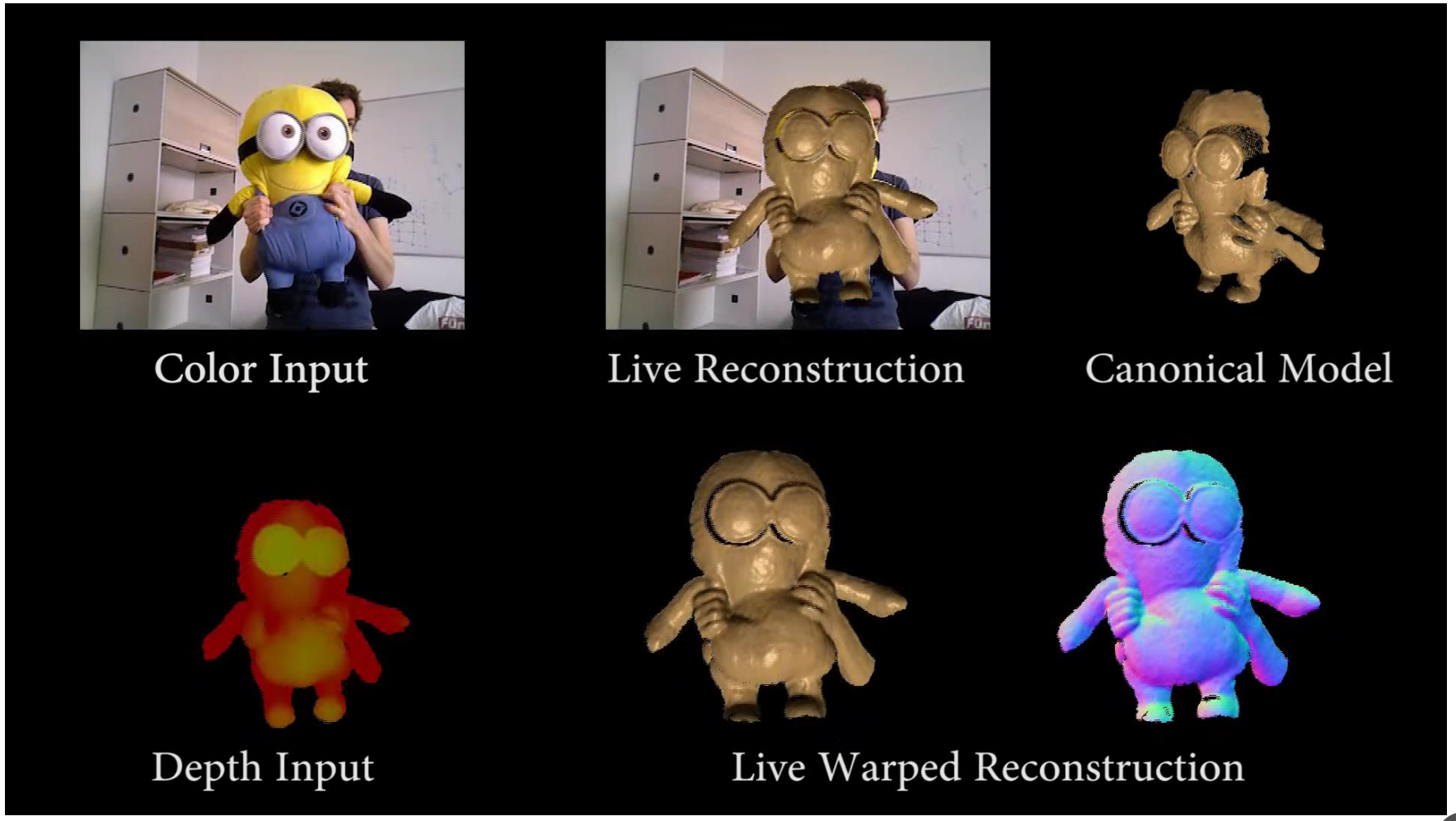
BF: Results

Dynamic Scenes



Today: Deformation and Non-rigid Surface Tracking

Dynamic Scenes



ECCV'16 [Innmann et al.]: VolumeDeform

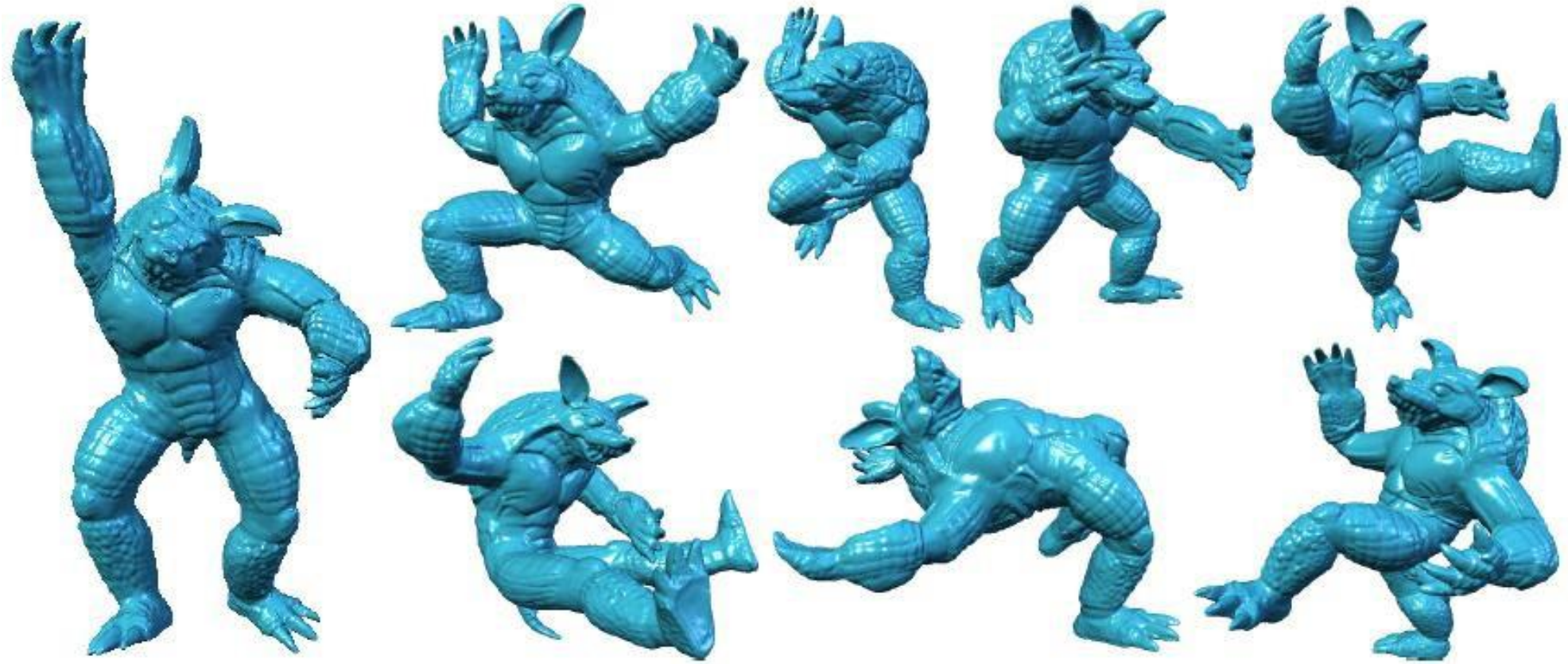
Goal

- Non-rigid 3D Reconstruction
  - Dynamic RGB-D sequence
  - Non-rigidly track over time
  - Accumulate data into shared model

- But not so easy ☺
  - Surface change or motion?
  - Let's start easy!

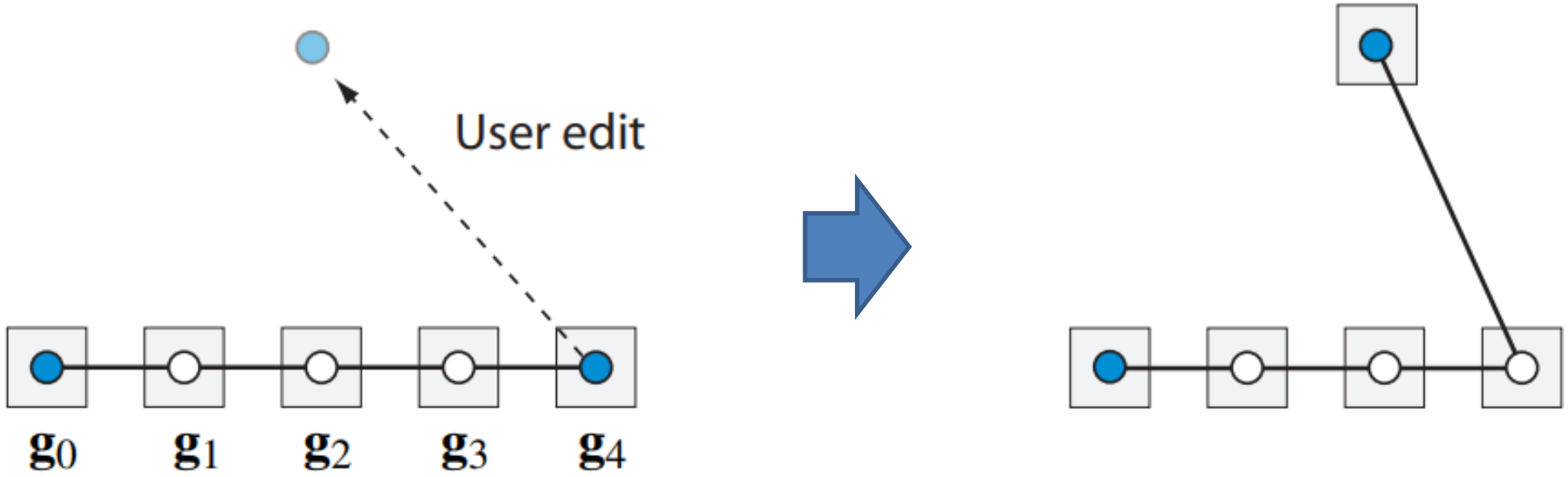
Mesh Deformation

- In graphics: e.g., character animation



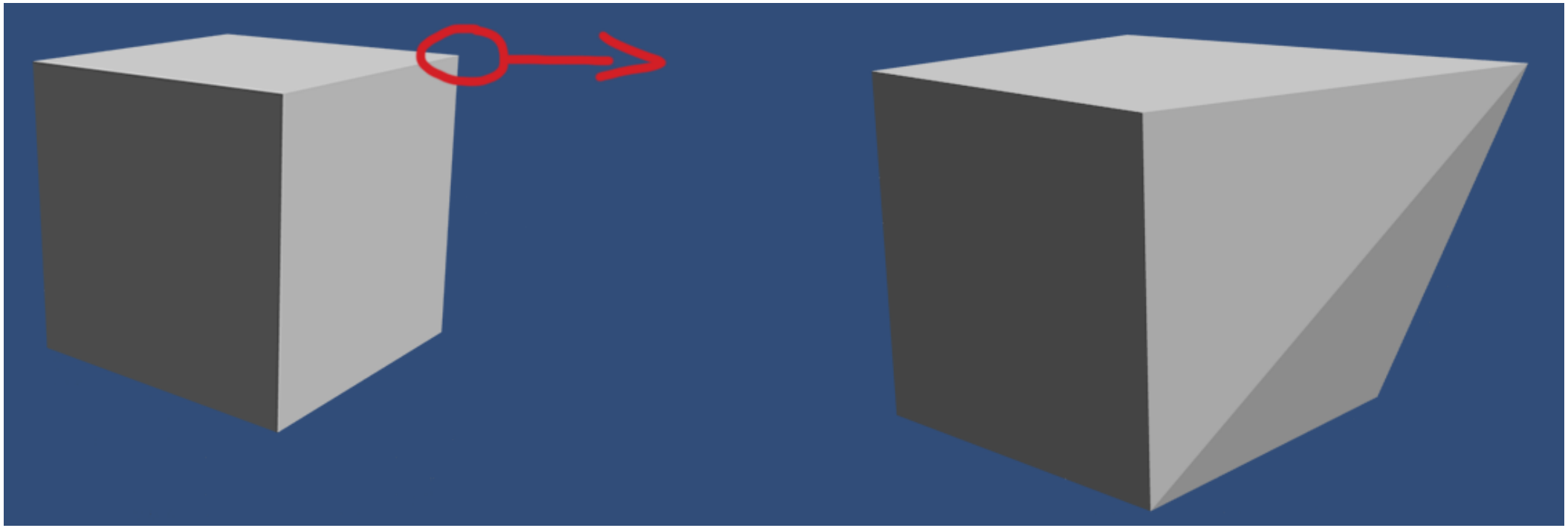
Mesh Deformation

- Simple idea
  - Just add handle to vertex and move it



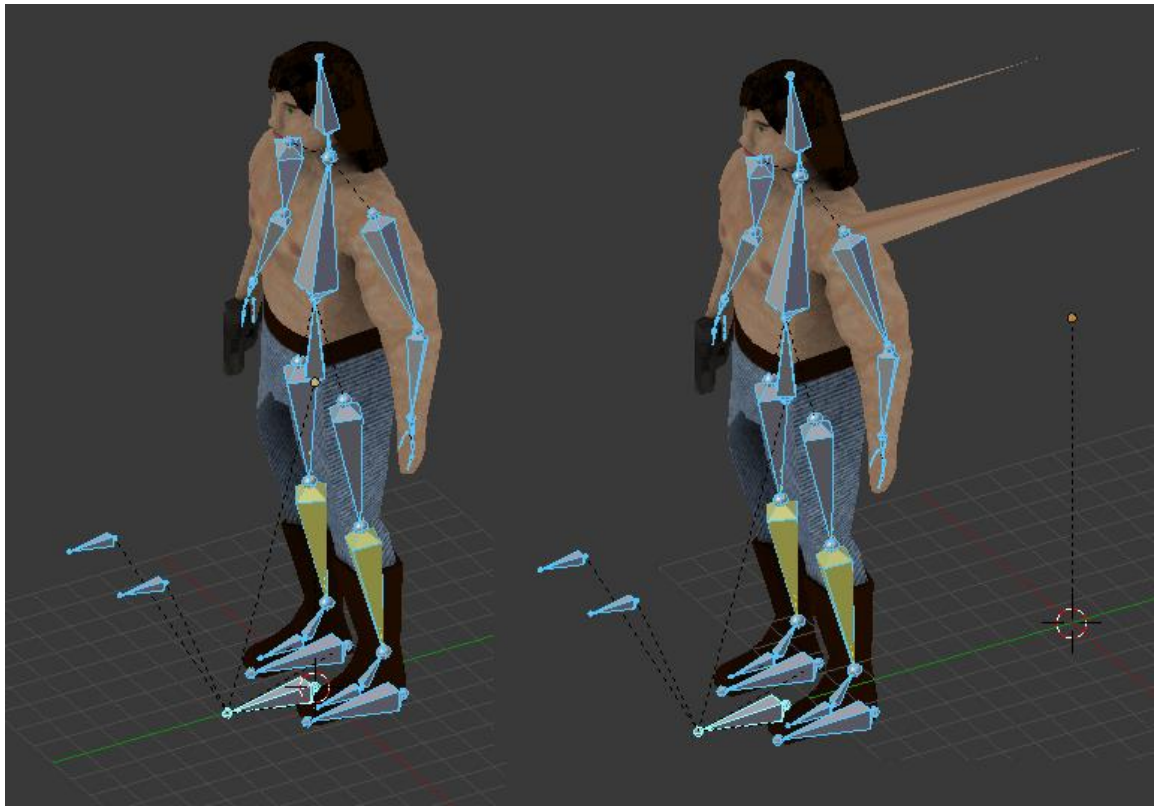
Mesh Deformation

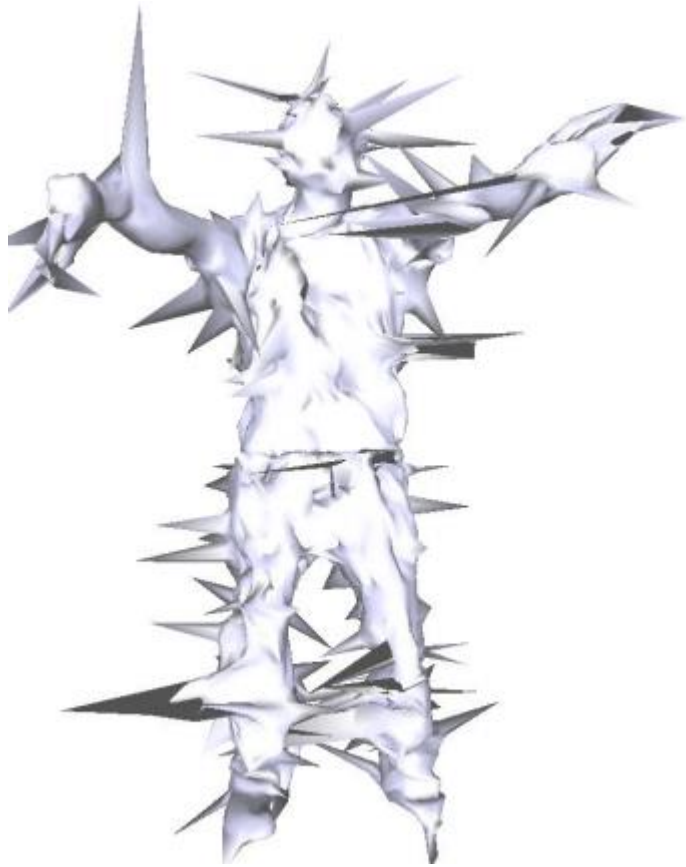
- Simple idea
  - Just add handle to vertex and move it



Mesh Deformation

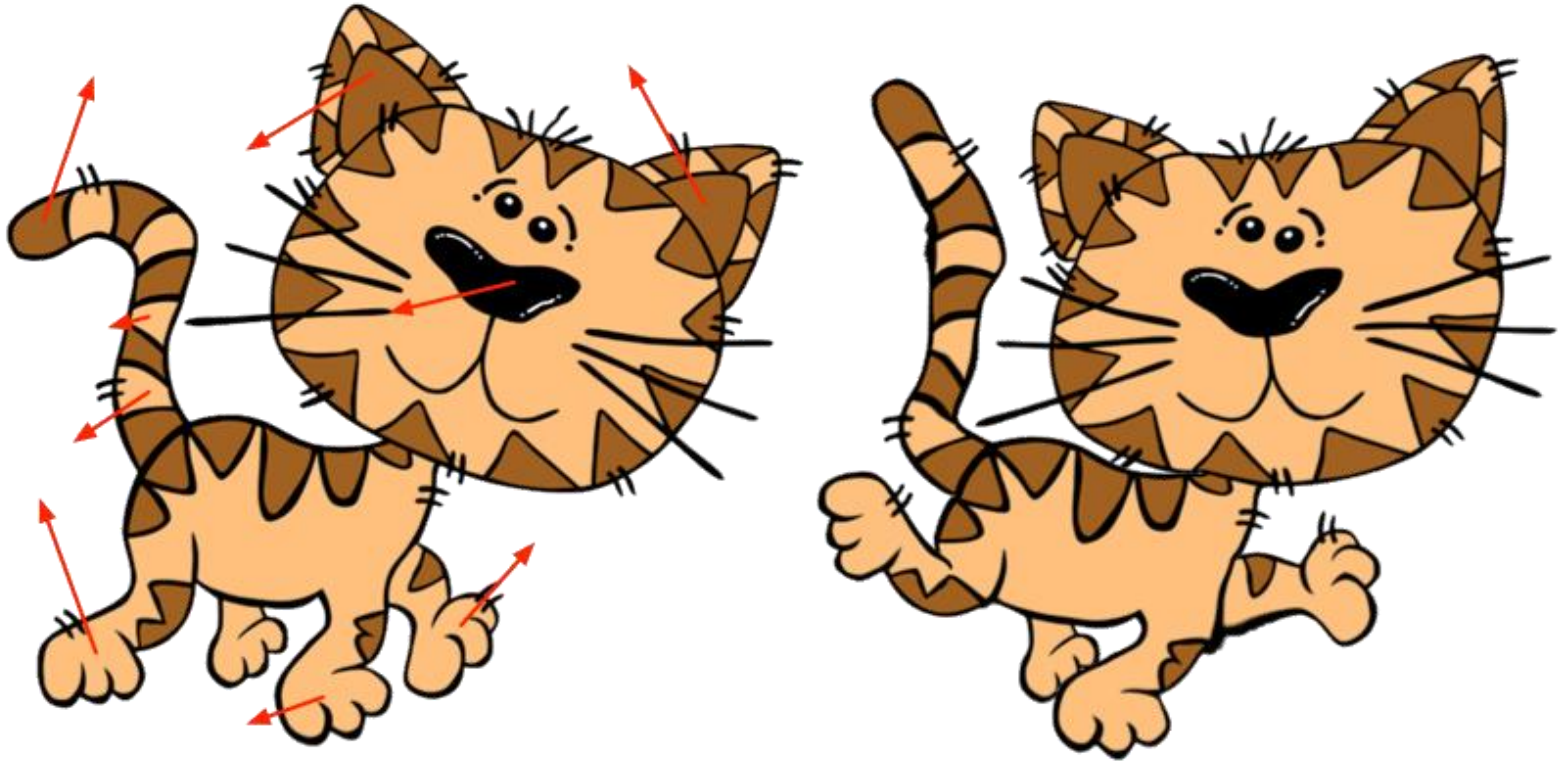
- Simple idea
  - Just add handle to vertex and move it





