15-869

Lecture 9
Body Representations

Leonid Sigal

Human Motion Modeling and Analysis

Fall 2012

Course Updates

Capture Project

- Now due next Monday (one week from today)

Final Project

- 3 minute pitches are due next class
- You should try to post your idea to the blog and get some feedback before then

Reading Signup

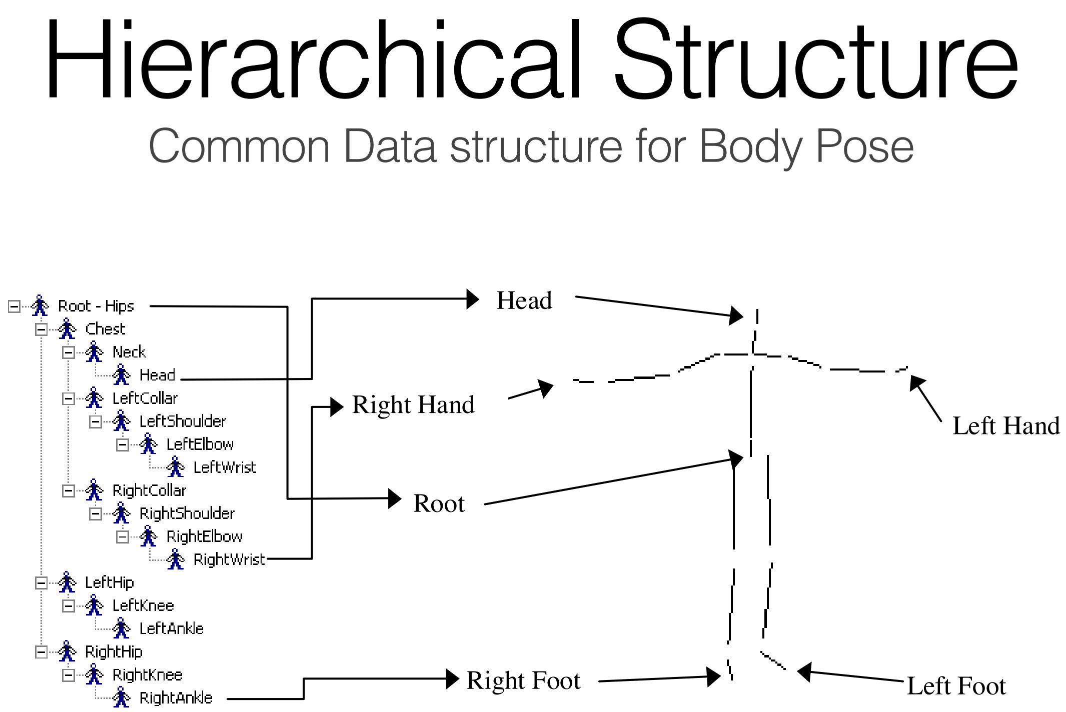
- Everyone has signed up by this point?

Plan for Today's Class

- Review representation of skeleton motion
- Modeling shape and geometry of the body
  - Skinning (rigid, linear, dual quaternion)
  - Data-driven body models (SCAPE)
- Applications and discussion
  - Shape estimation from images
  - Image reshaping

Skeleton Animations

Skeleton (tree hierarchy)

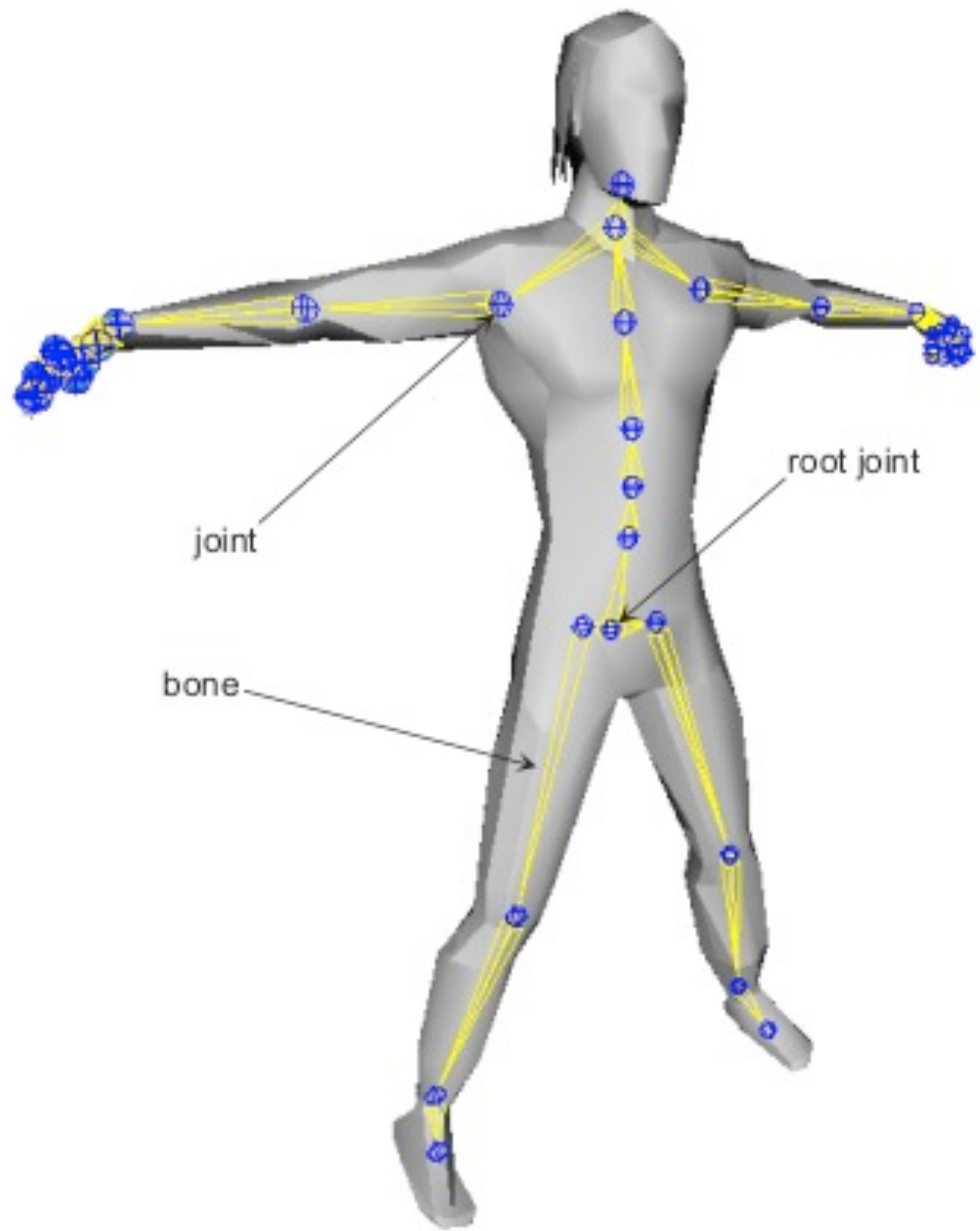


Source: Meredith and Maddock, Motion Capture File Formats Explained

Skeleton Animations

Skeleton (tree hierarchy)

- Nodes represent joints
- Joints are local coordinate systems (frames)
- Edges represent bones



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

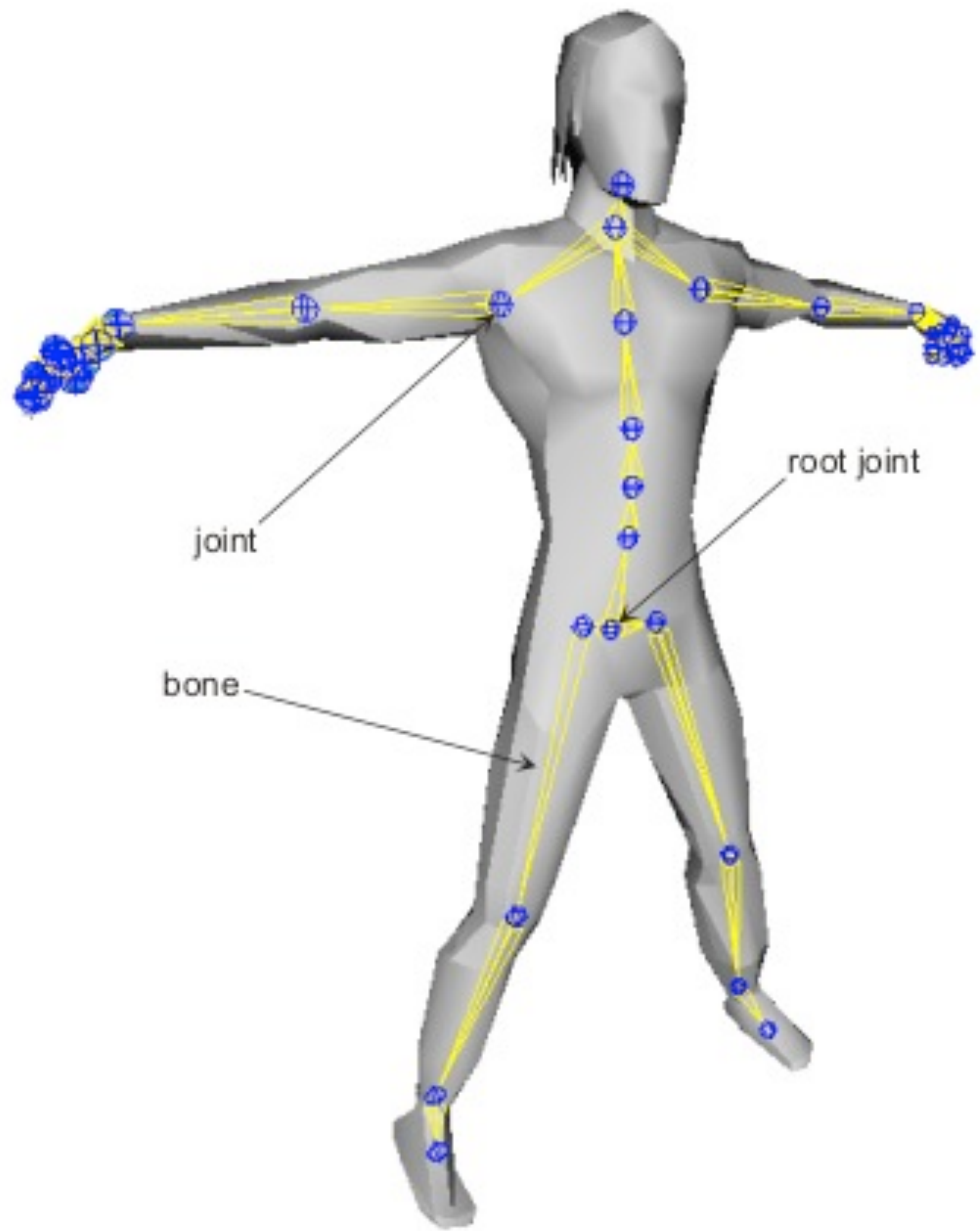
Skeleton Animations

Skeleton (tree hierarchy)

- Nodes represent joints
- Joints are local coordinate systems (frames)
- Edges represent bones

Skin

- 3D model driven by the skeleton



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

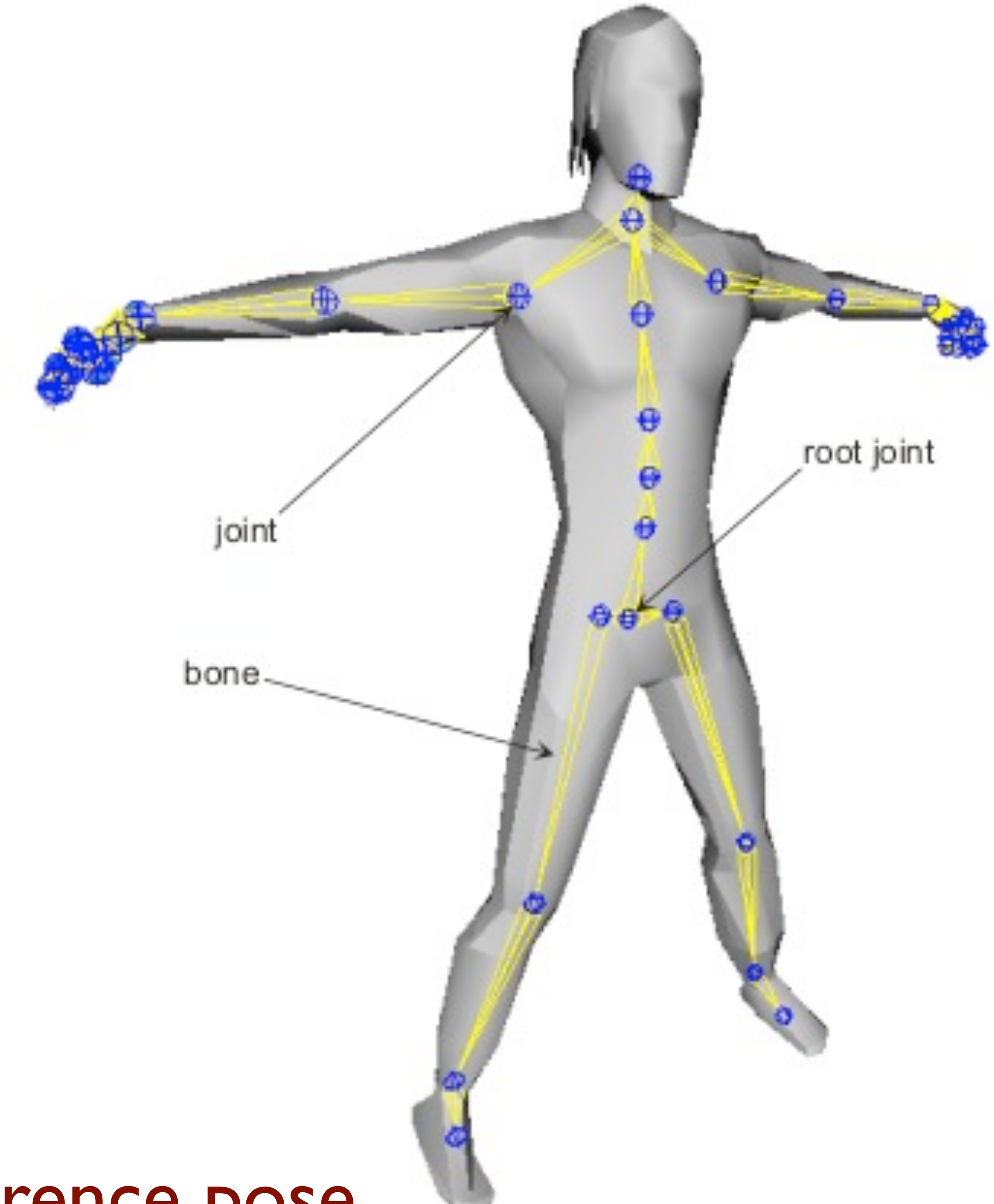
Skeleton Animations

Skeleton (tree hierarchy)

- Nodes represent joints
- Joints are local coordinate systems (frames)
- Edges represent bones

Skin

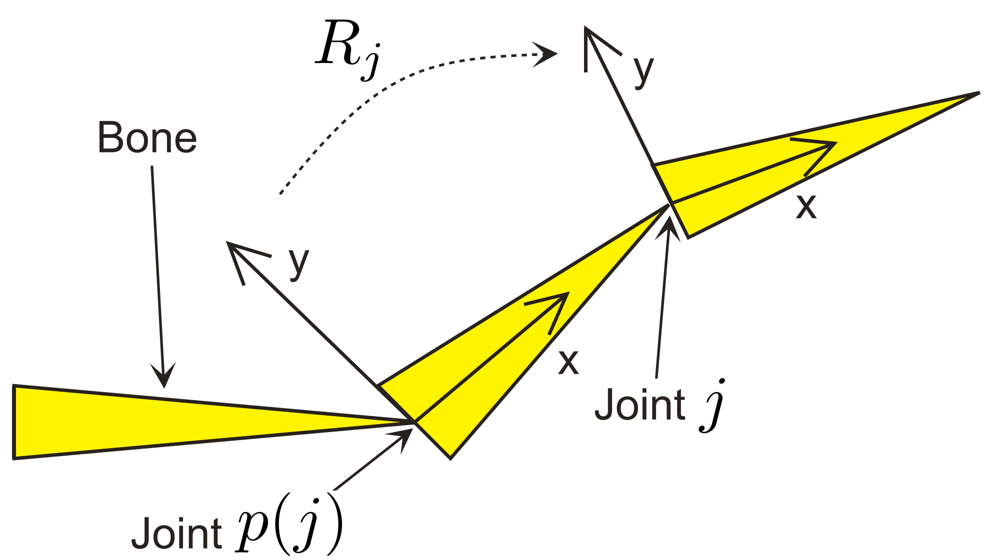
- 3D model driven by the skeleton



Both are typically designed in a reference pose

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Skeleton in Reference Posture

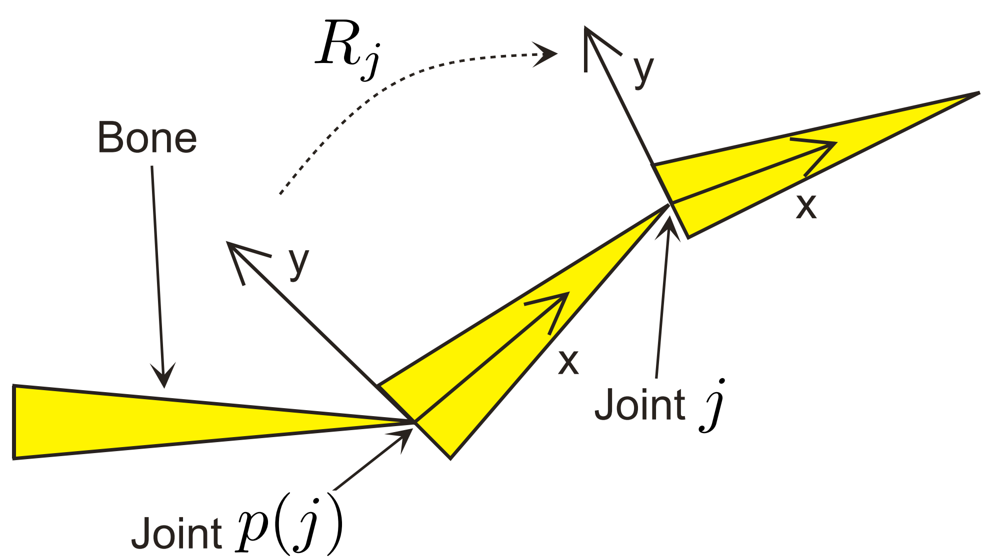


N-joint skeleton in reference frame is given by

- Root frame expressed with respect to the world - R_0
- Relative joint coordinate frames - $R_1, R_2, R_3, \ldots, R_N$

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Skeleton in Reference Posture



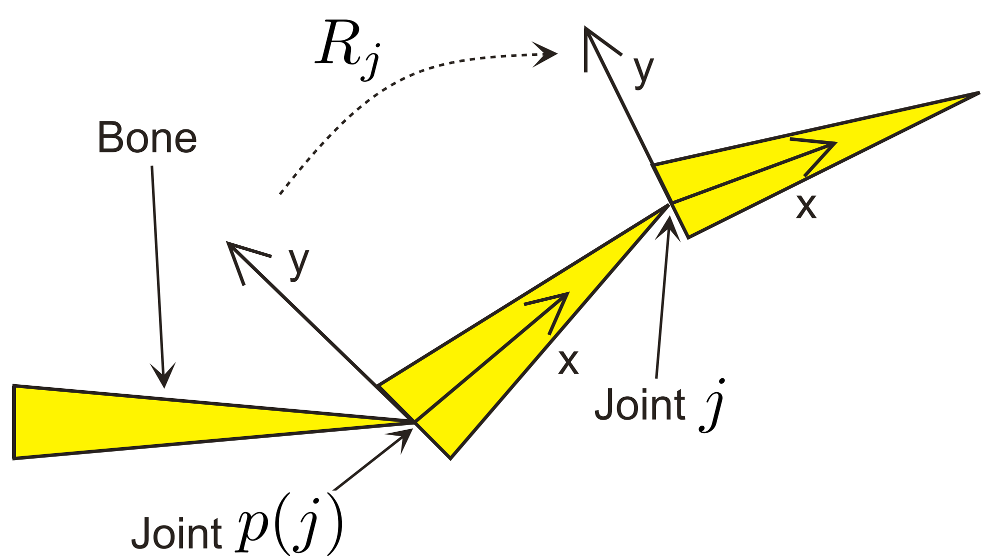
N-joint skeleton in reference frame is given by

- Root frame expressed with respect to the world - R_0
- Relative joint coordinate frames - $R_1, R_2, R_3, \ldots, R_N$

$$R_j = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Skeleton in Reference Posture



N-joint skeleton in reference frame is given by

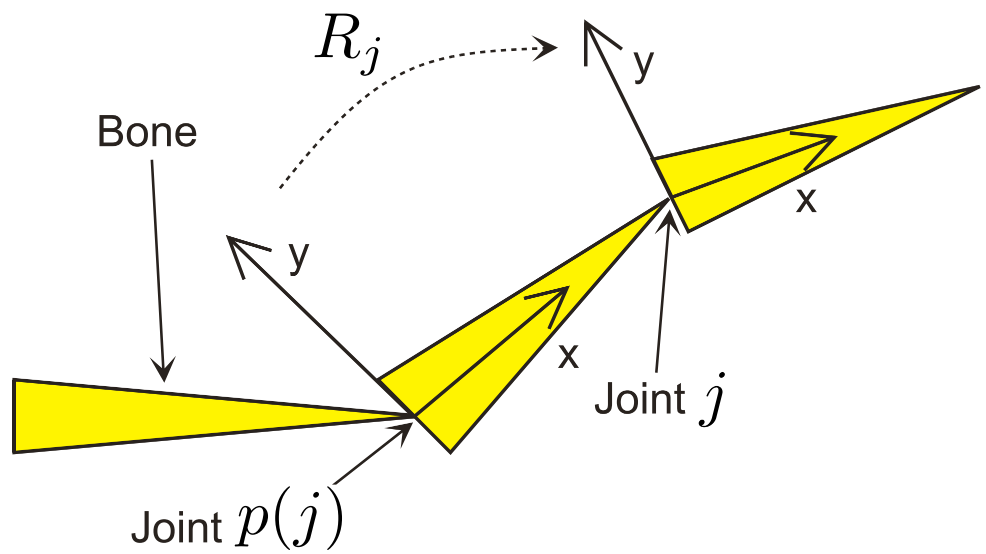
- Root frame expressed with respect to the world - R_0
- Relative joint coordinate frames - $R_1, R_2, R_3, \ldots, R_N$

Mapping from world to a coordinate frame of joint j

$$A_j = R_0 \cdots R_{p(j)} R_j$$

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Skeleton in Reference Posture



N-joint skeleton in reference frame is given by

- Root frame expressed with respect to the world - R_0
- Relative joint coordinate frames - $R_1, R_2, R_3, \ldots, R_N$

Mapping from world to a coordinate frame of joint j

$$A_j = R_0 \cdots R_{p(j)} R_j$$

parent of joint j

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Animating Skeleton

Achieved by rotating each joint from it's reference posture

- Note: joint rotation effects the entire sub-tree (e.g., rotation at the shoulder will induce motion of the whole arm)

Rotation at joint j is described by:

$$T_j = \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Animating Skeleton

Mapping from world to a coordinate frame of joint j in reference pose:

$$A_j = R_0 \cdots R_{p(j)} R_j$$

Mapping from world to a coordinate frame of joint j in animated pose:

$$F_j = R_0 T_0 \ldots R_{p(j)} T_{p(j)} R_j T_j$$

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Animating Skeleton

Mapping from world to a coordinate frame of joint j in reference pose:

$$A_j = R_0 \cdots R_{p(j)} R_j$$

Mapping from world to a coordinate frame of joint j in animated pose:

$$F_j = R_0 T_0 \dots R_{p(j)} T_{p(j)} R_j T_j$$

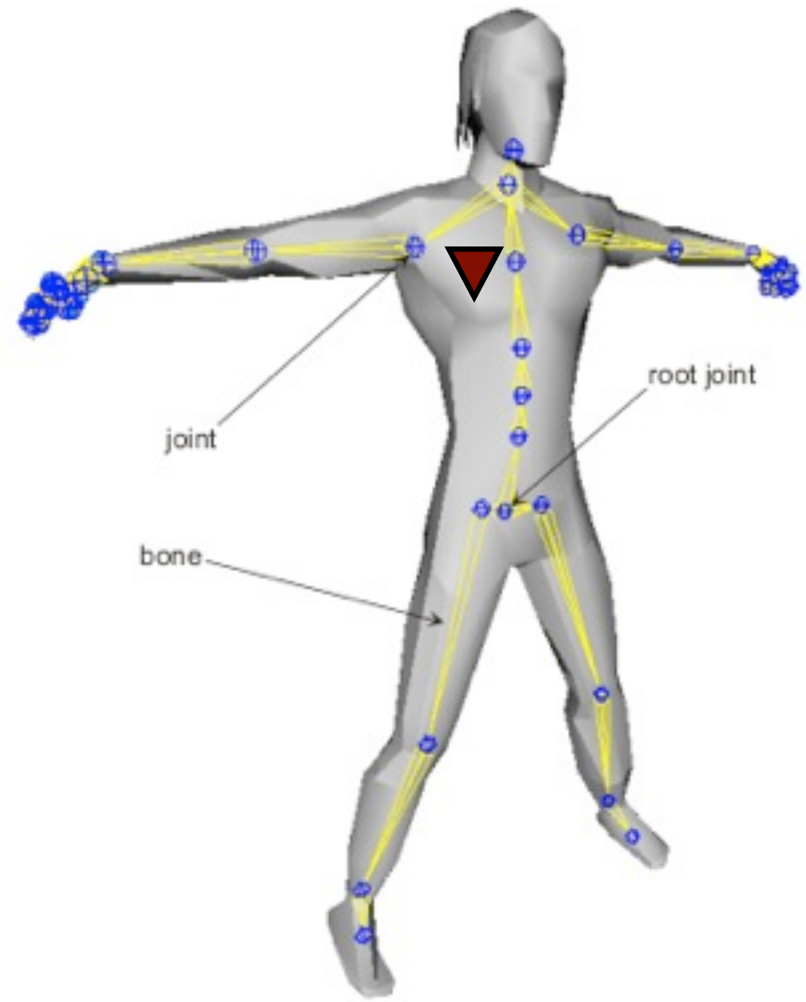
Note: if $T_j = I_{4 \times 4}$ then $F_j = A_j$

