Contour People:

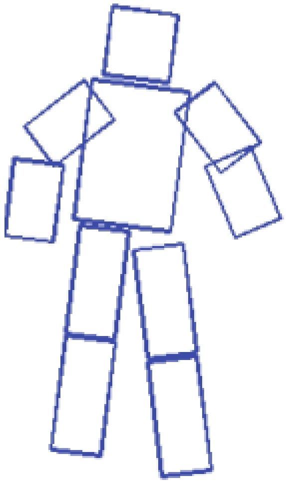
A Parameterized Model of 2D Articulated Human Shape

O. Freifeld S. Zuffi A. Weiss M.J. Black

Brown University

CVPR 2010

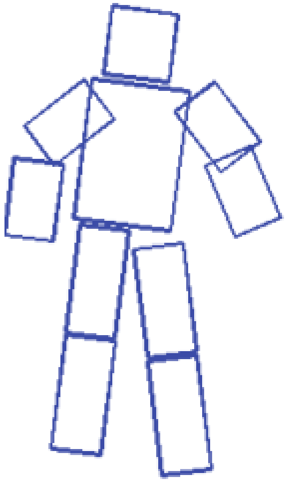
2D generative models of humans



- In wide use in Computer Vision
  Pose estimation; tracking; gesture analysis
- Go back a long way
  [Fischler & Elschlager 73, Hinton 76, Hogg 76, Ju et al. 96]
- Computationally efficient
  Pictorial Structure (PS) and Belief Propagation (BP)
  [Felzenszwalb & Huttenlocher, IJCV ’05]
  [Andriluka et al. CVPR ‘09]

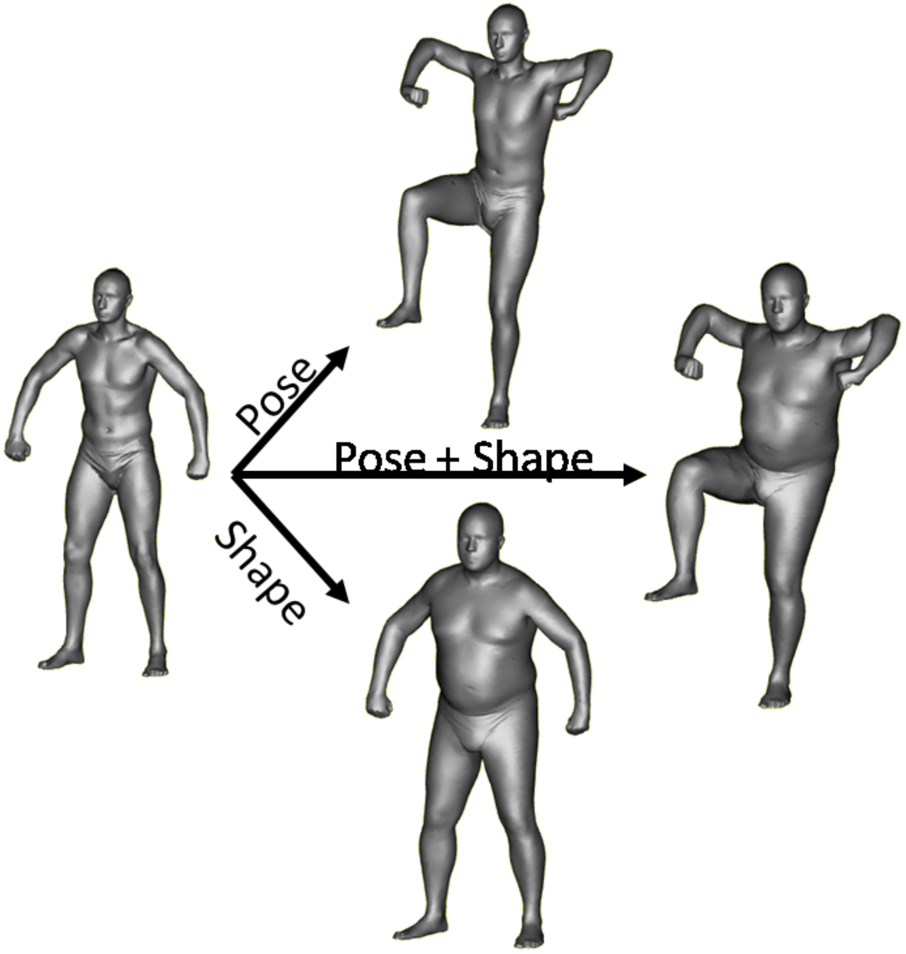
Problem

- Lack of realism (no detailed shape)
- Body shape estimation has many applications
  Gaming, clothing industry, security
- A good model of shape can improve pose estimation