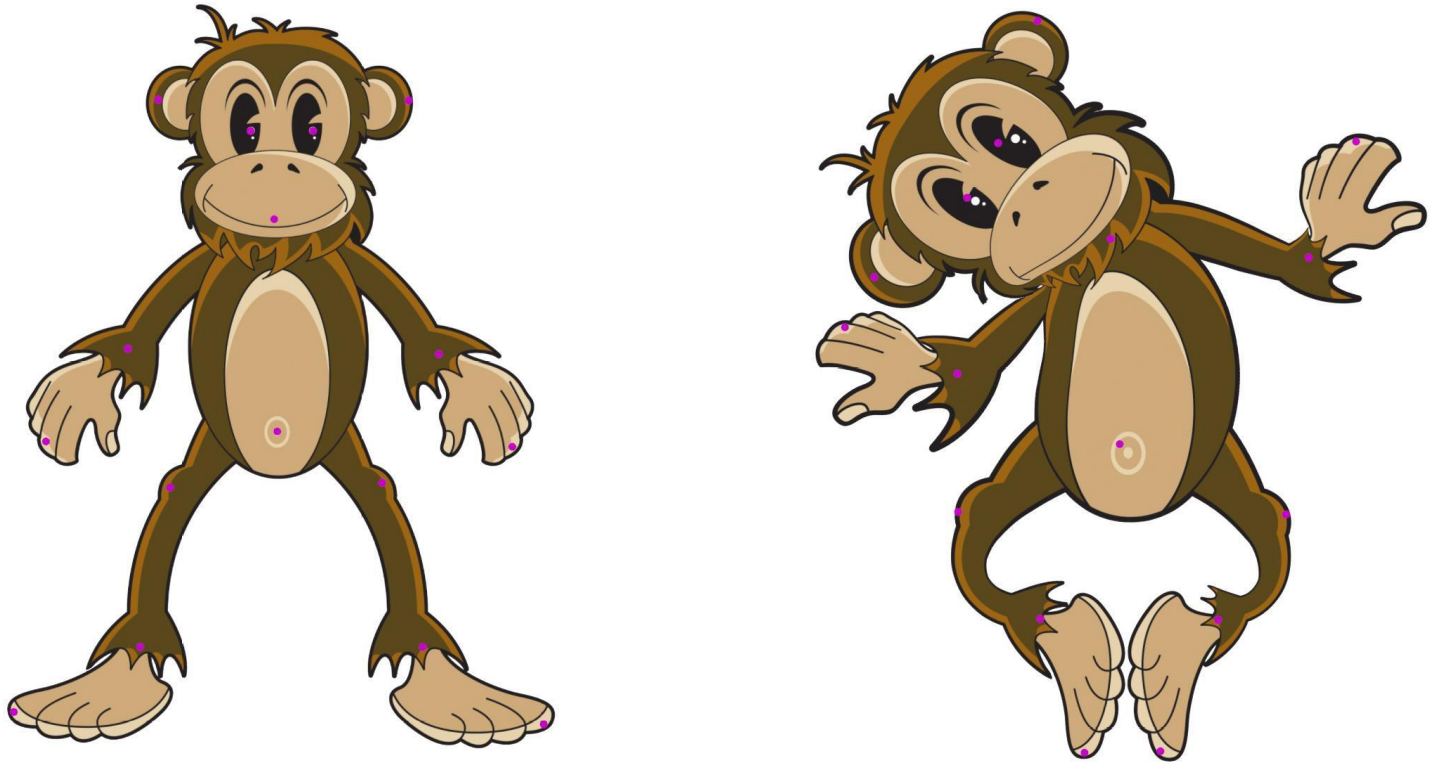
Mesh Deformation

- We also want adjacent vertices to move!
  - Pick some points, drag them around



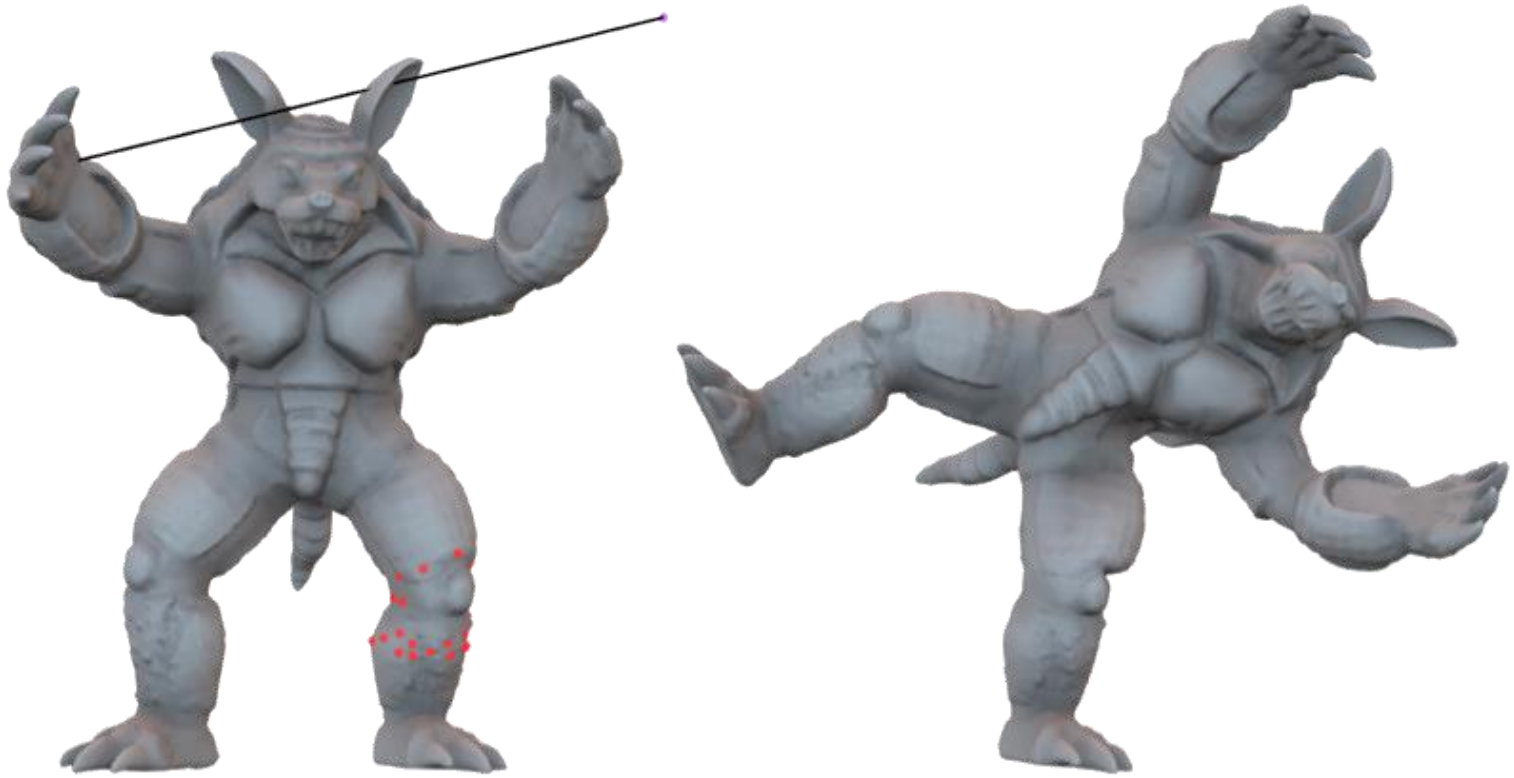
Mesh Deformation

- We also want adjacent vertices to move!
  - Pick some points, drag them around



Mesh Deformation

- We also want adjacent vertices to move!
  - Pick some points, drag them around



Mesh Deformation

- $E_{fit}(\boldsymbol{V}) = \sum_{i \in C} \left|\left| c_i - v_i \right|\right|_2^2$

Picked correspondence

Target pos. of correspondence

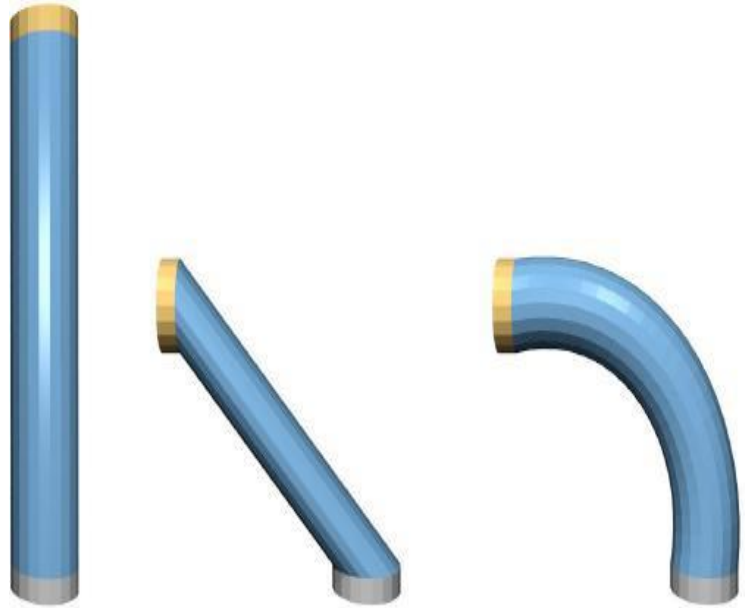
Loop over all picked points by user

- $E_{reg}(\boldsymbol{V}, X) = \cdots$ ← Somehow make neighbors move!

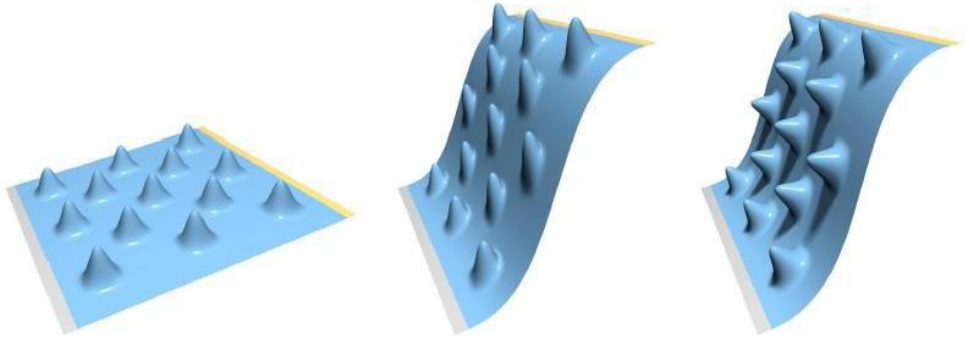
- Minimize $\lambda_{fit} E_{fit}(\boldsymbol{V}) + \lambda_{reg} E_{reg}(\boldsymbol{V}, X)$

Mesh Deformation

- How to choose a regularizer?
- Computed with respect to base mesh V
- Intuitive and natural behavior
  - Smooth global deformation
  - Global changes should preserve local detail
  - Should be zero in un-deformed state (base mesh)



- $E_{reg}(\boldsymbol{V}, X) = \cdots$



Laplacian Surface Editing

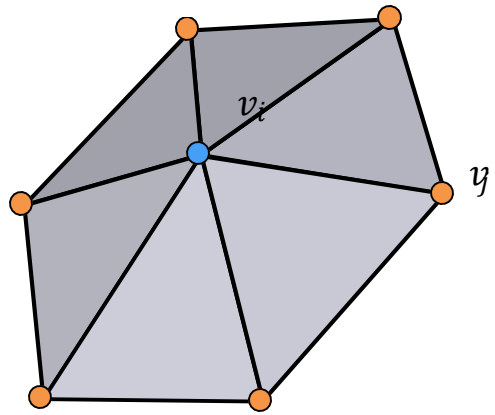
- Enforce Smoothness on deformations

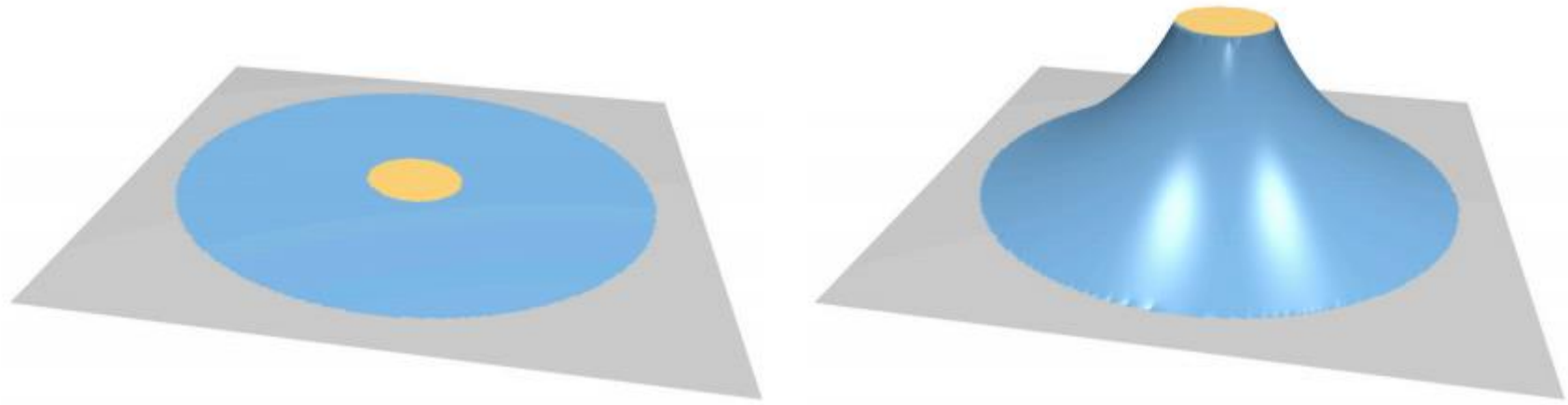
- $E_{reg}(\boldsymbol{V}, X) = \sum_i \sum_{j \in \mathcal{N}(i)} \left|\left|(v_i' - v_i) - (v_j' - v_j)\right|\right|_2^2$

Sum over all vertices

Sum over 1-ring (vertex fan)

Deformation difference



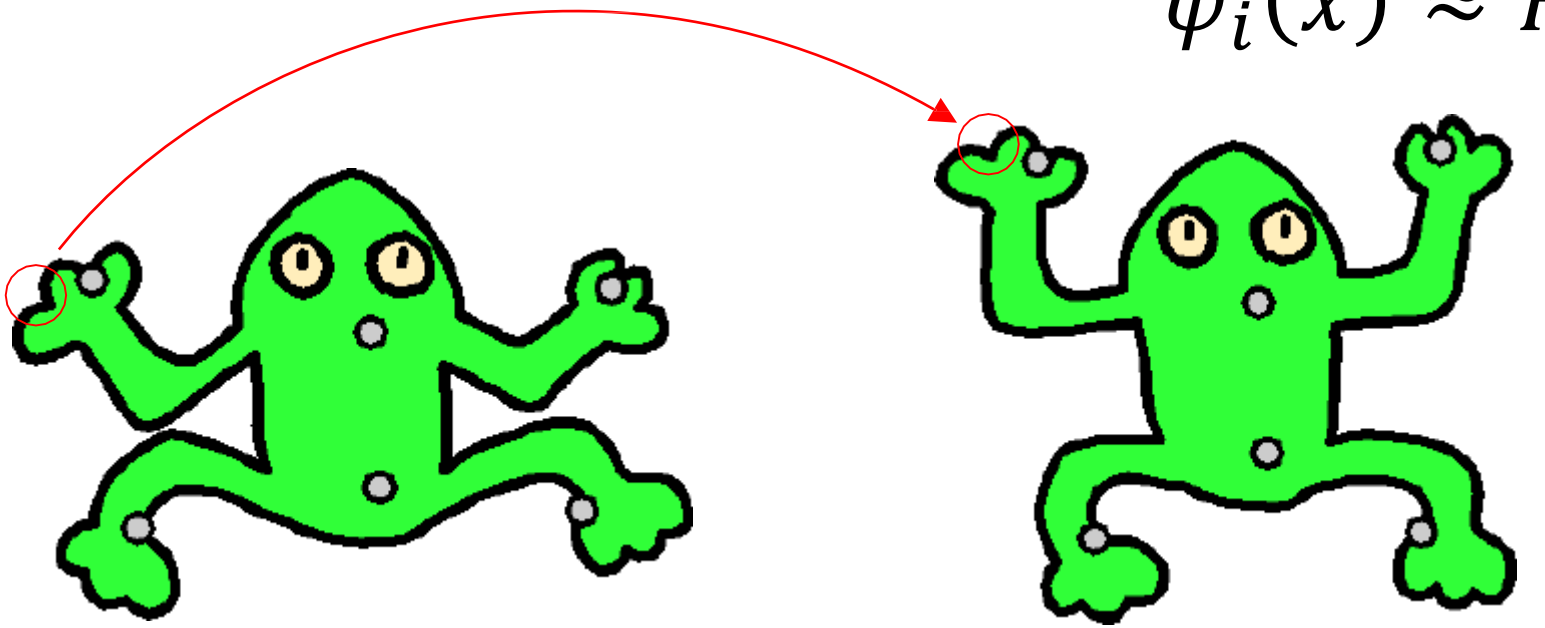


[Sorkine and Botsch]

As-Rigid-as-Possible Deformation

- ARAP Idea:
  - maximize local rigidity
  - Local transformations ψ_i almost rigid

$$\psi_i(x) \approx Rx + t$$

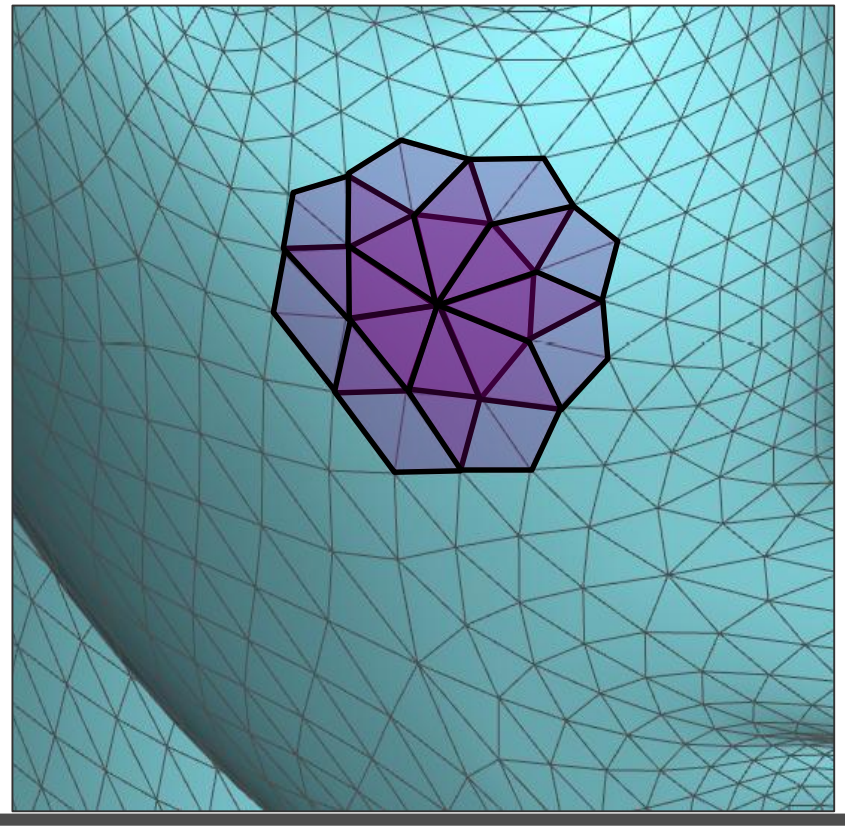


[Sorkine et al. 08]

As-Rigid-as-Possible Deformation

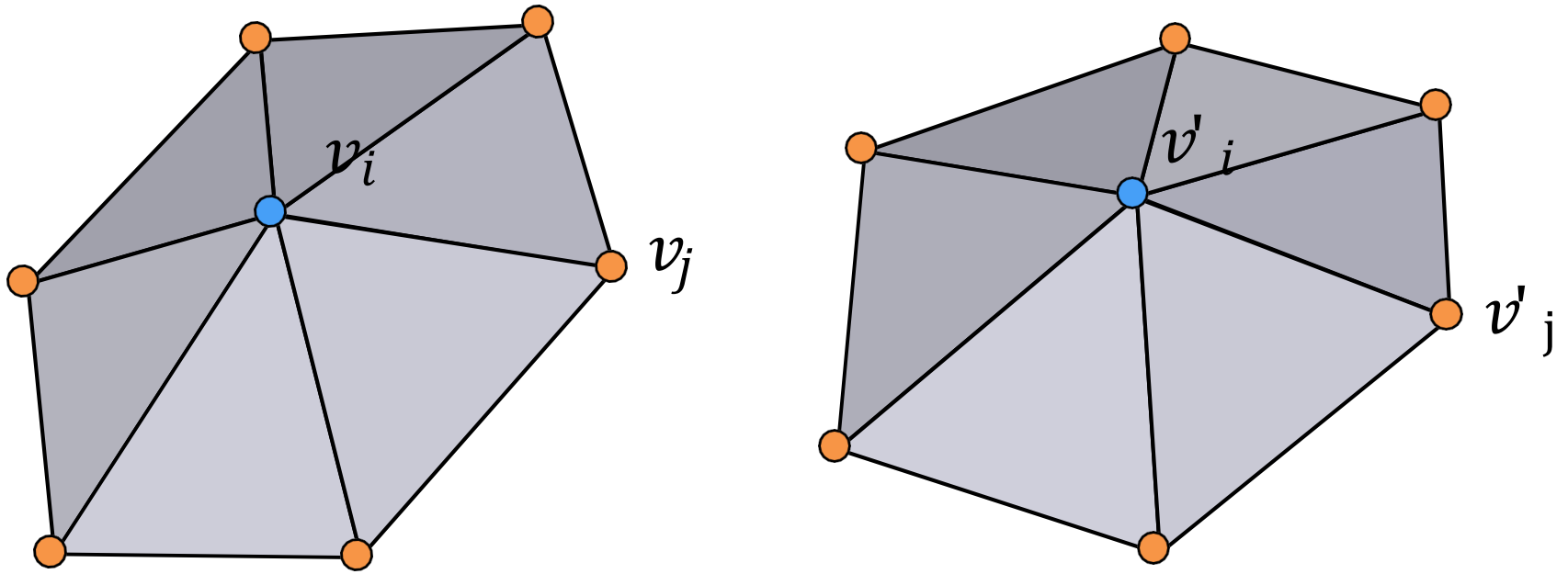
- Discrete setting:
  - Local cells (i.e., vertex fan) should be transformed w/ rigid transformation

Each fan has local matrix



As-Rigid-as-Possible Deformation

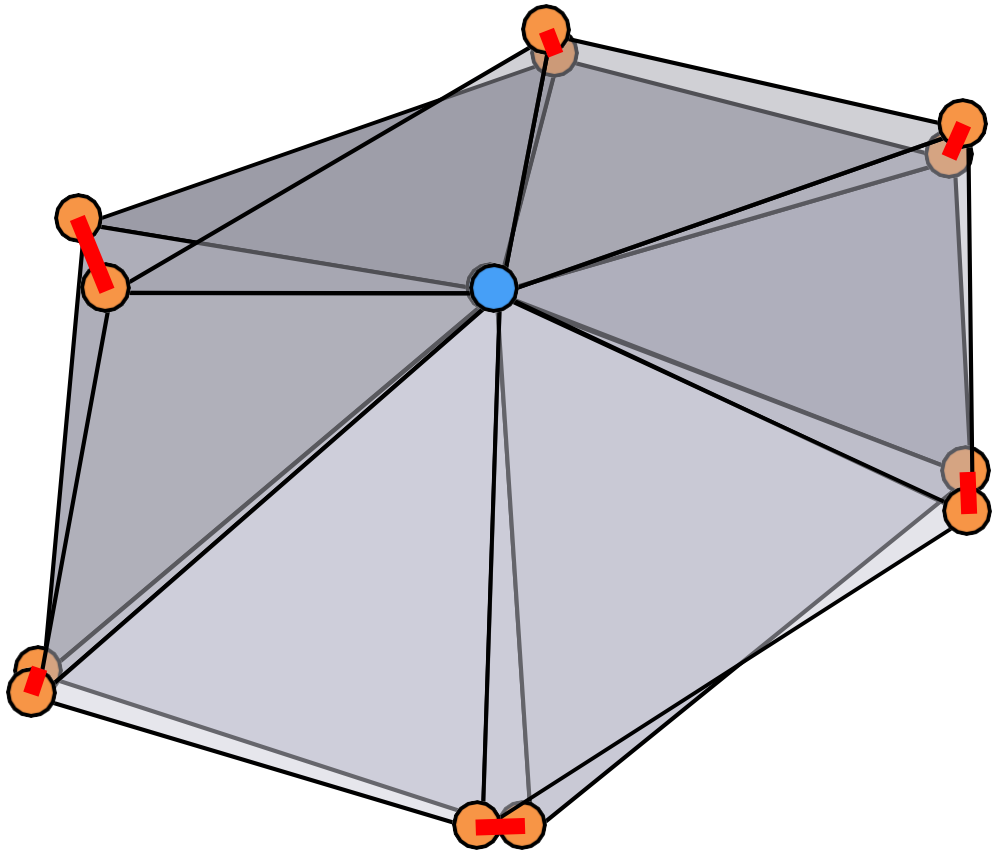
- For each vertex fan, measure deviation from rigidity



[Sorkine et al. 08]

As-Rigid-as-Possible Deformation

- For each vertex fan, measure deviation from rigidity



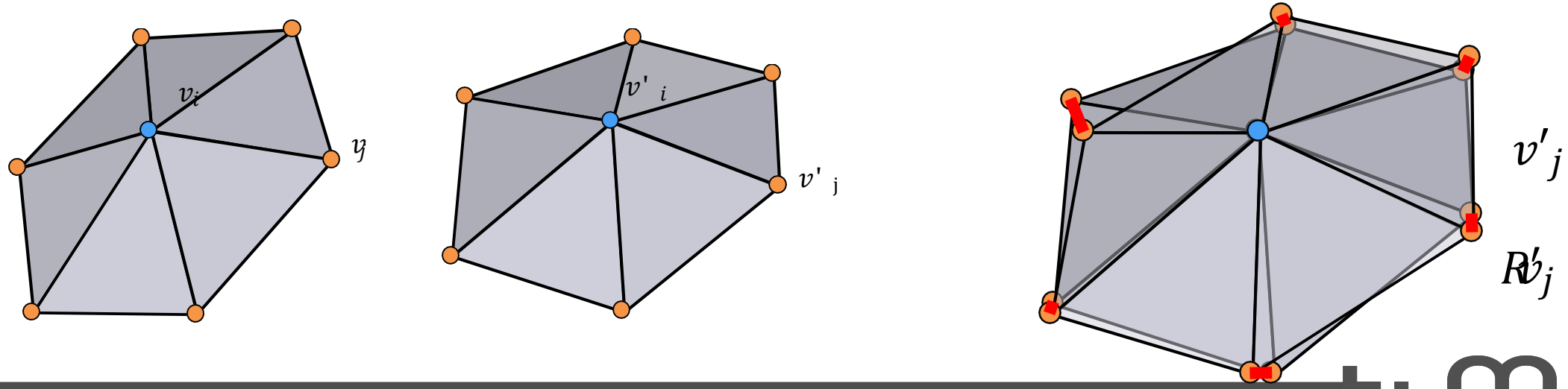
[Sorkine et al. 08]

As-Rigid-as-Possible Deformation

- One rotation matrix per fan; sum up deviations from rigidity
- $E_{ARAP}(\boldsymbol{R},\boldsymbol{V}) = \sum_i \sum_{j\in\mathcal{N}(i)} \left\| (v_i - v_j) - R_i(v'_i - v'_j) \right\|_2^2$

Edge (i.e., cancels translation)