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Character Rigging

1. Embed the skeleton into a 3D mesh (skin)

2. Assign vertices of the mesh to one or more bones to allow skin to move with the skeleton



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

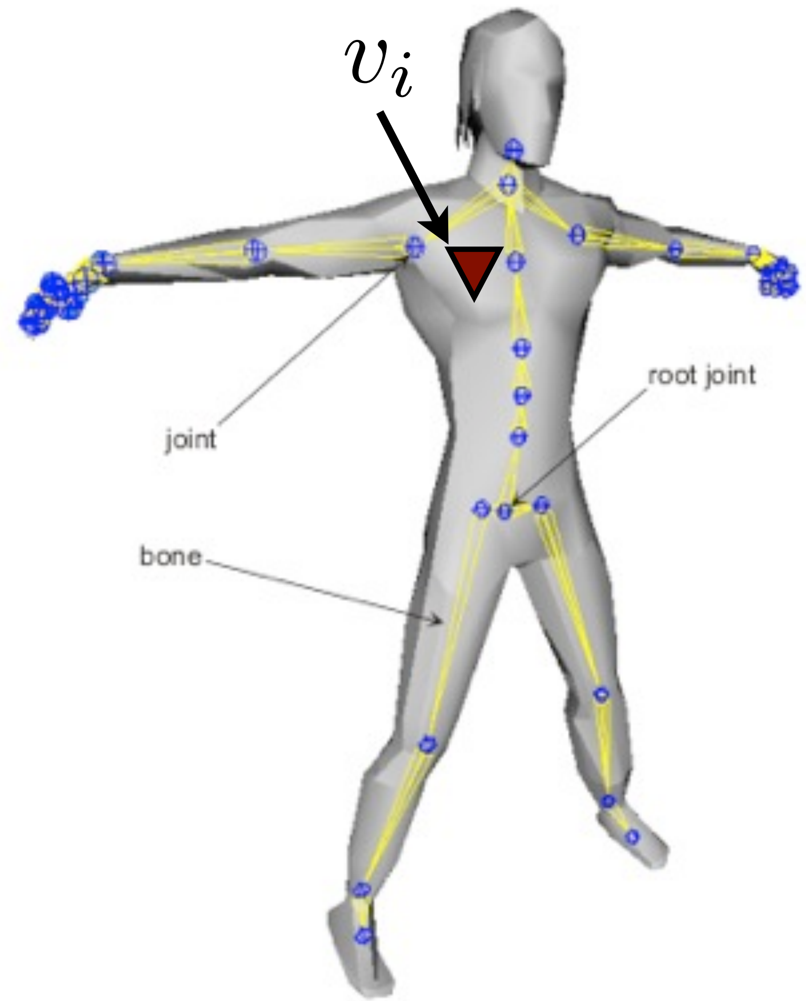
Rigid Skinning

1. Embed the skeleton into a 3D mesh (skin)

2. Assign vertices of the mesh to one or more bones to allow skin to move with the skeleton

Assign each vertex to one bone/joint - *j*

$$\hat{v}_i = F_j (A_j)^{-1} v_i$$



v_i - position of vertex in reference mesh

A_j - joint j in reference mesh

F_j - joint j in animated mesh

$\hat{v}_i$ - position of vertex in animated mesh

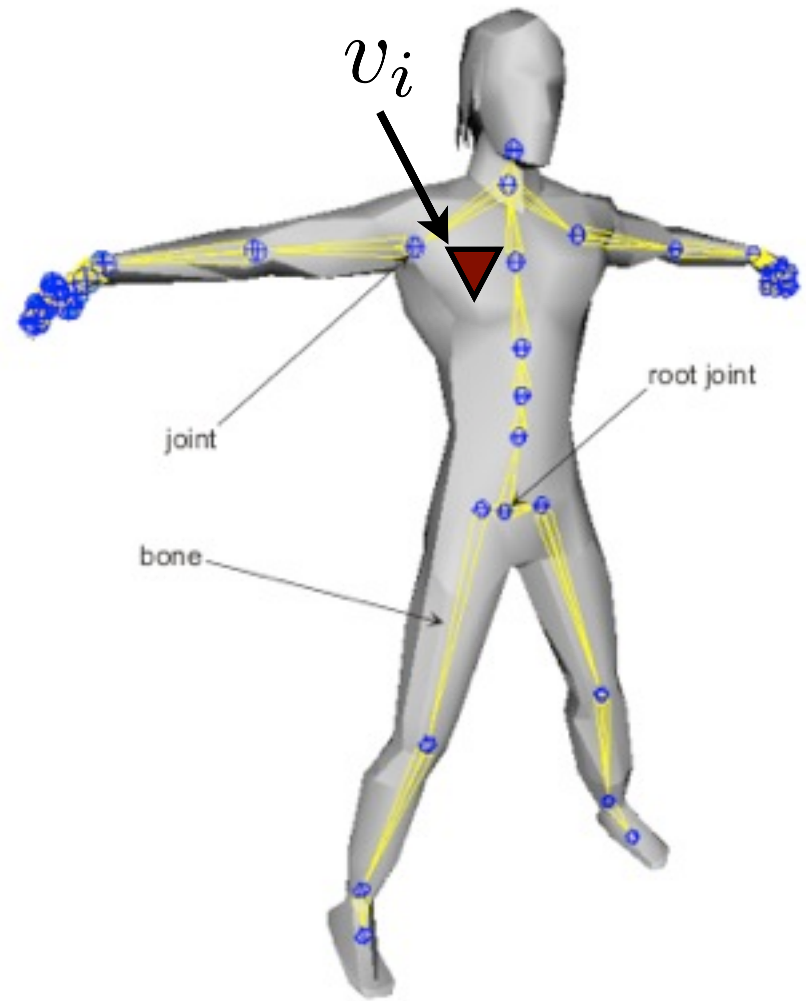
Rigid Skinning

Requires assignment of vertices on 3D mesh to joints on skeleton

- Often done manually
- Assigning to joint influencing the closest bone is usually a good automatic guess

Assign each vertex to one bone/joint - j

$$\hat{v}_i = F_j(A_j)^{-1} v_i$$



v_i - position of vertex in reference mesh

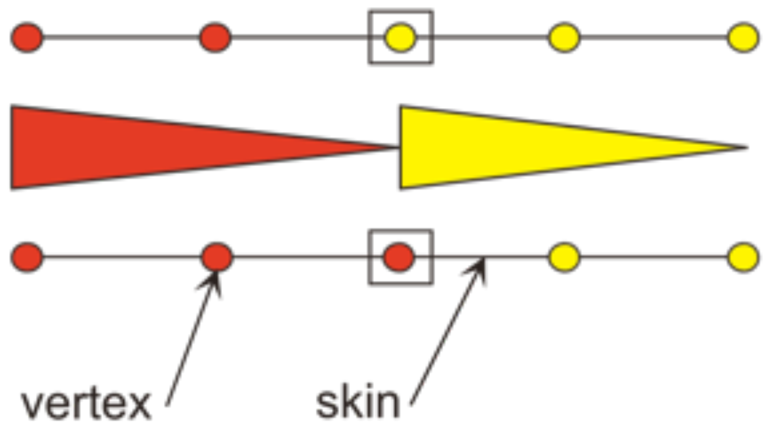
A_j - joint j in reference mesh

F_j - joint j in animated mesh

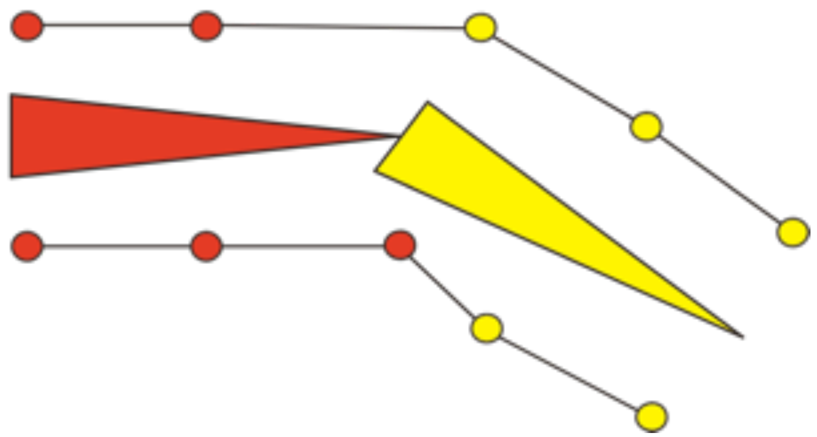
$\hat{v}_i$ - position of vertex in animated mesh

Rigid Skinning Limitations

In reference pose:



In animated pose:



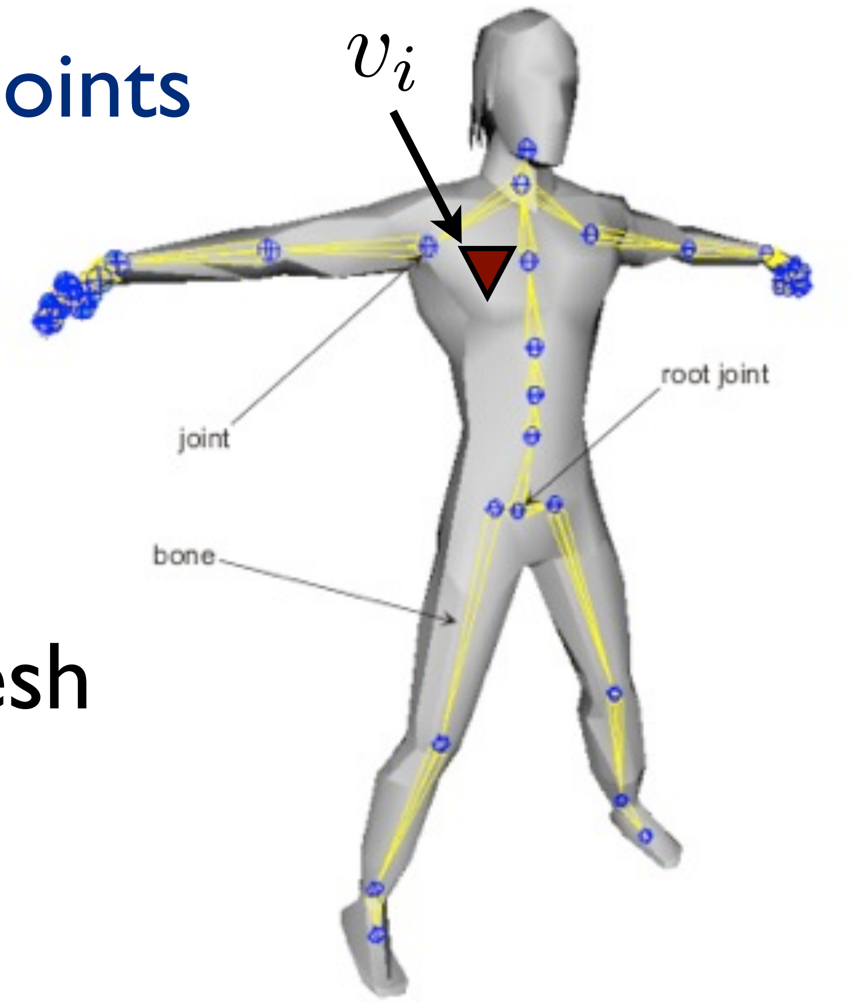
- Works well away from the ends of the bone (joints)
- Leads to unrealistic non-smooth deformations near joints

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Linear Blend Skinning

Each vertex is assigned to multiple bone/joints

$$\hat{v}_i = \sum_{j=1}^{N} w_{ji} F_j (A_j)^{-1} v_i$$



v_i - position of vertex i in reference mesh

A_j - joint j in reference mesh

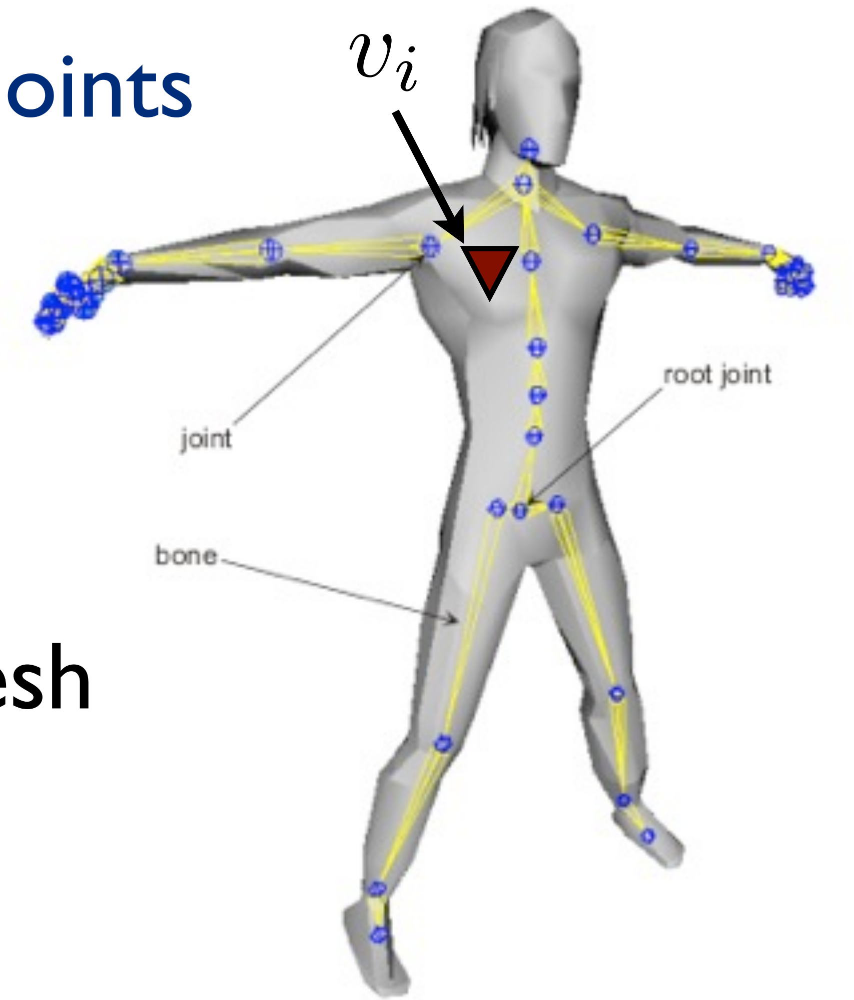
F_j - joint j in animated mesh

$\hat{v}_i$ - position of vertex i in animated mesh

w_{ji} - influence of joint j on the vertex

Linear Blend Skinning

Each vertex is assigned to multiple bone/joints



$$\hat{v}_i = \sum_{j=1}^{N} w_{ji} F_j (A_j)^{-1} v_i$$

v_i - position of vertex i in reference mesh

A_j - joint j in reference mesh

F_j - joint j in animated mesh

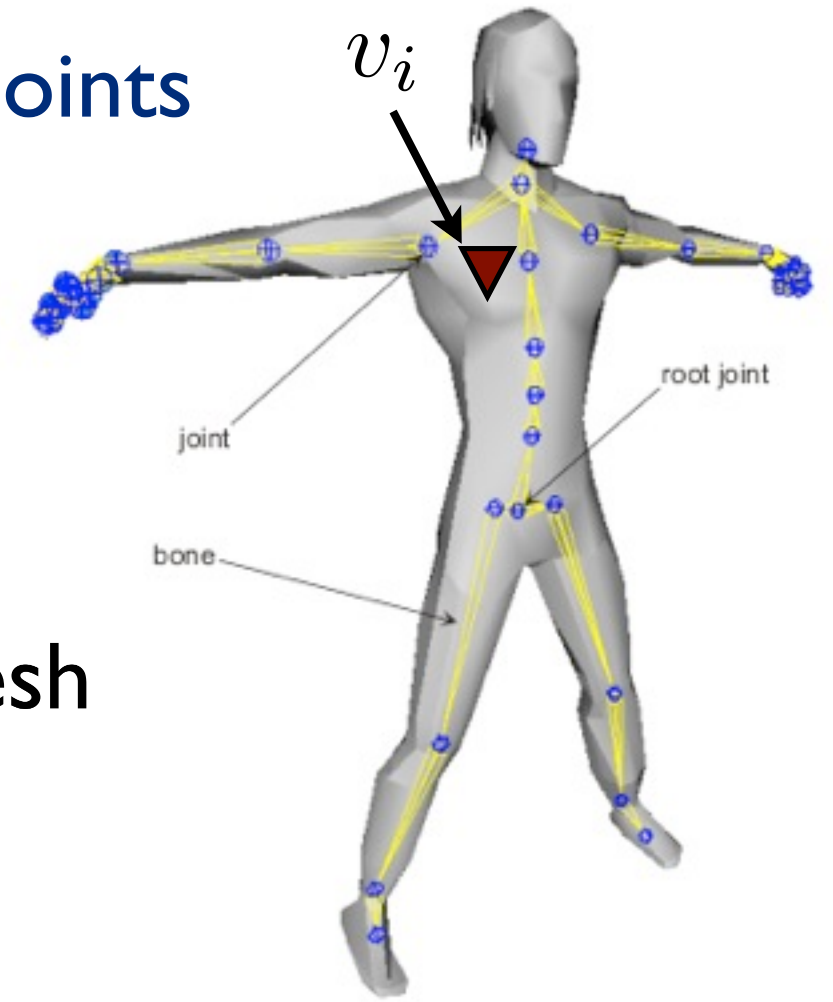
$\hat{v}_i$ - position of vertex i in animated mesh

w_{ji} - influence of joint j on the vertex

Weights need to be convex: $\sum_{j=1}^{N} w_{ji} = 1, w_{ji} \geq 0$

Linear Blend Skinning

Each vertex is assigned to multiple bone/joints



$$\hat{v}_i = \sum_{j=1}^{N} w_{ji} F_j (A_j)^{-1} v_i$$

v_i - position of vertex i in reference mesh

A_j - joint j in reference mesh

F_j - joint j in animated mesh

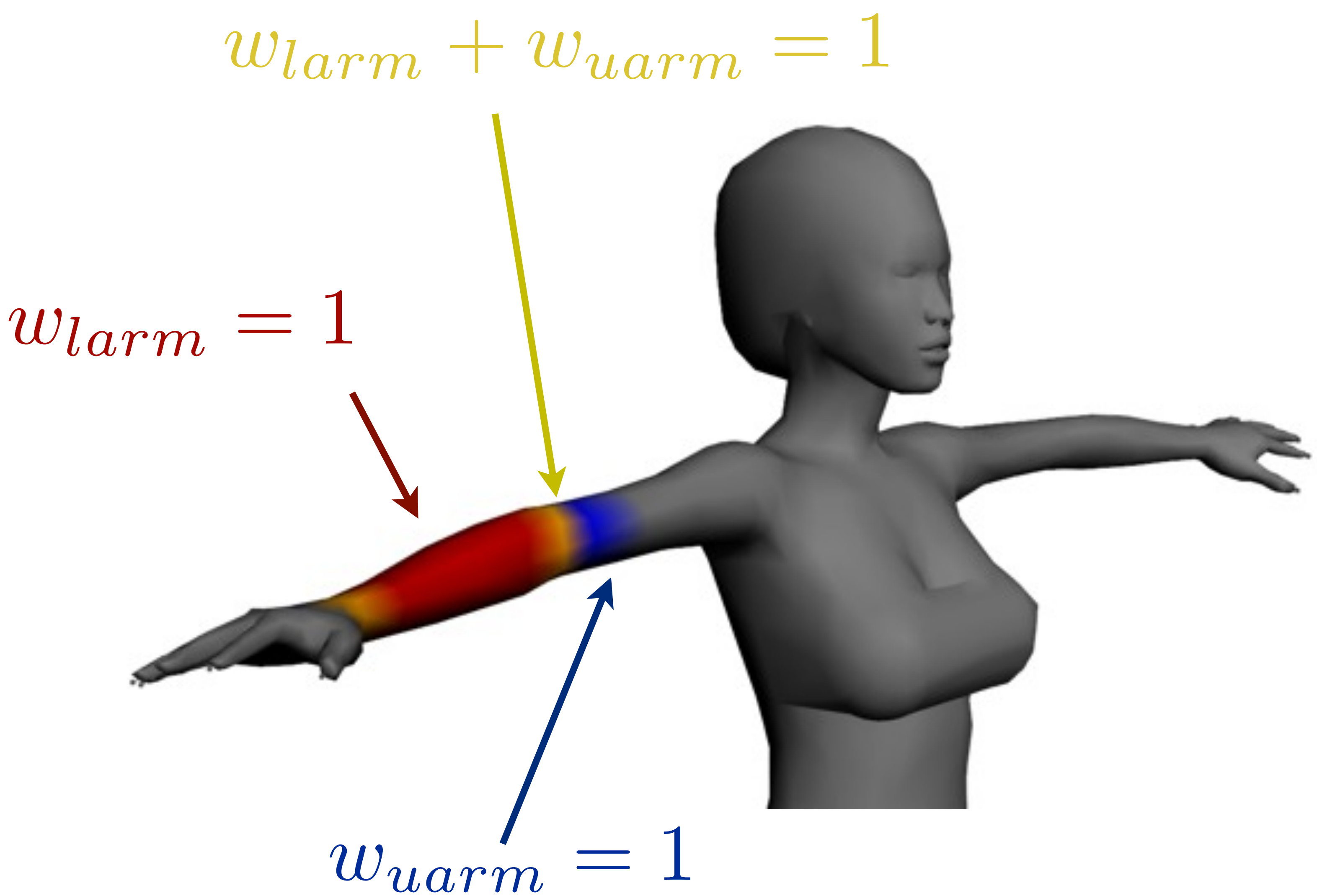
$\hat{v}_i$ - position of vertex i in animated mesh

w_{ji} - influence of joint j on the vertex

painted on or based on distance to joints

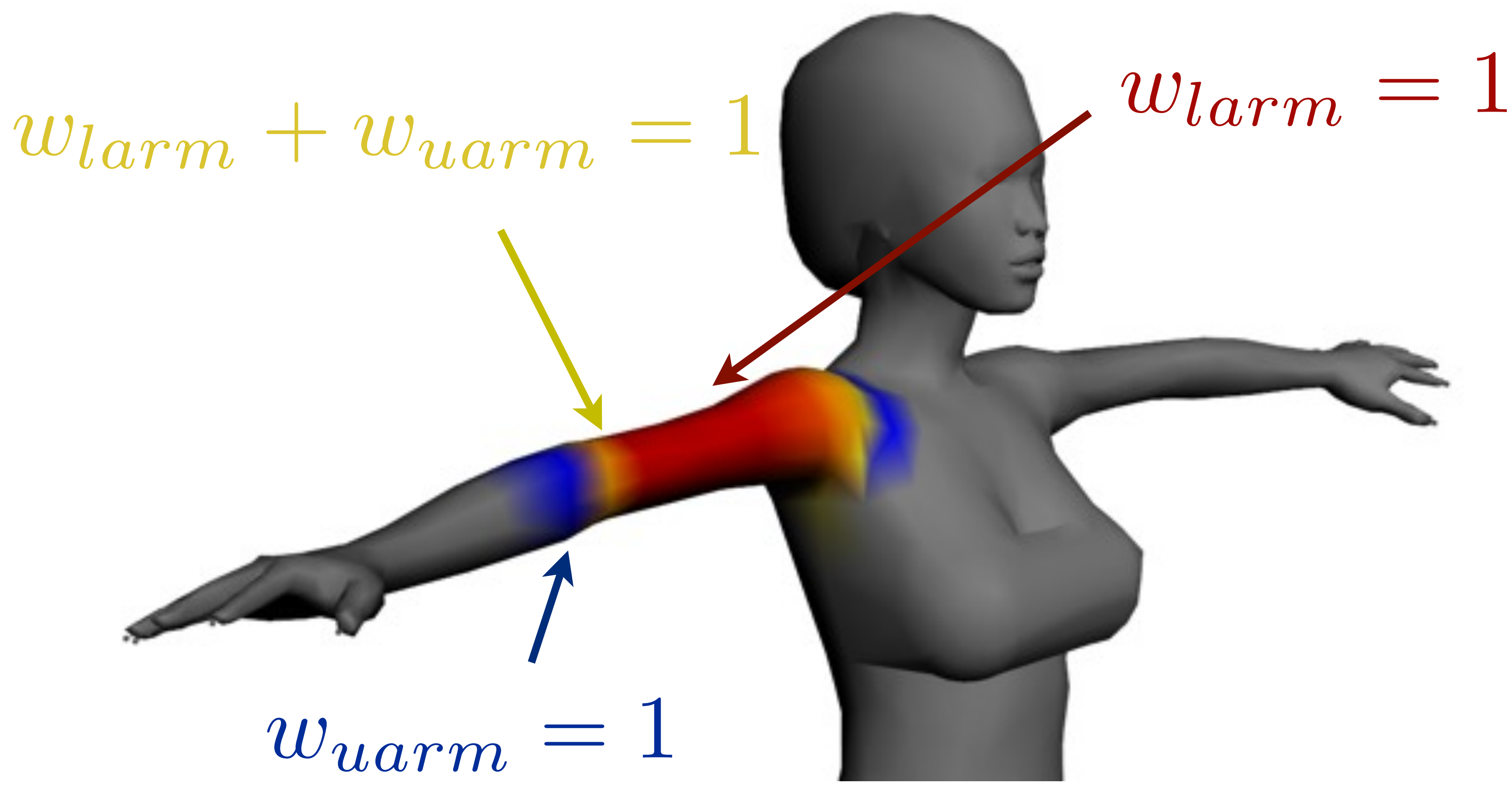
Weights need to be convex: $\sum_{j=1}^{N} w_{ji} = 1, \boxed{w_{ji}} \geq 0$