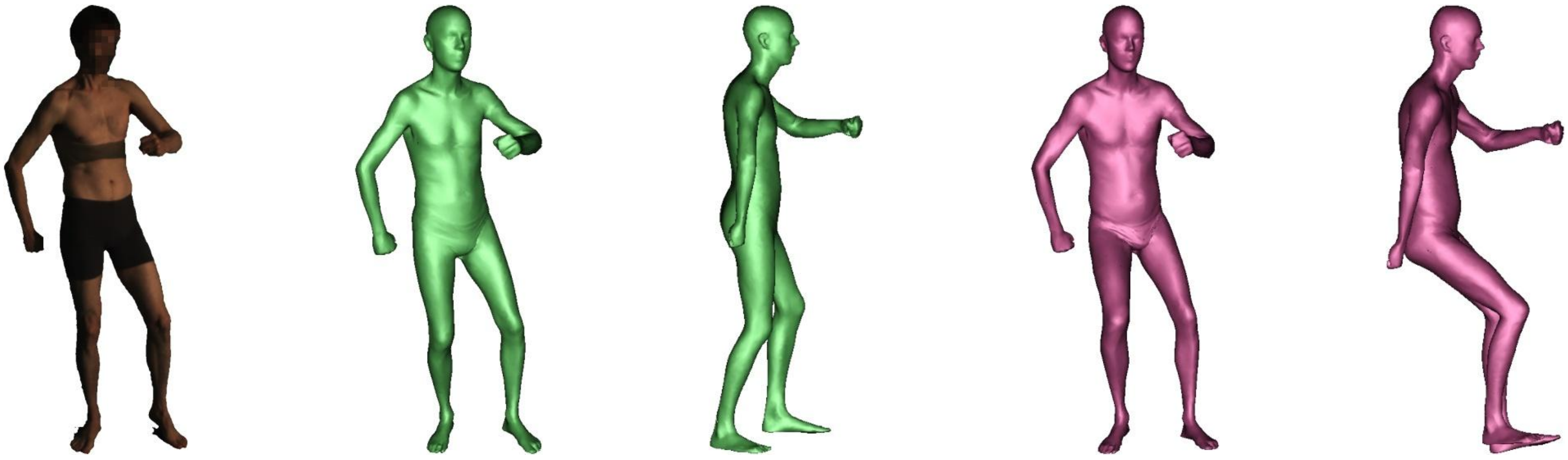
Possible solution: SCAPE

- A 3D graphics model
  [Anguelov et al. Siggraph '05]
- Used in computer vision for shape and pose estimation from multiple calibrated cameras
  [Balan et al. '07]



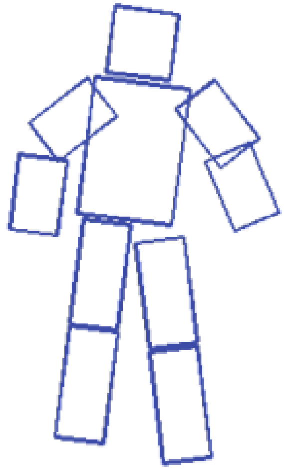
Problem

- Inference is computationally expensive
- Assumes calibrated cameras
- Ambiguous in a single view



[Guan et al. ICCV '09]