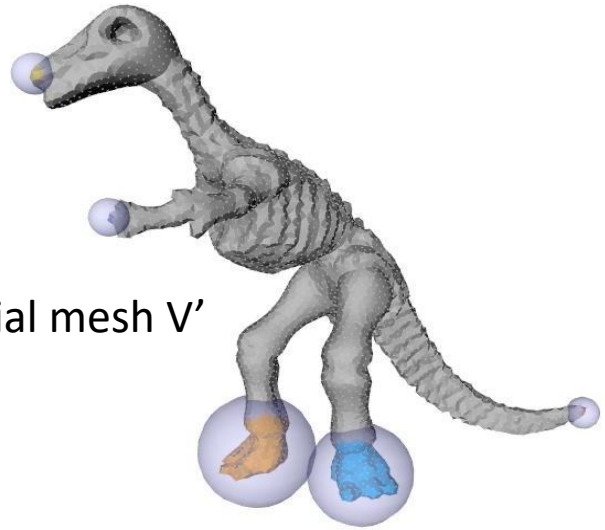
Find best rotation matrix that maps undeformed fan to deformed vertex fan



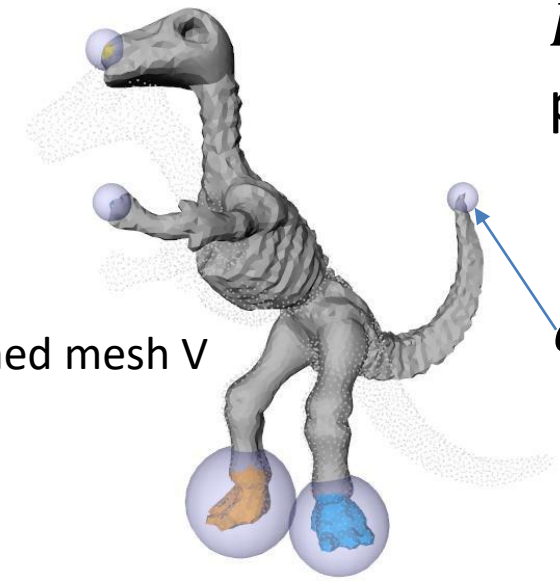
[Sorkine et al. 08]

As-Rigid-as-Possible Deformation

- One rotation matrix per fan; sum up deviations from rigidity
- $E_{ARAP}(\boldsymbol{R},\boldsymbol{V}) = \sum_i \sum_{j\in\mathcal{N}(i)} \left|\left|(v_i - v_j) - R_i(v_i' - v_j')\right|\right|_2^2$
- $E_{fit}(\boldsymbol{V}) = \sum_{i\in C} ||c_i - v_i||_2^2$



Initial mesh V'



Deformed mesh V

R_i is rotation matrix; e.g., parametrized by Euler angles α, β, γ

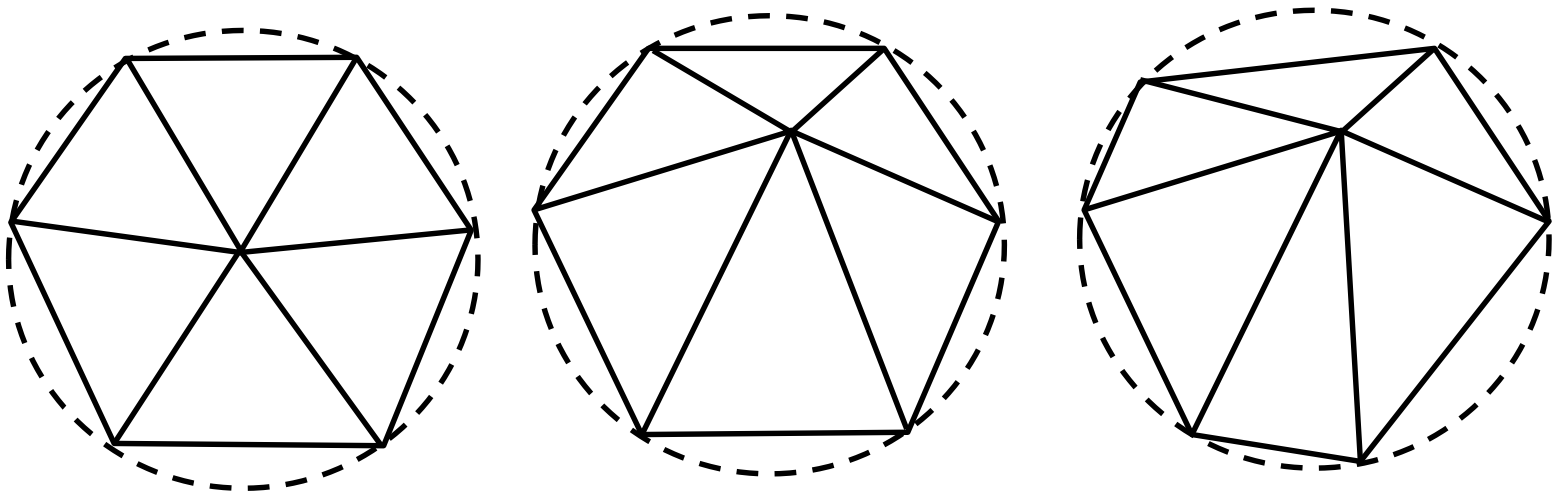
c_i is the target location

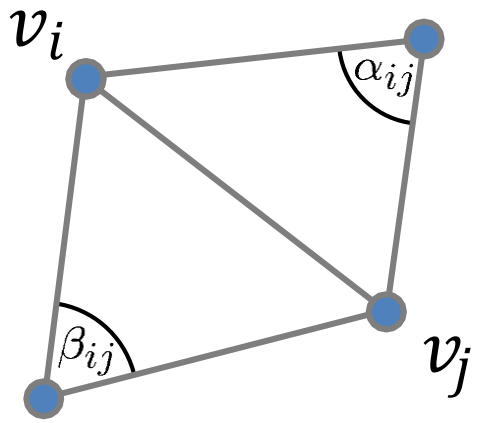
[Sorkine et al. 08]

As-Rigid-as-Possible Deformation

- Could weight: close neighbors should count more
- $E_{ARAP}(\boldsymbol{R},\boldsymbol{V}) = \sum_i \sum_{j\in\mathcal{N}(i)} w_{ij} \left|\left|(v_i - v_j) - R_i(v_i' - v_j')\right|\right|_2^2$

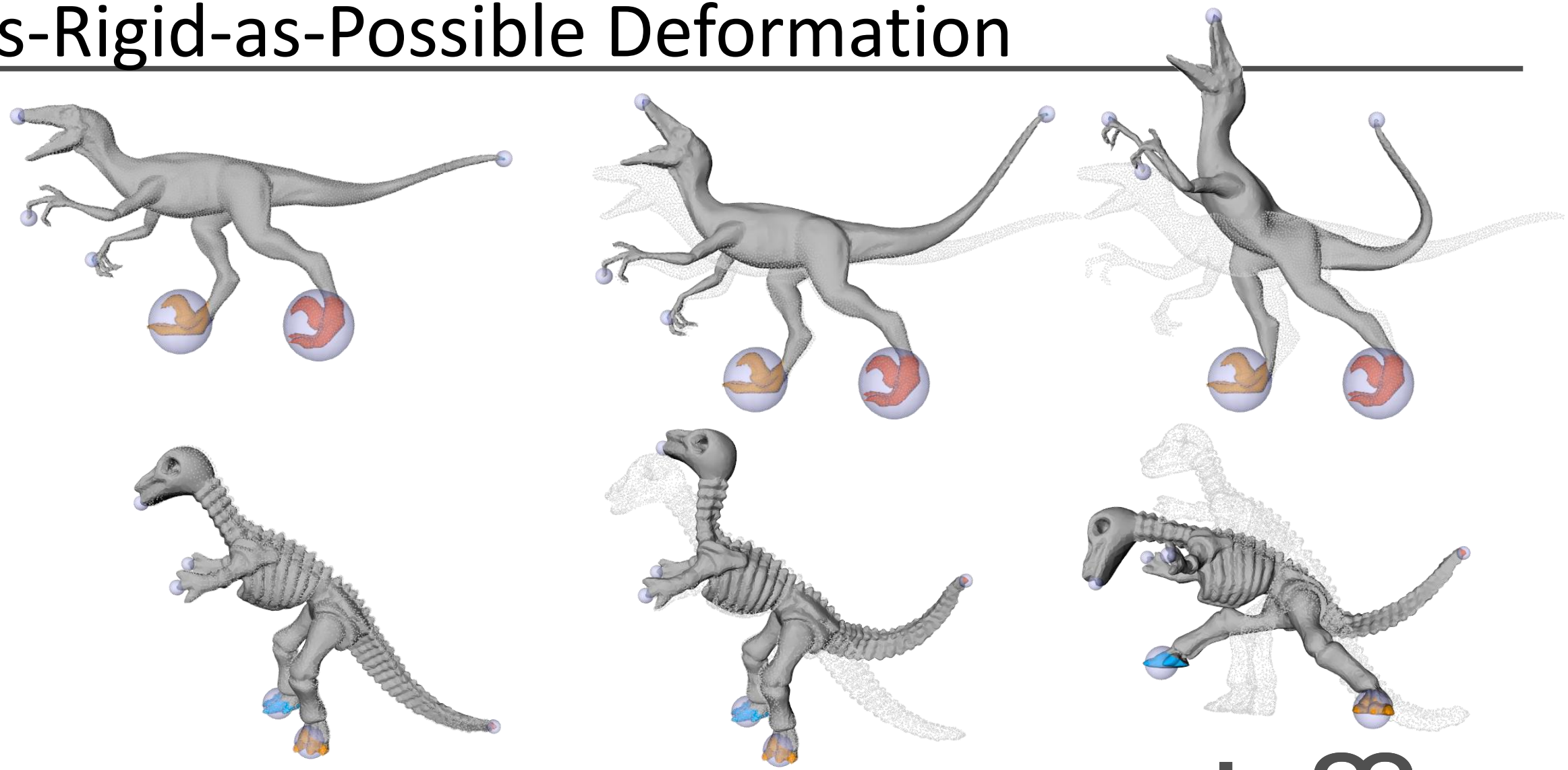
  $-\ w_{ij} = \frac{1}{2}(\cot\alpha_{ij} + \cot\beta_{ij})$



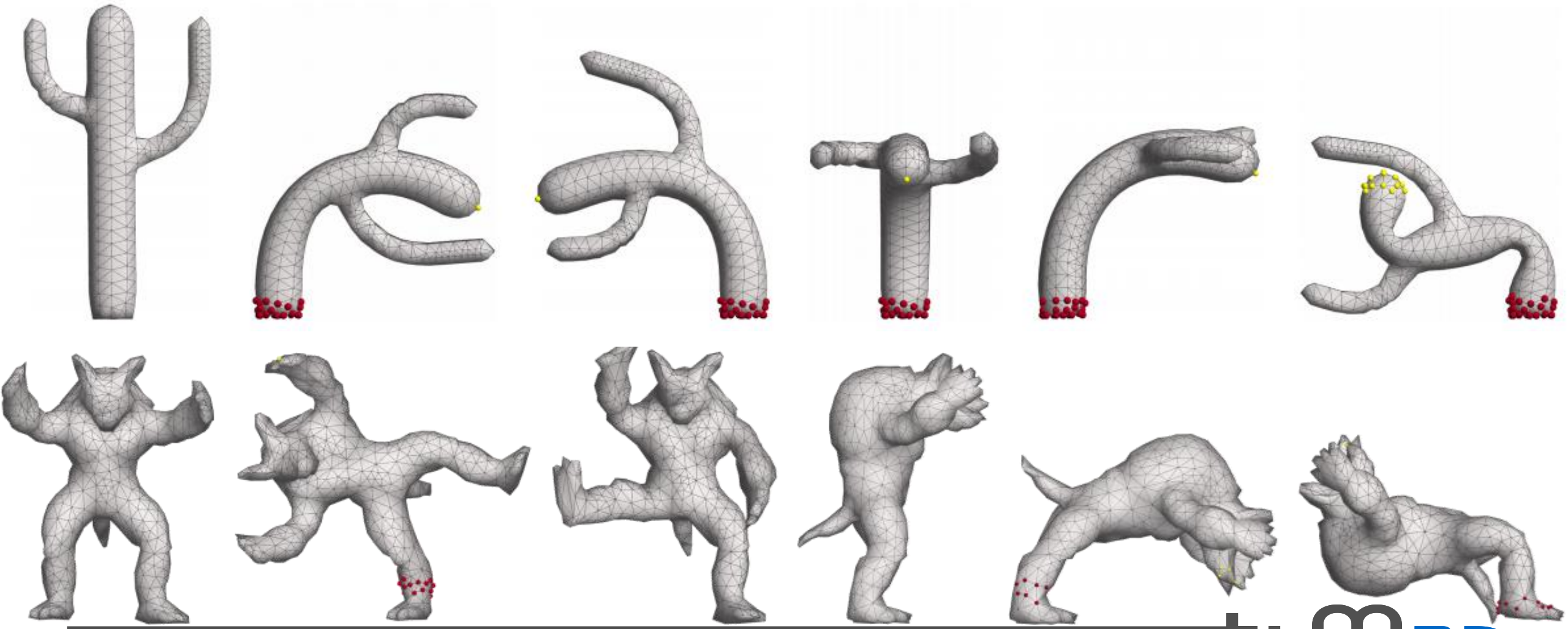


[Sorkine et al. 08]

As-Rigid-as-Possible Deformation



As-Rigid-as-Possible Deformation



[Sorkine et al. 08]

As-Rigid-as-Possible Deformation

As-Rigid-as-Possible Deformation

- Given a mesh $\in \mathbb{R}^3$ with n Vertices:

How many unknowns?

How many constraints?

[Sorkine et al. 08]

As-Rigid-as-Possible Deformation

- Given a mesh $\in \mathbb{R}^3$ with n Vertices:

How many unknowns?

$$3 \cdot n + 3 \cdot n$$

translations / offsets

rotations

How many constraints?

$$3 \cdot n \cdot valence + 3 \cdot c$$

[Sorkine et al. 08]

As-Rigid-as-Possible Deformation

- Given a mesh $\in \mathbb{R}^3$ with n Vertices:

How many unknowns?

$$3 \cdot n + 3 \cdot n$$

translations / offsets — rotations

How many constraints?

$$3 \cdot n \cdot valence + 3 \cdot c$$

Use iterative flip-flop optimization
- Assume vertices are fixed
  → solve for rotations
- Assume rotations are fixed
  → solve for vertices

[Sorkine et al. 08]

Embedded Deformation

- Allow deviations from rotations: given $M \in \mathbb{R}^{3\times3}$
- $E_{ED}(\boldsymbol{M},\boldsymbol{V}) = \sum_i \sum_{j\in\mathcal{N}(i)} \left|\left|(v_i - v_j) - M_i(v_i' - v_j')\right|\right|_2^2$
- $E_{rot}(\boldsymbol{M}) = \sum_i Rot(M_i)$
- $Rot(M) = (c_1 \cdot c_2)^2 + (c_1 \cdot c_3)^2 + (c_2 \cdot c_3)^2 + (c_1 \cdot c_1 - 1)^2 + (c_2 \cdot c_2 - 1)^2 + (c_3 \cdot c_3 - 1)^2$

$c_{1,2,3}$ are columns of matrix M

[Sumner et al. 07]

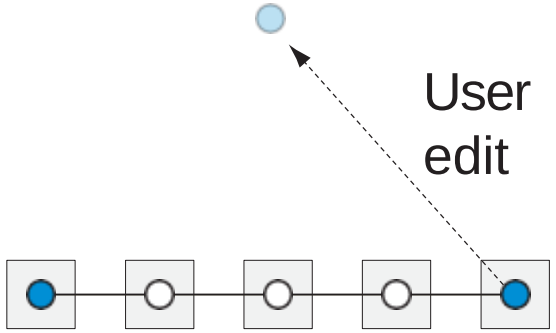
Embedded Deformation

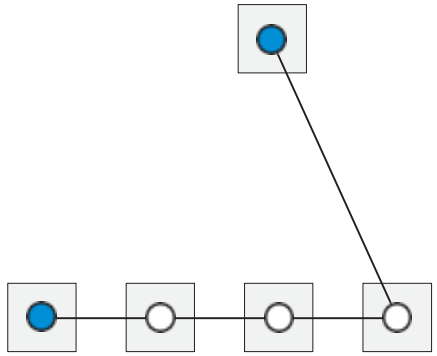
- Allow deviations from rotations: given $M \in \mathbb{R}^{3\times3}$
- $E_{ED}(\boldsymbol{M},\boldsymbol{V}) = \sum_i \sum_{j\in\mathcal{N}(i)} \left|\left|(v_i - v_j) - M_i(v_i' - v_j')\right|\right|_2^2$
- $\boxed{E_{rot}(\boldsymbol{M}) = \sum_i Rot(M_i)}$ ← soft-constraint for rotation
- $Rot(M) = (c_1 \cdot c_2)^2 + (c_1 \cdot c_3)^2 + (c_2 \cdot c_3)^2 + (c_1 \cdot c_1 - 1)^2 + (c_2 \cdot c_2 - 1)^2 + (c_3 \cdot c_3 - 1)^2$

$c_{1,2,3}$ are columns of matrix M

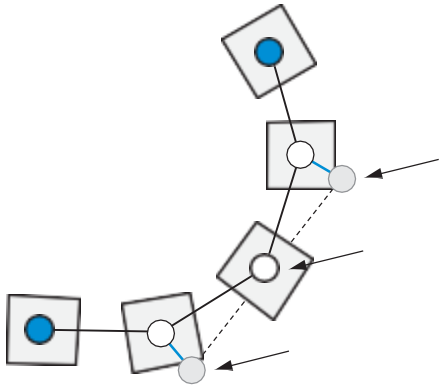
[Sumner et al. 07]

Embedded Deformation

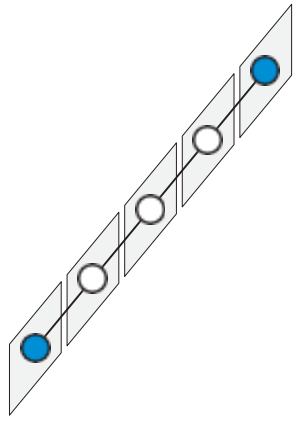




E_{con}



$E_{con} + E_{reg} + E_{rot}$

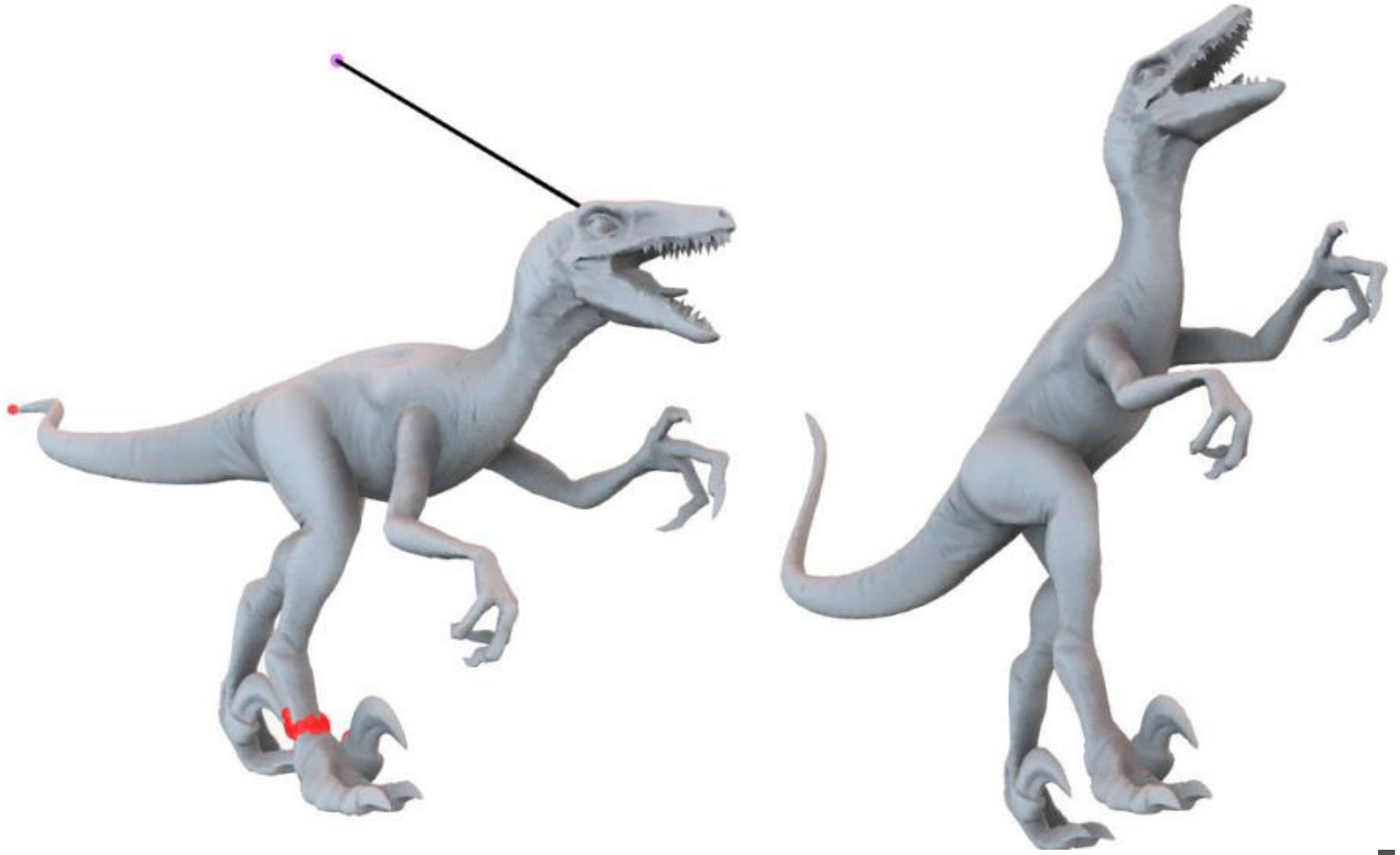


$E_{con} + E_{reg}$

ensures neighboring nodes having similar transformations

[Sumner et al. 07]

Embedded Deformation



[Sumner et al. 07]

ARAP vs ED

- ARAP: hard-constraints for rotations
  - Forced rotation matrices → 3 unknowns per local matrix (6 per vertex)
  - Can be efficiently optimized with flip-flop (local SVDs / Procrustes)

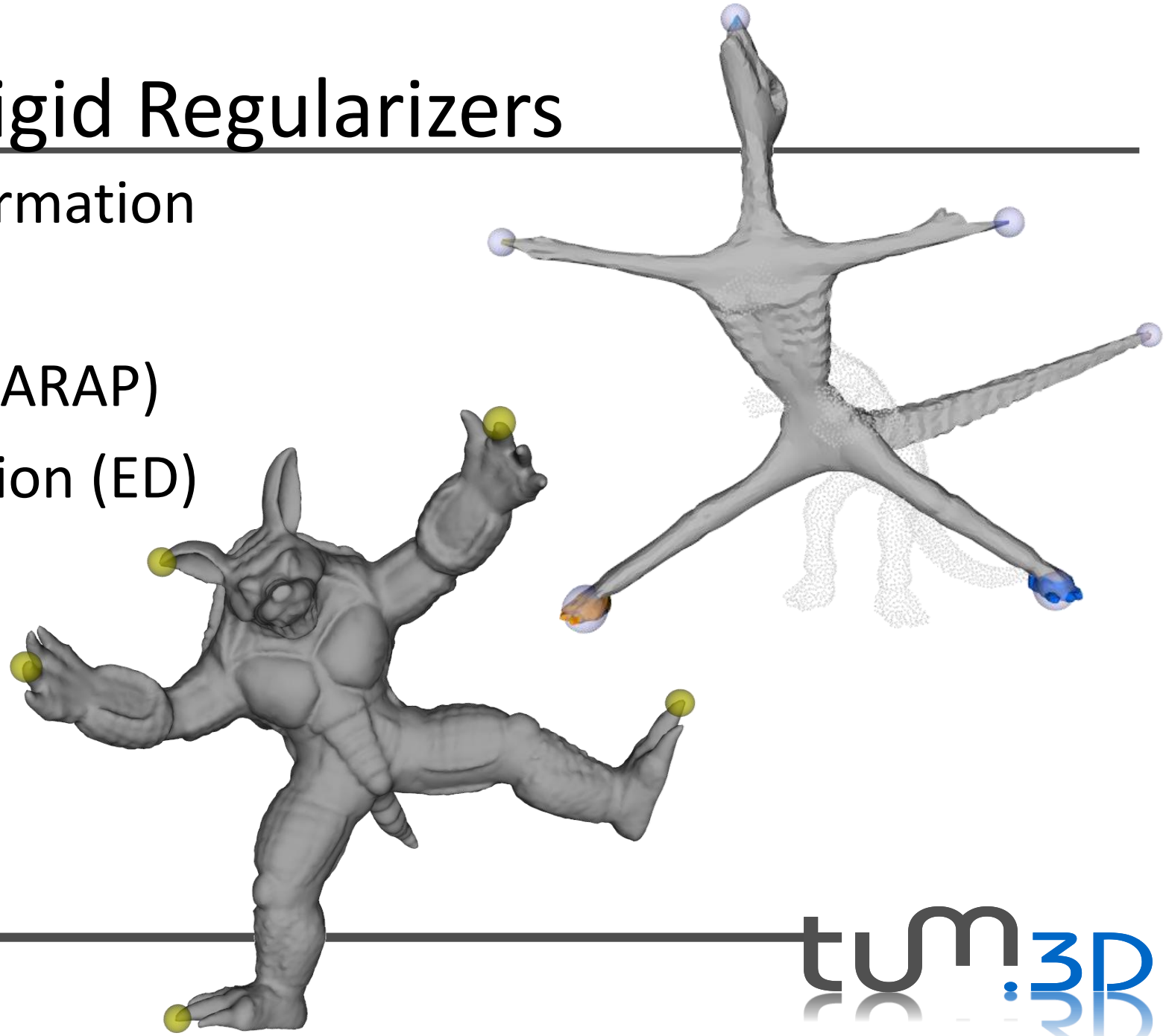
- ED: soft-constraints for rotations → more DoFs
  - I.e., → 9 unknowns per local matrix (12 per vertex)
  - Harder to optimize (actually a quartic problem)