Linear Blend Skinning Weights



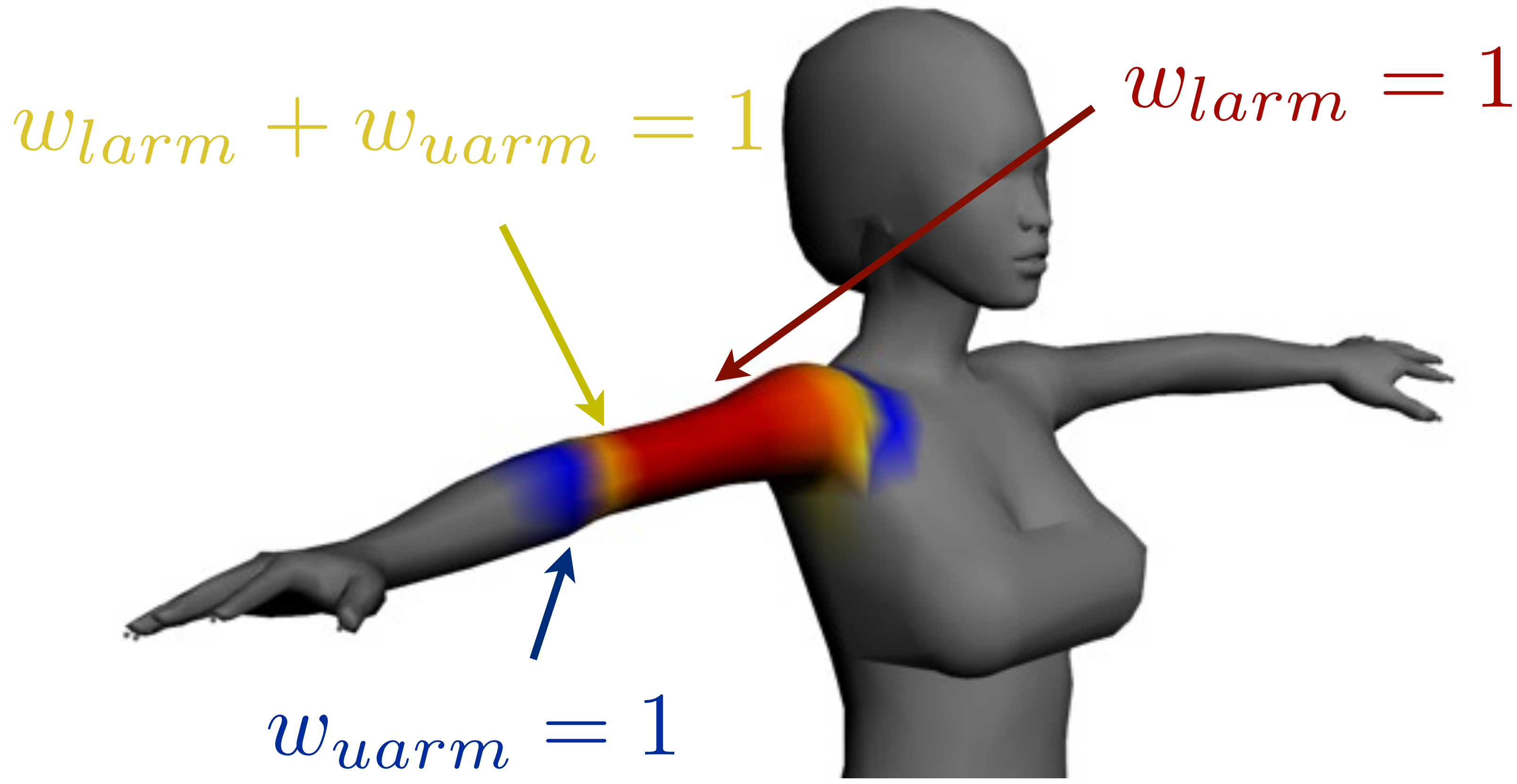
[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Linear Blend Skinning Weights



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Linear Blend Skinning Weights

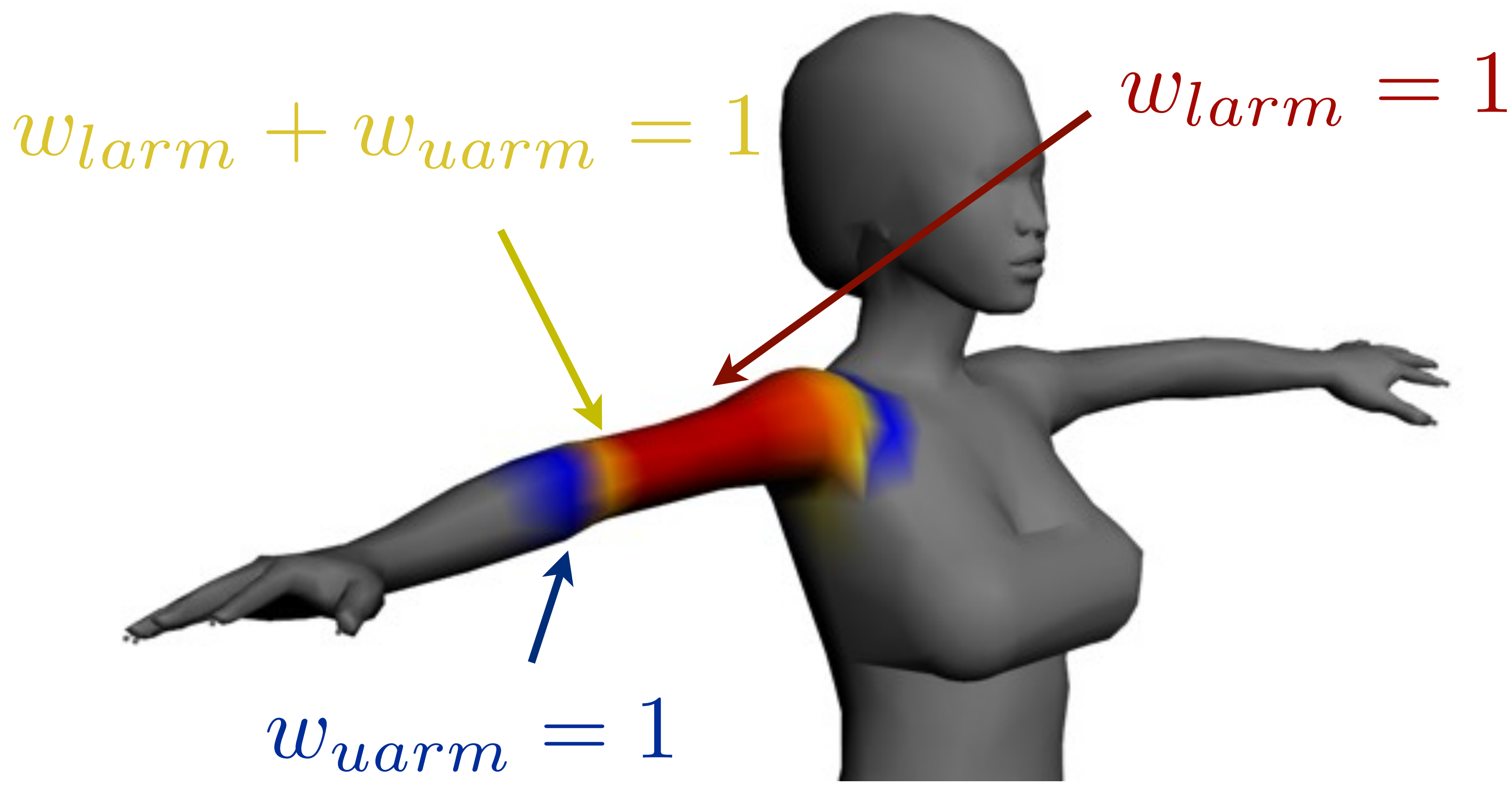


Why weights must convex?

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Linear Blend Skinning Weights

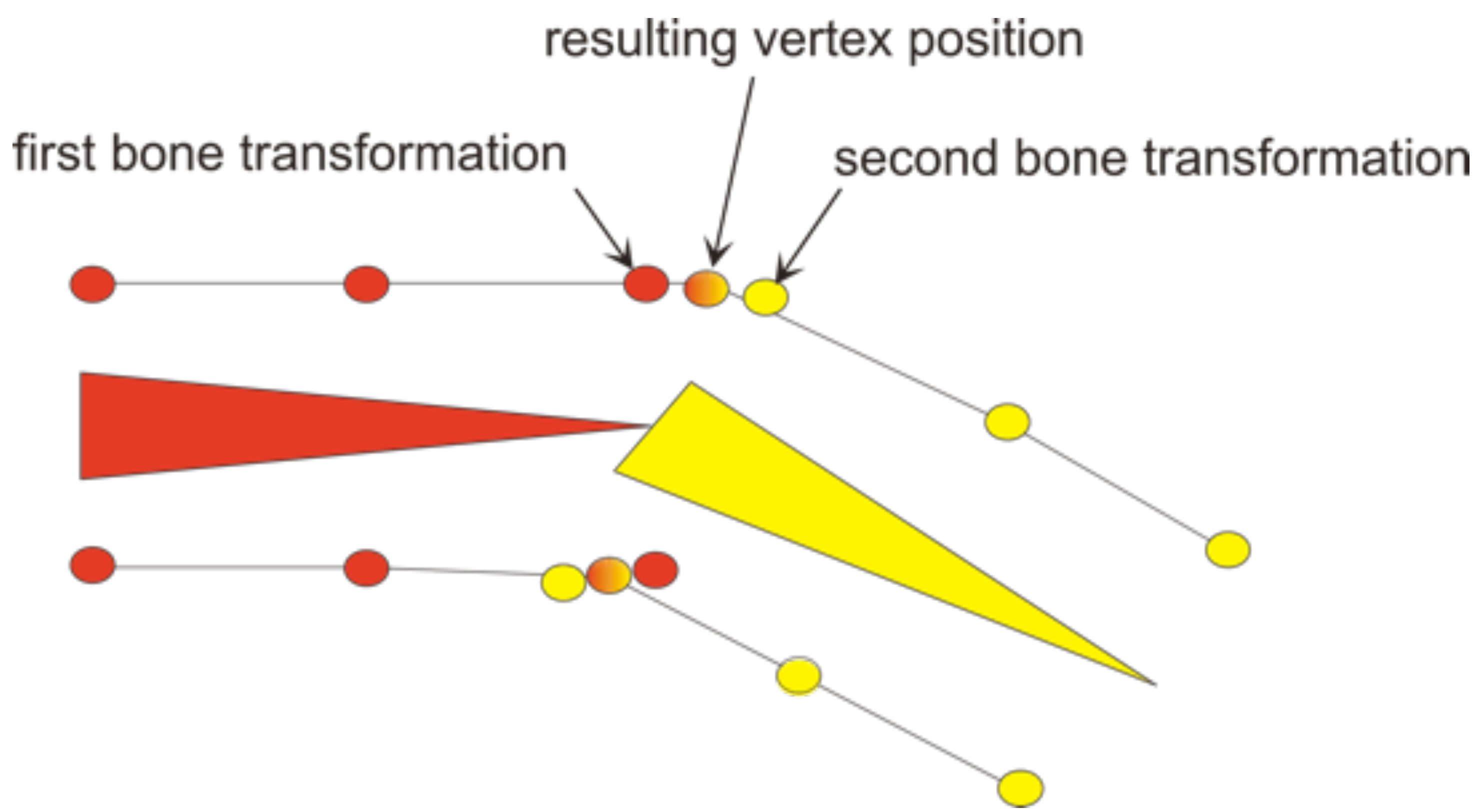
$$\hat{v}_i = \sum_{j=1}^{N} w_{ji} F_j (A_j)^{-1} v_i$$



Why weights must convex?

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

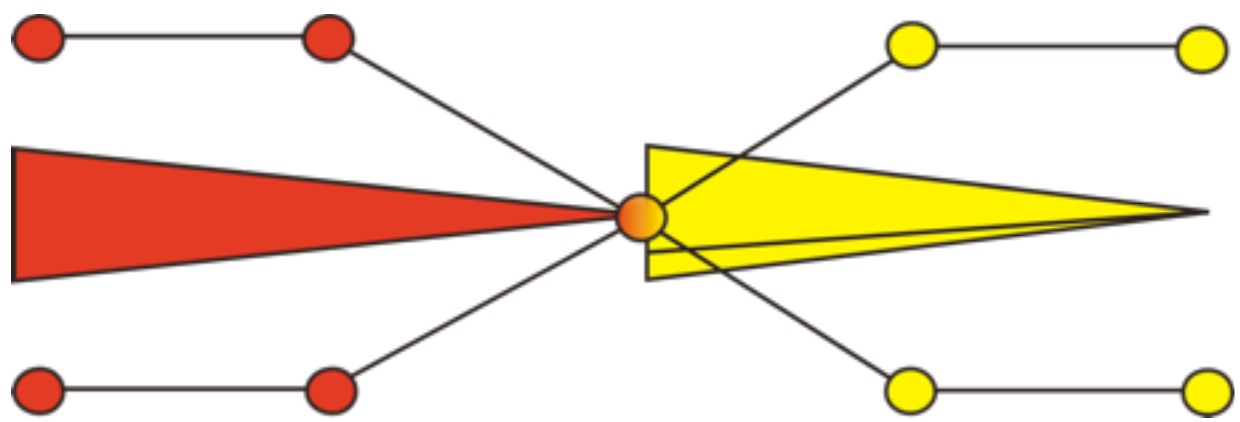
Linear Blend Skinning Example



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Linear Blend Skinning Limitations

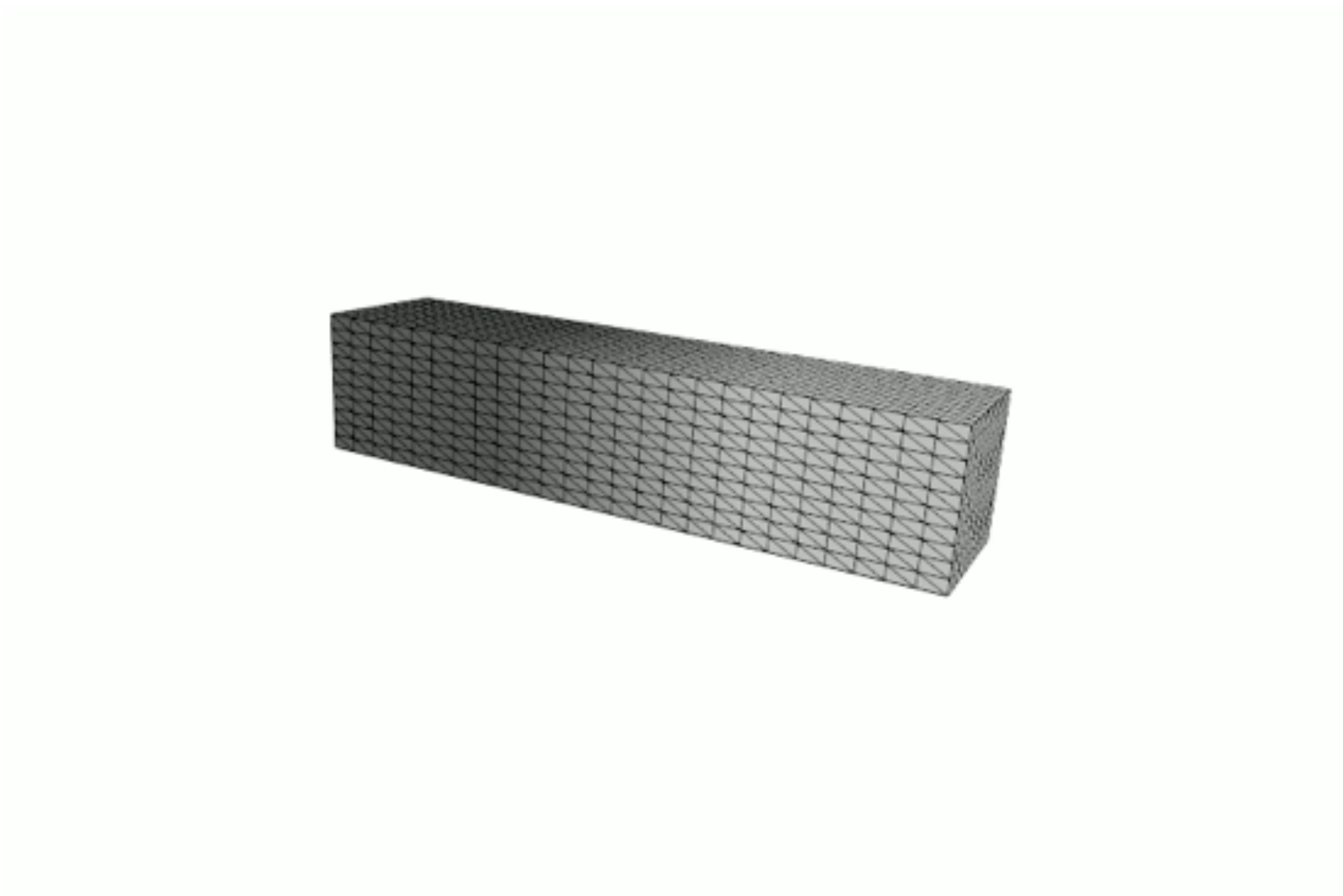
Joint twisting 180 degrees



produces a collapsing effect where skin collapses to a single point
("candy-wrapper" artifact)

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Linear Blend Skinning Limitations



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Linear Blend Skinning Limitations

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

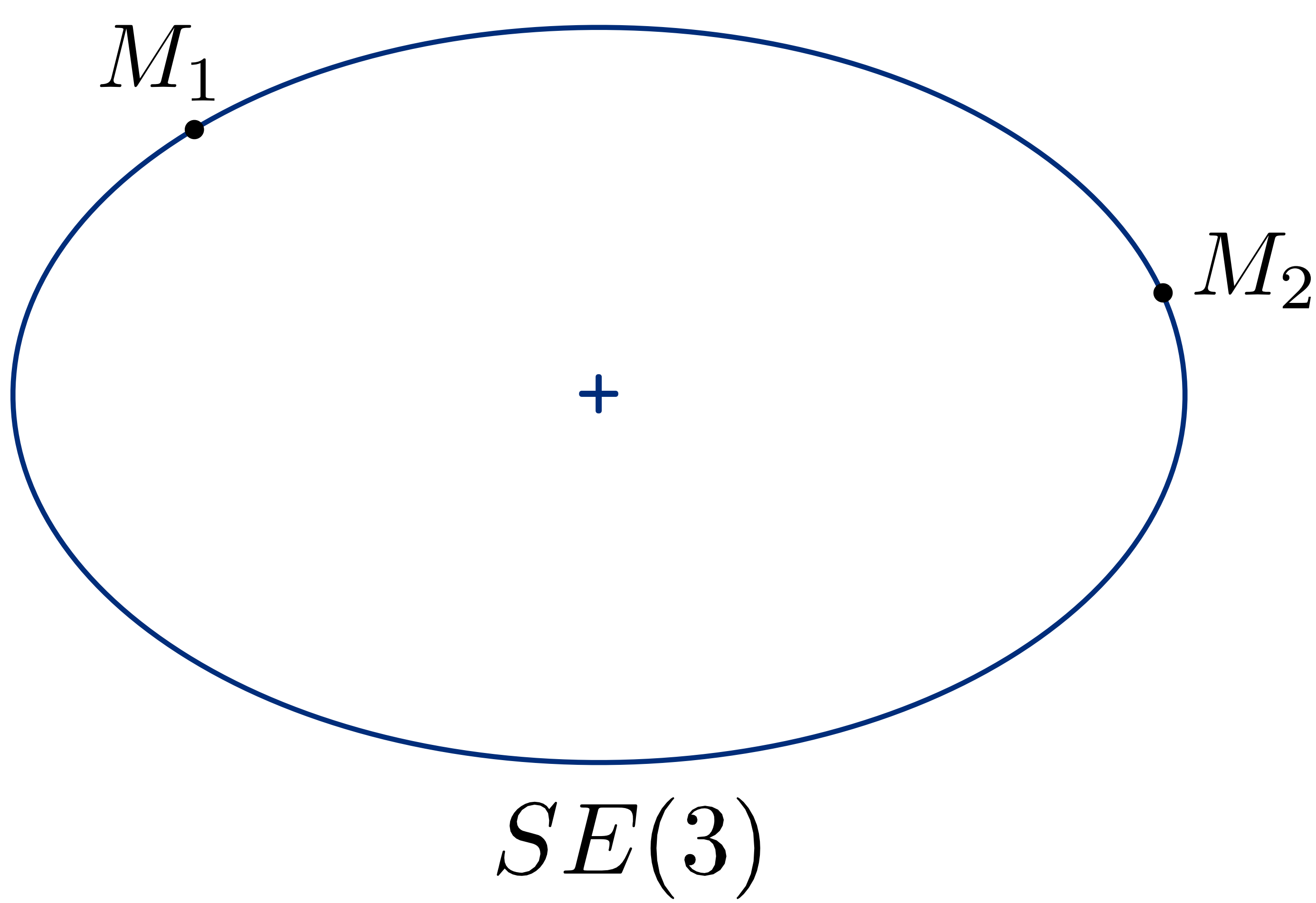
Why LBS Produce Artifacts?

$$\hat{v}_i = \sum_{j=1}^{N} w_{ji} F_j (A_j)^{-1} v_i \longleftrightarrow \hat{v}_i = \left(\sum_{j=1}^{N} w_{ji} M_j \right) v_i$$

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Why LBS Produce Artifacts?

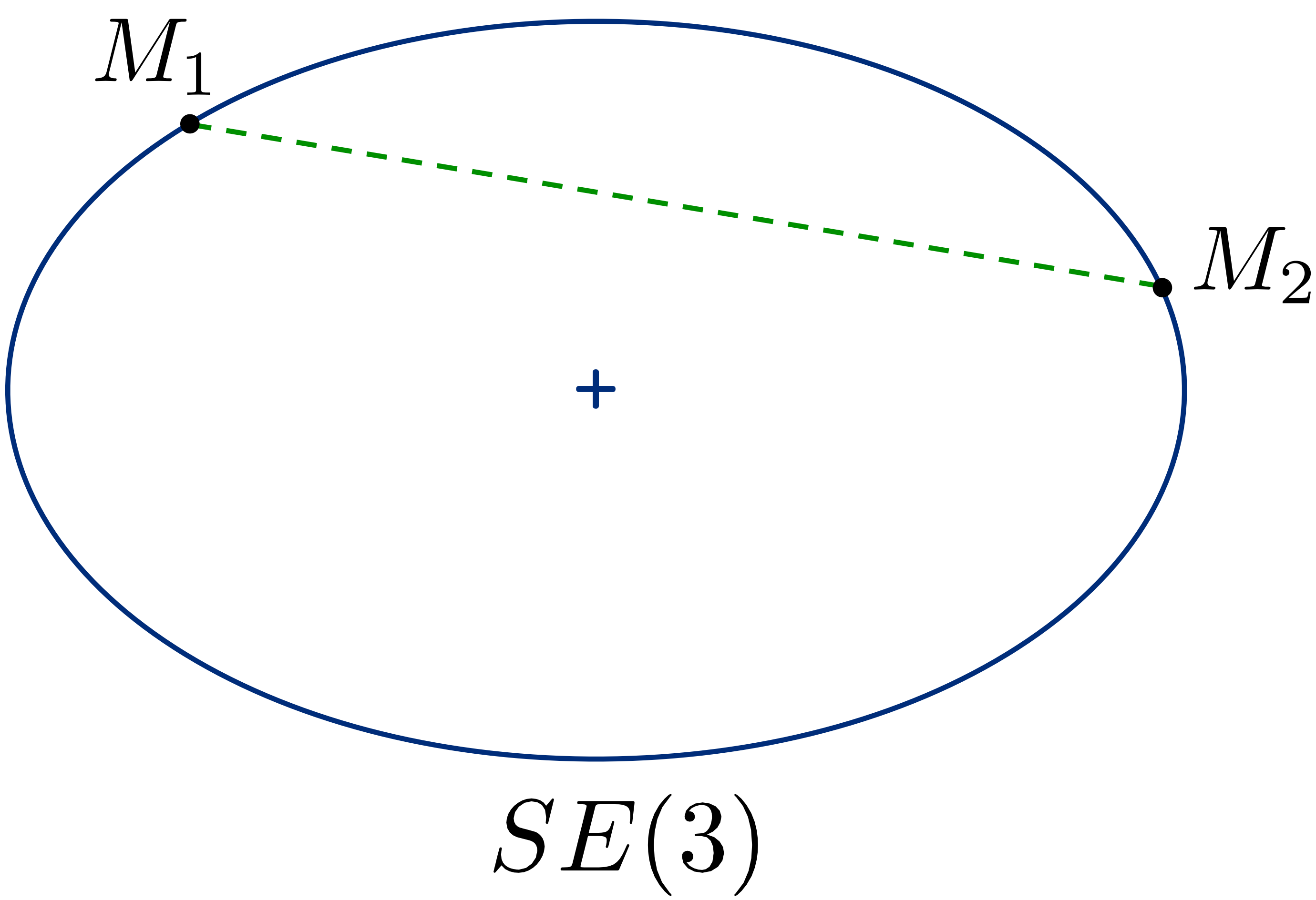
$$\hat{v}_i = \sum_{j=1}^{N} w_{ji} F_j (A_j)^{-1} v_i \longleftrightarrow \hat{v}_i = \left(\sum_{j=1}^{N} w_{ji} M_j \right) v_i$$



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Why LBS Produce Artifacts?