Goal: The best of both

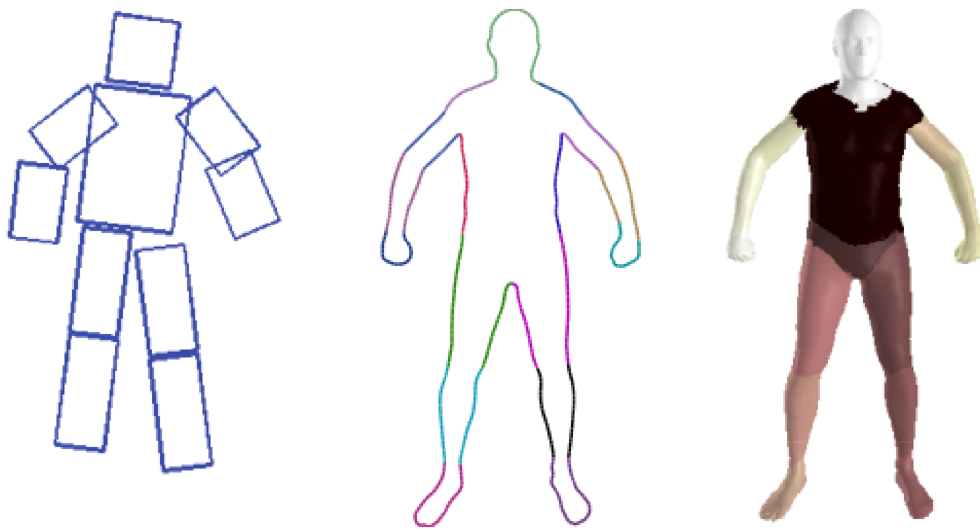


2D 3D

parameters: 12-40 60-100

Goal: The best of both

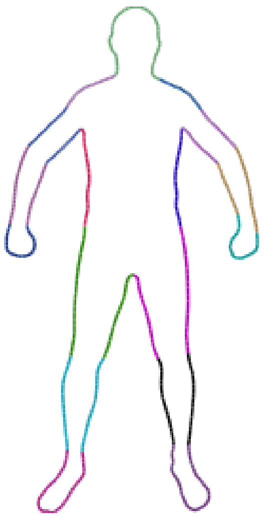
- Learn an articulated 2D model of detailed shape that is lower-dimensional than SCAPE



2D		3D

# parameters:	12-40	25-60	60-100

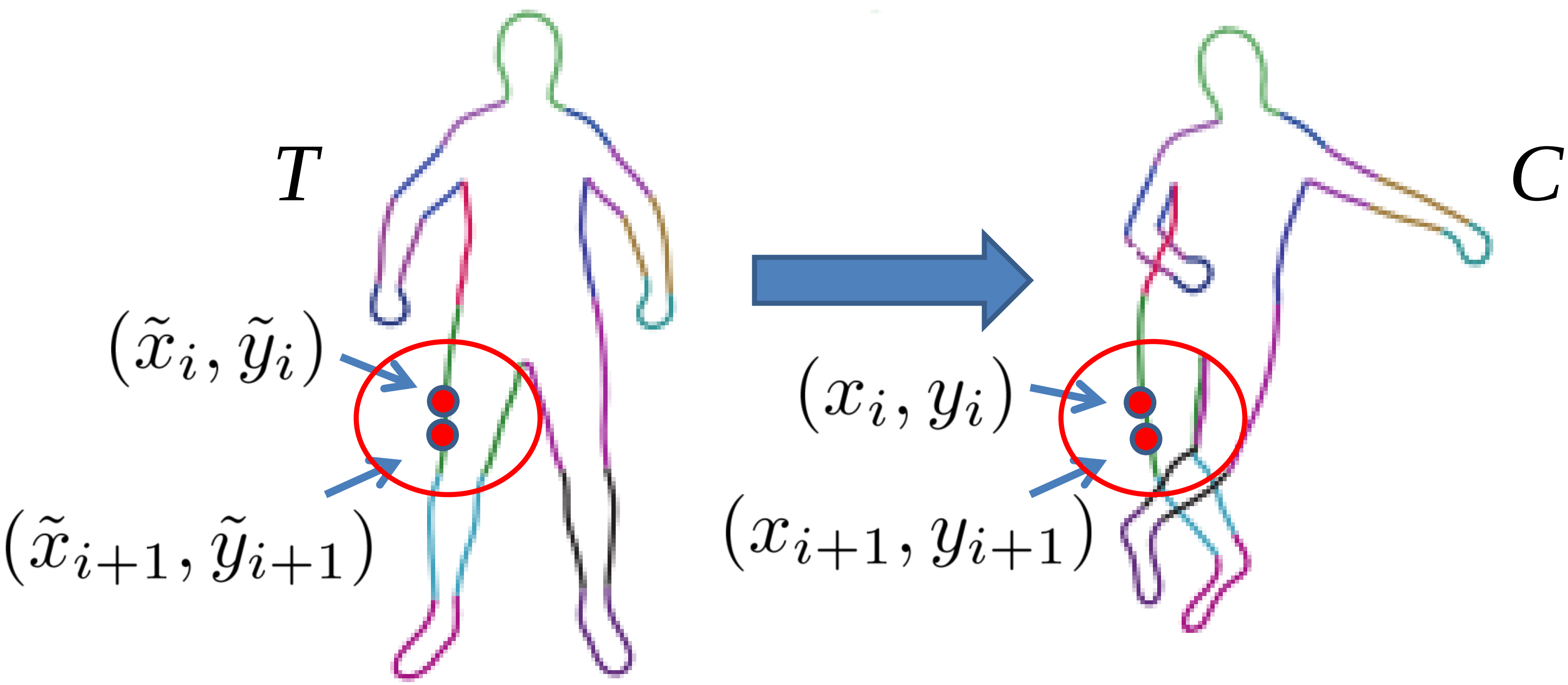
The Contour Person (CP) model



- A part-based articulated deformable template model
- It factorizes shape, pose and camera variations
- A 2D model that captures the detailed shape of the human body
- The model is learned from 3D SCAPE