Summary: Non-rigid Regularizers

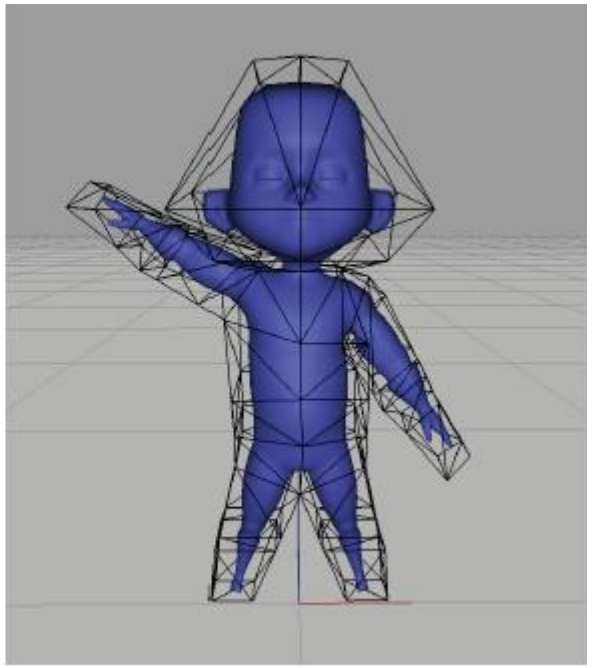
- Laplacian Mesh Deformation

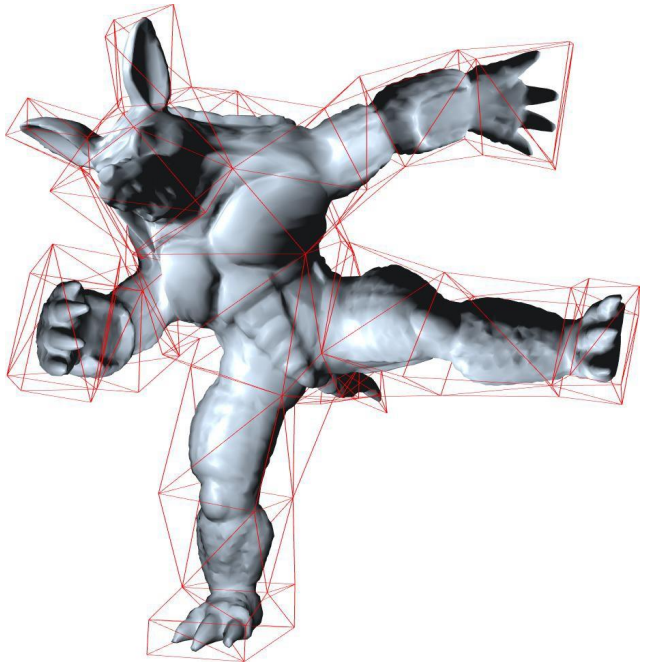
- As-Rigid-as-Possible (ARAP)
- Embedded Deformation (ED)

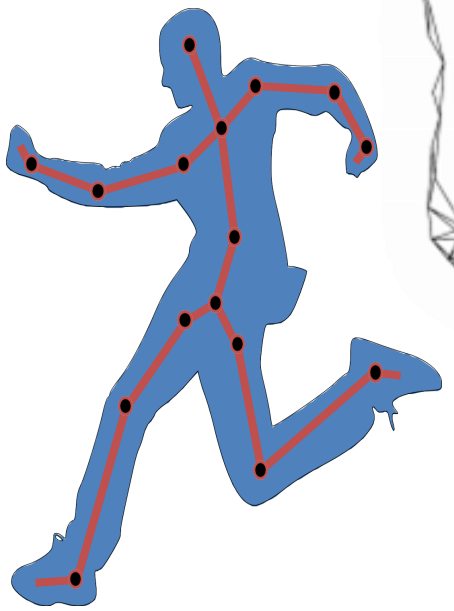
- Many others!

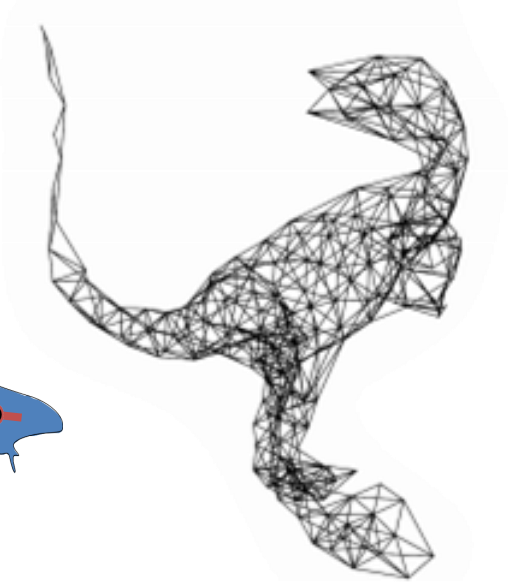


Deformation Proxies



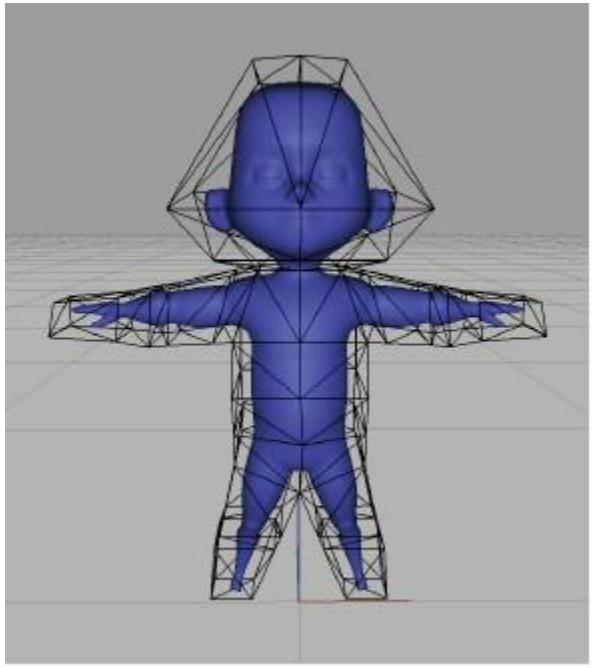


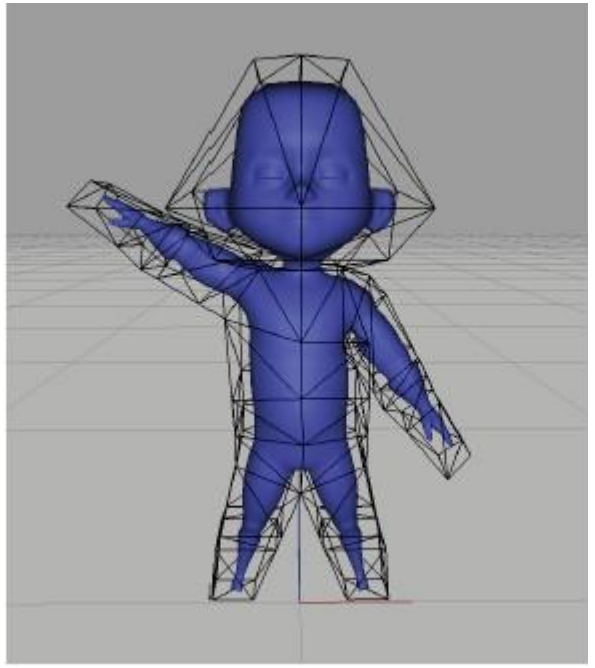




Regularizers on Deformation Proxies

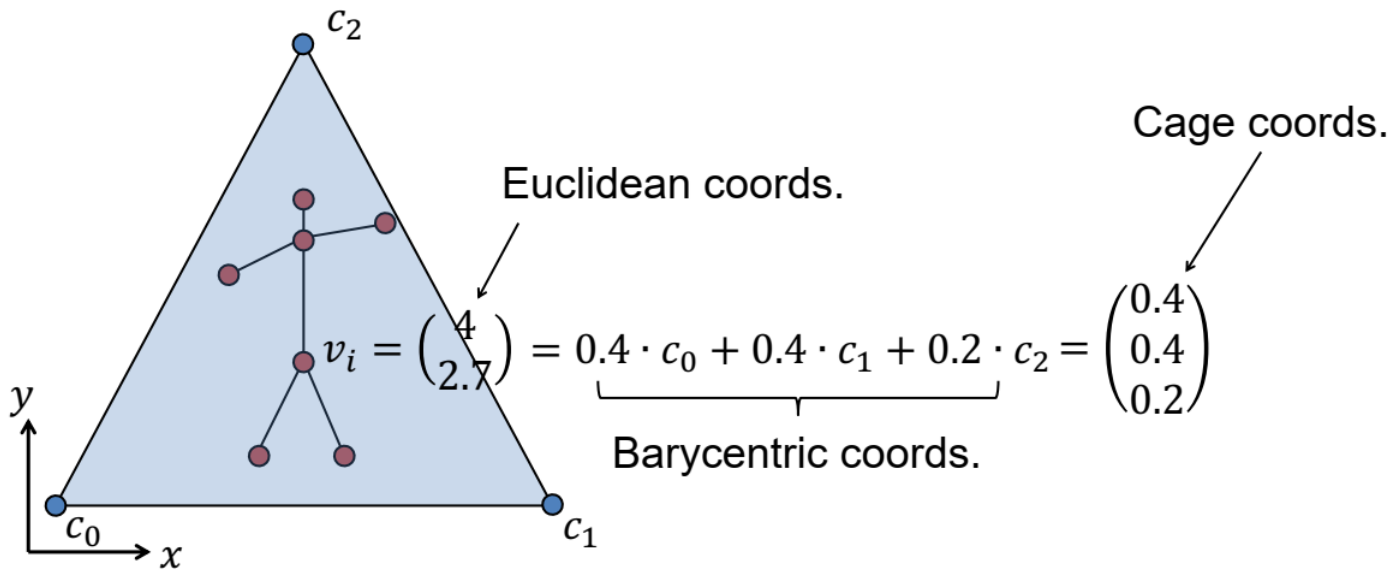
- Idea: deformation energy is computed on proxy
- Make it sparser!
  - Fewer degrees of freedom for deformation + more geometric detail





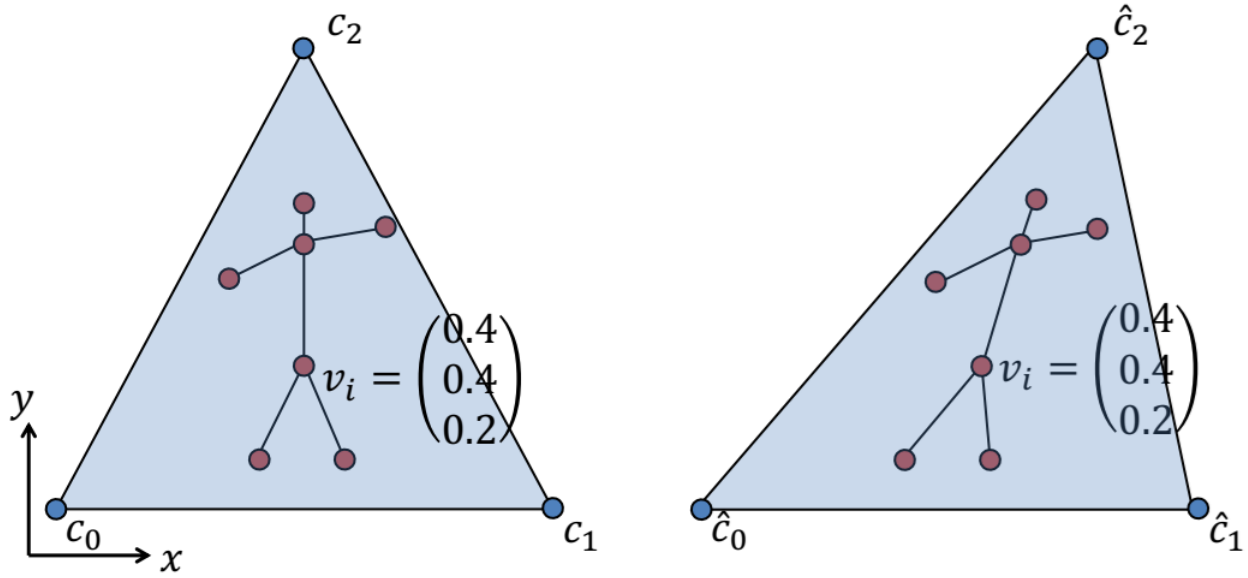
Deformation Proxies: Cage

- Extend the concept of barycentric coordinates to shapes (e.g., harmonic coordinates)
- Vertices are a linear combination with respect to cage



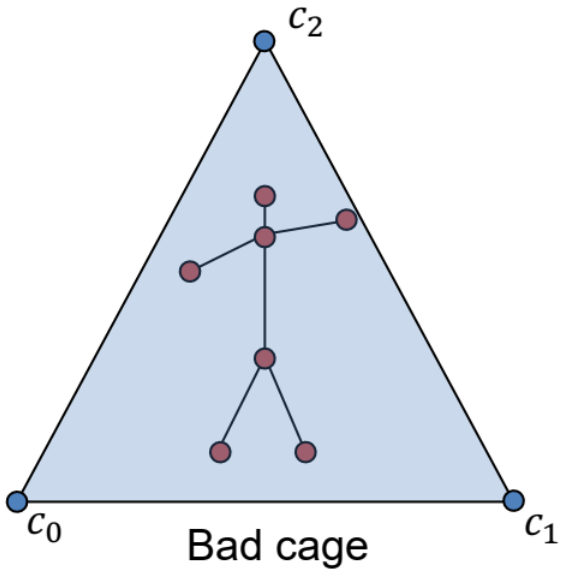
Deformation Proxies: Cage

- Extend the concept of barycentric coordinates to shapes (e.g., harmonic coordinates)
- Vertices are a linear combination with respect to cage

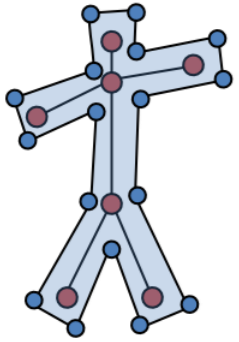


Deformation Proxies: Cage

- Extend the concept of barycentric coordinates to shapes (e.g., harmonic coordinates)
- Vertices are a linear combination with respect to cage



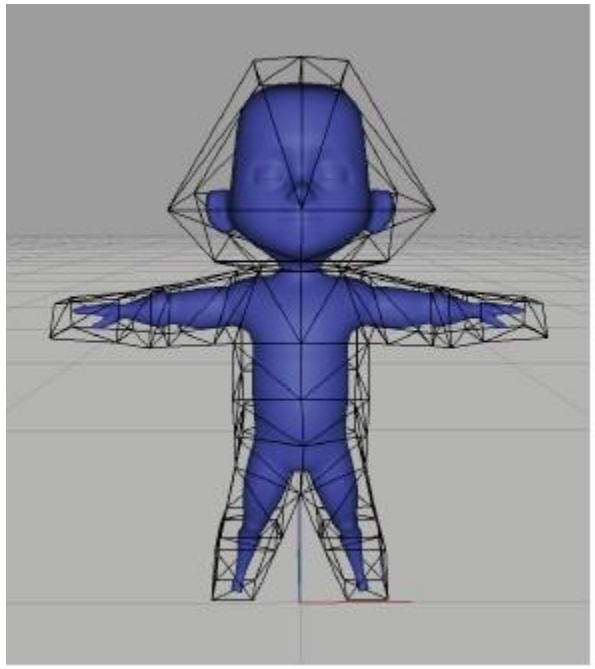
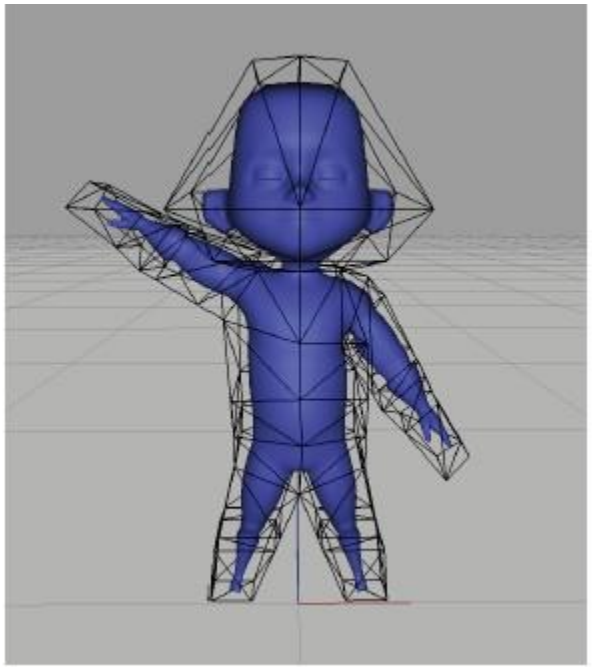
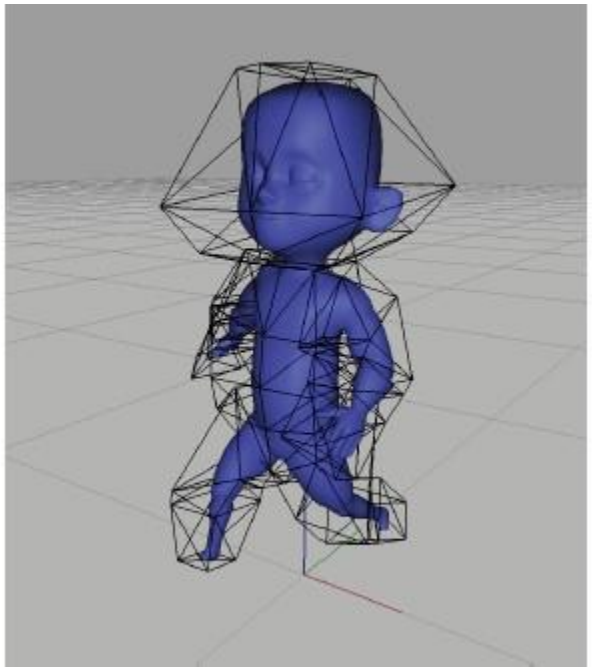
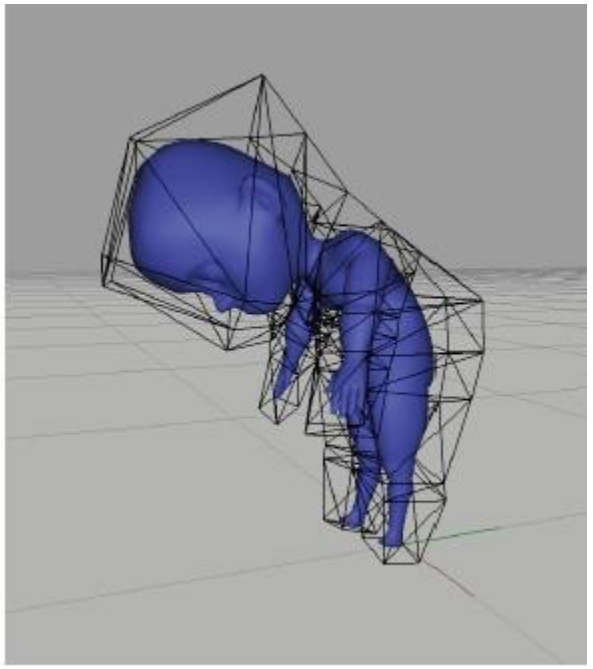
Bad cage


Good cage

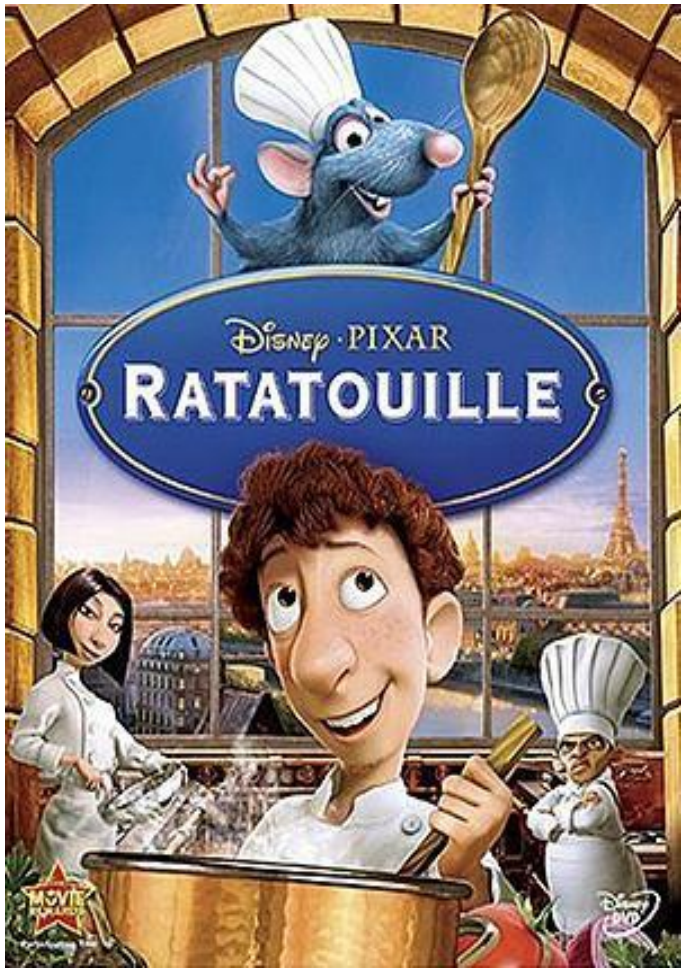
Mesh should be tightly enclosed in the cage.
It should also offer sufficient flexibility to pose a model