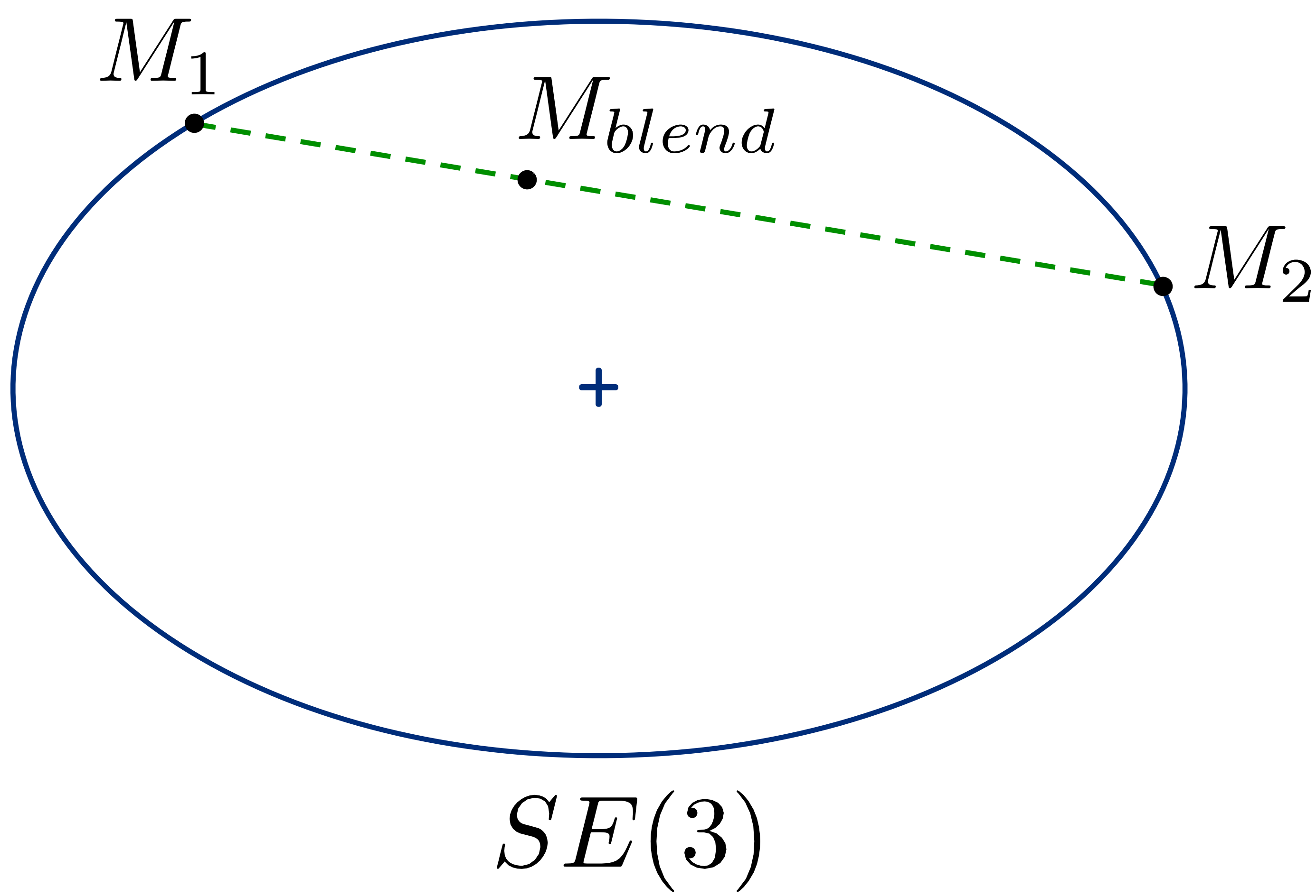
$$\hat{v}_i = \sum_{j=1}^{N} w_{ji} F_j (A_j)^{-1} v_i \longleftrightarrow \hat{v}_i = \left(\sum_{j=1}^{N} w_{ji} M_j \right) v_i$$



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

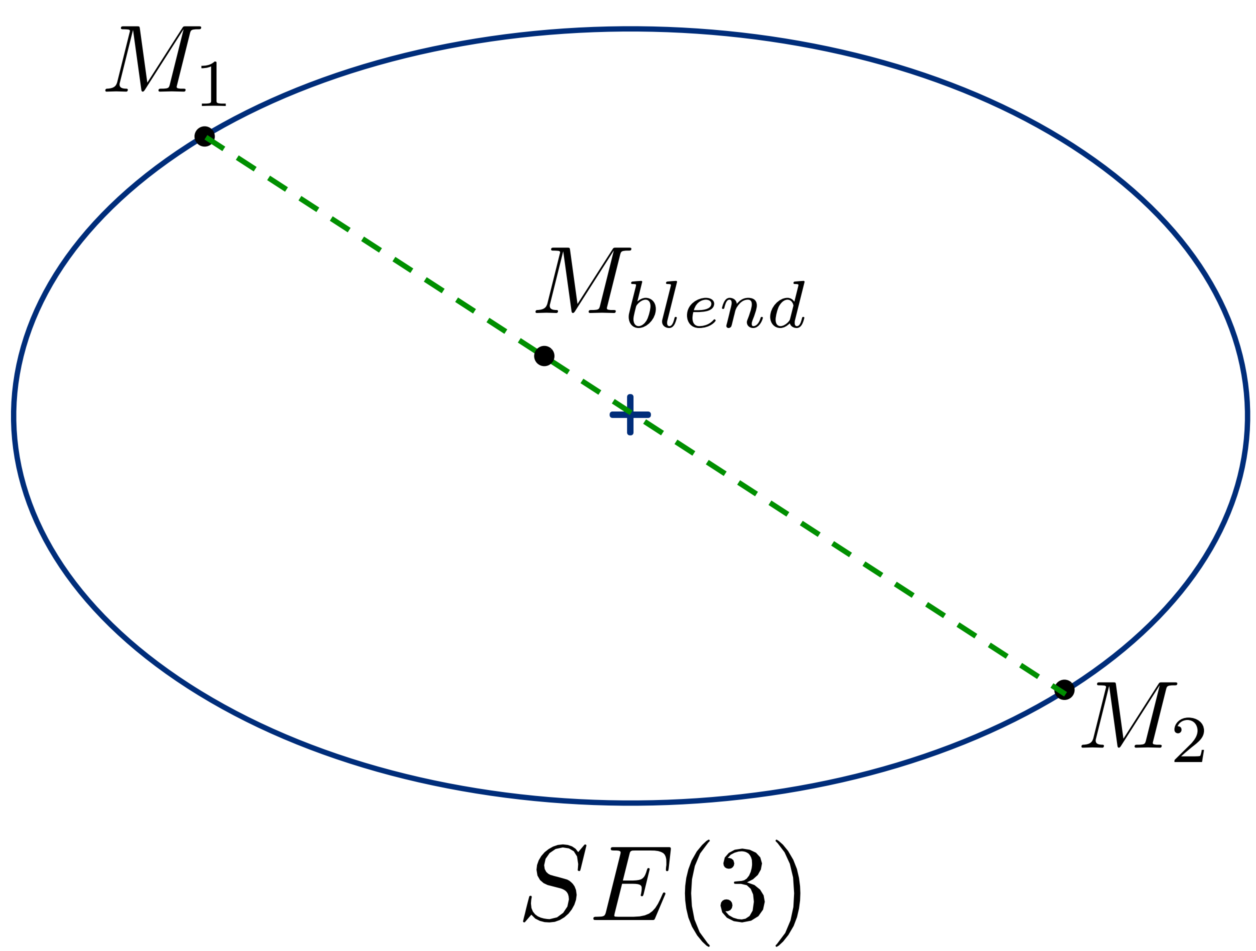
Why LBS Produce Artifacts?

$$\hat{v}_i = \sum_{j=1}^{N} w_{ji} F_j (A_j)^{-1} v_i \longleftrightarrow \hat{v}_i = \left(\sum_{j=1}^{N} w_{ji} M_j \right) v_i$$



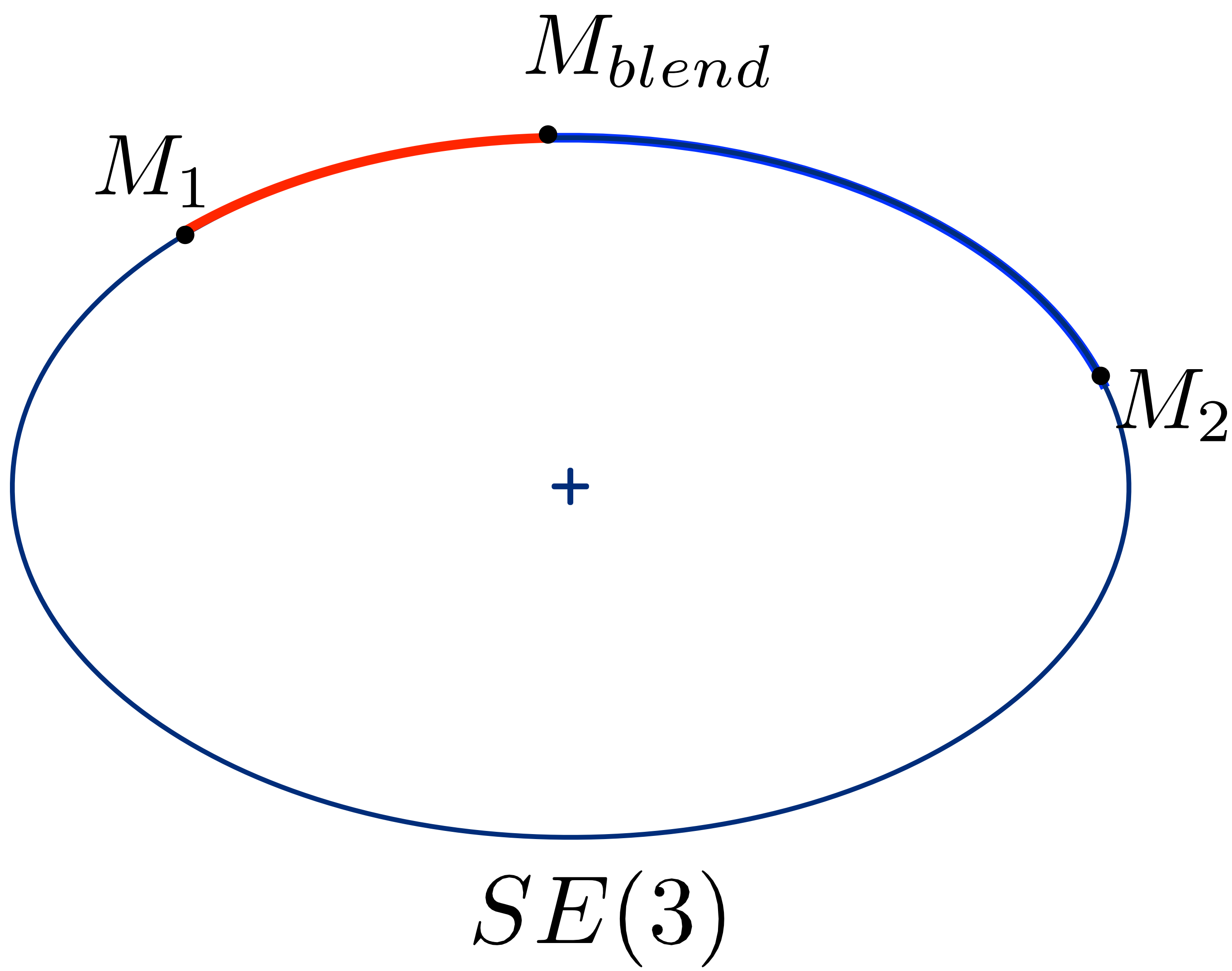
[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Why LBS Produce Artifacts?



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

SE(3) Intrinsic Blending

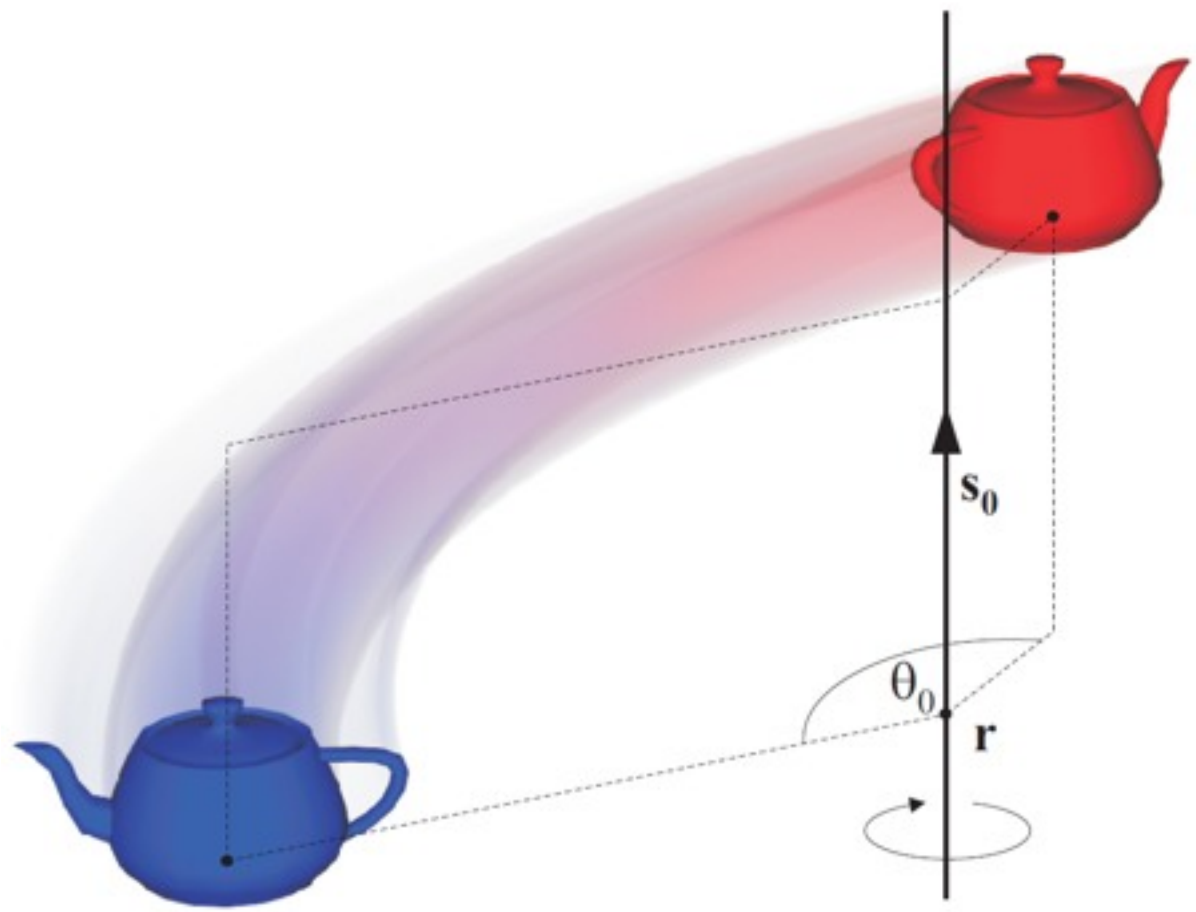


[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Dual Quaternion Skinning

[Kavan et al., ACM TOG 2008]

Closed form approximation of SE(3) blending

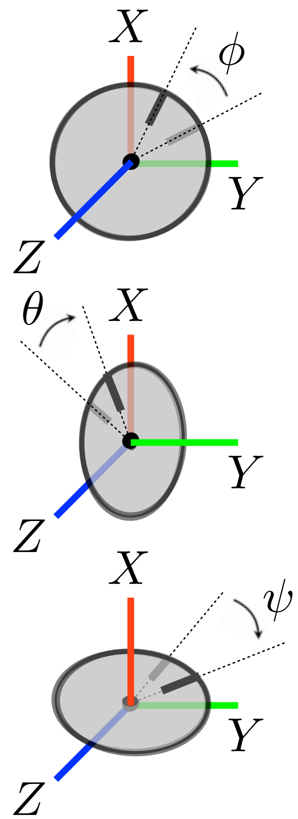


[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Regular Quaternions

Remember: Euler angles

Rotation in 3D



$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(\phi) & \sin(\phi) & 0 \\ -\sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{R}_z} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}}_{\mathbf{R}_y} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi) & \sin(\psi) \\ 0 & -\sin(\psi) & \cos(\psi) \end{bmatrix}}_{\mathbf{R}_x} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Note: I've overloaded the use of \phi in these slides. Earlier I used \phi to denote the original orientation. Here I am using it to denote the rotation about the Z-axis.

Interpolating Euler angles has similar issues as LBS skinning

Regular Quaternions

- Quaternions are alternative representation for orientations (defined using complex algebra)
- Represents orientation using 4 tuples (roughly speaking one for amount of rotation and 3 for the axis)

$$q = w + xi + yi + zi$$

- However, there are only 3 degrees of freedom for a rotation
- Hence, to be a valid rotation, quaternion must be unit norm

$$||q|| = 1$$

Regular Quaternions

- Quaternions are alternative representation for orientations (defined using complex algebra)

- Represents orientation using 4 tuples (roughly speaking one for amount of rotation and 3 for the axis)

$$q = w + xi + yi + zi$$

- However, there are only 3 degrees of freedom for a rotation

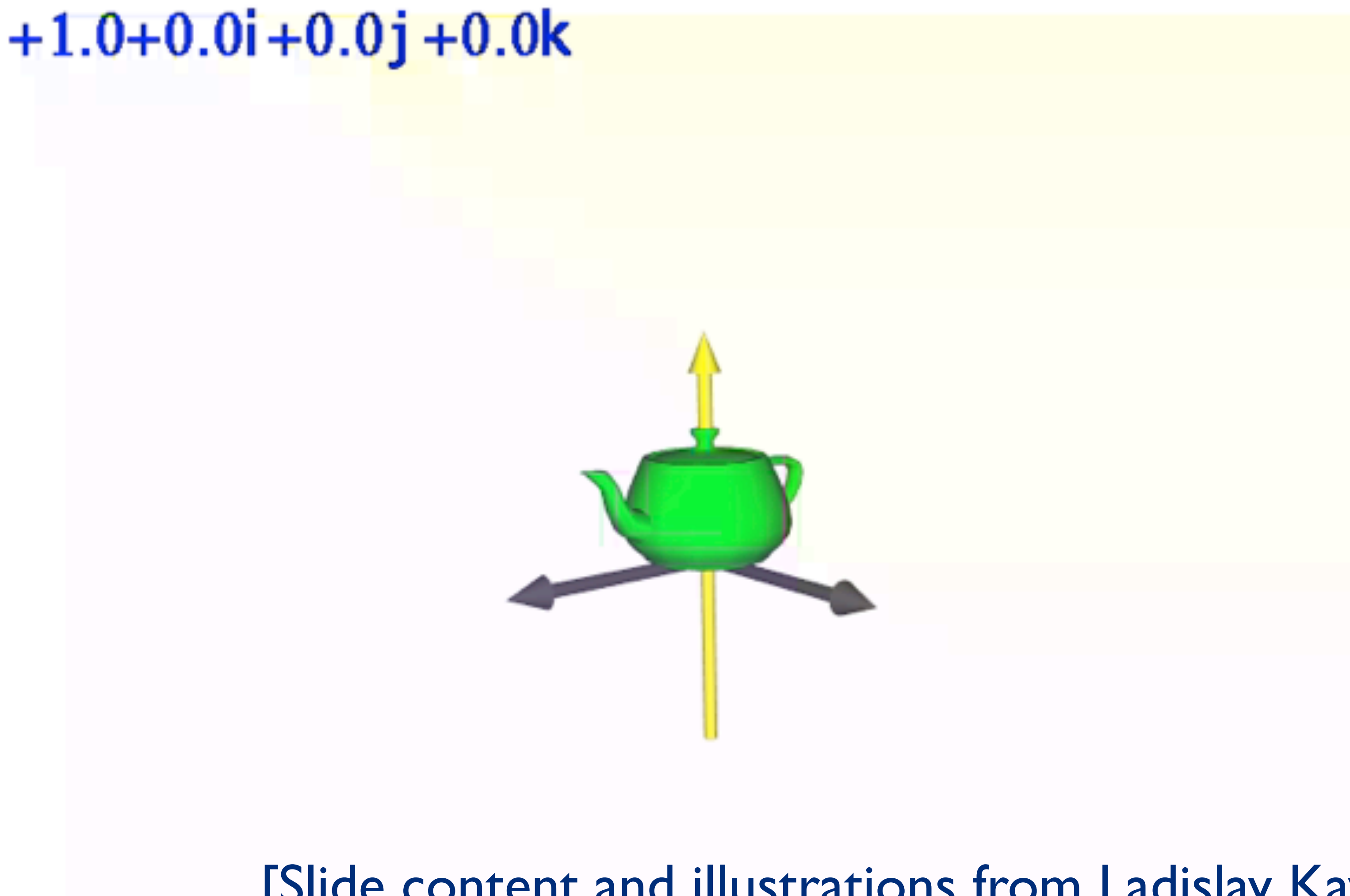
- Hence, to be a valid rotation, quaternion must be unit norm

$$||q|| = 1$$

Interpretation: Quaternions live on a sphere in a 4D space

Dual Quaternions [Clifford 1873]

- Dual quaternions are able to model rigid transformations (rotation + translation)
- Map a 6 dimensional manifold in an 8 dimensional space
- Need to be unit length to represent a valid rigid transform



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Dual Quaternions [Clifford 1873]

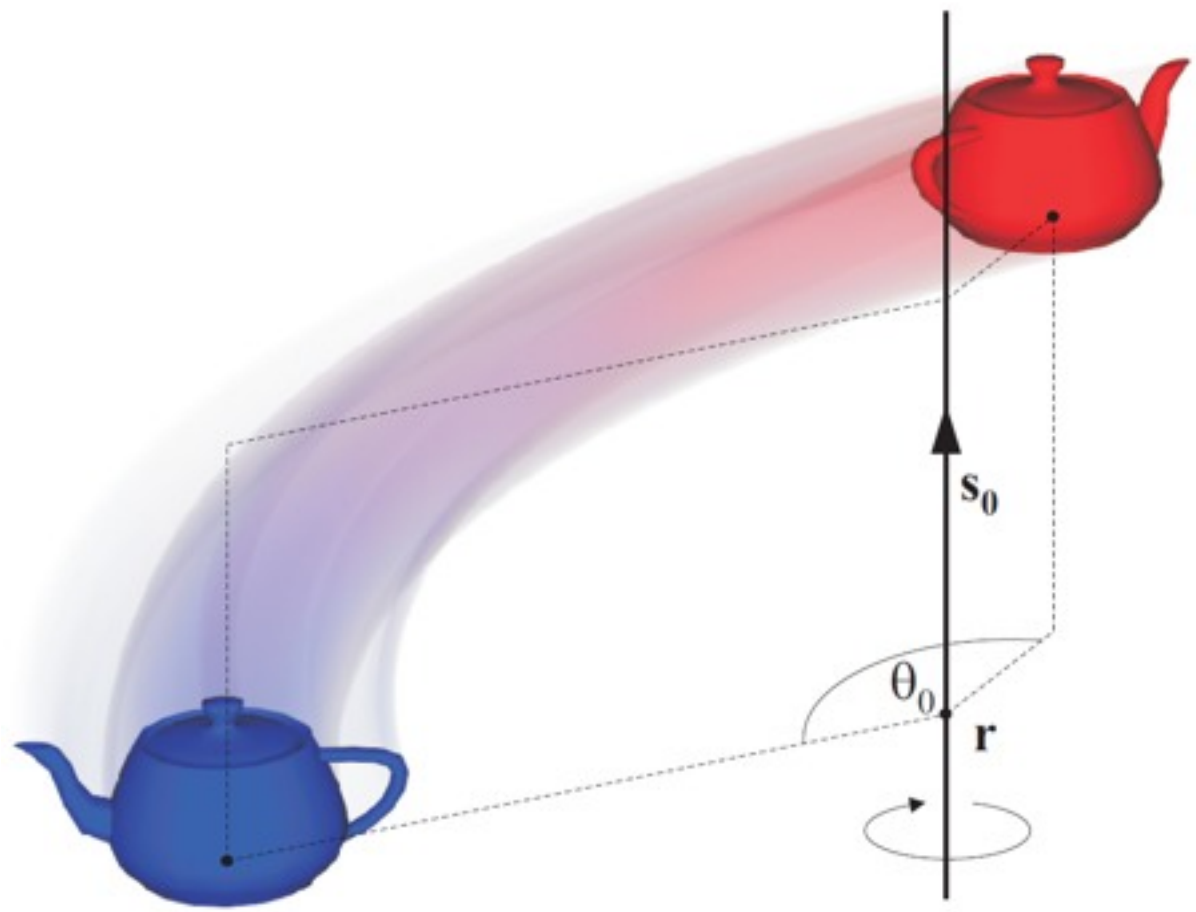
- Dual quaternions are able to model rigid transformations (rotation + translation)
- Map a 6 dimensional manifold in an 8 dimensional space
- Need to be unit length to represent a valid rigid transform

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Dual Quaternion Skinning

[Kavan et al., ACM TOG 2008]

Closed form approximation of SE(3) blending



[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Comparison: Linear Blend Skinning

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Comparison: Dual Quaternion

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Dual Quaternion Skinning

[Slide content and illustrations from Ladislav Kavan and Olga Sorkine]

Skinning Limitations

- All skinning methods assume a fixed relationship between skeleton motion and the mesh
- Humans are more complex (e.g., muscles lead to local deformations)
- Skinning only allows animation of predefined body geometry (created by an animator), do not help us create this geometry

Data-driven Body Shape Models

[Cyberware]

Idea: Let's scan real people and figure out how their body deforms and what body types are possible

Data-driven Body Shape Models

[Cyberware]

Pipeline

- Register all the scans
- Create (typically parametric) model of shape
- Use that model for an interesting application