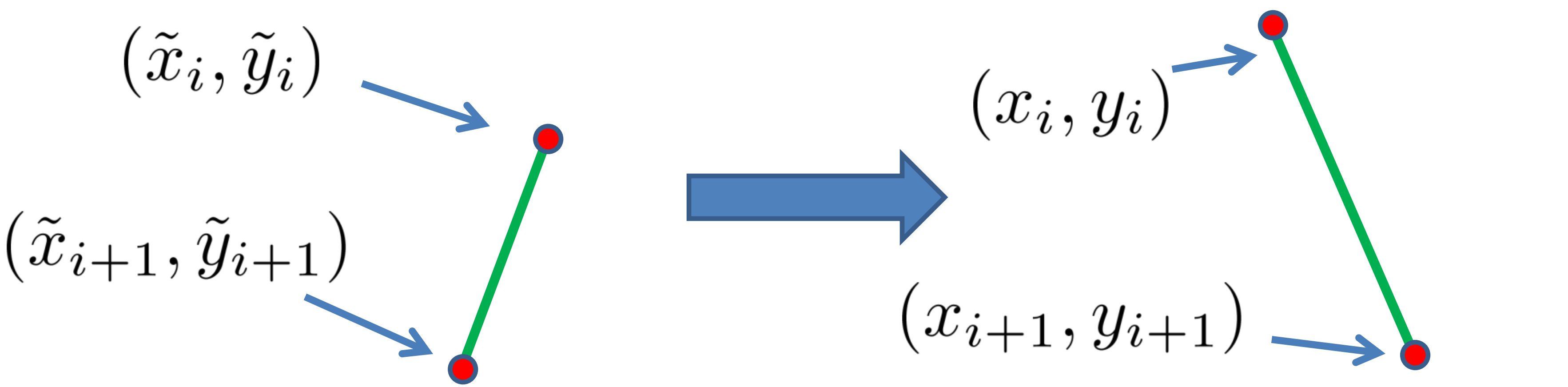
Deformable template

$$T = \begin{pmatrix} \tilde{x}_1 & \tilde{y}_1 & \ldots & \tilde{x}_n & \tilde{x}_n \end{pmatrix}^T \qquad C = \begin{pmatrix} x_1 & y_1 & \ldots & x_n & y_n \end{pmatrix}^T$$



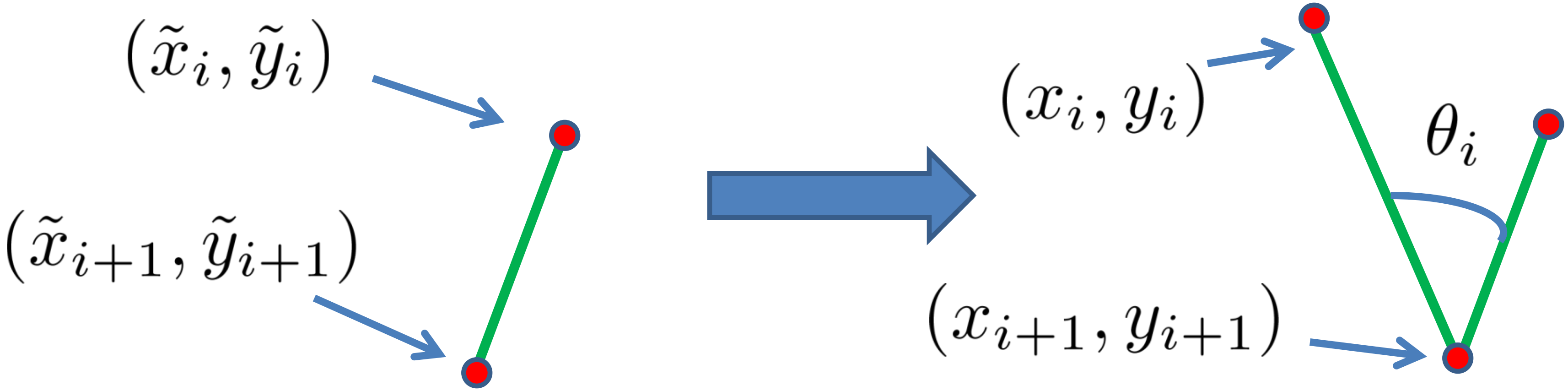
Deformation of a line segment

$$T = \begin{pmatrix} \tilde{x}_1 & \tilde{y}_1 & \dots & \tilde{x}_n & \tilde{x}_n \end{pmatrix}^T \quad C = \begin{pmatrix} x_1 & y_1 & \dots & x_n & y_n \end{pmatrix}^T$$