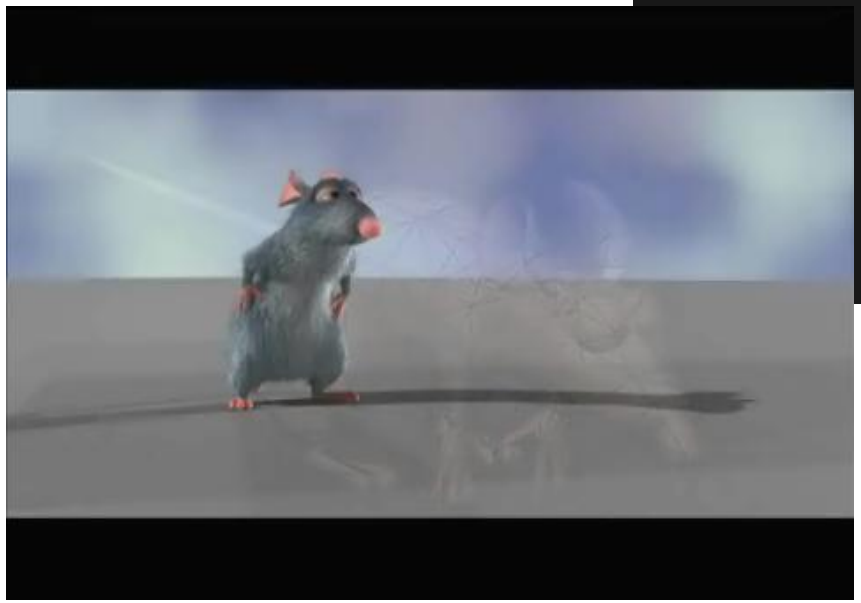
Deformation Proxies: Cage

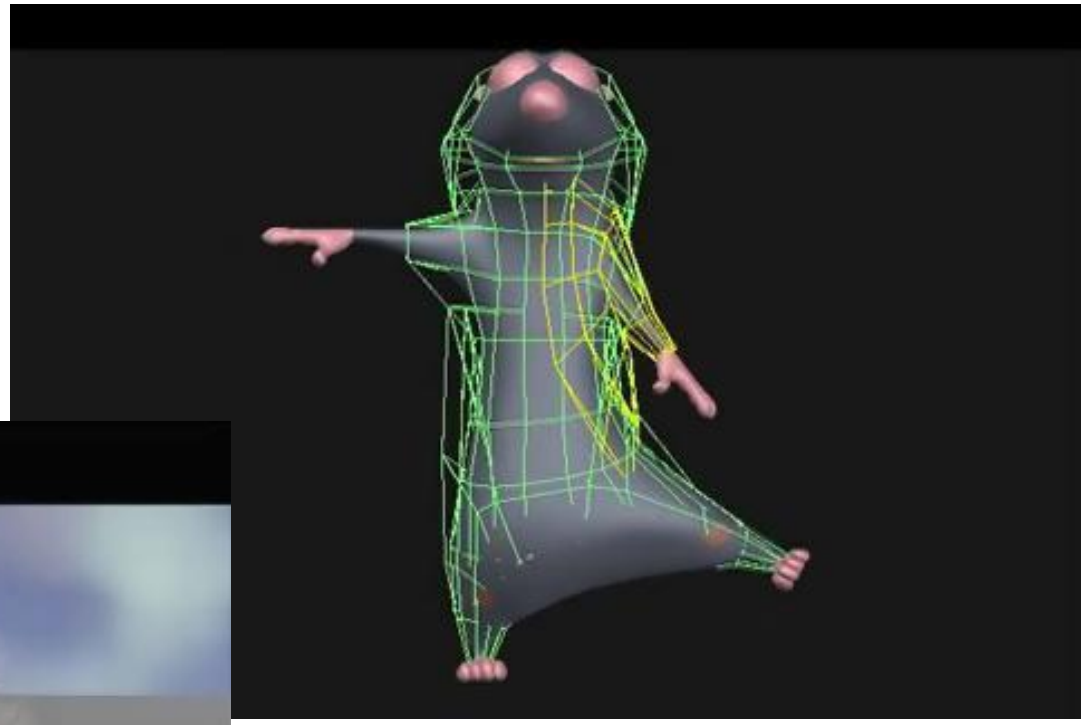
- Extend the concept of barycentric coordinates to shapes (e.g., harmonic coordinates)
- Vertices are a linear combination with respect to cage

Deformation Proxies: Cage

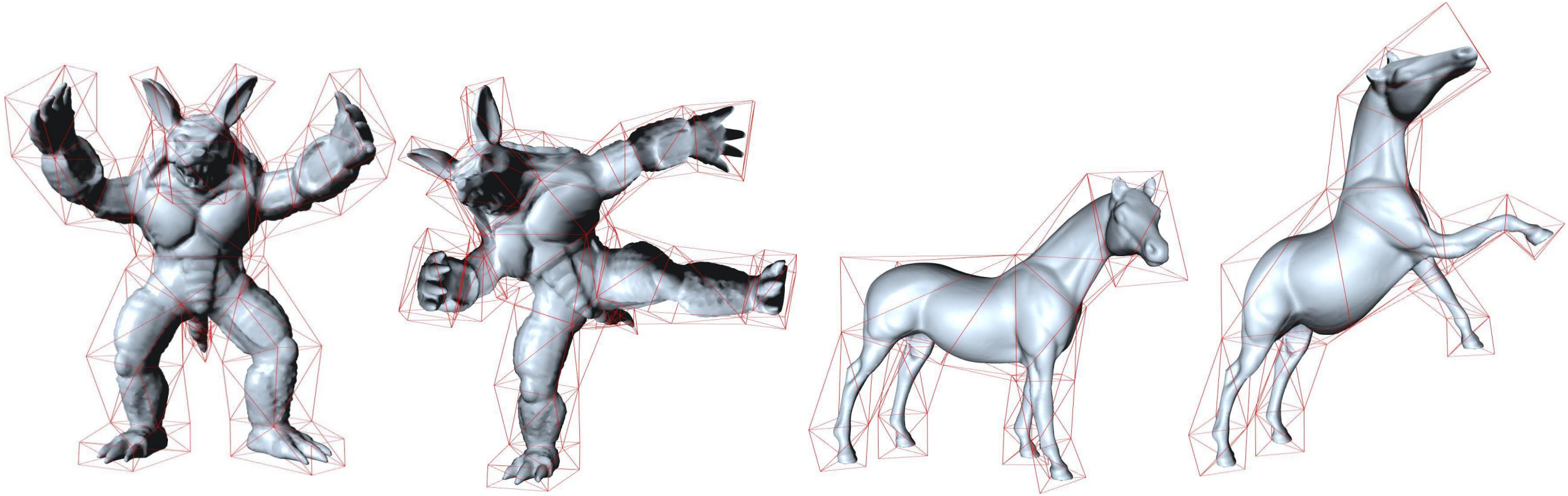




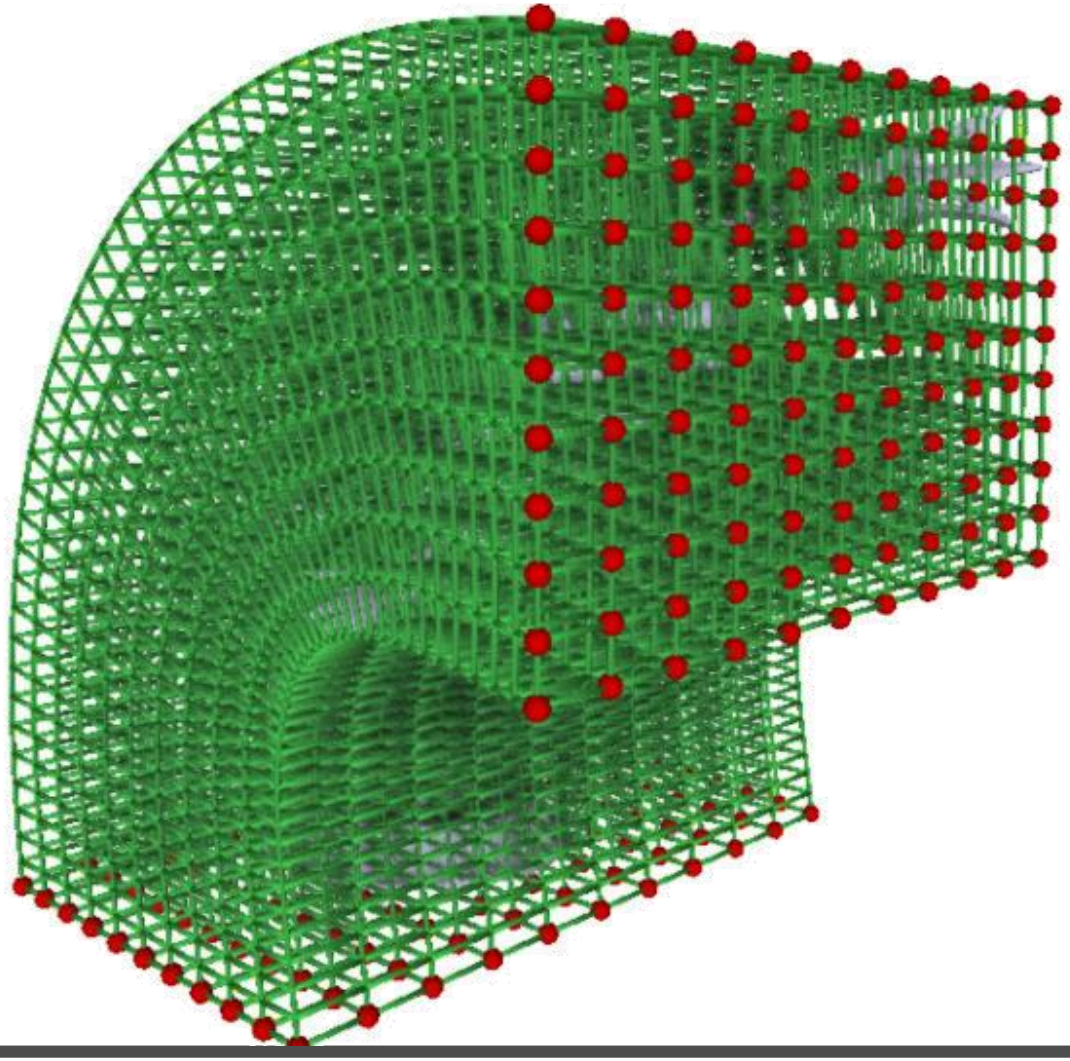


Deformation Proxies: Cage

- Once cage and weights are computed, mesh can be deformed by animating the cage vertices



ARAP on 3D Grids

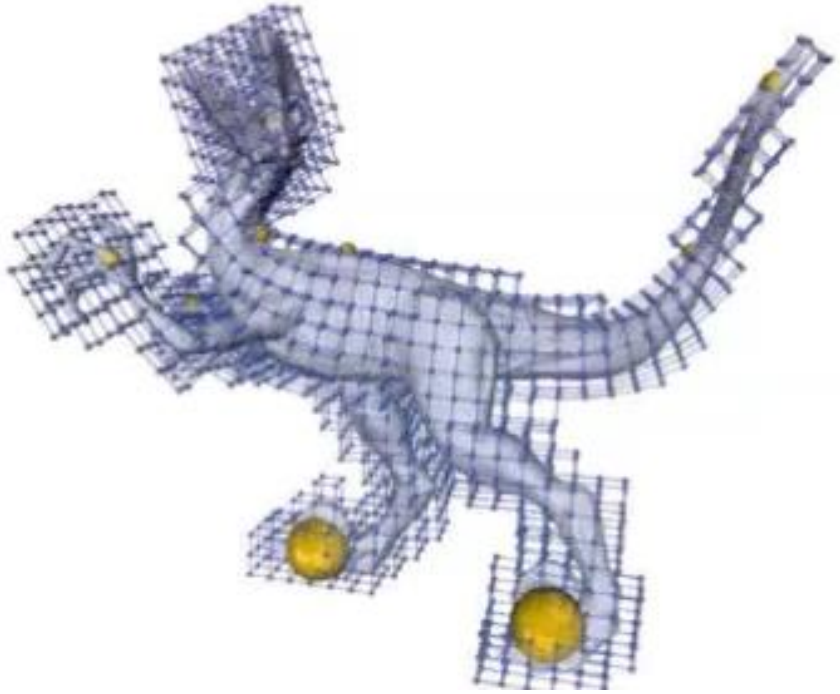


ARAP on 3D Grids

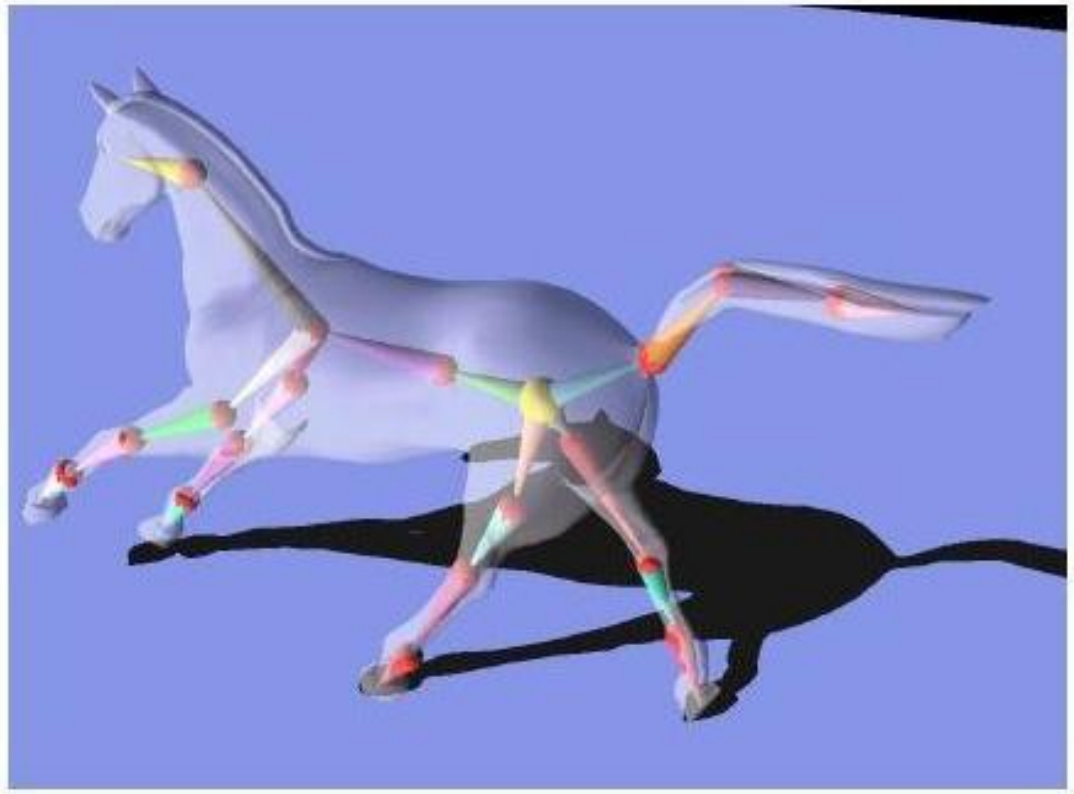
GPU based ARAP Deformation using Volumetric Lattices

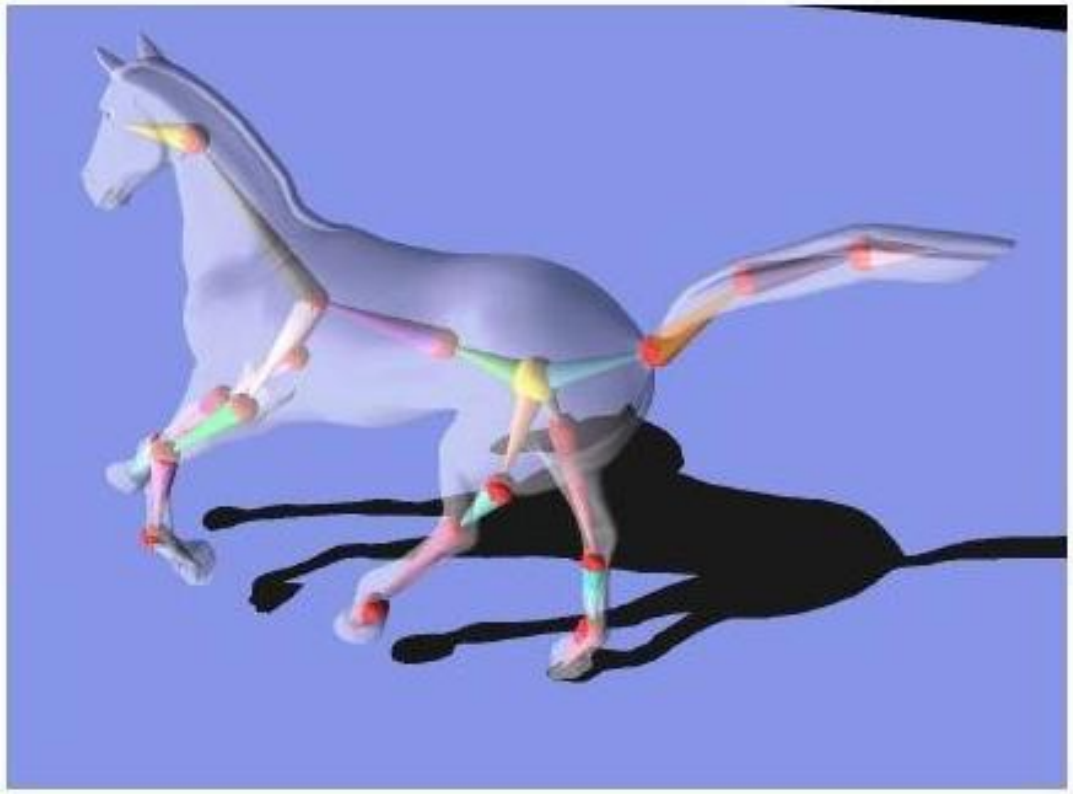
Michael Zollhöfer, Ezgi Sert, Günther Greiner and Jochen Süßmuth

Computer Graphics Group, University Erlangen-Nuremberg, Germany



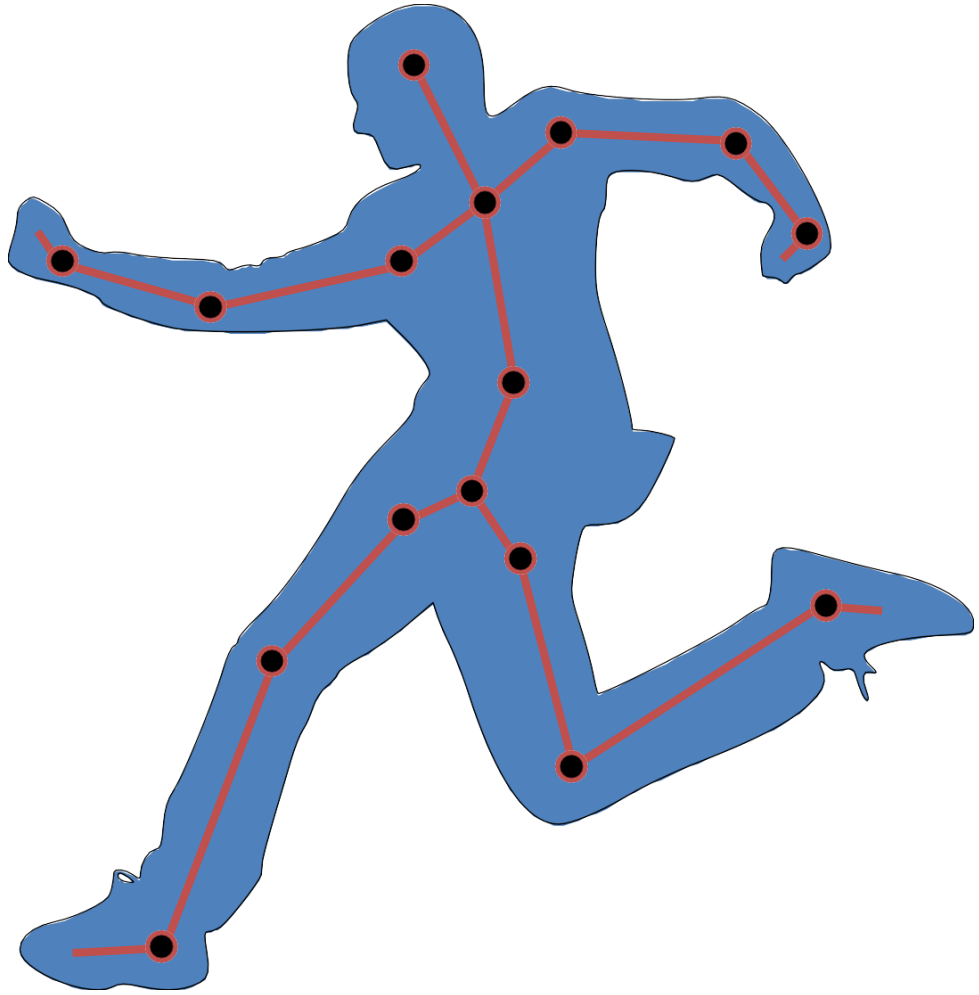
Deformation Proxies: Skeleton





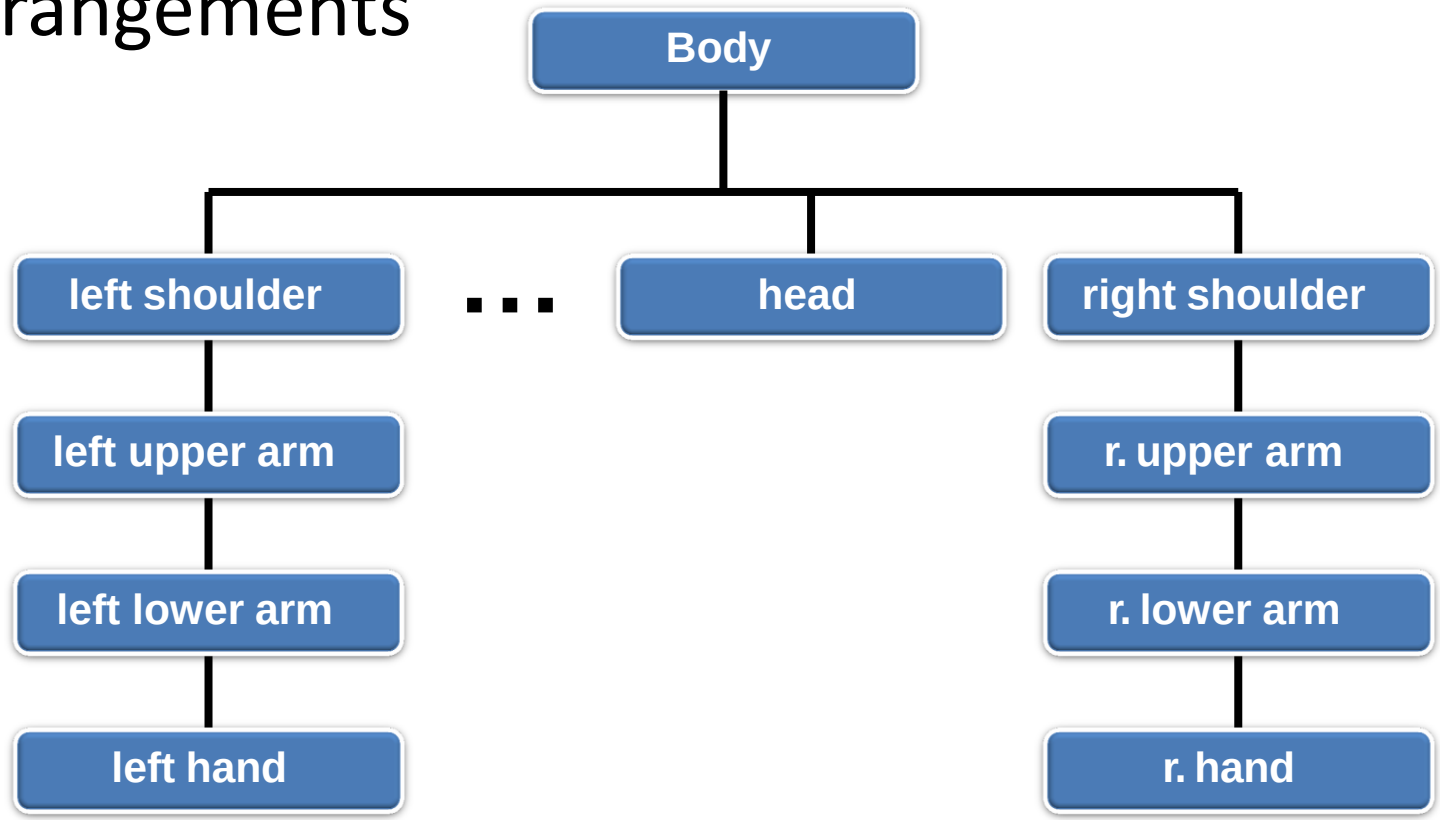
Deformation Proxies: Skeleton

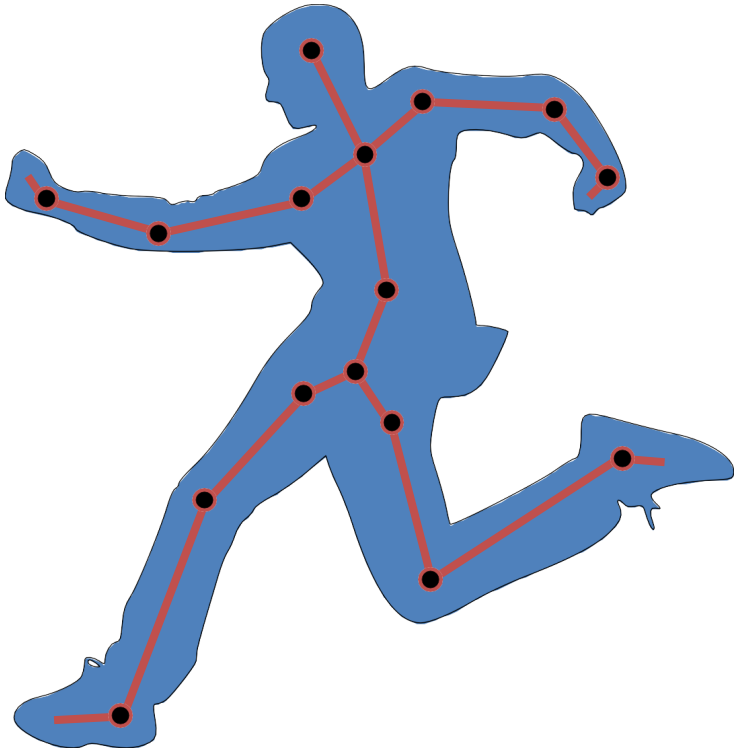
- Basic idea:
  - Embed skeleton in mesh
  - Assign mesh vertices to bones
  - Compute deformed mesh from bones



Deformation Proxies: Skeleton

- Skeleton is typically created by hand and has hierarchical arrangements



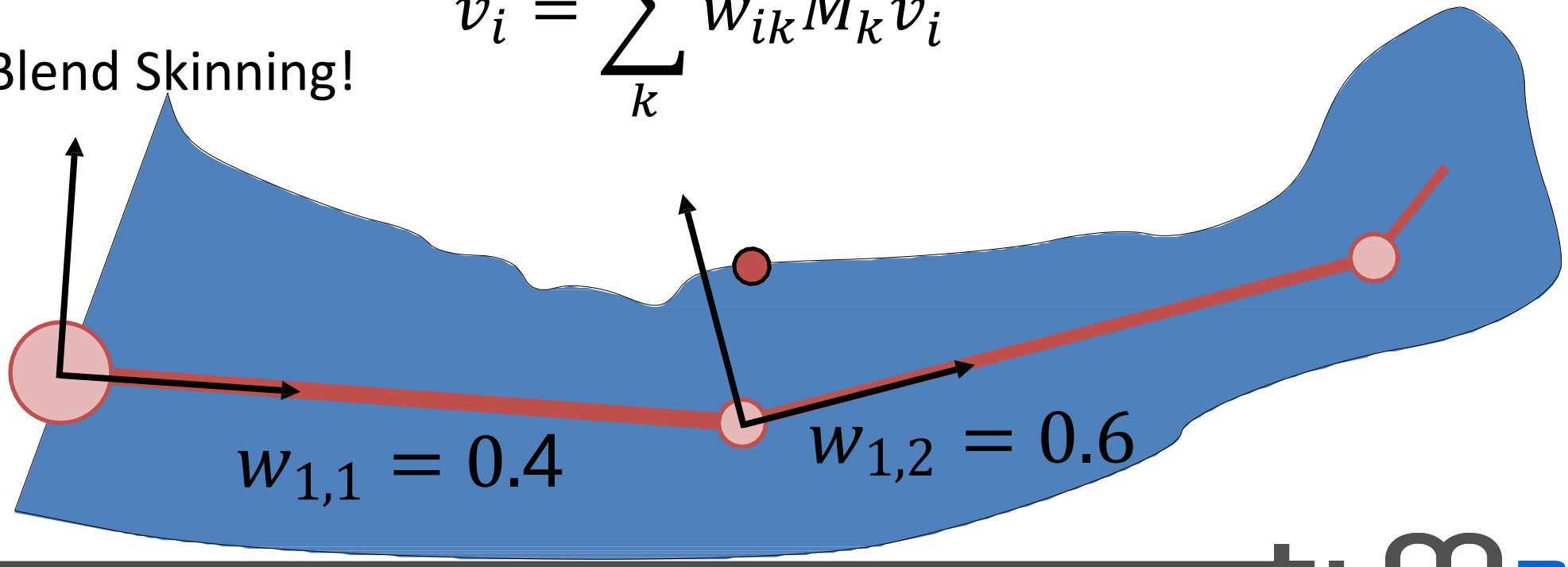


Deformation Proxies: Skeleton

- Each vertex is linear combination of bone matrices
- Assign weights for each vertex for each matrix

$$v_i = \sum_k w_{ik} M_k v_i'$$

Linear Blend Skinning!