Registration / Correspondence

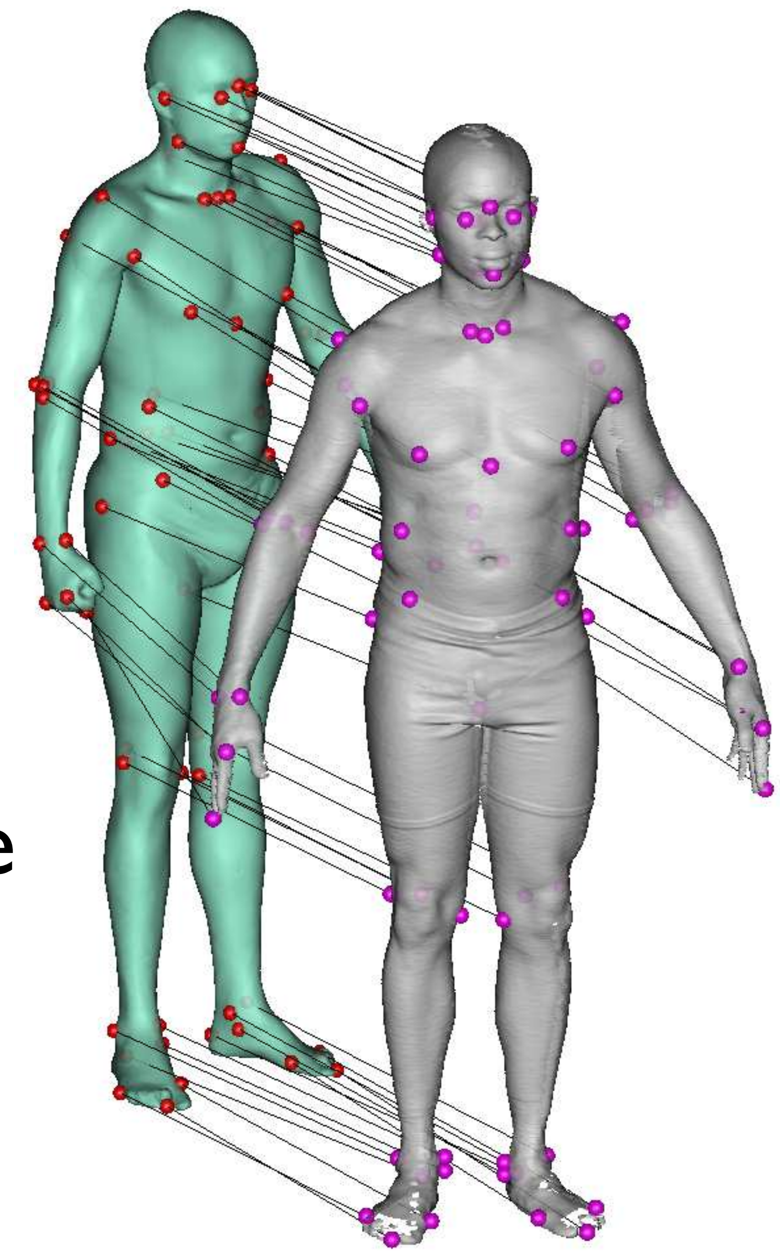
[Allen et al., 2003]

Marker-based Non-rigid Iterative Closest Point Registration

Goal: Fit a template mesh to triangulated 3D point cloud

$$E = \alpha E_d + \beta E_s + \gamma E_m$$

Amounts to estimating a 4x4 transform for every vertex through optimization of above



Registration / Correspondence

[Allen et al., 2003]

Marker-based Non-rigid Iterative Closest Point Registration

Goal: Fit a template mesh to triangulated 3D point cloud

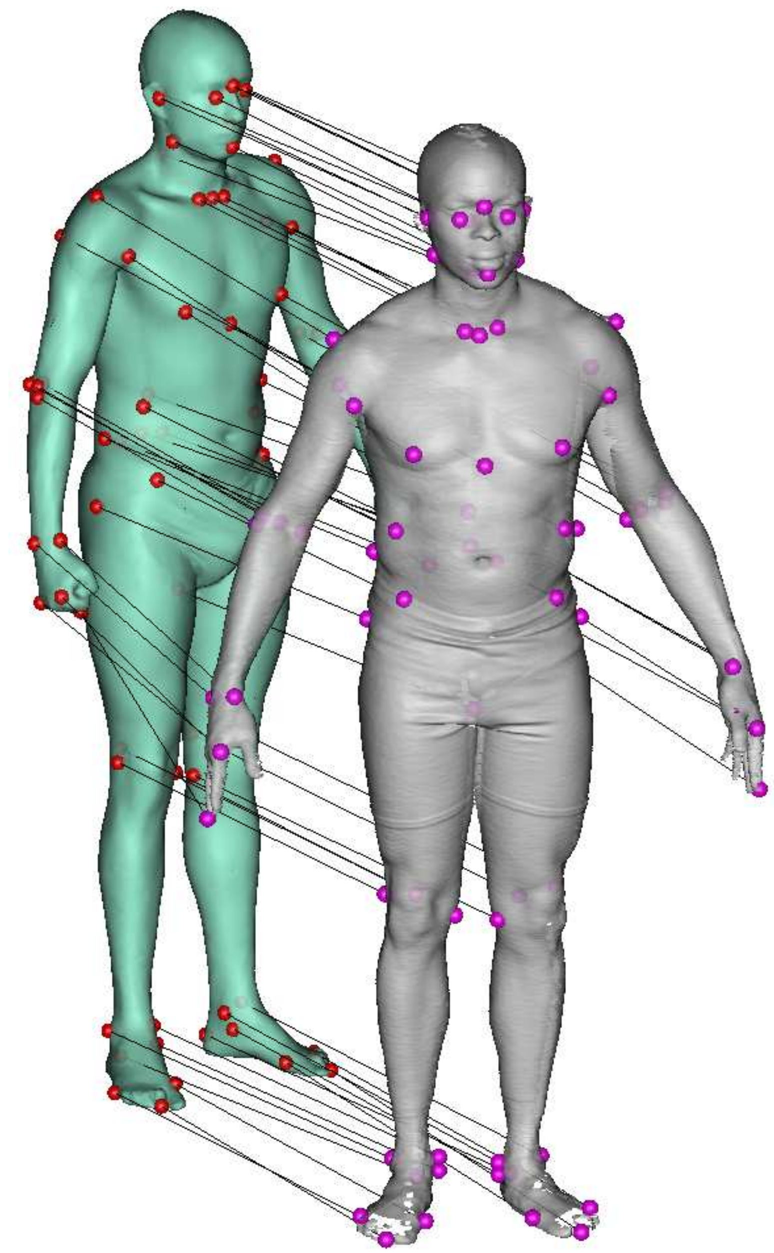
$$E = \alpha E_d + \beta E_s + \gamma E_m$$

data term

$$E_d = \sum_{i=1}^{V} w_i \,\mathrm{gap}^2(\mathbf{T}_i \vec{v}_i, \mathcal{D})$$

Distance from the reference mesh point to closest compatible vertex on observed surface

- Angle between surface normals
- Robust measure for dealing with holes



Registration / Correspondence

[Allen et al., 2003]

Marker-based Non-rigid Iterative Closest Point Registration

Goal: Fit a template mesh to triangulated 3D point cloud

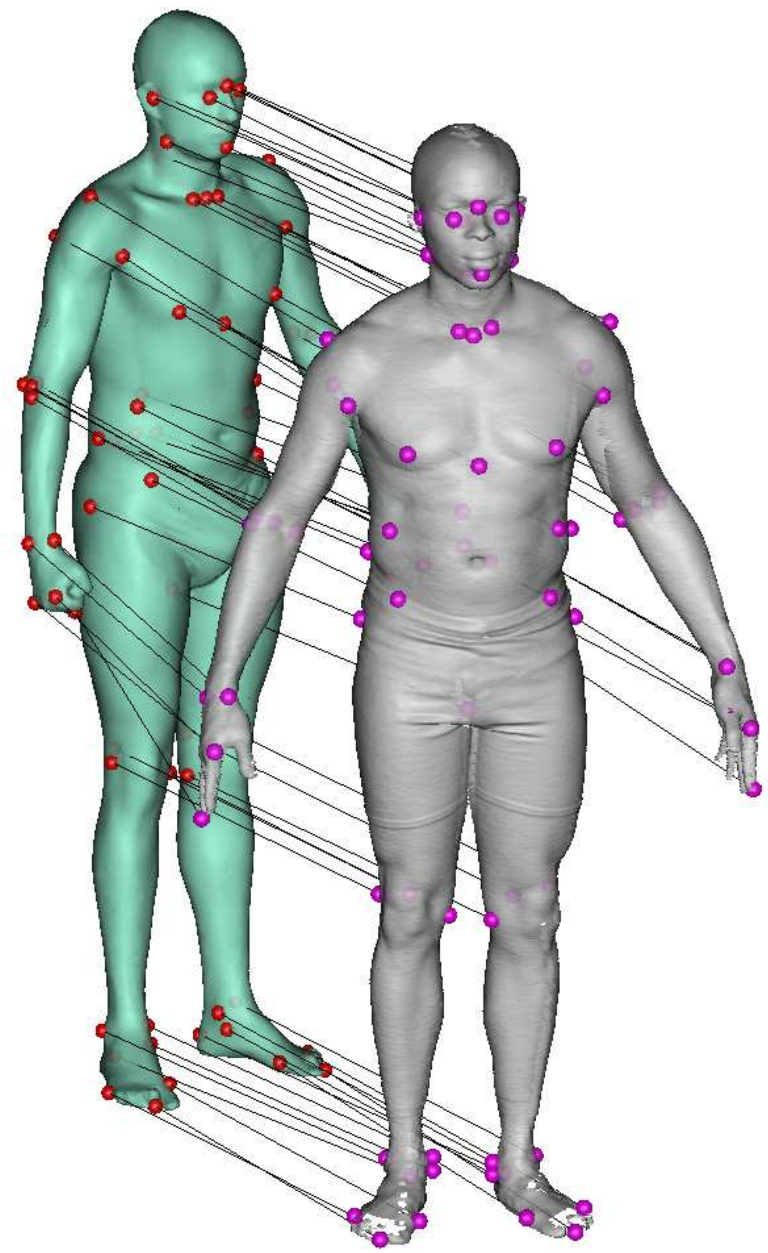
$$E = \alpha \underline{E_d} + \beta \underline{E_s} + \gamma E_m$$

data term

smoothness term

$$E_s = \sum_{\{i,j | (\vec{v}_i, \vec{v}_j) \in \text{edges}(\mathcal{T})\}} ||\mathbf{T}_i - \mathbf{T}_j||_F^2$$

Ensures that transforms on near by vertices are similar (induces local smoothness)



Registration / Correspondence

[Allen et al., 2003]

Marker-based Non-rigid Iterative Closest Point Registration

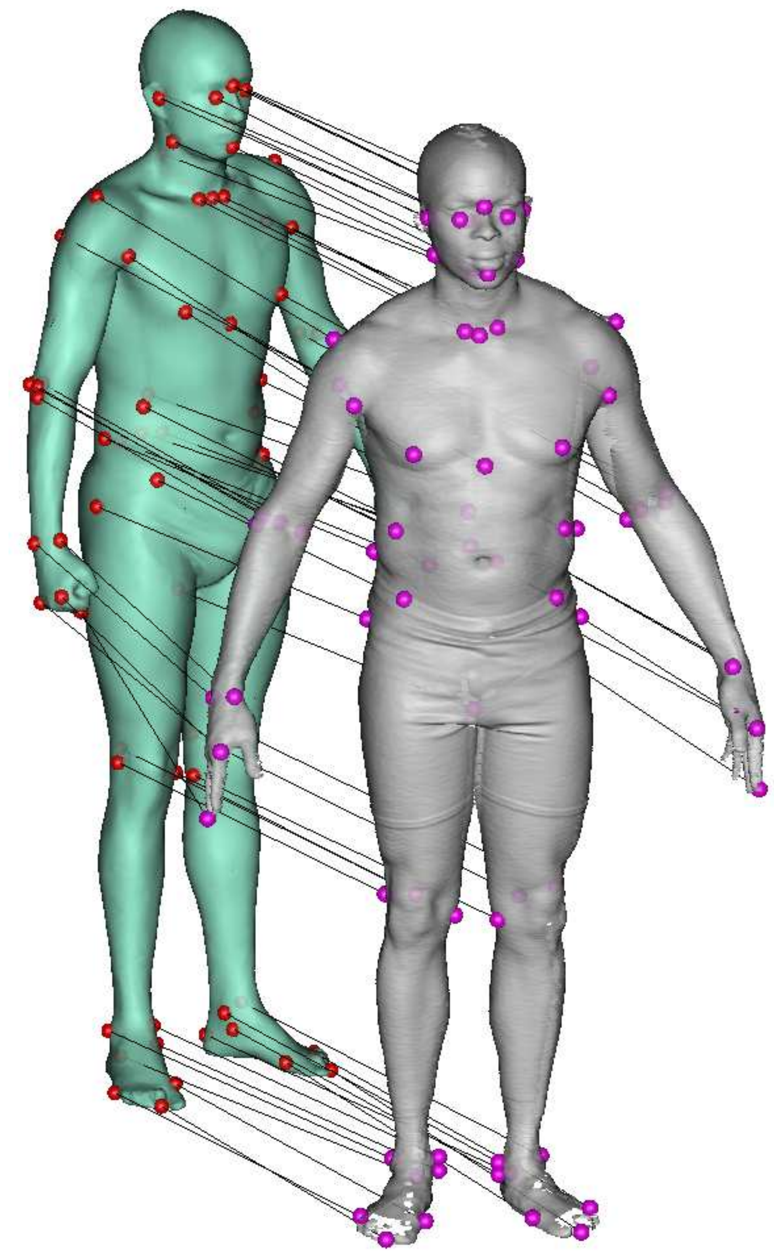
Goal: Fit a template mesh to triangulated 3D point cloud

$$E = \alpha E_d + \beta E_s + \gamma E_m$$

data term — smoothness term — marker anchor term

$$E_m = \sum_{i=1}^{M} ||\mathbf{T}_{\kappa_i} \vec{v}_{\kappa_i} - \vec{m}_i||^2$$

Ensures that transforms on near by vertices are similar (induces local smoothness)



Registration / Correspondence

[Allen et al., 2003]

Marker-based Non-rigid Iterative Closest Point Registration

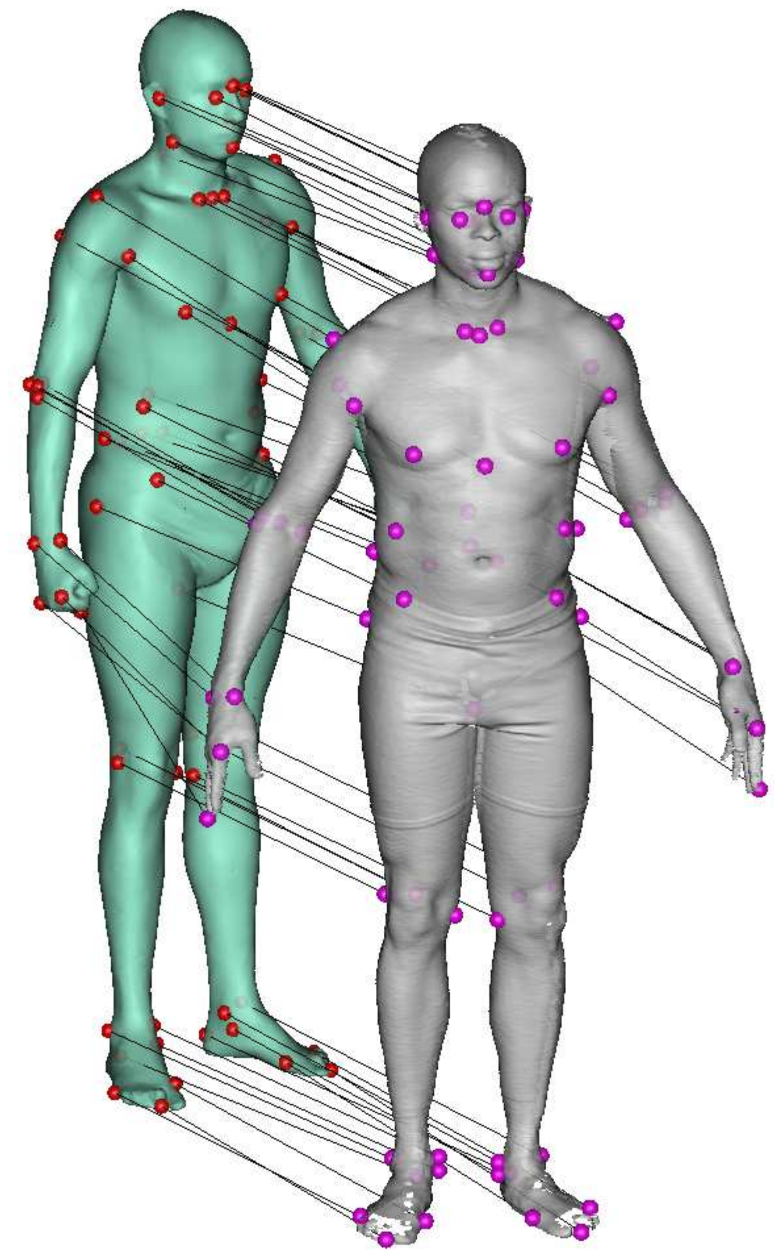
Goal: Fit a template mesh to triangulated 3D point cloud

$$E = \alpha \underline{E_d} + \beta \underline{E_s} + \gamma \underline{E_m}$$

data term — smoothness term — marker anchor term

Solved using gradient descent

Initialized by aligning centers of mass



Registration

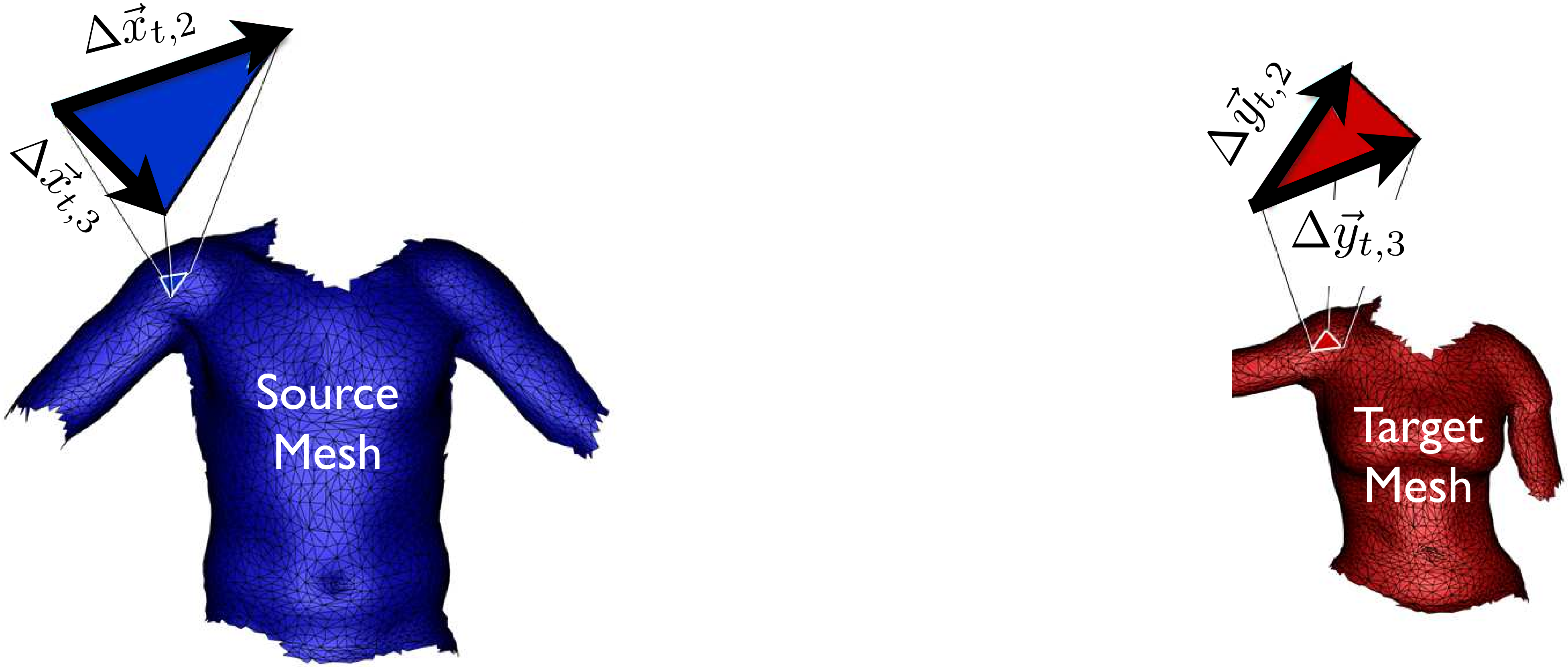
[Image from Alexandru Balan]

Mesh Deformation Gradients

[Sumner and Popovic, 2004]

3x3 transform for every triangle

$$\mathbf{A}_t\,[\Delta\vec{x}_{t,2}\ ,\ \Delta\vec{x}_{t,3}] = [\Delta\vec{y}_{t,2}\ ,\ \Delta\vec{y}_{t,3}]$$



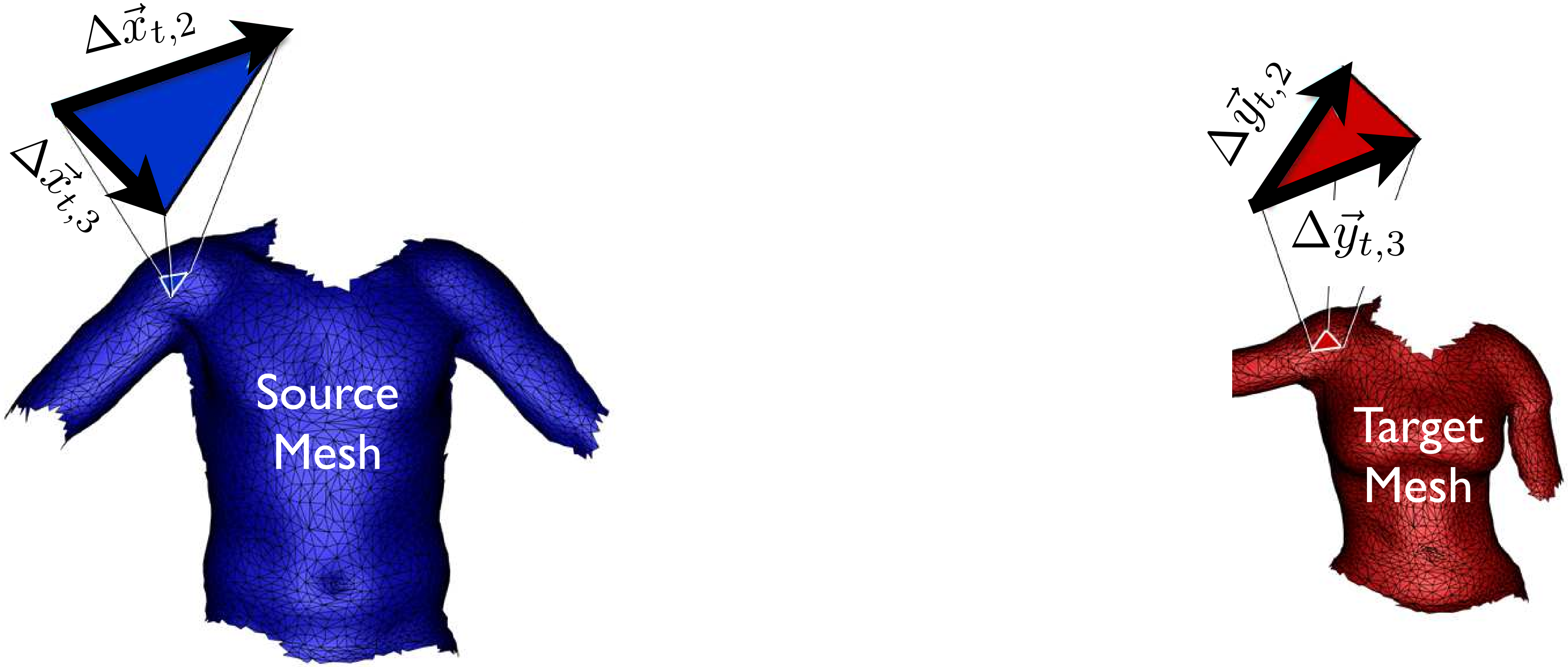
[Image from Alexandru Balan]

Mesh Deformation Gradients

[Sumner and Popovic, 2004]

3x3 transform for every triangle

$$\mathbf{A}_t \, [\Delta\vec{x}_{t,2} \; , \; \Delta\vec{x}_{t,3}] = [\Delta\vec{y}_{t,2} \; , \; \Delta\vec{y}_{t,3}]$$



[Image from Alexandru Balan]

Estimation:

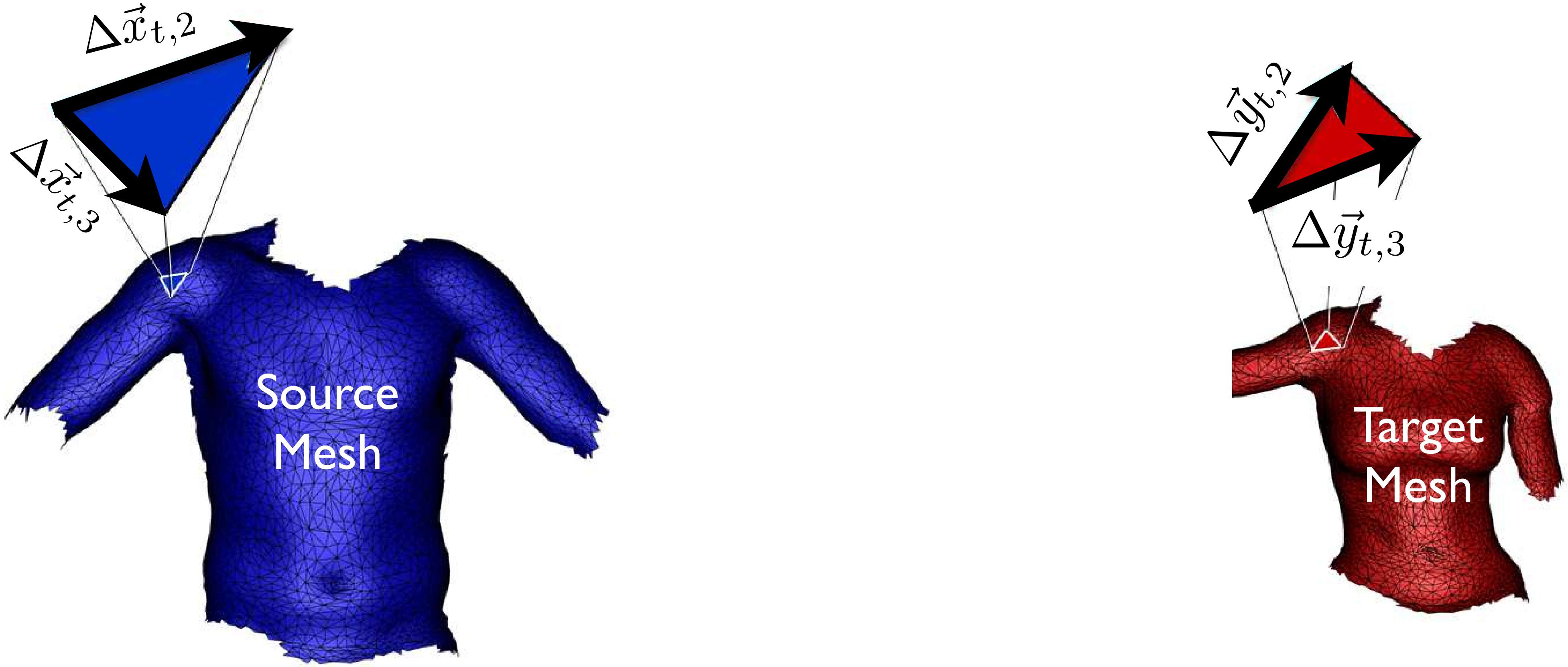
$$\arg\min_{\{\mathbf{A}_1, \cdots, \mathbf{A}_T\}} \sum_{t=1}^{T} \sum_{k=2,3} ||\mathbf{A}_t \Delta\vec{x}_{t,k} - \Delta\vec{y}_{t,k}||^2$$