Deformation of a line segment

$$T = \begin{pmatrix} \tilde{x}_1 & \tilde{y}_1 & \ldots & \tilde{x}_n & \tilde{x}_n \end{pmatrix}^T \qquad C = \begin{pmatrix} x_1 & y_1 & \ldots & x_n & y_n \end{pmatrix}^T$$



S_i is the ratio of the lengths

$$D_i^{2\times 2} = S_i \begin{pmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{pmatrix}$$

Template contour $T = \begin{pmatrix} \tilde{x}_1 & \tilde{y}_1 & \dots & \tilde{x}_n & \tilde{x}_n \end{pmatrix}^T$

New contour $C = \begin{pmatrix} x_1 & y_1 & \dots & x_n & y_n \end{pmatrix}^T$

Connectivity matrix

$$E = \begin{pmatrix} -1 & 0 & +1 & 0 & \dots & 0 & 0 & 0 \\ 0 & -1 & 0 & +1 & \dots & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -1 & 0 & +1 \\ +1 & 0 & 0 & 0 & \cdots & 0 & -1 & 0 \\ 0 & +1 & 0 & 0 & \cdots & 0 & 0 & -1 \end{pmatrix}$$

Template contour $\quad T = \begin{pmatrix} \tilde{x}_1 & \tilde{y}_1 & \dots & \tilde{x}_n & \tilde{x}_n \end{pmatrix}^T$

New contour $\quad C = \begin{pmatrix} x_1 & y_1 & \dots & x_n & y_n \end{pmatrix}^T$

Connectivity matrix

$$E = \begin{pmatrix} -1 & 0 & +1 & 0 & \dots & 0 & 0 & 0 \\ 0 & -1 & 0 & +1 & \dots & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -1 & 0 & +1 \\ +1 & 0 & 0 & 0 & \cdots & 0 & -1 & 0 \\ 0 & +1 & 0 & 0 & \cdots & 0 & 0 & -1 \end{pmatrix}$$

Template line segments

$$E\,T = \begin{pmatrix} \tilde{x}_2 - \tilde{x}_1 & \tilde{y}_2 - \tilde{y}_1 & \dots & \tilde{x}_1 - \tilde{x}_n & \tilde{y}_1 - \tilde{x}_n \end{pmatrix}^T$$

New line segments

$$E\,C = \begin{pmatrix} x_2 - x_1 & y_2 - y_1 & \dots & x_1 - x_n & y_1 - y_n \end{pmatrix}^T$$

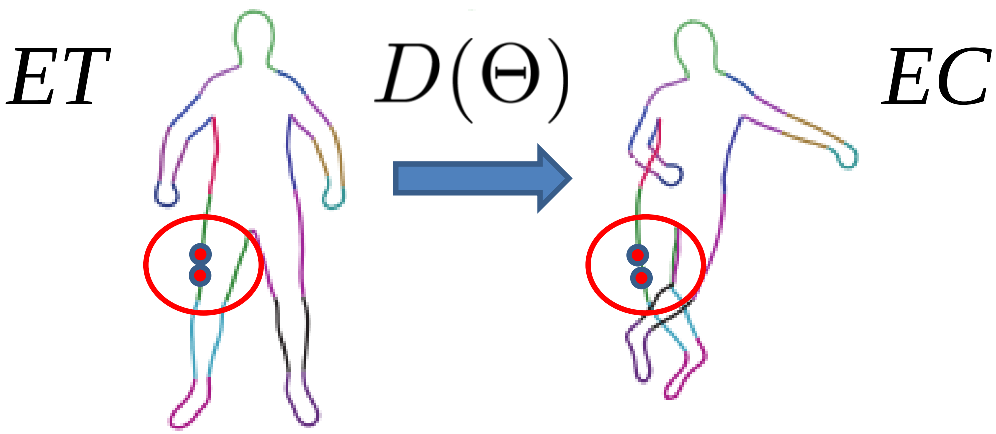
Deformation of T

$$D(\Theta) = \left(\begin{array}{cccc} D_1^{2\times 2} & 0 & \dots & 0 \\ 0 & D_2^{2\times 2} & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & D_n^{2\times 2} \end{array} \right)$$

$$D_i^{2\times 2} = S_i \left(\begin{array}{cc} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{array} \right)$$