Deformation Proxies: Skeleton

EG'17 [von Marcard et al.]: Sparse Inertial Poser

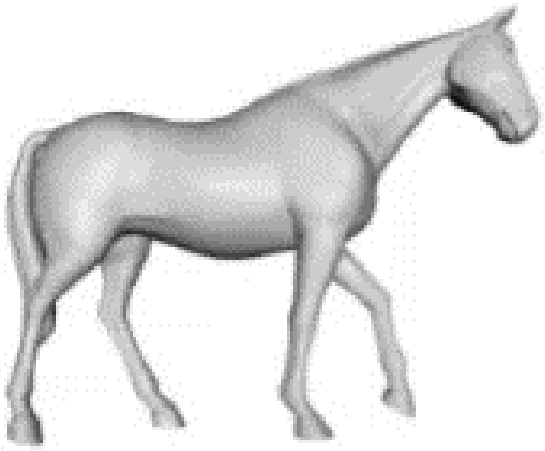
Deformation Proxies: Skeleton

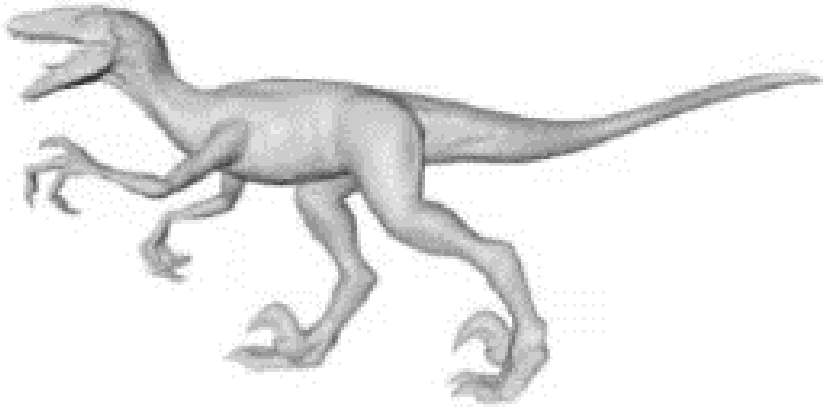
EG'17 [von Marcard et al.]: Sparse Inertial Poser

ARAP on Skeletons

Pseudo-Skeleton based ARAP Mesh Deformation

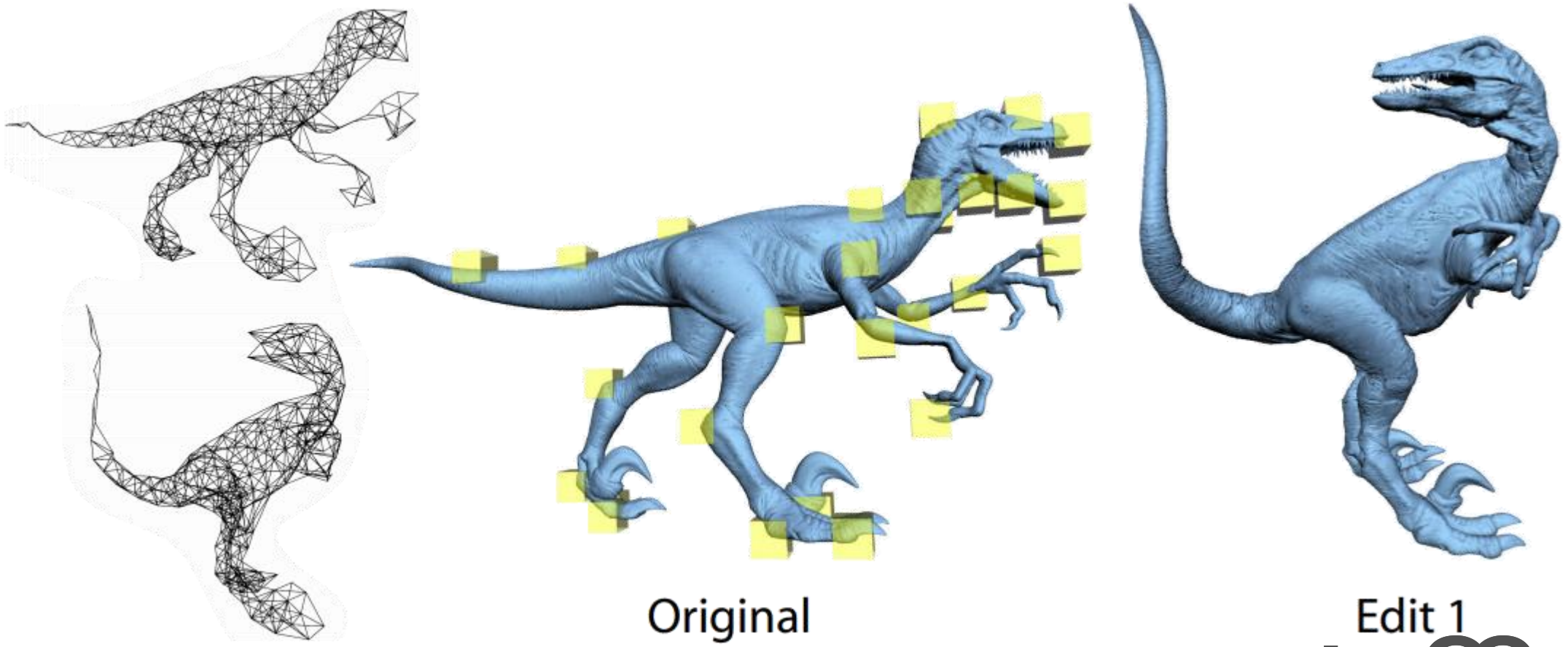
M. Zollhöfer, A. Vieweg, J. Süßmuth and G. Greiner



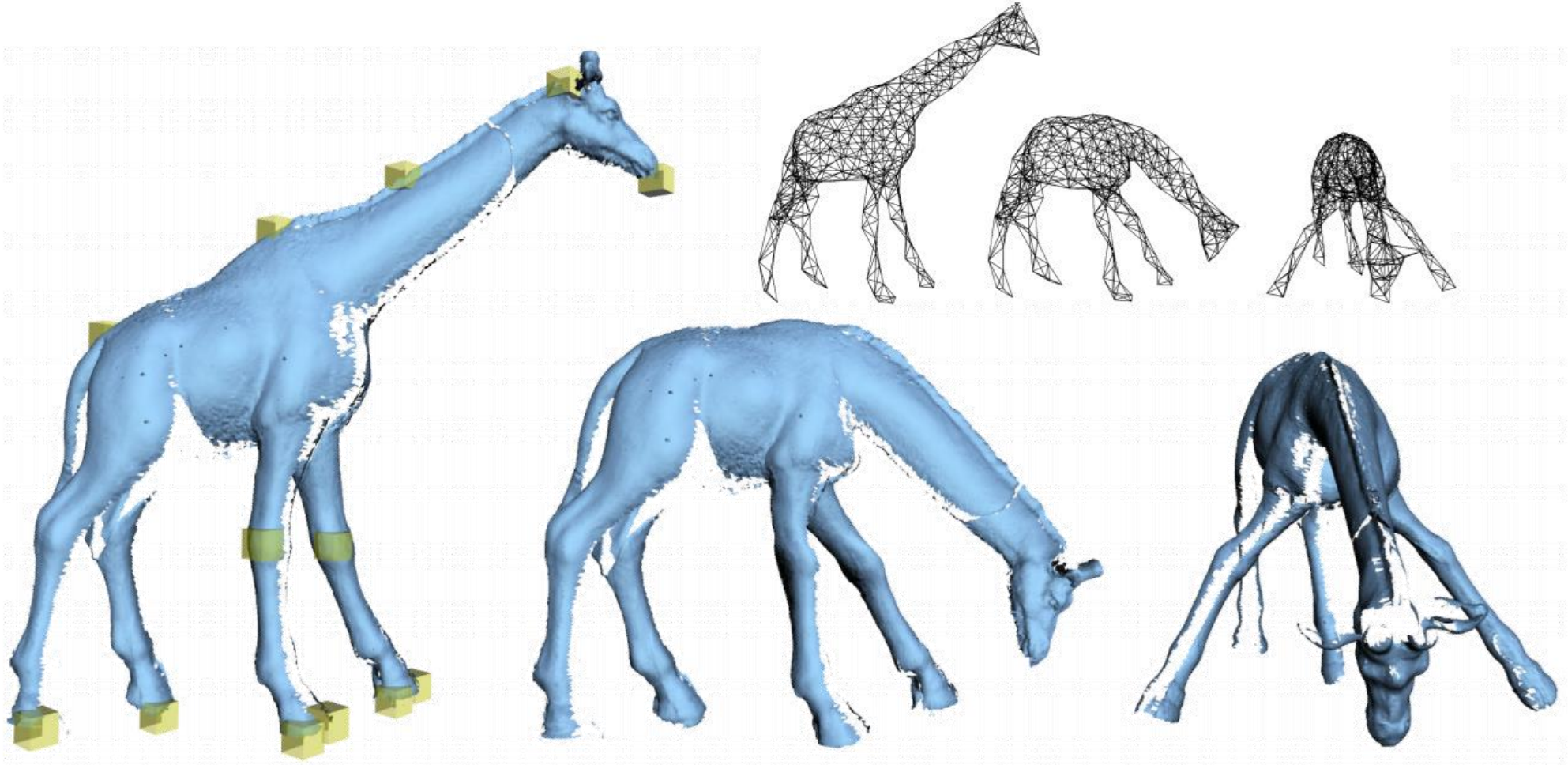


Deformation Graphs (ED)

- Introduced with the ED-paper

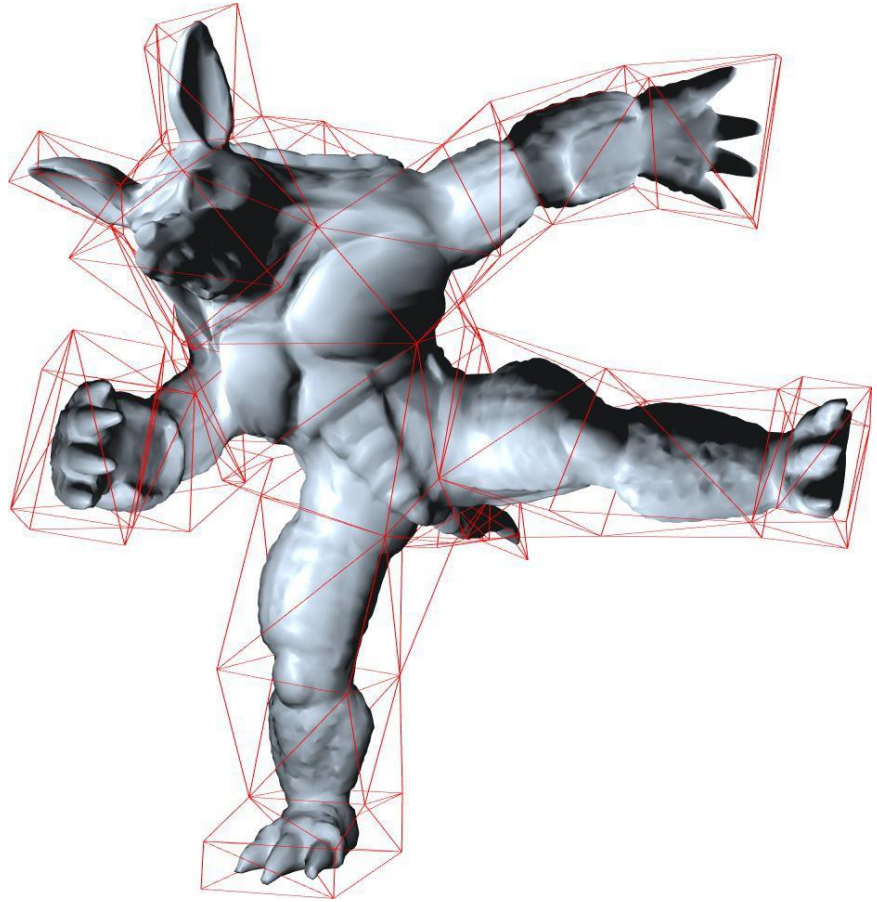


Deformation Graphs (ED)



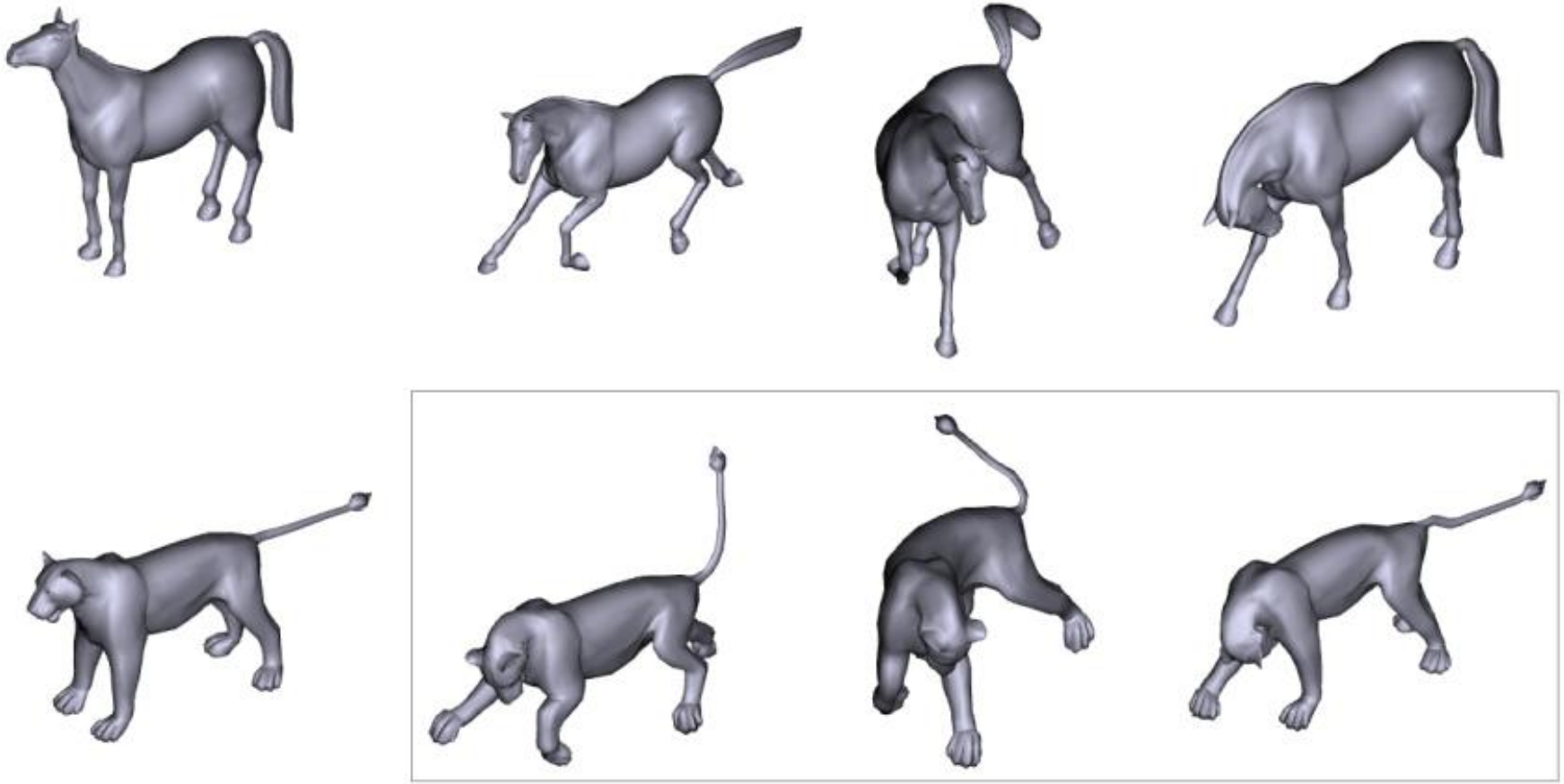
Summary: Deformation Proxies

- Dimensionality reduction on deformations



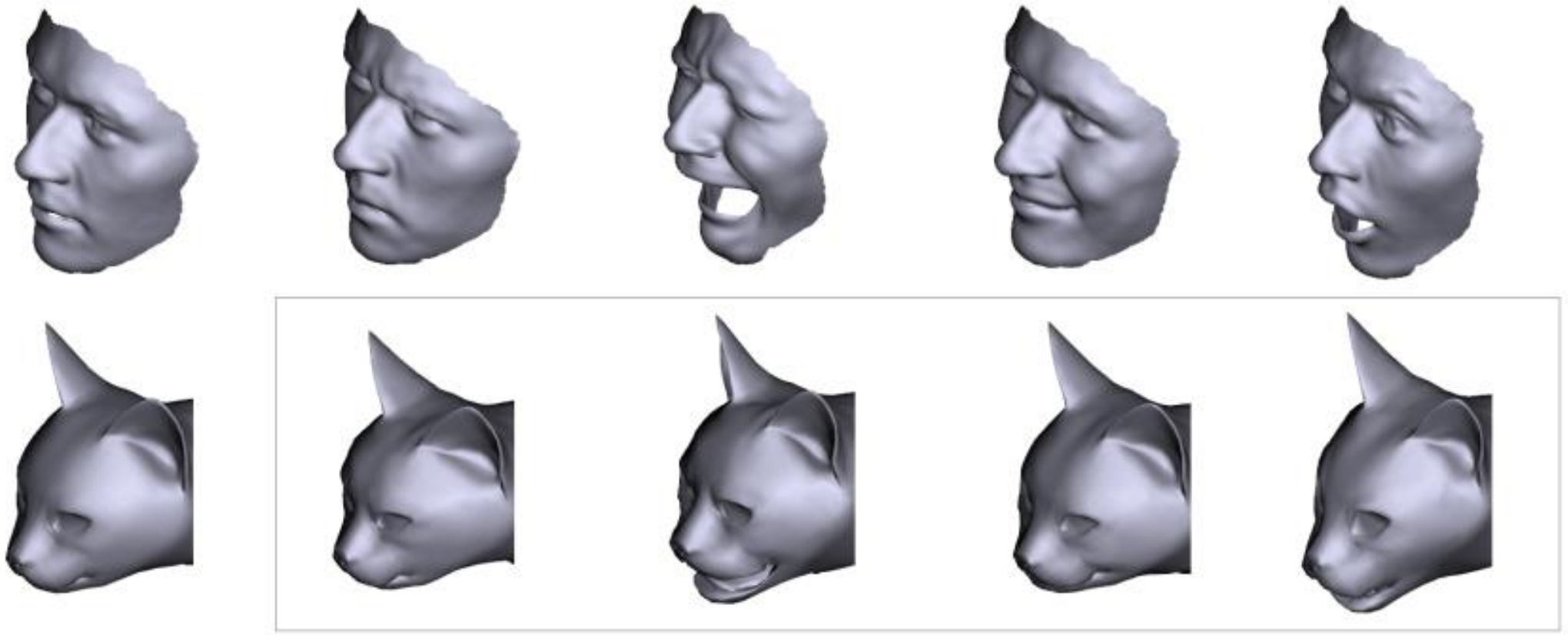
- Directly on Mesh
- Cages
- Grids
- Skeletons (e.g., Linear Blend Skinning)
- Deformation Graphs
- Tetrahedral Mesh (full volume)

Side Note: Deformation Transfer



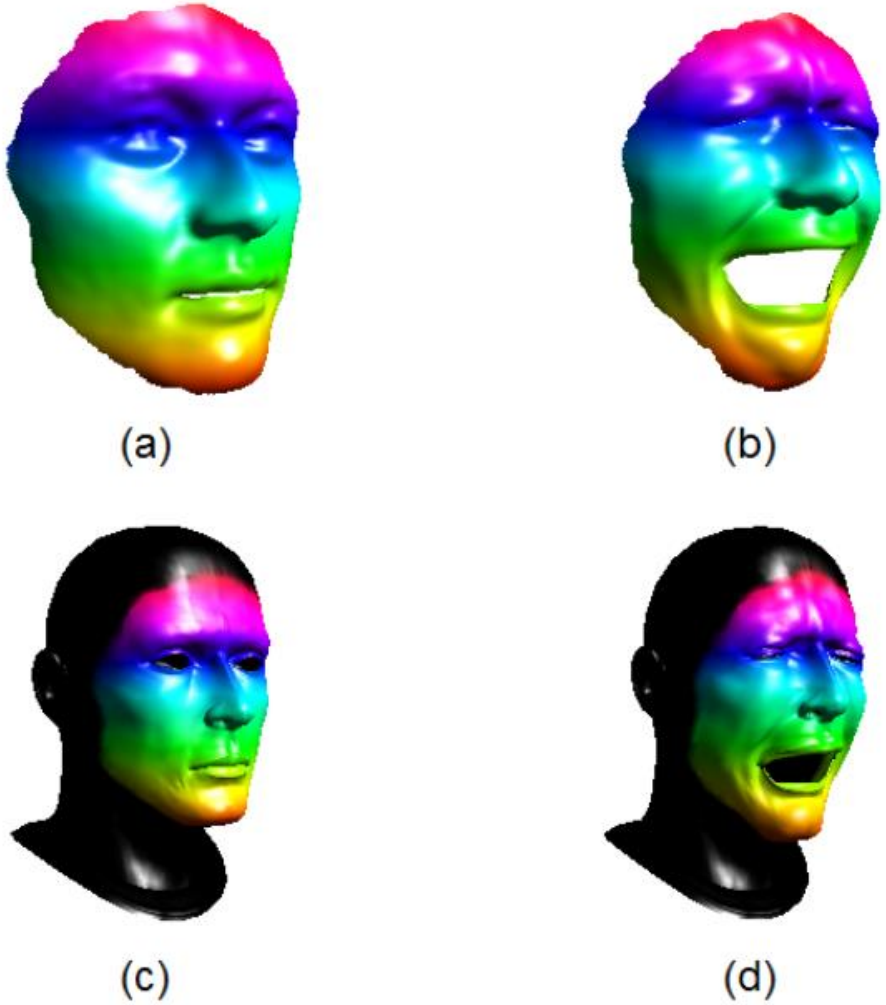
[Sumner et al. 06] [Thies 11]