Mesh Deformation Gradients

[Sumner and Popovic, 2004]

3x3 transform for every triangle

$$\mathbf{A}_t \, [\Delta\vec{x}_{t,2} \ , \ \Delta\vec{x}_{t,3}] = [\Delta\vec{y}_{t,2} \ , \ \Delta\vec{y}_{t,3}]$$



[Image from Alexandru Balan]

Estimation:

$$\arg\min_{\{\mathbf{A}_1,\cdots,\mathbf{A}_T\}} \sum_{t=1}^{T} \sum_{k=2,3} ||\mathbf{A}_t \Delta\vec{x}_{t,k} - \Delta\vec{y}_{t,k}||^2$$

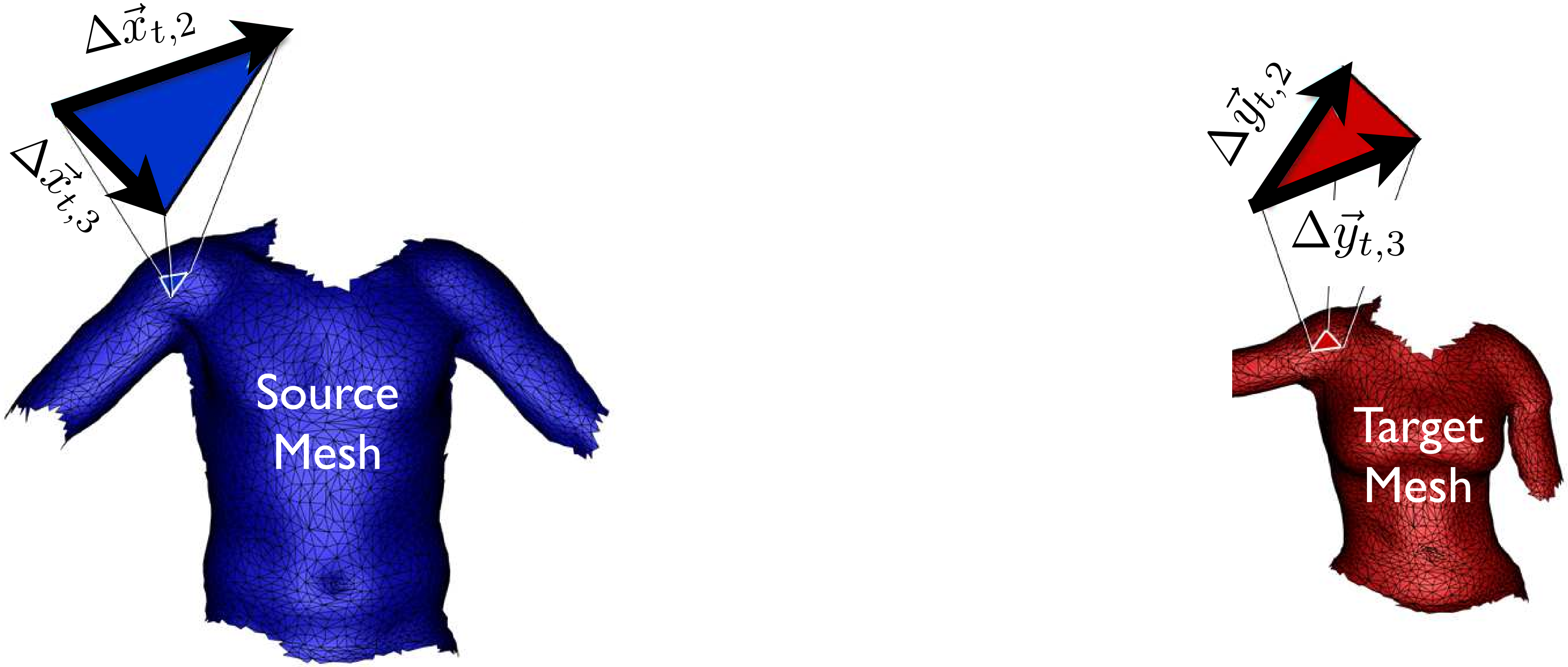
under-constrained

Mesh Deformation Gradients

[Sumner and Popovic, 2004]

3x3 transform for every triangle

$$\mathbf{A}_t \left[\Delta\vec{x}_{t,2}\ ,\ \Delta\vec{x}_{t,3}\right] = \left[\Delta\vec{y}_{t,2}\ ,\ \Delta\vec{y}_{t,3}\right]$$



[Image from Alexandru Balan]

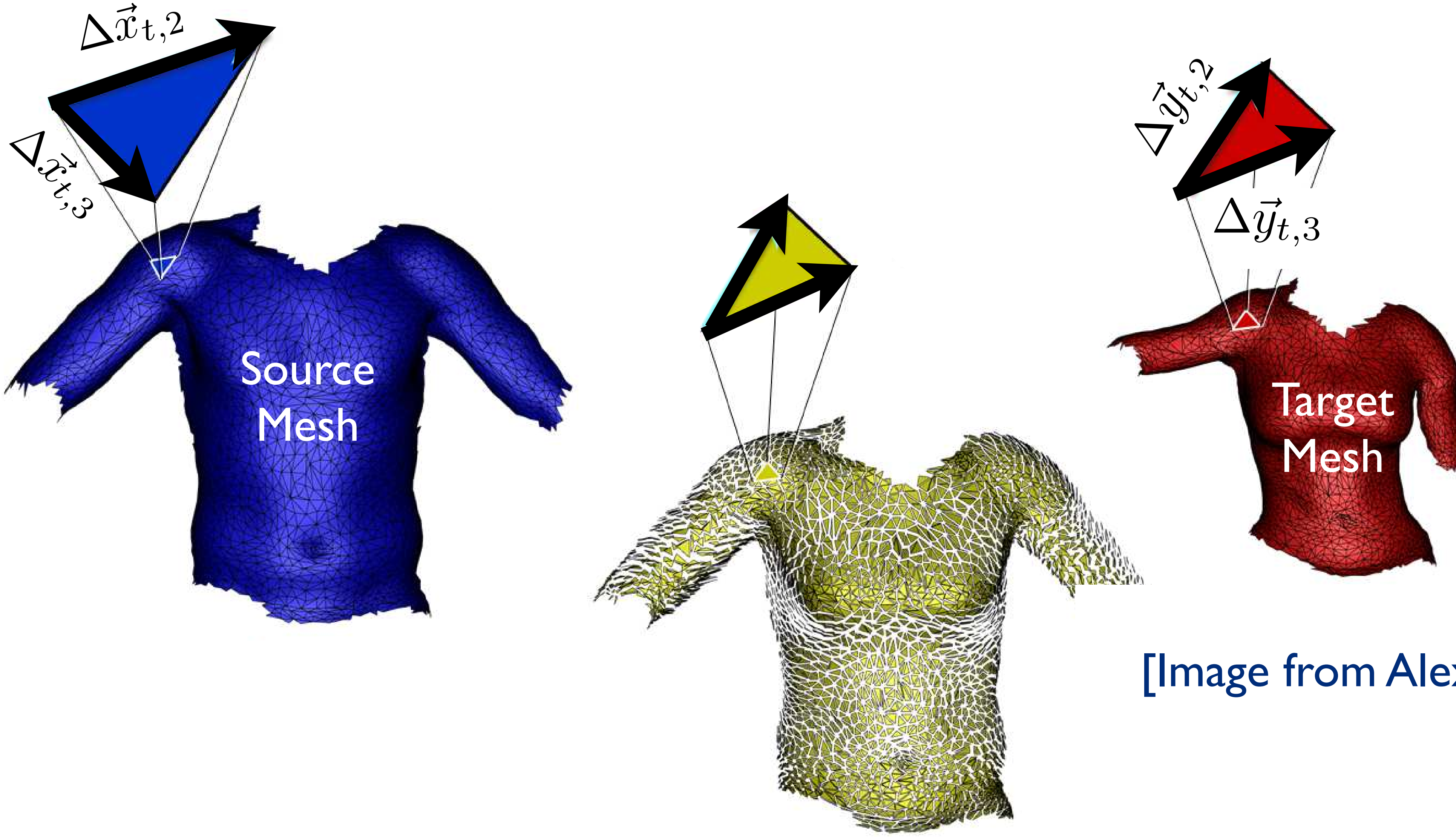
Estimation:

$$\arg\min_{\{\mathbf{A}_1,\cdots,\mathbf{A}_T\}} \sum_{t=1}^{T} \sum_{k=2,3} ||\mathbf{A}_t \Delta\vec{x}_{t,k} - \Delta\vec{y}_{t,k}||^2 + w_s \sum_{t_1,t_2 \text{ adj}} ||\mathbf{A}_{t_1} - \mathbf{A}_{t_2}||_F^2$$

Mesh Deformation Gradients

[Sumner and Popovic, 2004]

Applying deformation gradients typically leads to inconsistent edges and structures

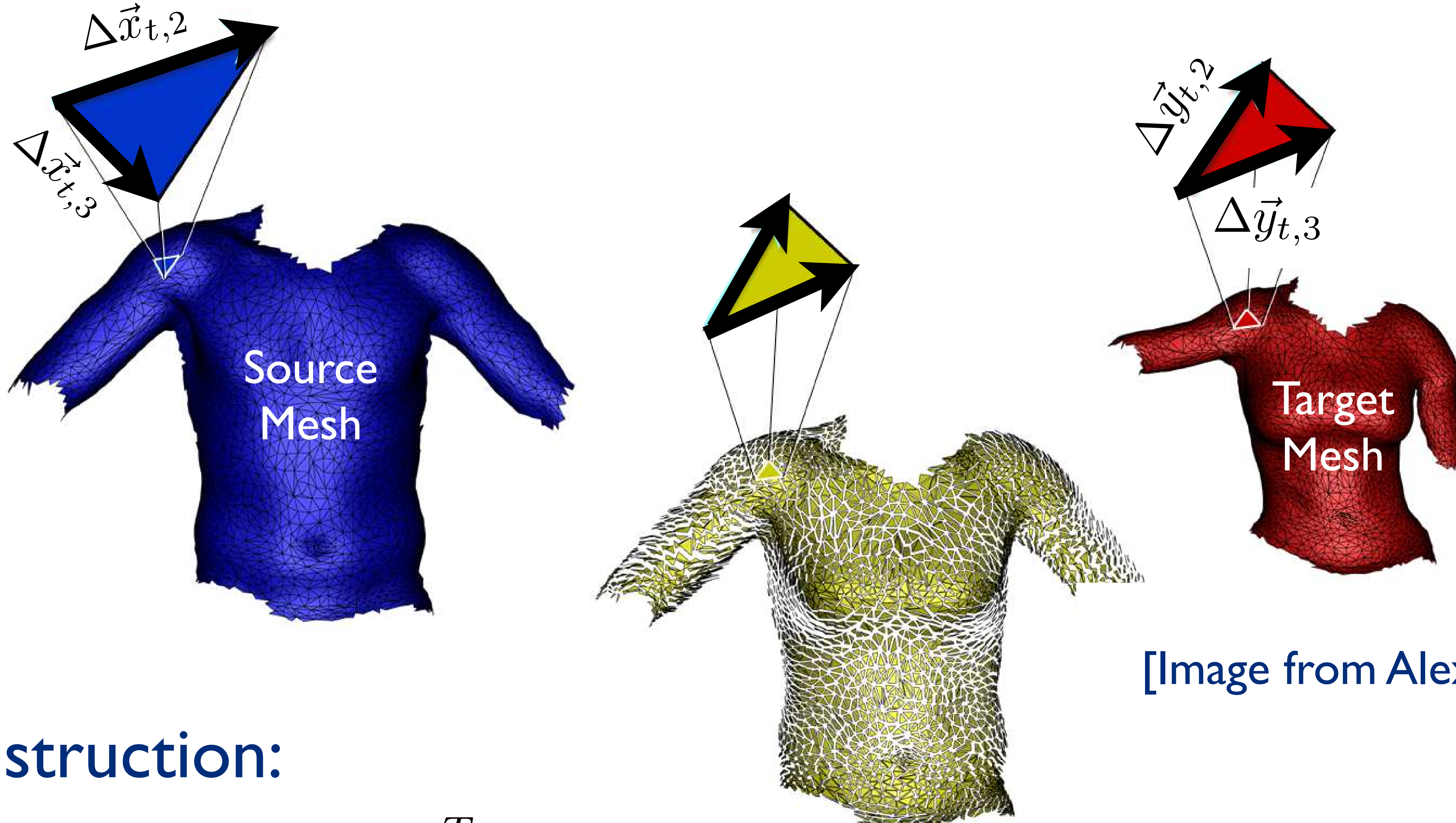


[Image from Alexandru Balan]

Mesh Deformation Gradients

[Sumner and Popovic, 2004]

Applying deformation gradients typically leads to inconsistent edges and structures



[Image from Alexandru Balan]

Reconstruction:

$$\arg\min_{\{\vec{y}_1,\cdots,\vec{y}_V\}} \sum_{t=1}^{T} \sum_{k=2,3} ||\mathbf{A}_t \Delta\vec{x}_{t,k} - \Delta\vec{y}_{t,k}||^2$$

SCAPE: Shape Completion and Animation of PEople

[Angelov et al., ACM TOG, 2005]

Key Idea: factor mesh deformation for a person into:

(1) Articulated rigid deformations

(2) Non-rigid deformations

(3) Body shape deformations

SCAPE: Shape Completion and Animation of PEople

[Angelov et al., ACM TOG, 2005]

Key Idea: factor mesh deformation for a person into:

(1) Articulated rigid deformations

(2) Non-rigid deformations

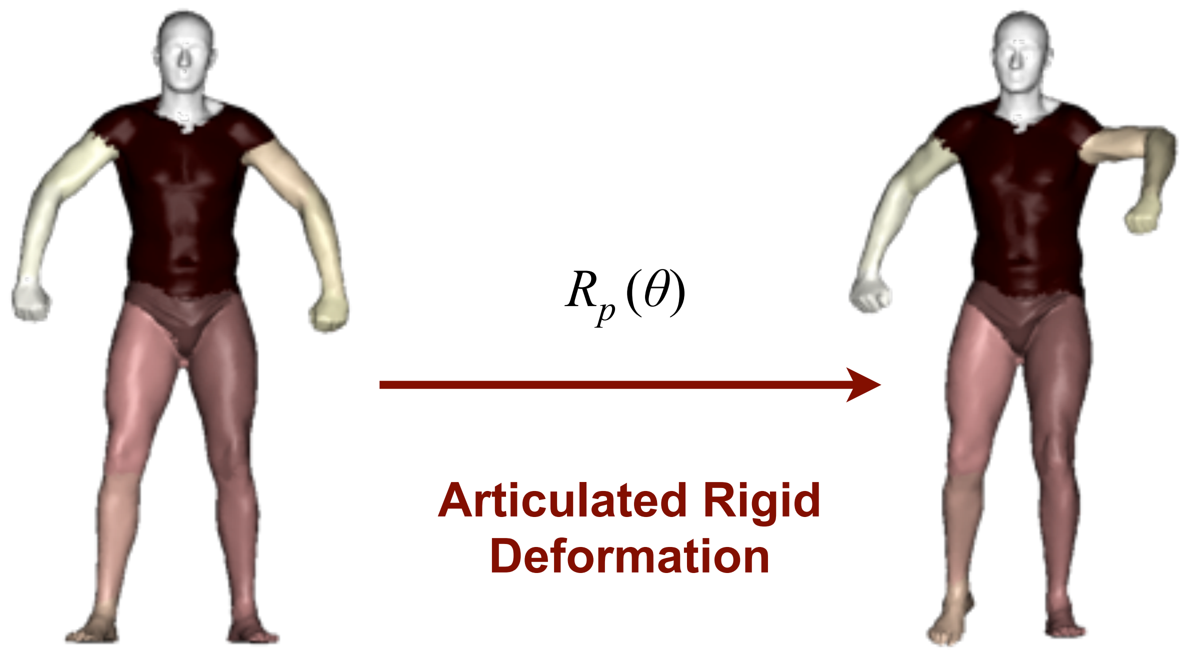
(3) Body shape deformations

Gives a parametric model of any person in any pose!

Articulated Rigid Deformation

[Angelov et al., ACM TOG, 2005]

This is basically rigid skinning



θ – joint angles

[Image from Alexandru Balan]

Non-rigid Deformations

[Angelov et al., ACM TOG, 2005]

From a dataset of one person in different poses, learn residual deformations

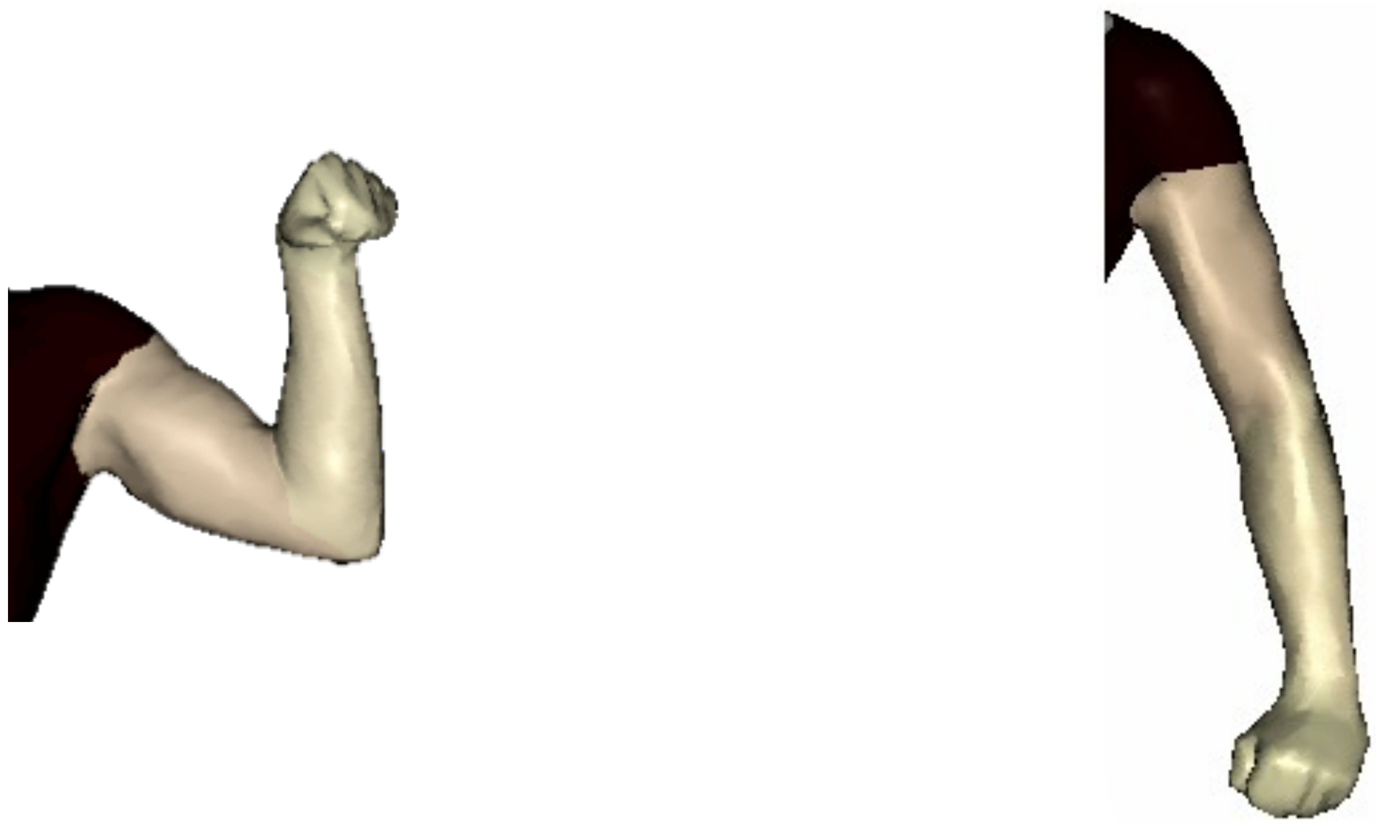
$$\underset{\{\mathbf{Q}_1^i,\cdots,\mathbf{Q}_T^i\}}{\arg\min} \sum_{t=1}^{T} \sum_{k=2,3} \left|\left|\mathbf{R}_{p[t]}^i \mathbf{Q}_t^i \Delta\vec{x}_{t,k} - \Delta\vec{y}_{t,k}^i\right|\right|^2 + w_s \sum_{\substack{t_1,t_2 \text{ adj} \\ p[t_1]=p[t_2]}} ||\mathbf{Q}_{t_1}^i - \mathbf{Q}_{t_2}^i||_F^2$$

[Image from Alexandru Balan]

Non-rigid Deformations

[Angelov et al., ACM TOG, 2005]

Let non-rigid deformations be linear functions of pose

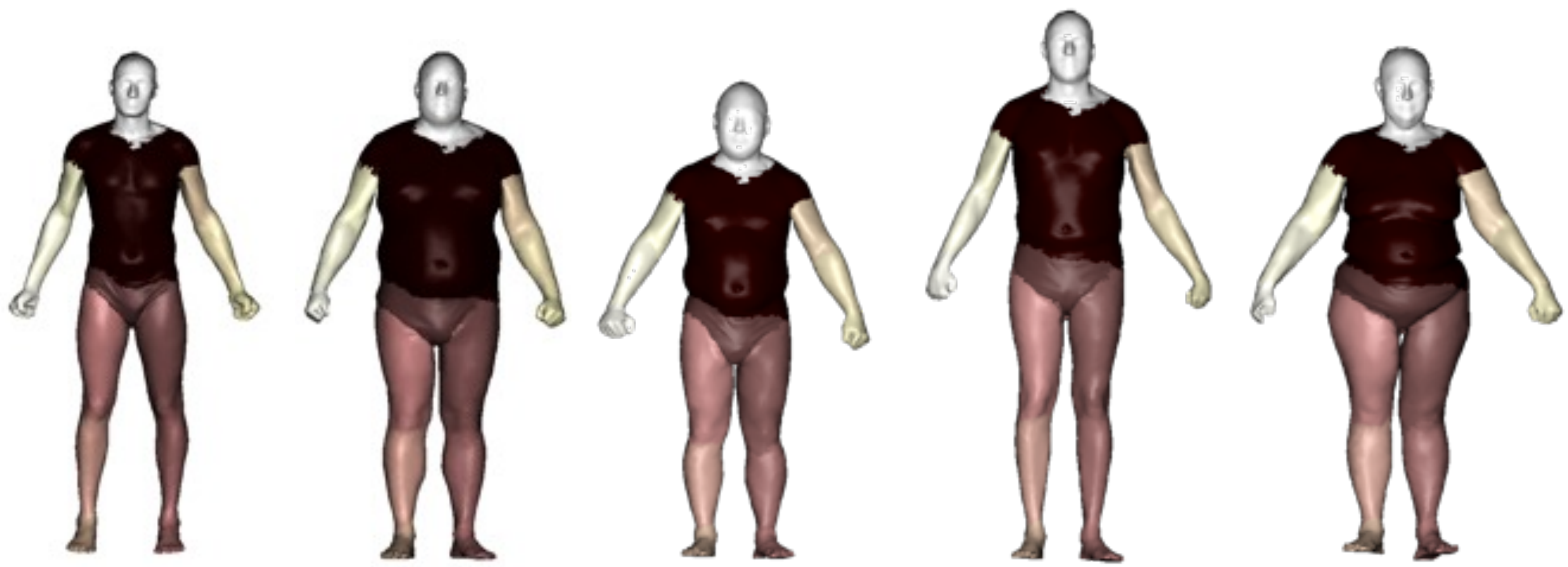


Models bulging of muscles, etc.