Deformation of T

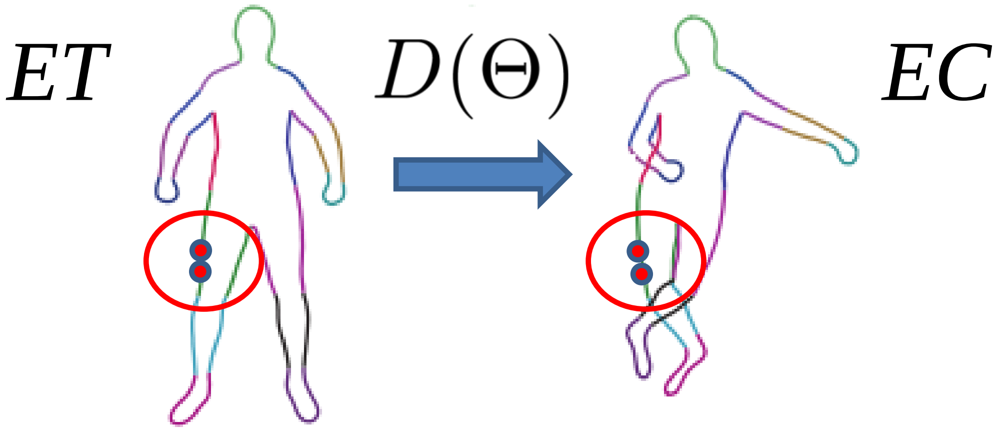
$$D(\Theta) = \begin{pmatrix} D_1^{2\times 2} & 0 & \dots & 0 \\ 0 & D_2^{2\times 2} & \dots & 0 \\ & & \ddots & \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & D_n^{2\times 2} \end{pmatrix}$$