Side Note: Deformation Transfer



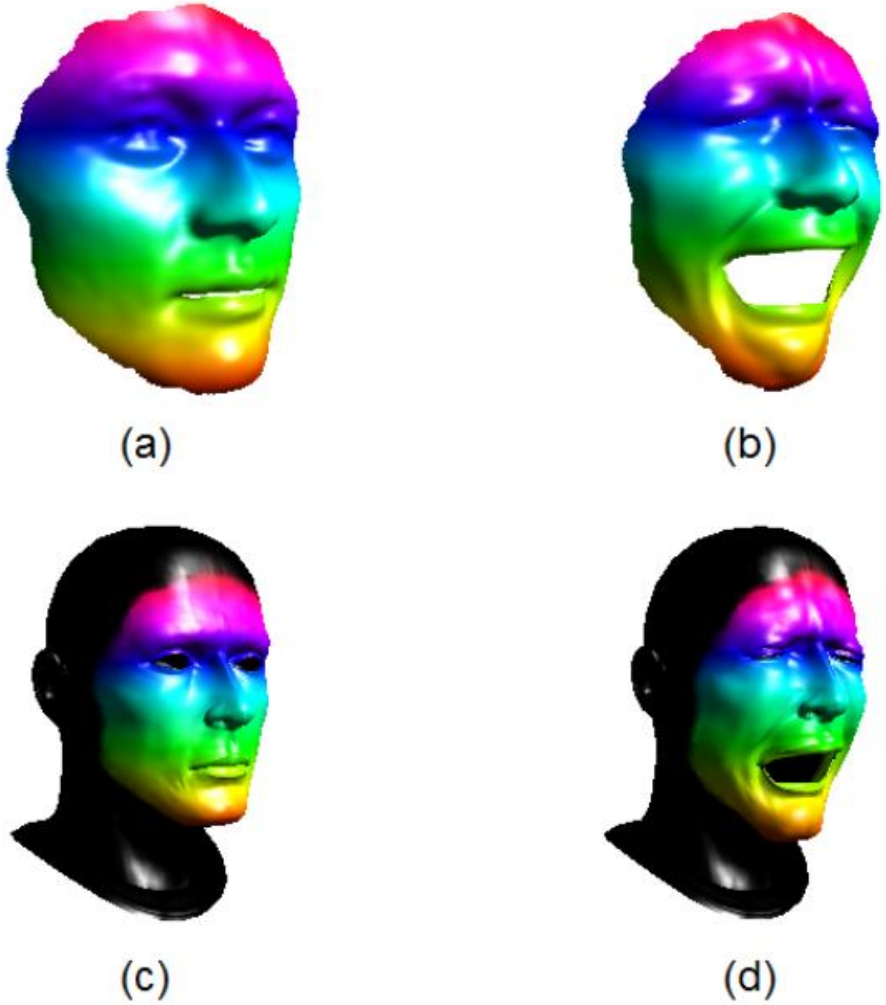
[Sumner et al. 06] [Thies 11]

Side Note: Deformation Transfer



[Sumner et al. 06] [Thies 11]

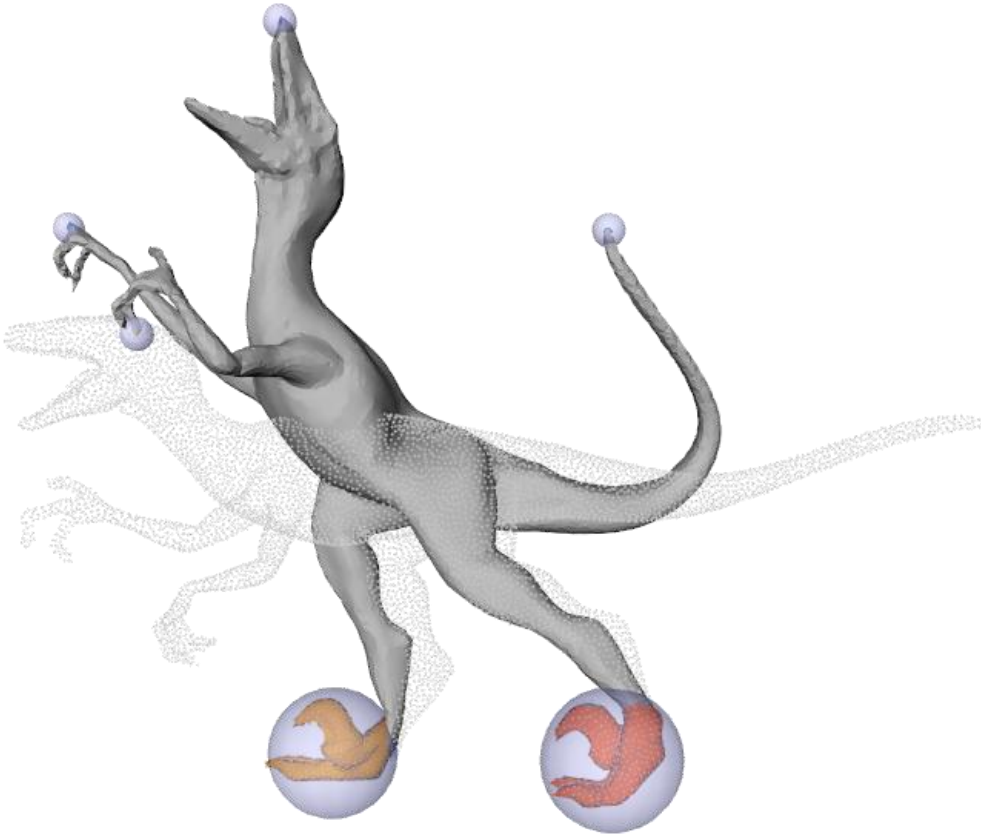
Side Note: Deformation Transfer



[Sumner et al. 06] [Thies et al. 16]

Non-rigid Tracking

- We know how to deform a mesh
  - Regularizers + Deformation proxies
  - User picks handles to edit the mesh
    - →constraints for deformation



- Tracking / Fitting:
  - Find these correspondences from data!
  - E.g., Non-rigid ICP
  - E.g., Sparse Features (e.g., SIFT, SURF)

Non-rigid ICP / Non-rigid Registration

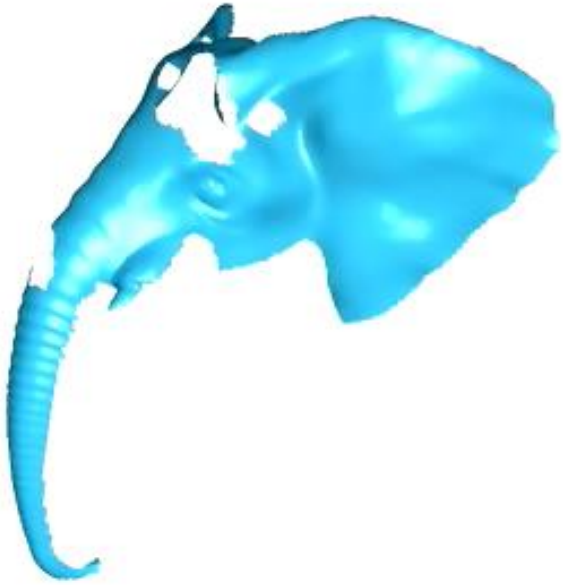
- Like rigid ICP (which has 6 DoF / frame)
  - Except we have many more degrees of freedom!

iterate

- Find correspondences
  - ➢closest points or projective correspondences
- Solve for deformation and vertices

Non-rigid ICP / Non-rigid Registration

Elephant (329 nodes, 21k vertices)



Source

CGF'08 / ToG'09 [Li et al. 08/09]: Robust Single-View Geometry and Motion Reconstruction

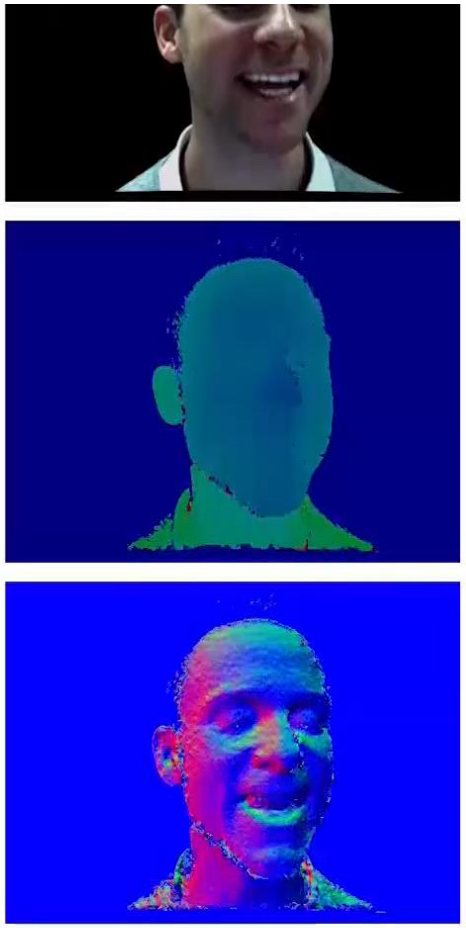
Non-rigid ICP / Non-rigid Registration

Bending Arm (138 nodes, 37k vertices)

Source

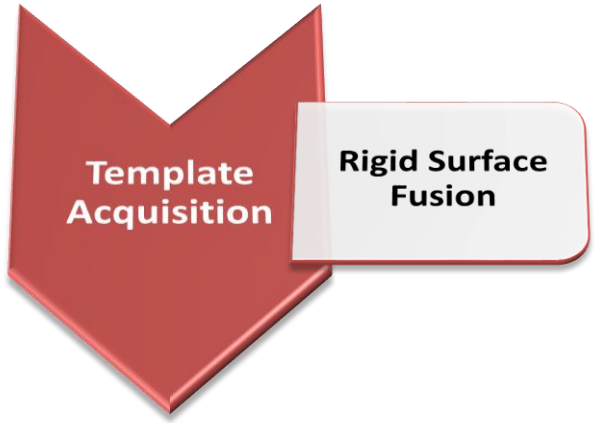
CGF'08 / ToG'09 [Li et al. 08/09]: Robust Single-View Geometry and Motion Reconstruction

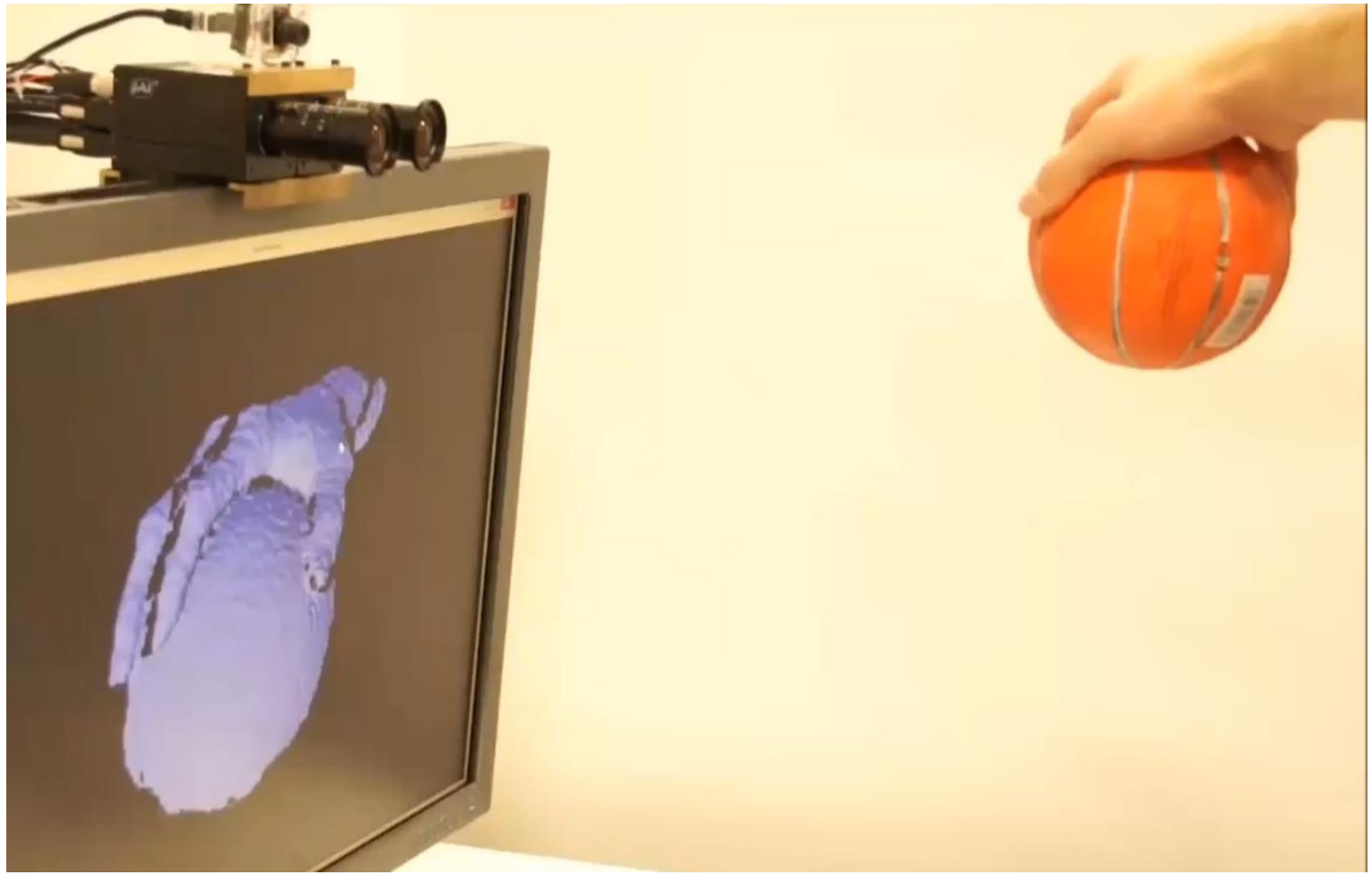
Non-rigid Tracking



TOG'14 [Zollhöfer et al.]: Deformables

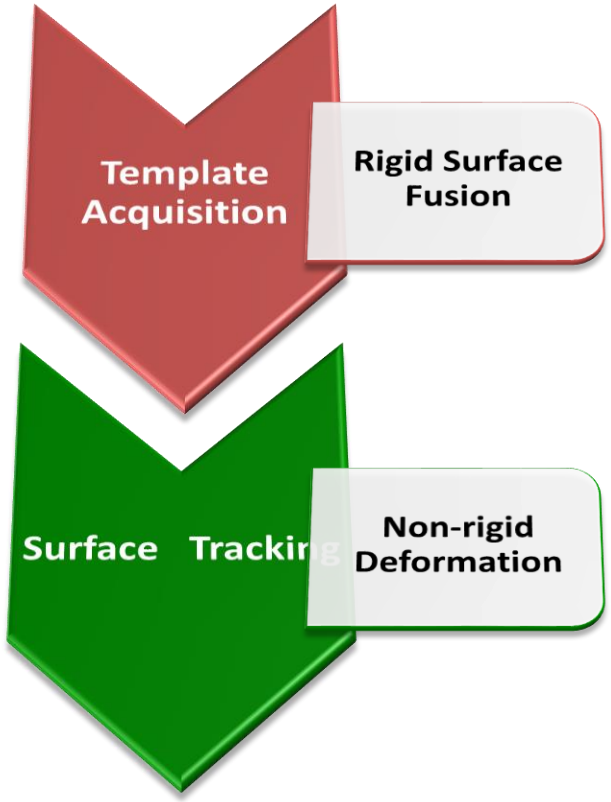
Non-rigid Tracking

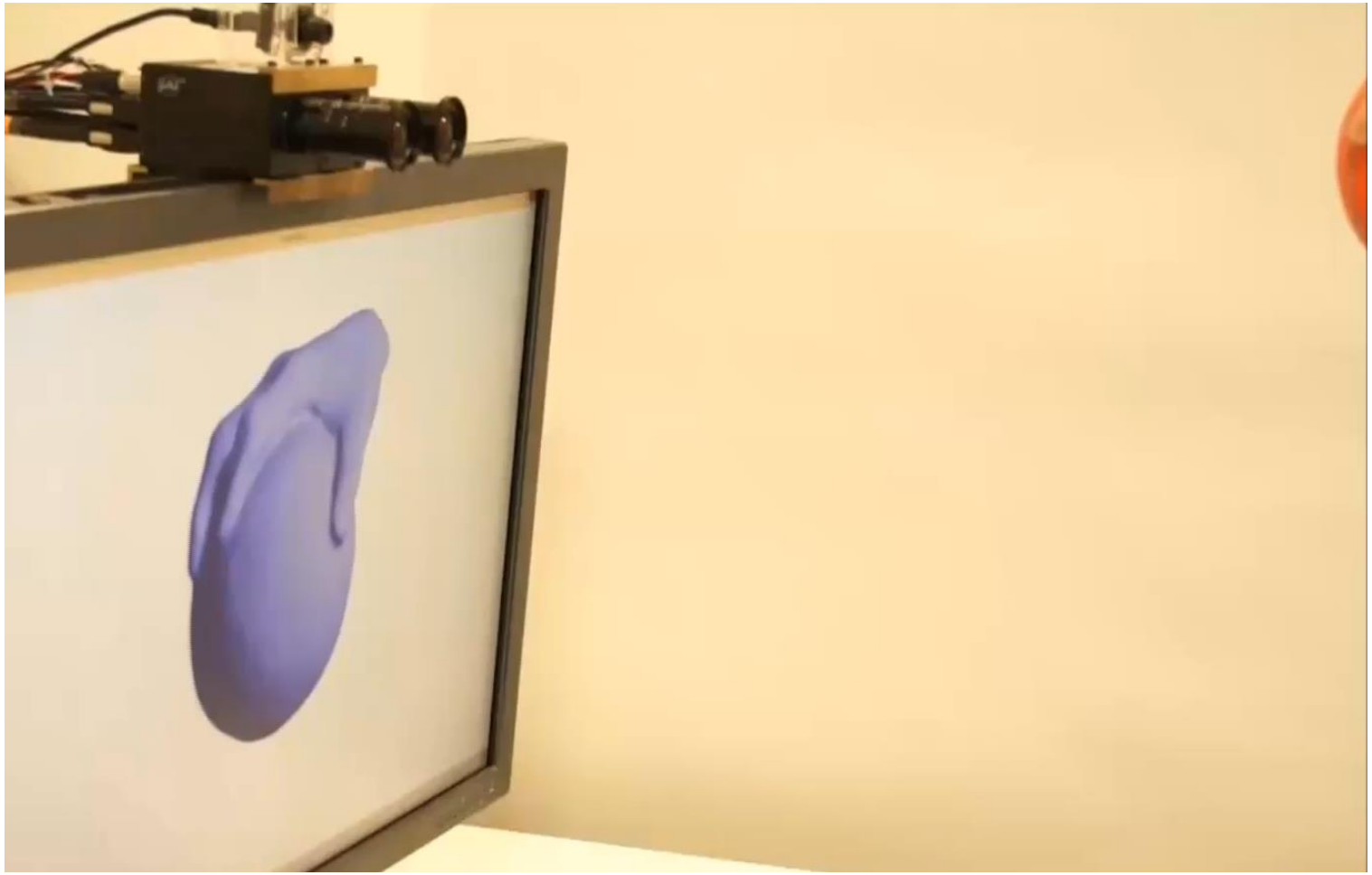




TOG'14 [Zollhöfer et al.]: Deformables

Non-rigid Tracking





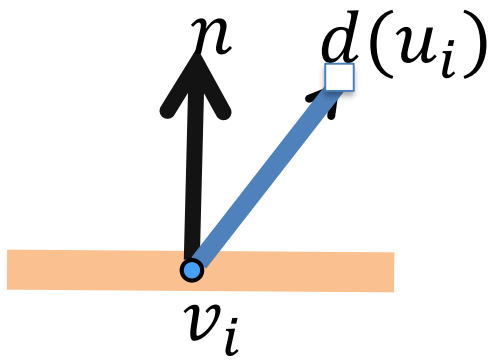
TOG'14 [Zollhöfer et al.]: Deformables

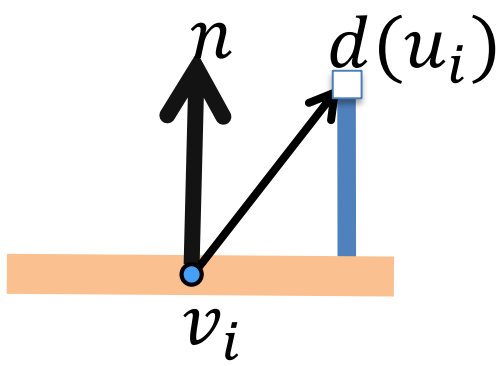
Non-rigid Tracking: Objective

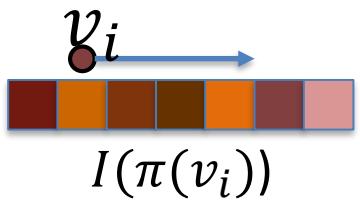
Data Terms

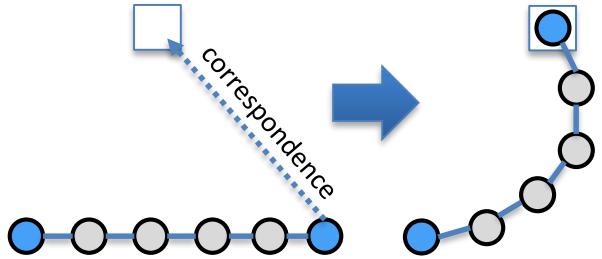
Regularization

$$E = \lambda_{point} E_{point} + \lambda_{plane} E_{plane} + \lambda_{color} E_{color} + \lambda_{reg} E_{reg}$$









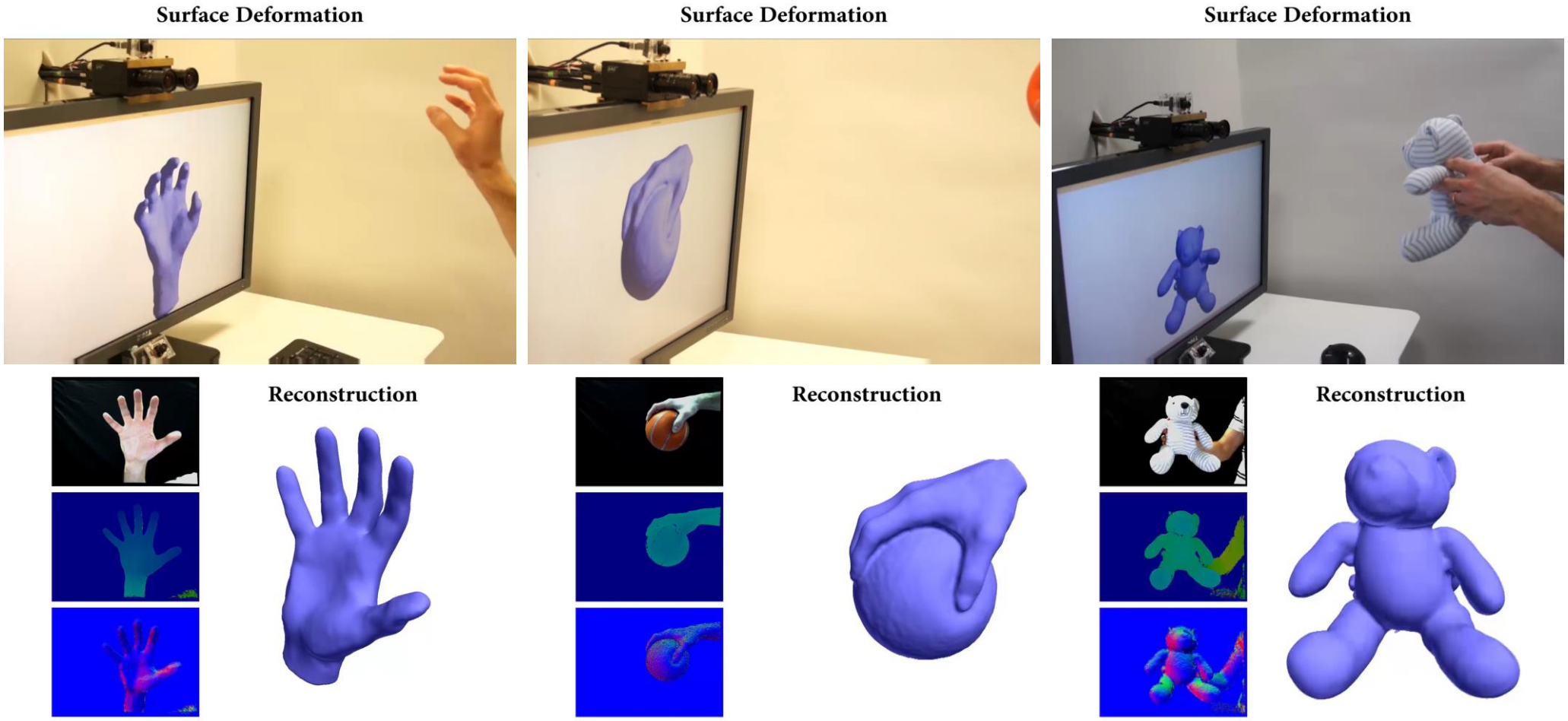
[Sorkine07]

Non-rigid Tracking: Objective

- $E_{point} = \sum_i \left|\left|v_i - Tv_i'\right|\right|_2^2$
- $E_{plane} = \sum_i \left[(v_i - Tv_i') \cdot n_{v_i}\right]^2$
- $E_{color} = \sum_i [I(\pi(v_i)) - I'(\pi(Tv_i))]^2$
- $E_{reg} = \text{ARAP} = \sum_i \sum_{j \in \mathcal{N}(i)} w_{ij} \left|\left|(v_i - v_j) - R_i(v_i' - v_j')\right|\right|_2^2$

Loop only over valid correspondences!