[Image from Alexandru Balan]

Body Shape

[Angelov et al., ACM TOG, 2005]

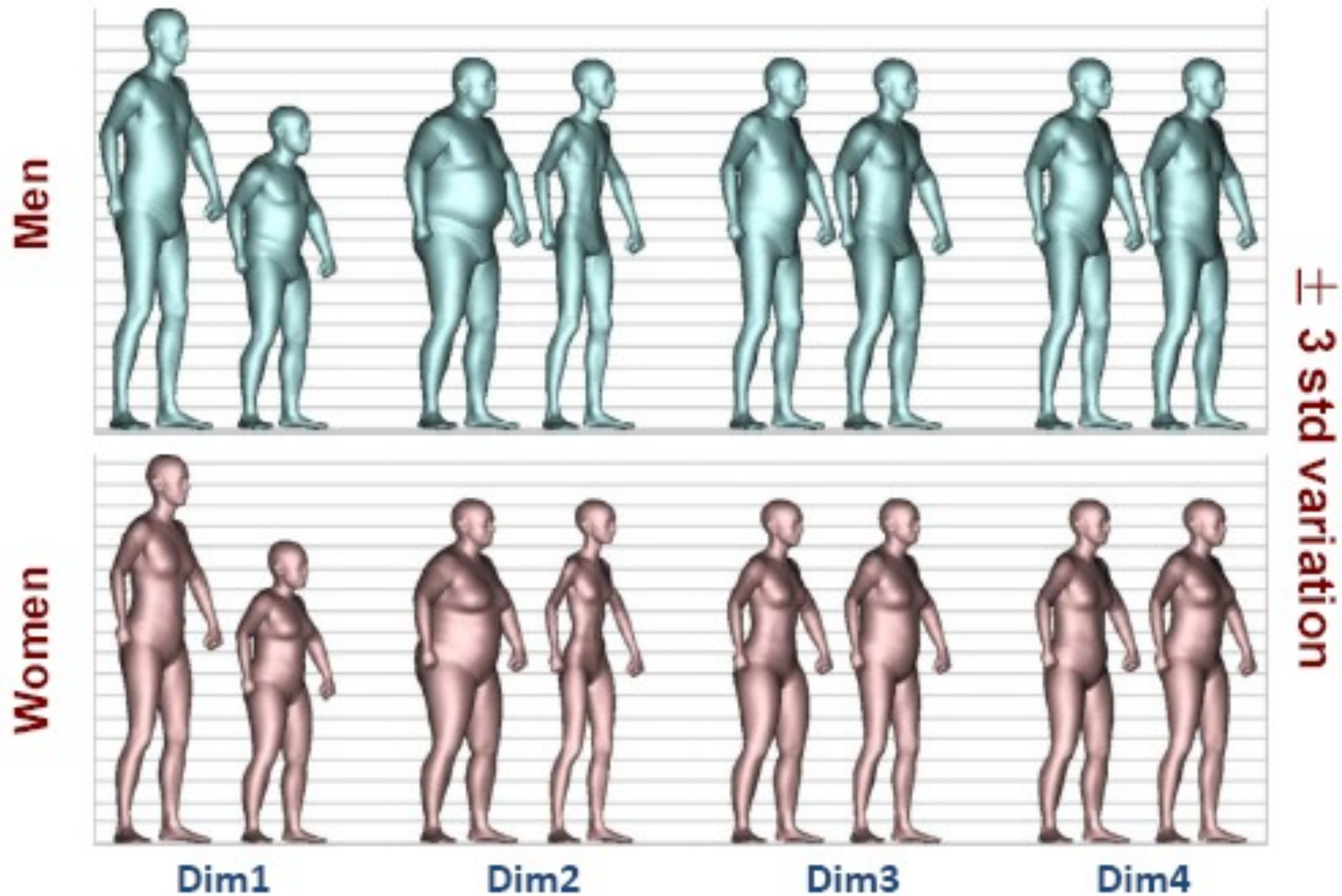


From a dataset of many people in one pose learn variations

- Extract mesh deformation gradients
- Do PCA on them (vectorizing them first)
  - 6 PCA dimensions can capture > 80% of variation

[Image from Alexandru Balan]

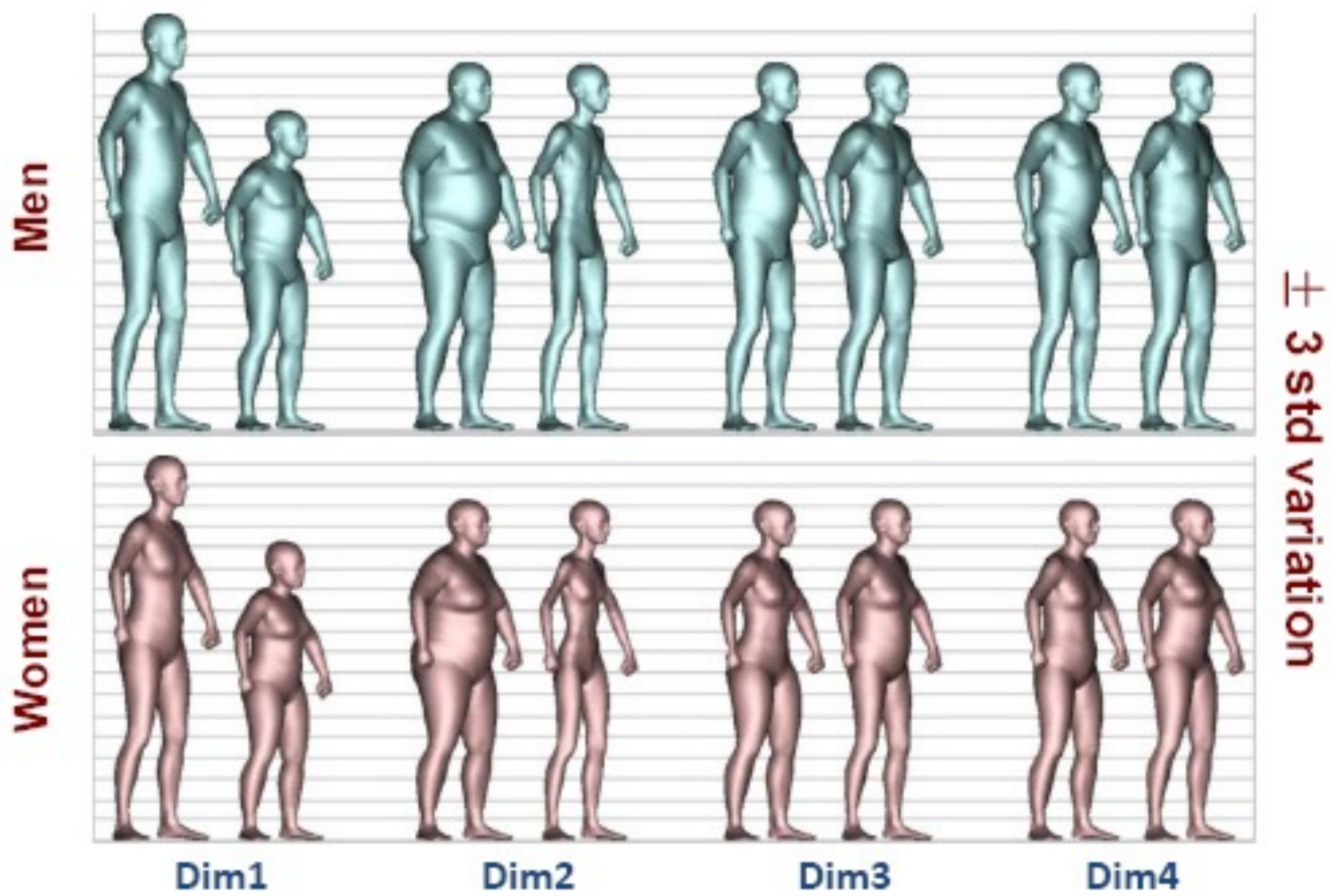
Body Shape



[Image from Peng Guan]

Body Shape

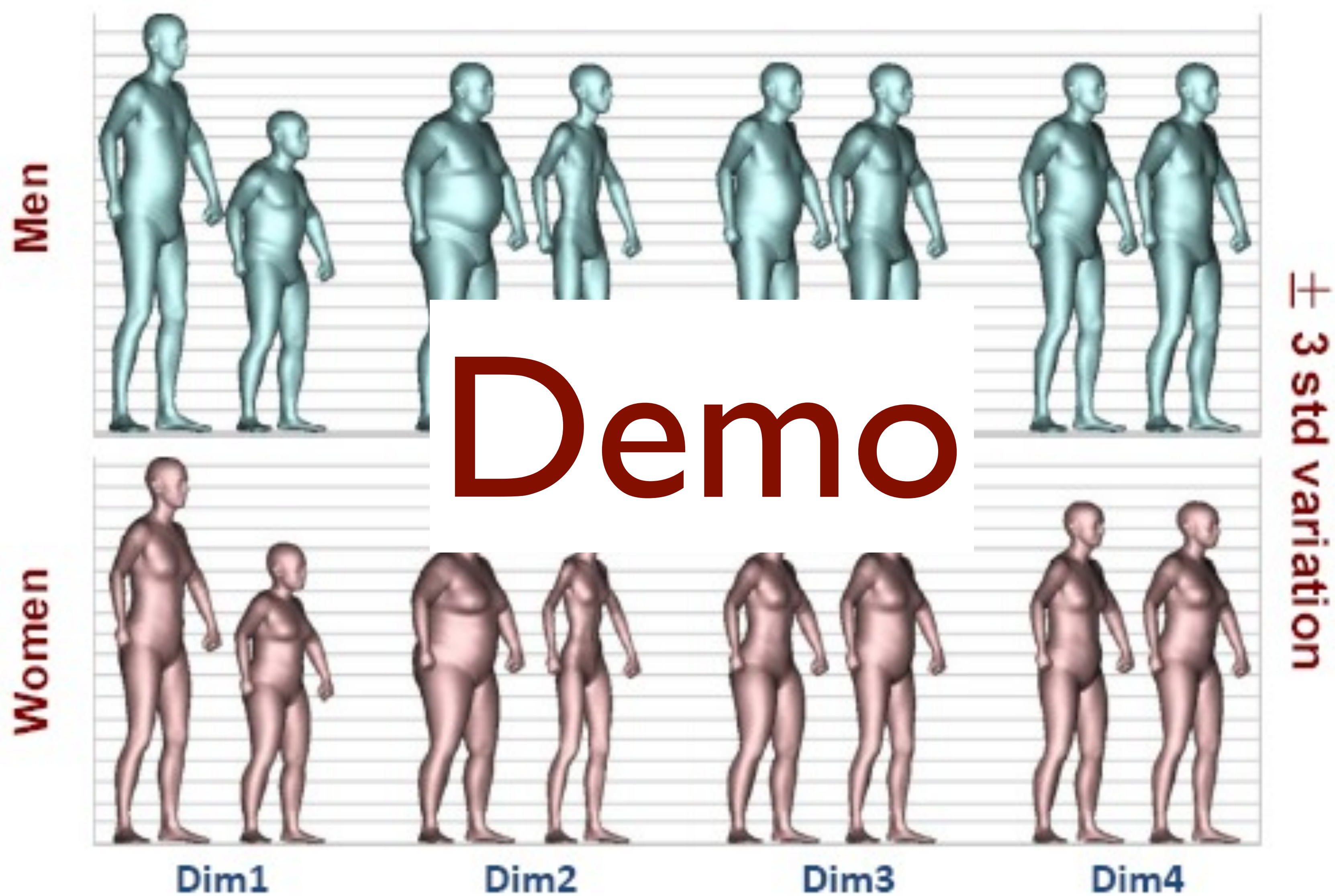
It is also possible to create semantic parameters and regress from them to PCA coefficients



[Image from Peng Guan]

Body Shape

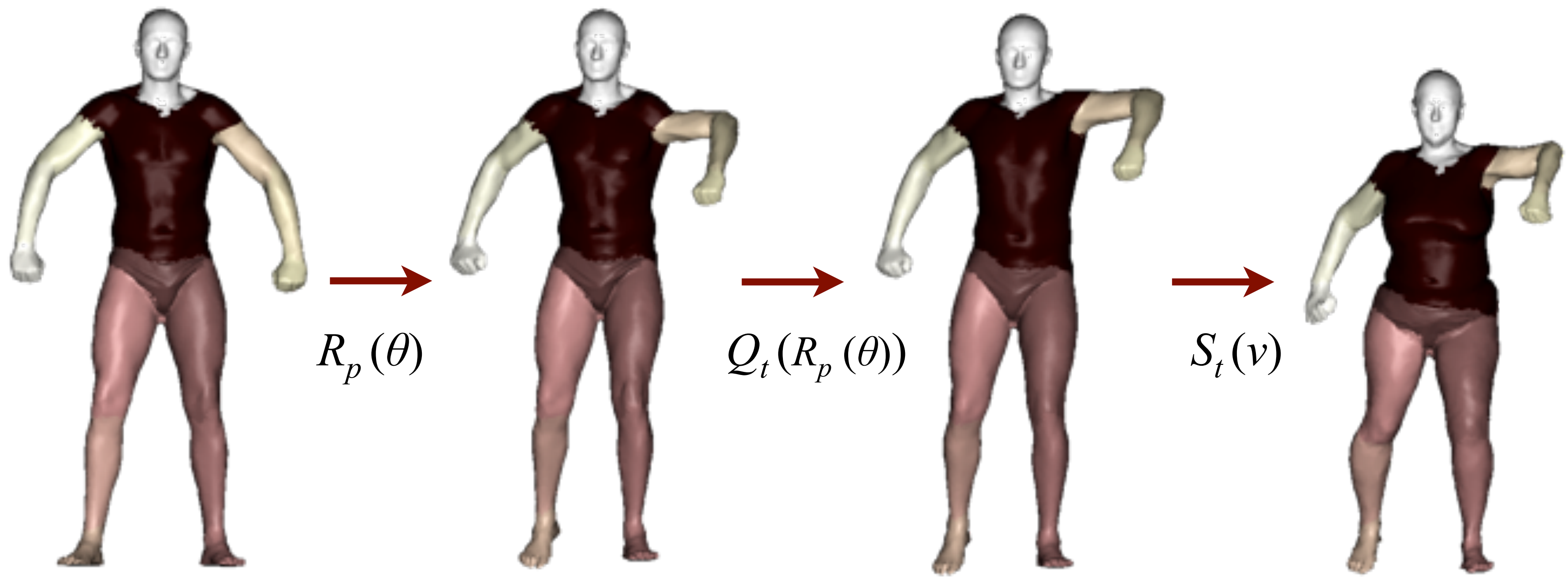
It is also possible to create semantic parameters and regress from them to PCA coefficients



[Image from Peng Guan]

SCAPE: Putting it all together

[Angelov et al., ACM TOG, 2005]

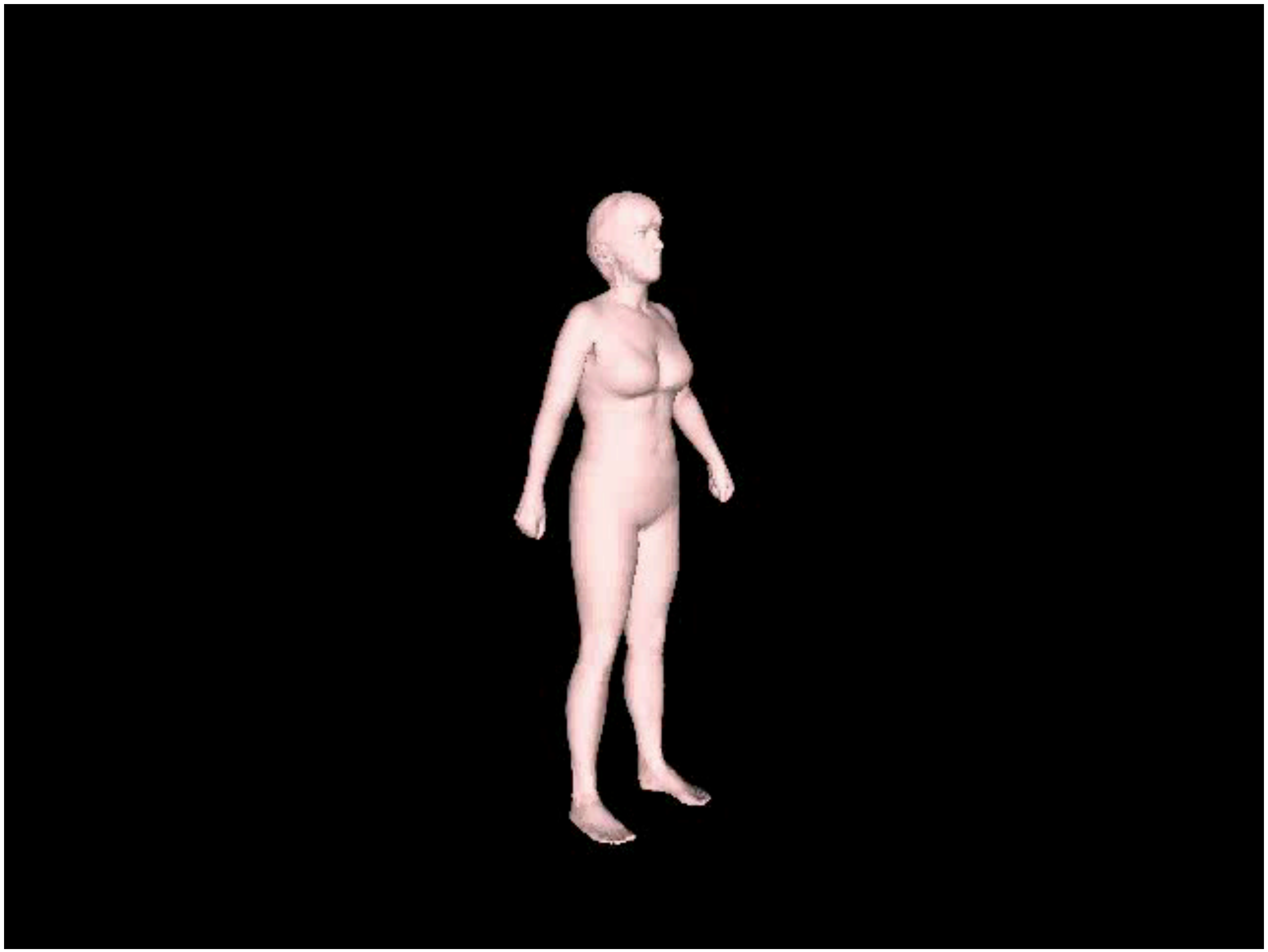


θ – joint angles

v – shape parameters

We can simply concatenate all the deformations by multiplying them together

SCAPE Model



[Video curtesy of Peng Guan]

Applications: Shape Estimation

[Sigal et al., NIPS'2007]

Goal: learn functional mapping from image features to pose and shape parameters of the SCAPE model

(from synthesized input-output pairs)

Applications: Shape Estimation

[Sigal et al., NIPS'2007]

Goal: learn functional mapping from image features to pose and shape parameters of the SCAPE model

(from synthesized input-output pairs)

Remember how Kinect works? Let's try that for shape

Applications: Shape Estimation

[Sigal et al., NIPS'2007]

Goal: learn functional mapping from image features to pose and shape parameters of the SCAPE model

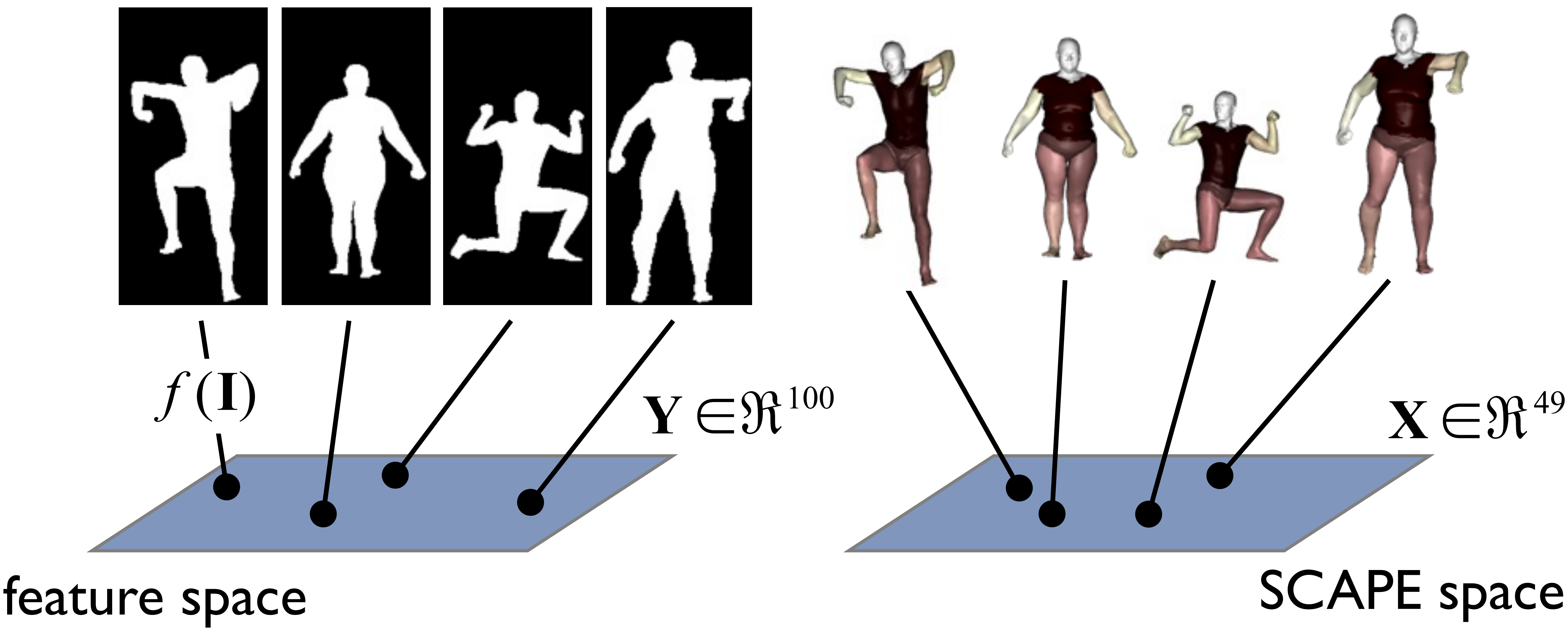
(from synthesized input-output pairs)

Applications: Shape Estimation

[Sigal et al., NIPS'2007]

Goal: learn functional mapping from image features to pose and shape parameters of the SCAPE model

(from synthesized input-output pairs)

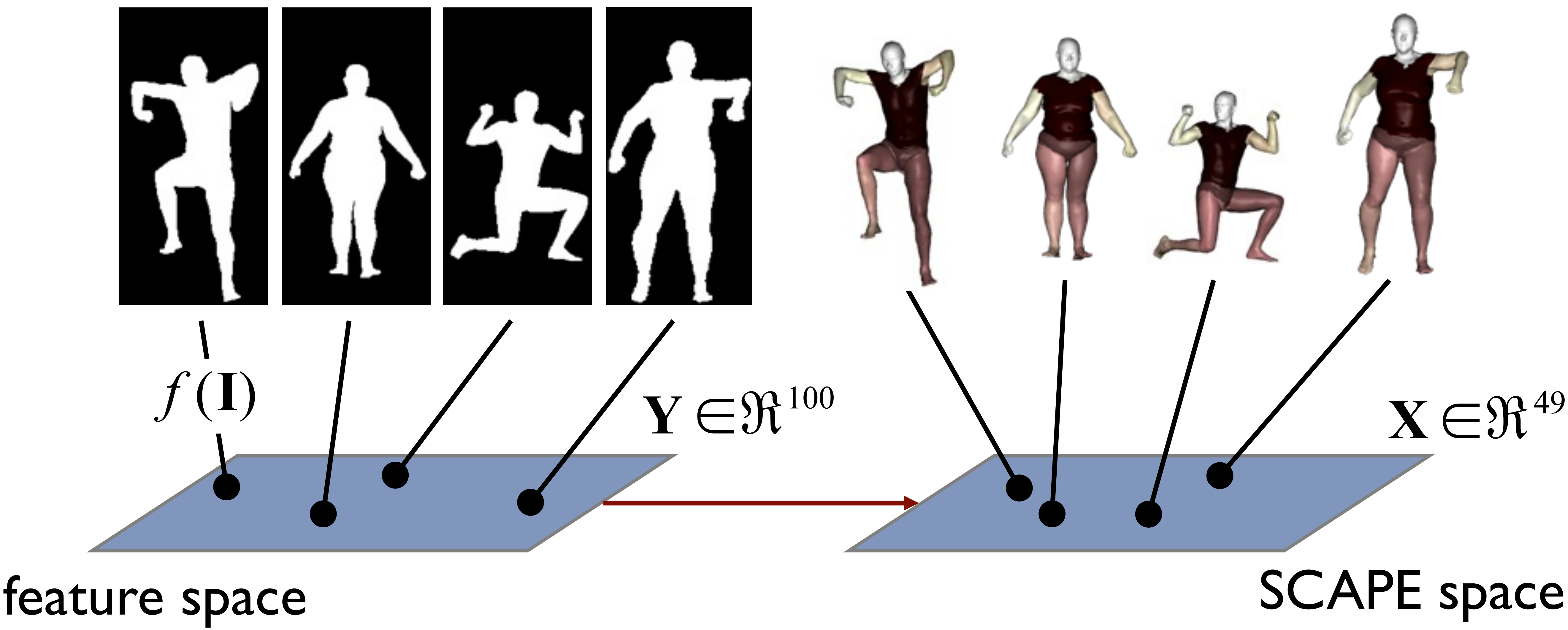


Applications: Shape Estimation

[Sigal et al., NIPS'2007]

Goal: learn functional mapping from image features to pose and shape parameters of the SCAPE model

(from synthesized input-output pairs)

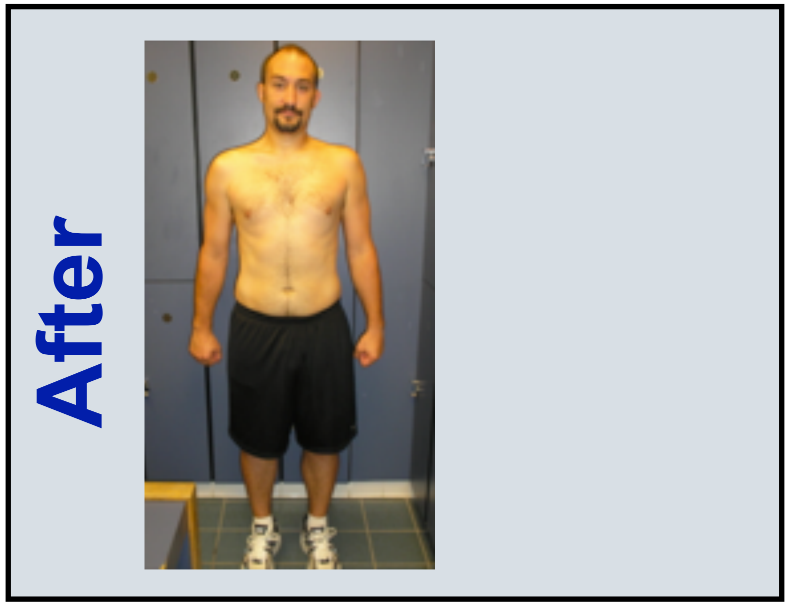


SCAPE parameters = g (features)

Proof of concept results

[Sigal et al., NIPS'2007]

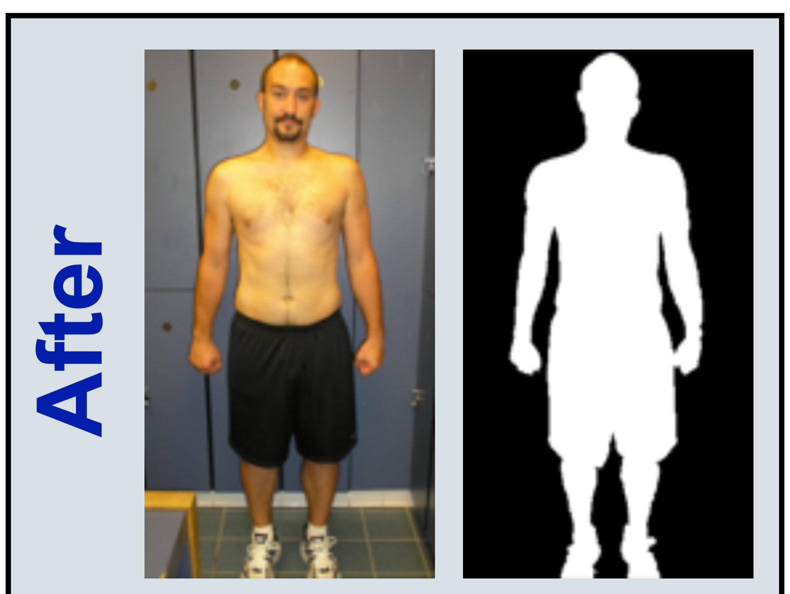
Can we estimate weight loss discriminatively from monocular images?