Non-rigid Tracking



TOG'14 [Zollhöfer et al.]: Deformables

Side Note: Active Stereo Depth Estimation

IR-Projector Pattern

Single lense RGB-IR cameras

1) PatchMatch stereo

2) Variational depth refinement

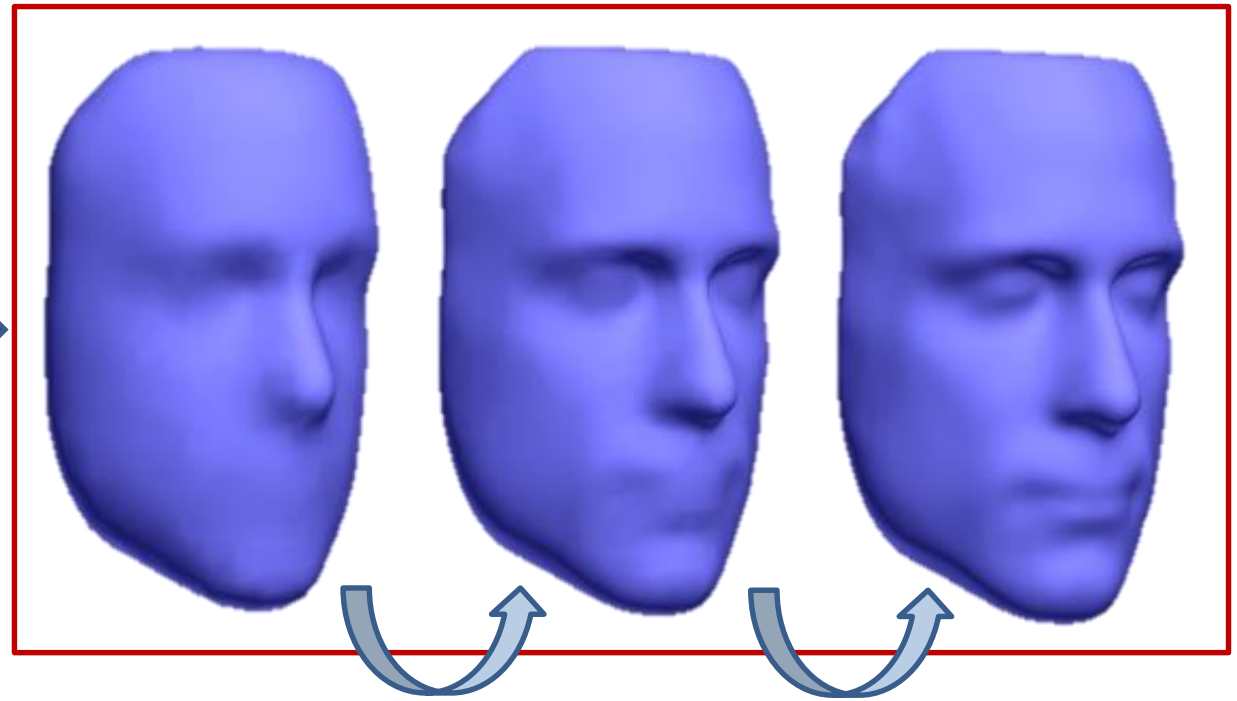
1)Template Acquisition (<1min)

Fused 3D model

[Nießner13]

Decimation & Fairing

Multi-resolution template hierarchy



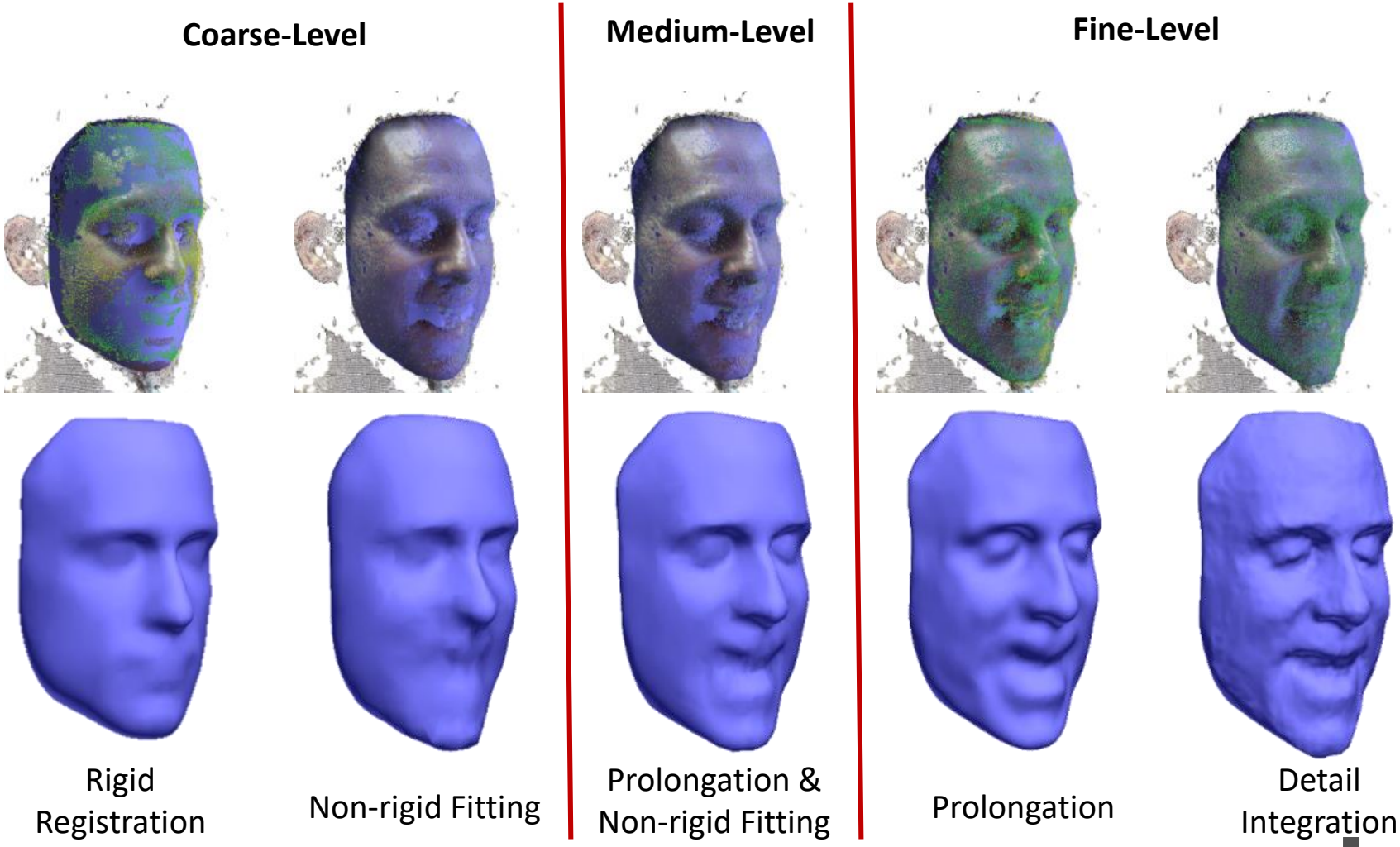
Prolongation Operator

$$v_i = \sum_{k=1}^{m} w_k(v_i)[R_k(\hat{v}_i - \hat{g}_k) + \hat{g}_k + t_k])$$

Automatic Skinning

[Sumner07]

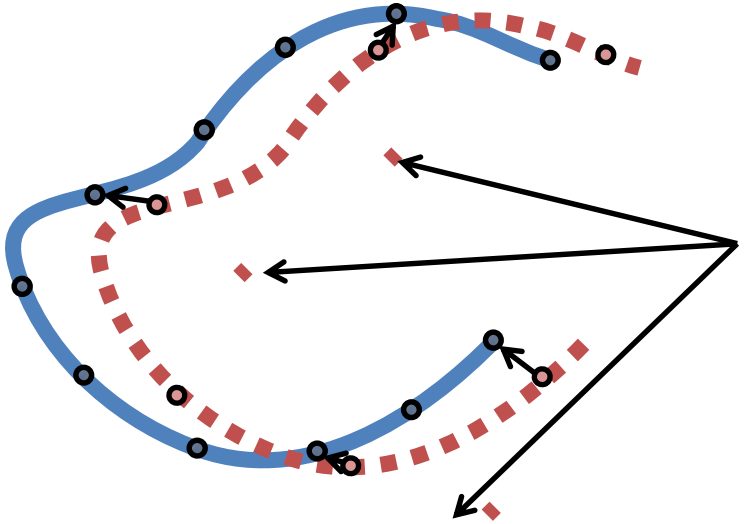
2) Surface Deformation (>30Hz)

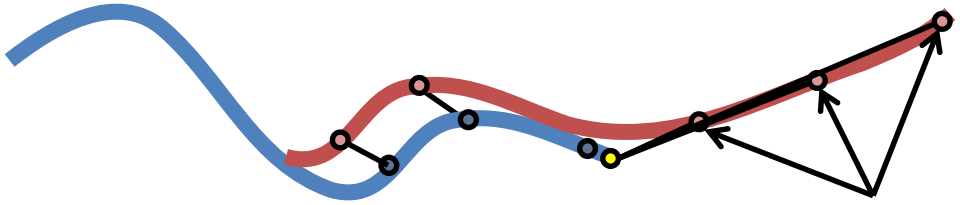


TOG'14 [Zollhöfer et al.]: Deformables

Correspondence Pruning

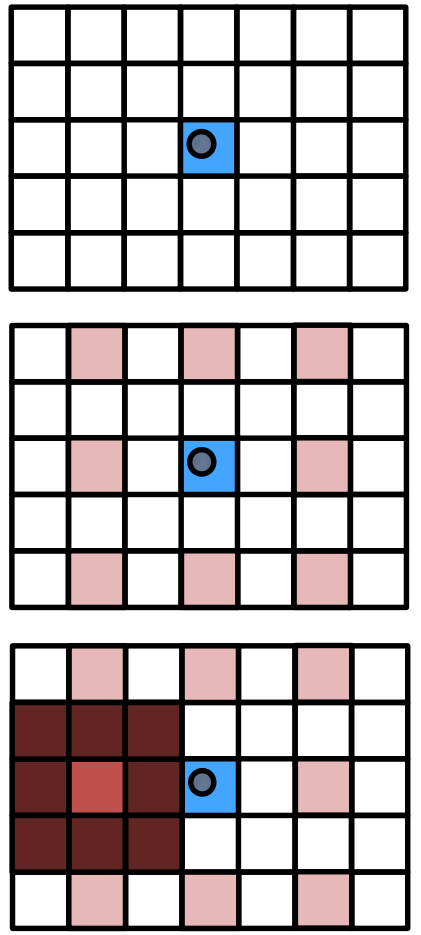
- Strategies:
  - Distance
  - Normal
  - Boundary
  - Viewing Angle
  - Occlusion





Correspondence Association

- Projective lookup
  - Search in kernel region
    - Only sample every $k-th$ element
  - One block per vertex
    - Minimum Reduction
- Refine result



Energy Minimization: Gauss-Newton on GPU

- Per frame overview:

```
for ( l ∈ |Levels| )
{
   for ( k ∈ |GaussNewtonSteps| )
   {
      precompute( blockDiag( [J^T(x^k)J(x^k)]^-1, J^T(x^k)f(x^k) );

      for ( i ∈ |PCGSteps| )
      {
            solve  ( J^T(x^k)J(x^k) h = J^T(x^k)f(x^k) );
            update( x^(k+1) = x^k − h );
            relax  ( λ_reg );
      }
   }
}
```

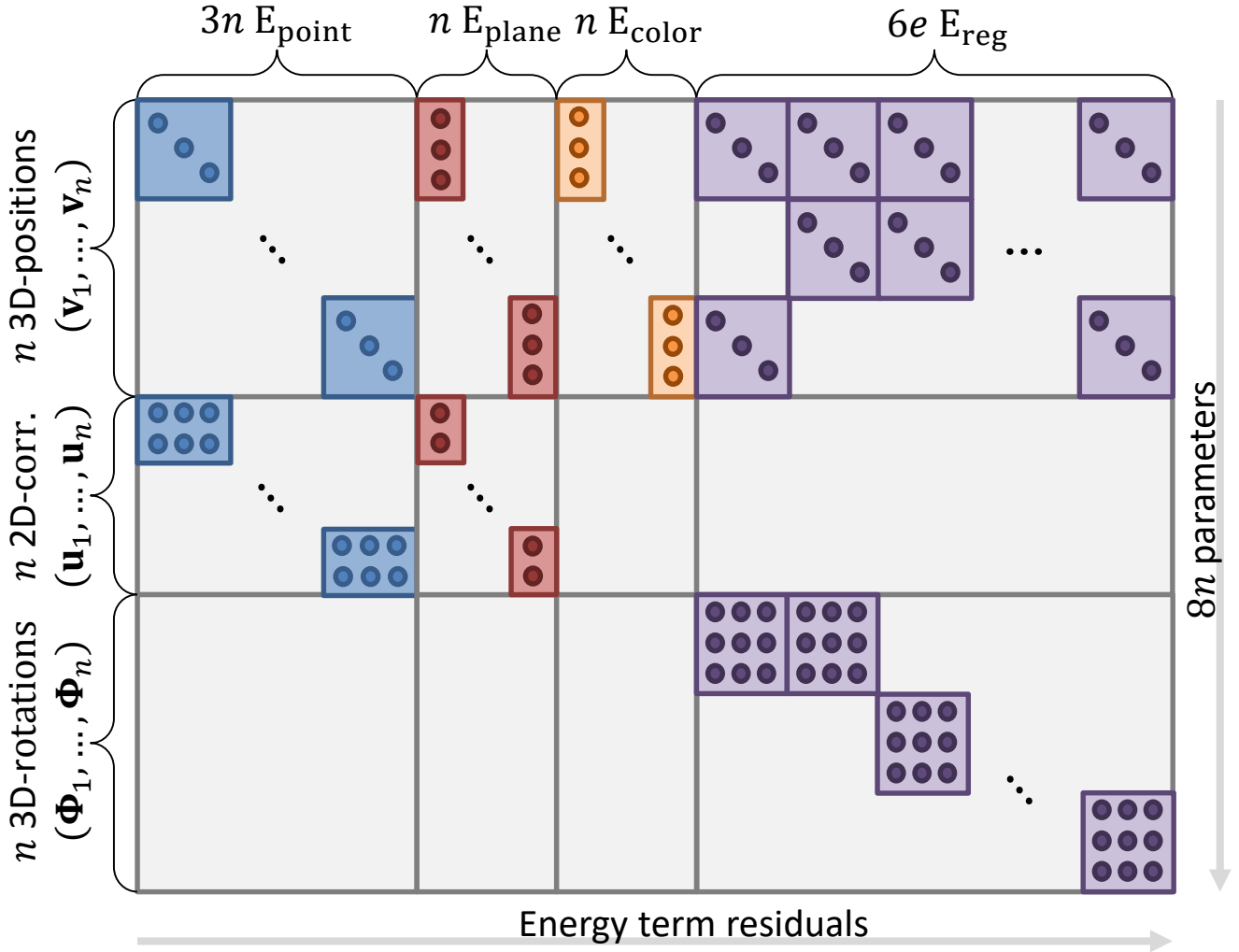
2(+1) hierarchy levels

5 Gauss-Newton iterations

10 PCG iterations

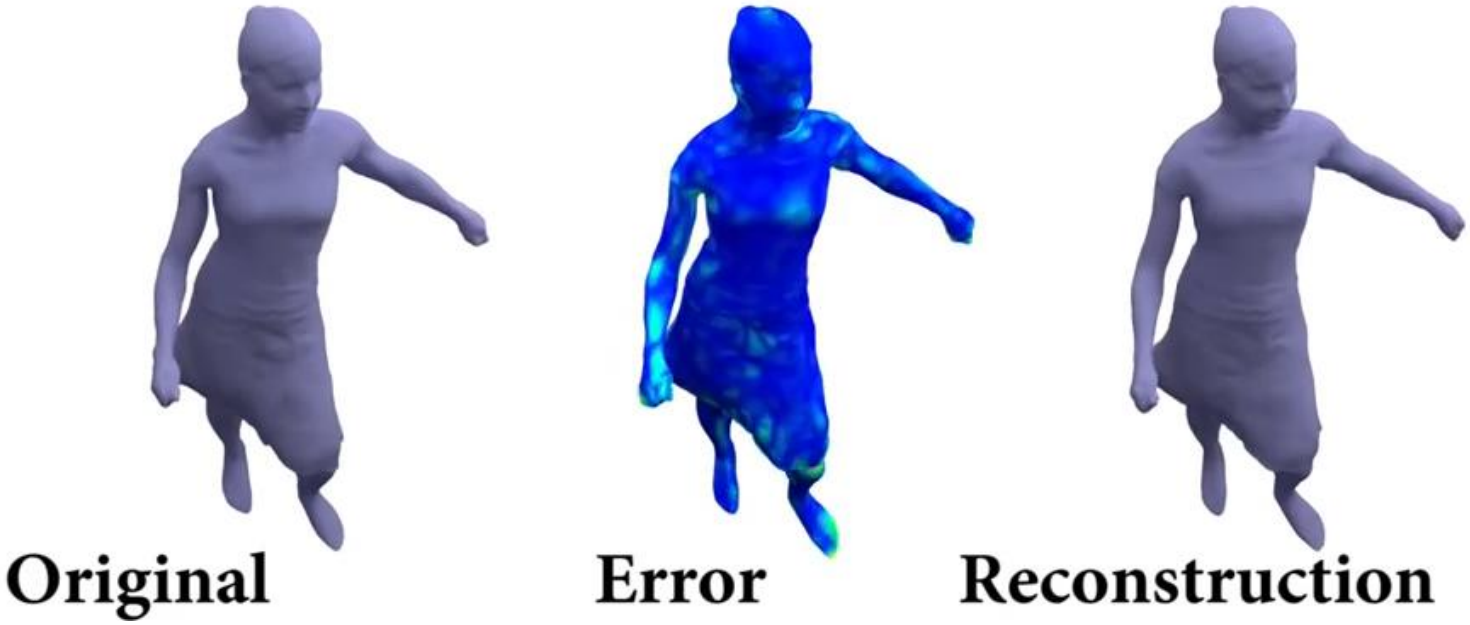
Complete runtime ~22ms!

Exploit Structure of J^T



Non-rigid Tracking

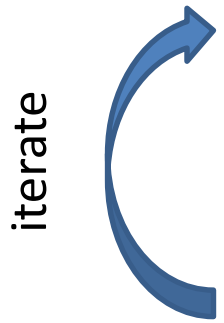
- More stable with more cameras (here 8)
  - Correspondences from each view



TOG'14 [Zollhöfer et al.]: Deformables

Non-rigid ICP / Non-rigid Registration

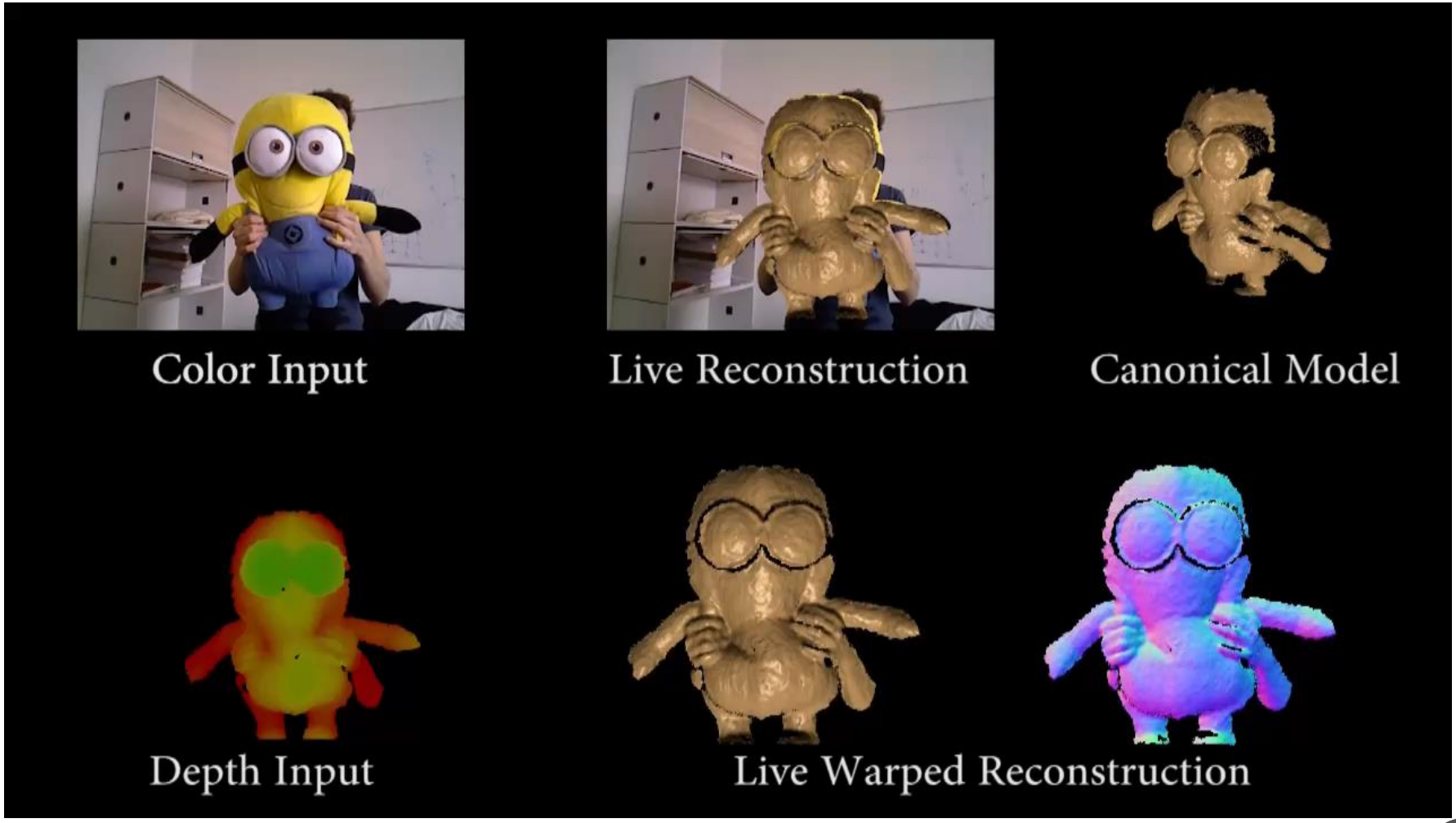
- Like rigid ICP (which has 6 DoF / frame)
  - Except we have many more degrees of freedom!



- Find correspondences
  - ➢closest points or projective correspondences
- Solve for deformation and vertices

- Needs a predefined template mesh!

Next Lecture: Non-Rigid Tracking & Reconstruction



ECCV'16 [Innmann et al.]: VolumeDeform

Administrative

- Reading Homework:
  - [Sorkine & Alexa 07] As-Rigid-As-Possible Surface Modeling
    https://diglib.eg.org/bitstream/handle/10.2312/SGP.SGP07.109-116/109-116.pdf
  - [Li et al. 08] Global Correspondence Optimization for Non-Rigid Registration of Depth Scans
    http://www.hao-li.com/publications/papers/sgp2008GCO.pdf
  - [Zollhoefer et al. 14] Real-time Non-rigid Reconstruction using an RGB-D Camera
    https://niessnerlab.org/papers/2014/5deformables/zollhoefer2014deformable.pdf

- Next week:
  - Non-Rigid Tracking and Reconstruction

Administrative

See you next week!