Proof of concept results

[Sigal et al., NIPS'2007]

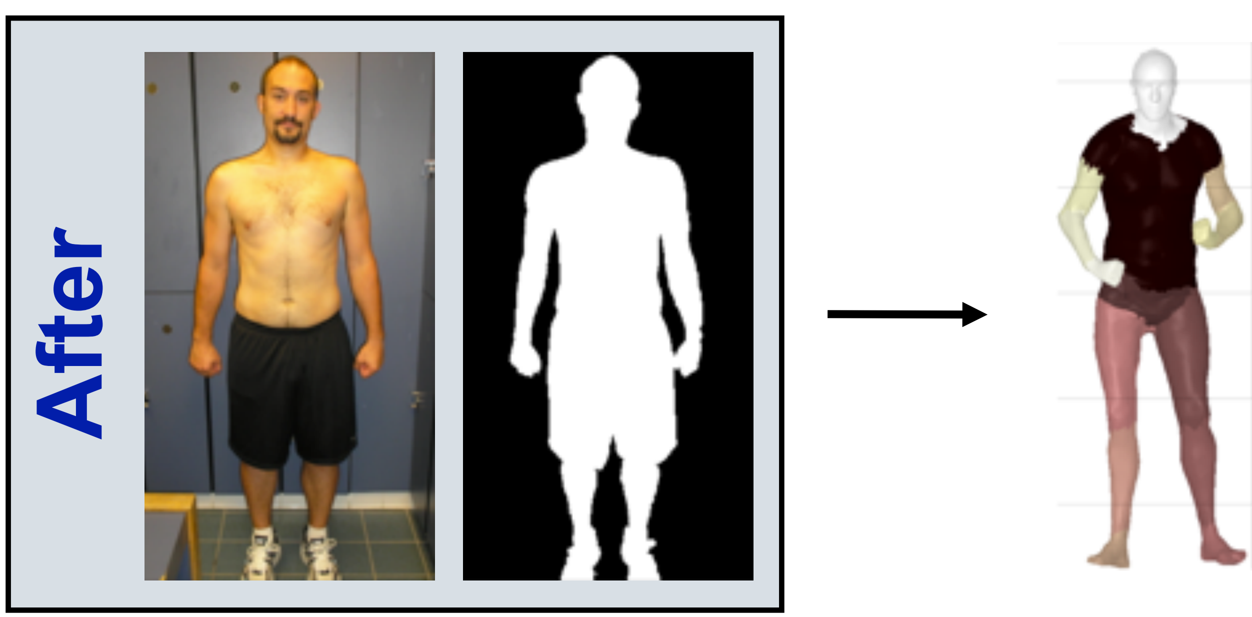
Can we estimate weight loss discriminatively from monocular images?



Proof of concept results

[Sigal et al., NIPS'2007]

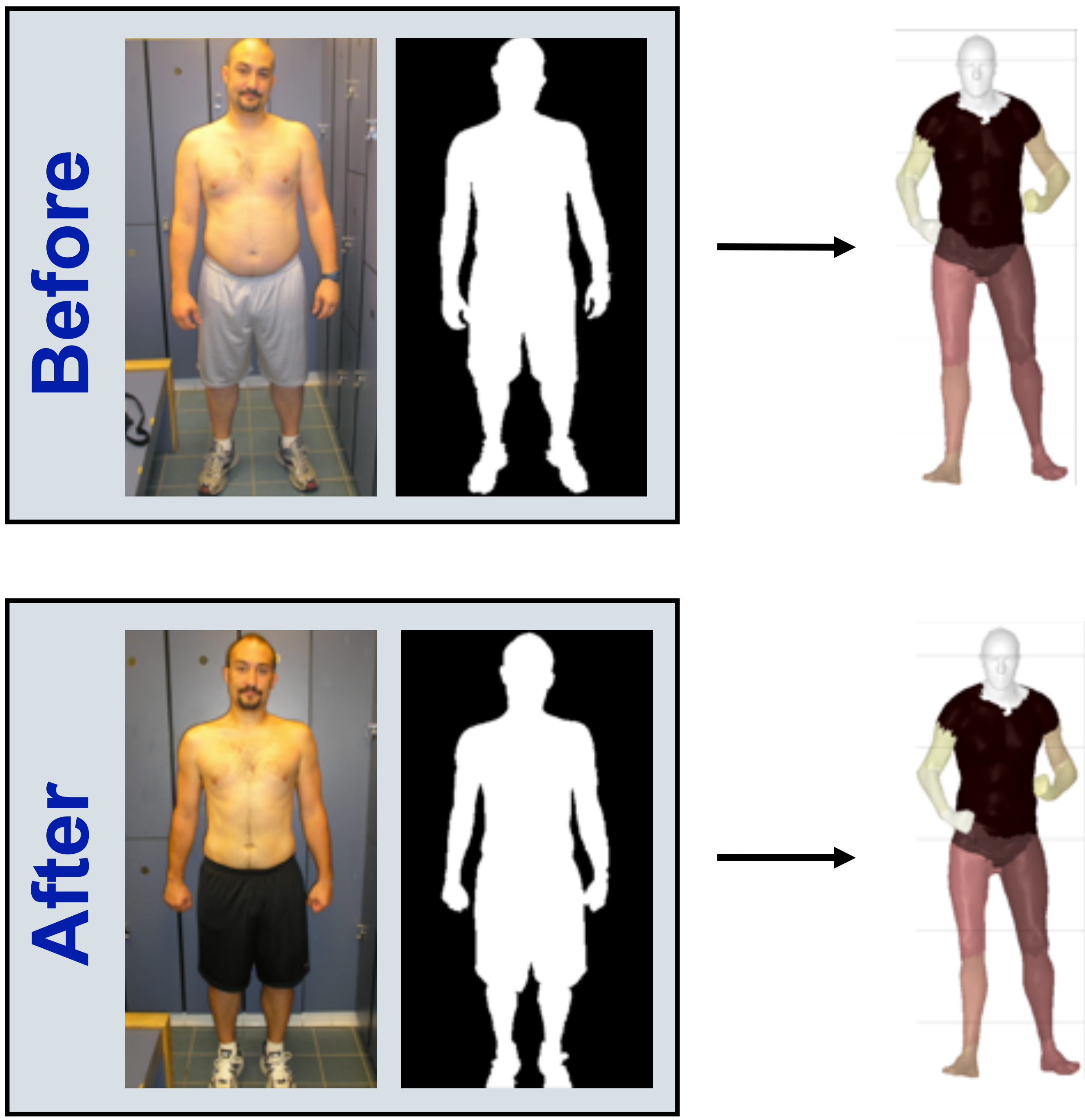
Can we estimate weight loss discriminatively from monocular images?



Proof of concept results

[Sigal et al., NIPS'2007]

Can we estimate weight loss discriminatively from monocular images?

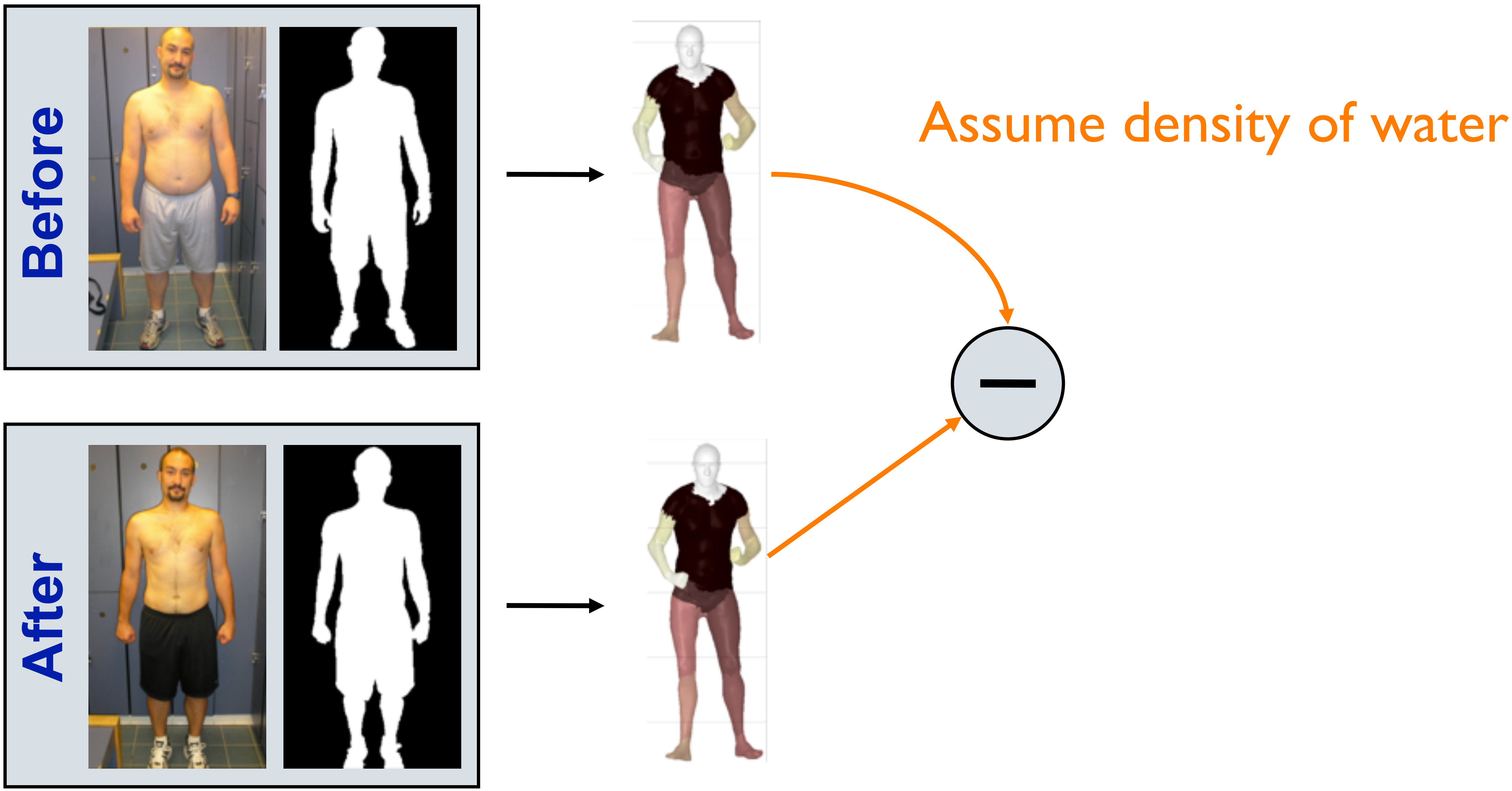


Assume density of water

Proof of concept results

[Sigal et al., NIPS‘2007]

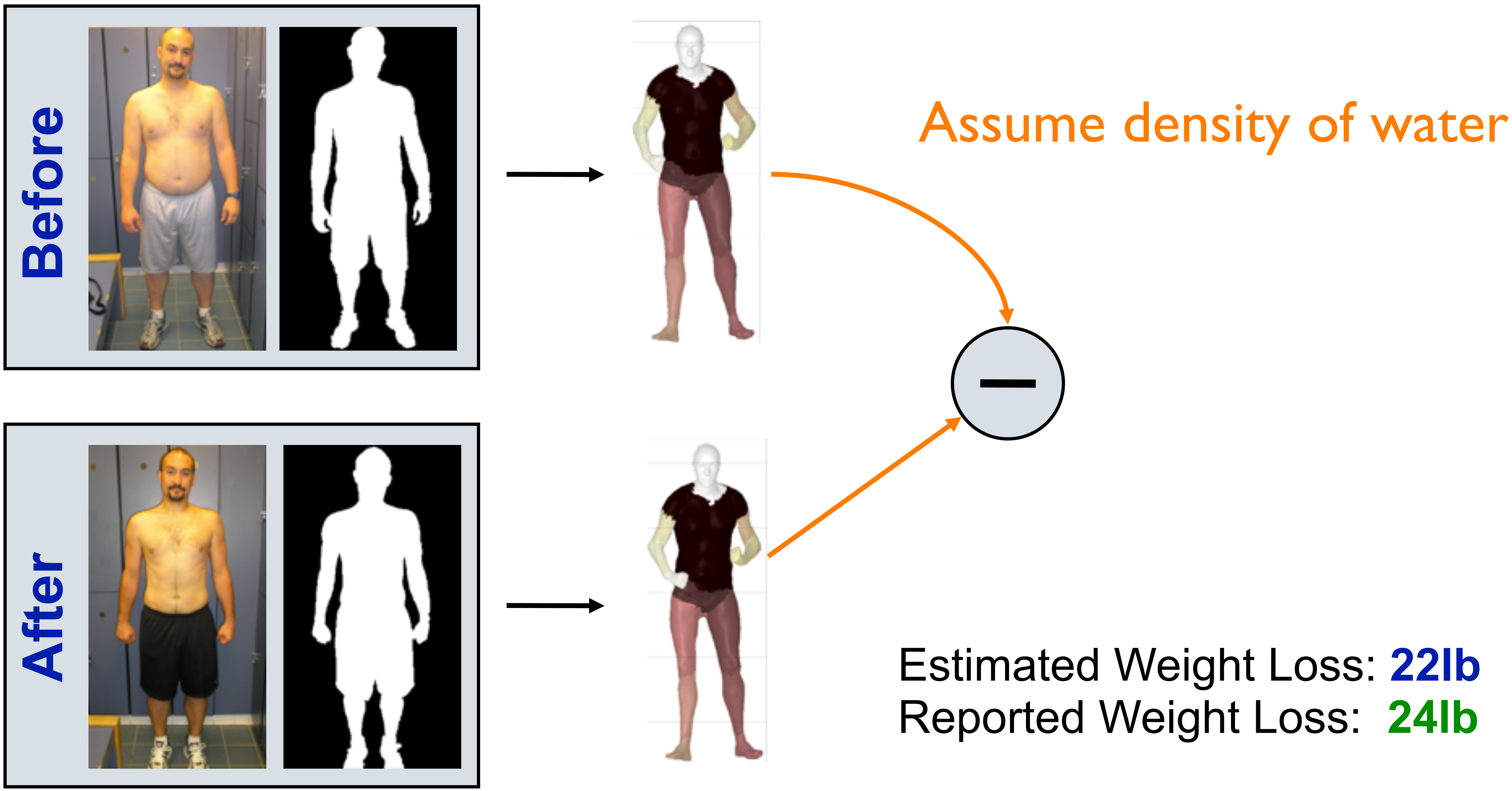
Can we estimate weight loss discriminatively from monocular images?



Proof of concept results

[Sigal et al., NIPS‘2007]

Can we estimate weight loss discriminatively from monocular images?



Proof of concept results

[Sigal et al., NIPS'2007]

Can we estimate weight loss discriminatively from monocular images?

Assume density of water

Estimated Weight Loss: **32lb**
Reported Weight Loss: **64lb**

Silhouettes is not enough

[Guan et al., ICCV, 2009]

Silhouettes are ambiguous

Silhouettes is not enough

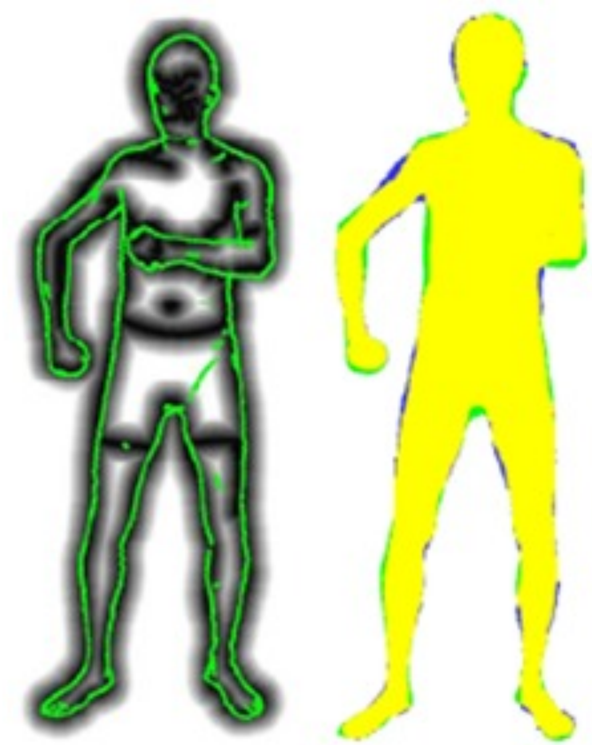
[Guan et al., ICCV, 2009]

Silhouettes are ambiguous

Silhouettes is not enough

[Guan et al., ICCV, 2009]

Silhouettes are ambiguous

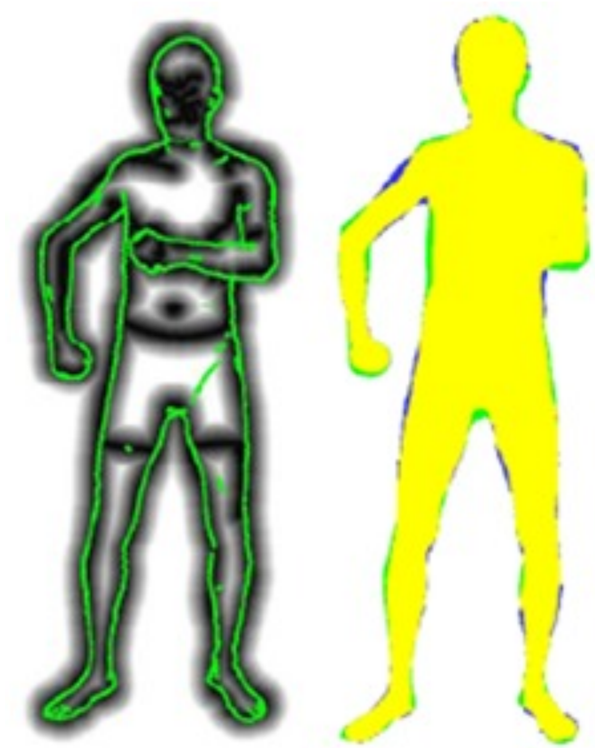


Adding edges

Silhouettes is not enough

[Guan et al., ICCV, 2009]

Silhouettes are ambiguous

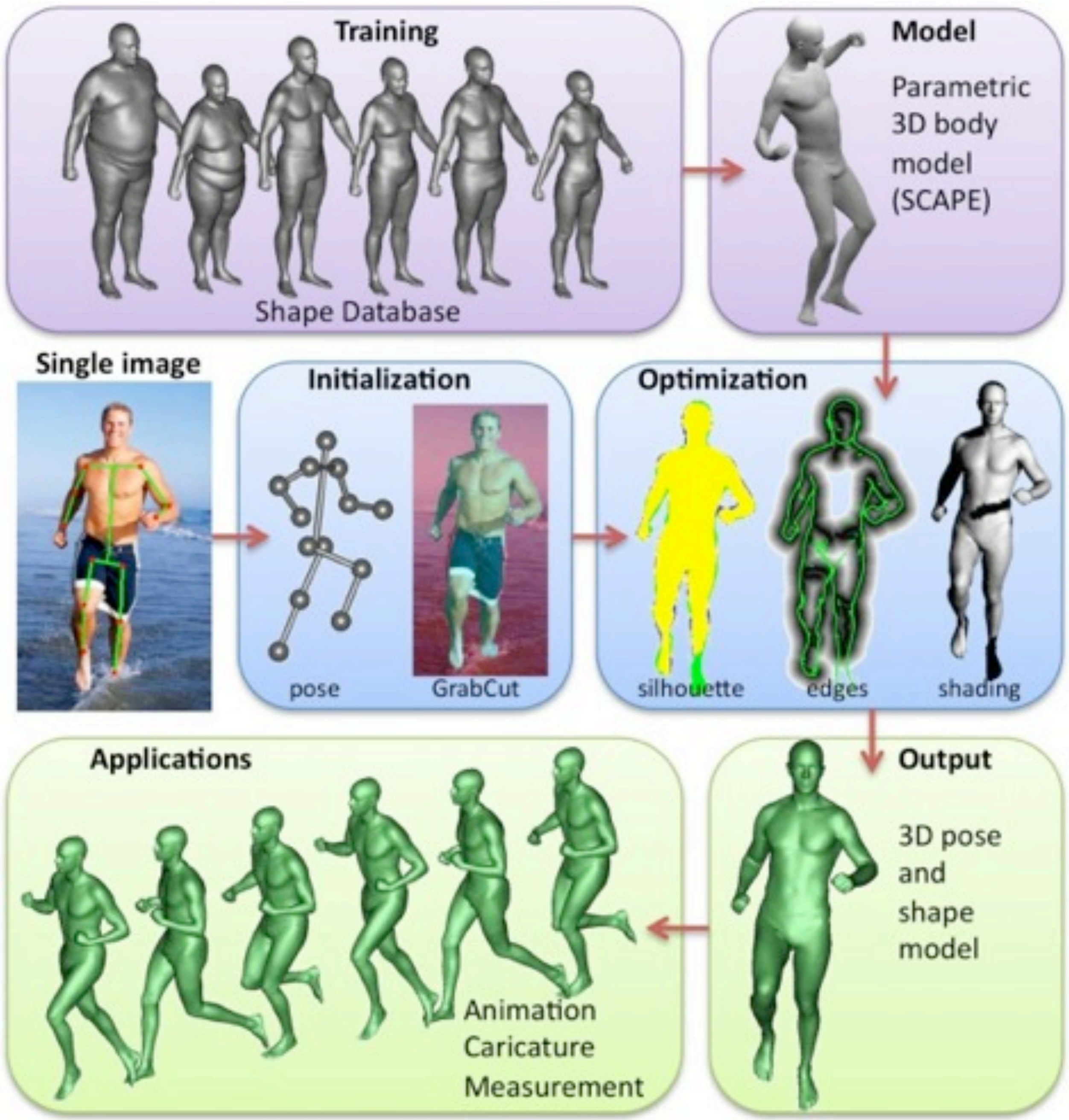


Adding edges

They also add shading cues

Estimating Pose and Shape from Image

[Guan et al., ICCV, 2009]



Fun Results

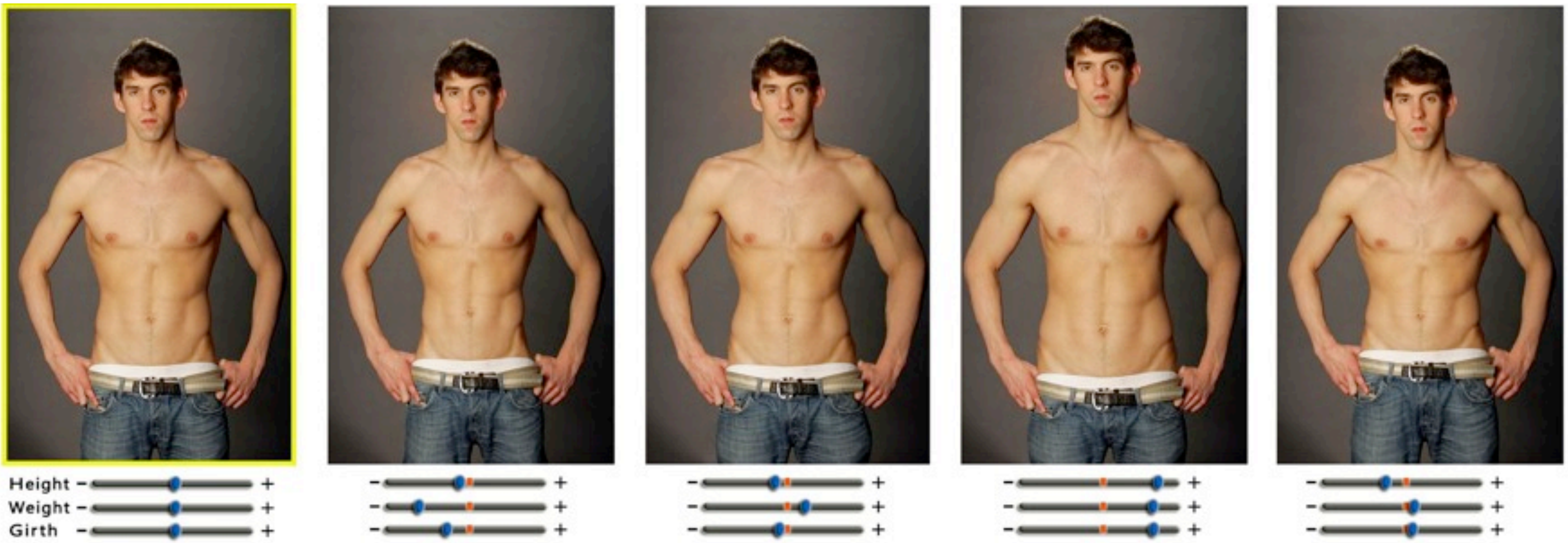
[Guan et al., ICCV, 2009]

Internet Images:

Paintings:

Parametric Body Reshaping

[Zhou et al., ACM TOG, 2010]



Parametric Body Reshaping

[Zhou et al., ACM TOG, 2010]