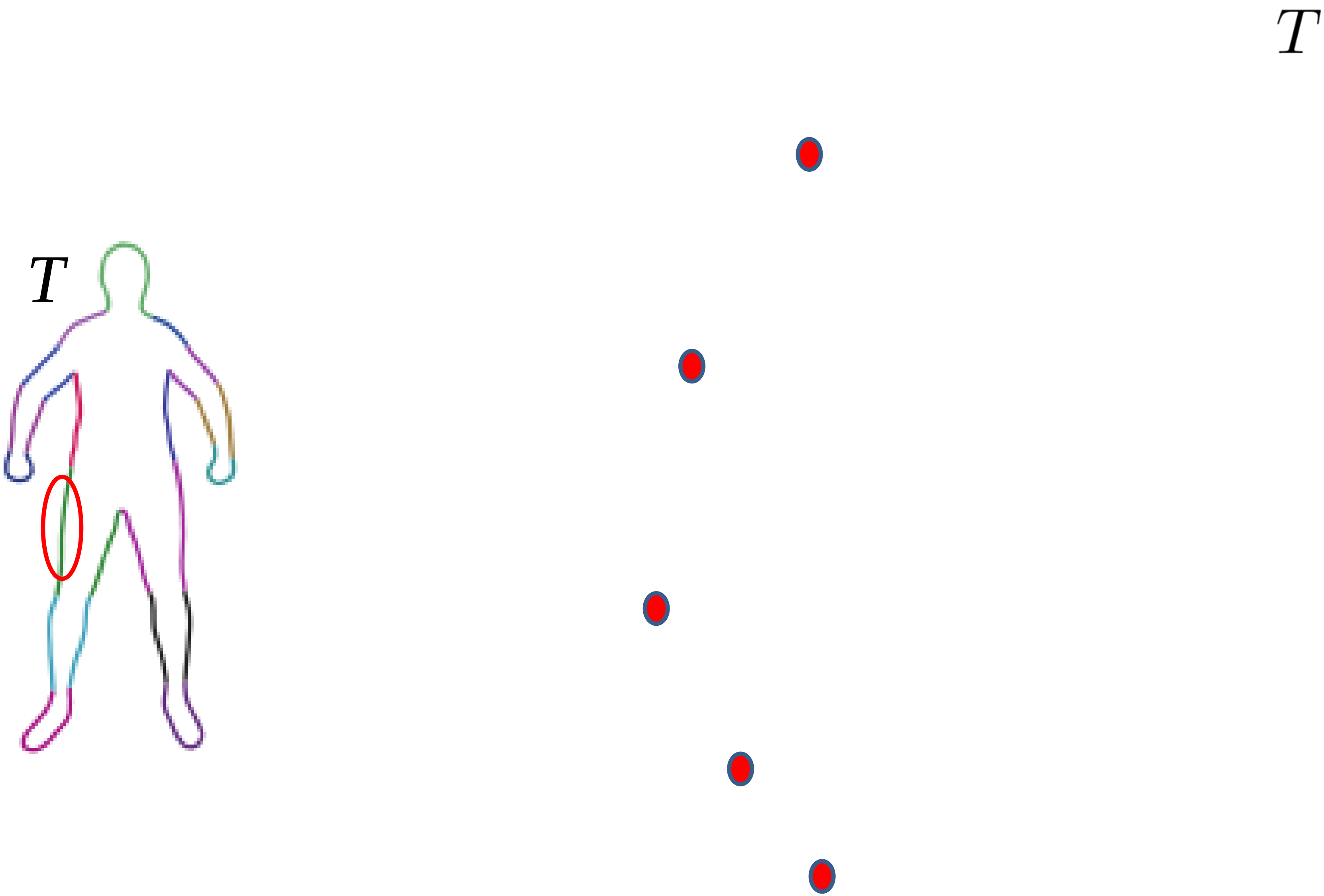
$$E\,C \;=\; D(\Theta)E\,T$$

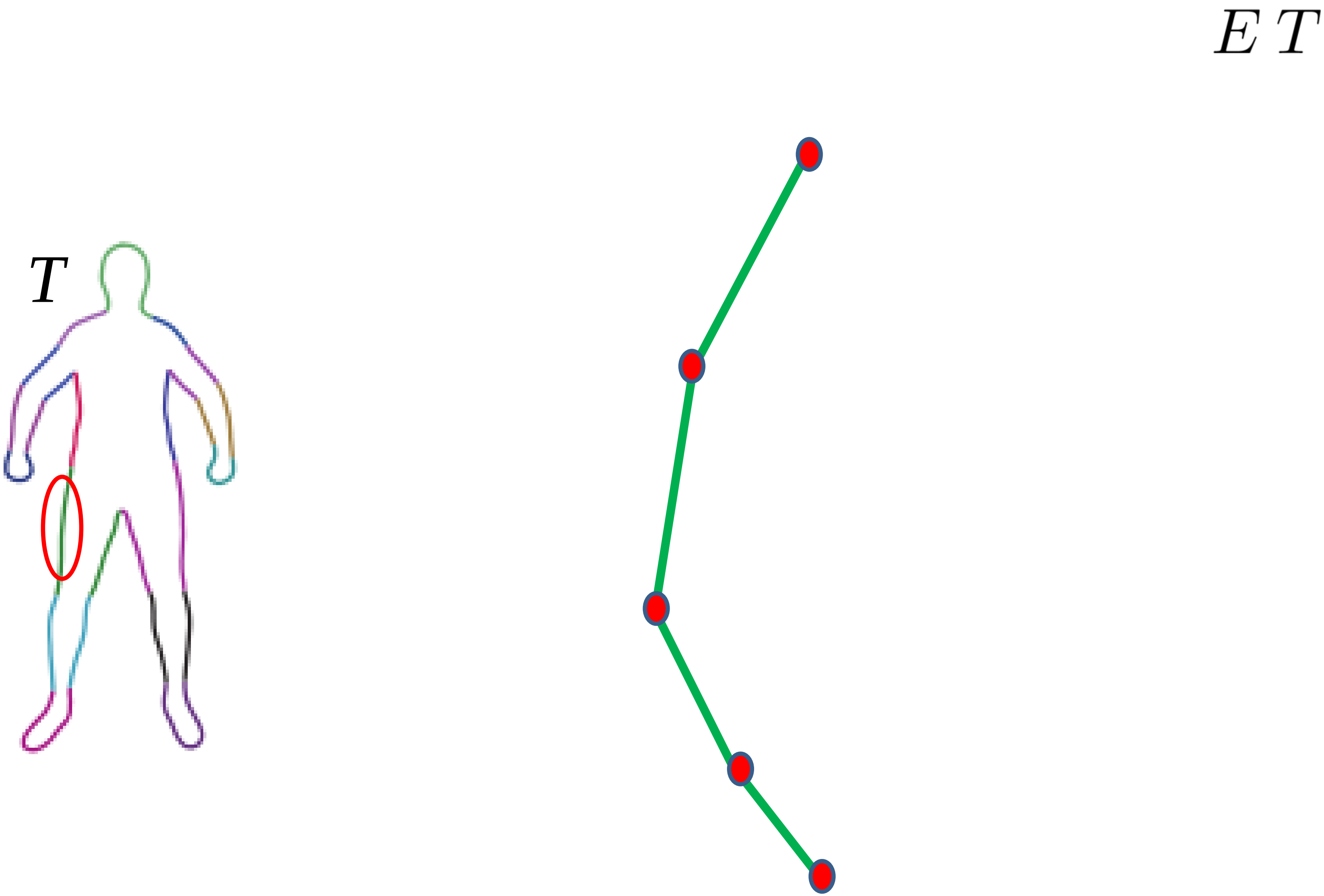
Deformation of T

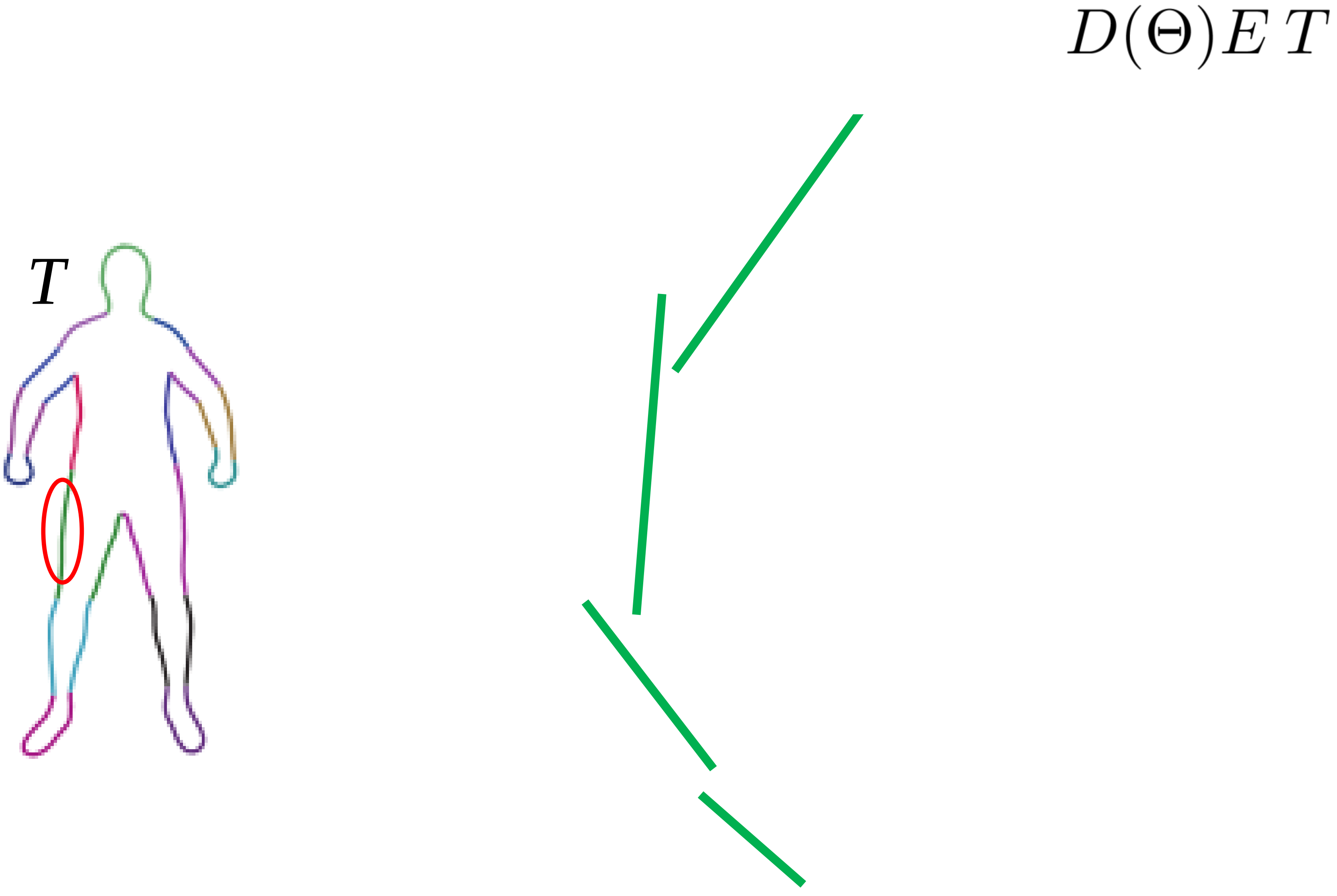
$$D(\Theta) = \begin{pmatrix} D_1^{2\times2} & 0 & \cdots & 0 \\ 0 & D_2^{2\times2} & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & & 0 \\ 0 & 0 & \cdots & D_n^{2\times2} \end{pmatrix}$$



$$E\,C \approx D(\Theta)E\,T$$


T
T


T
E T

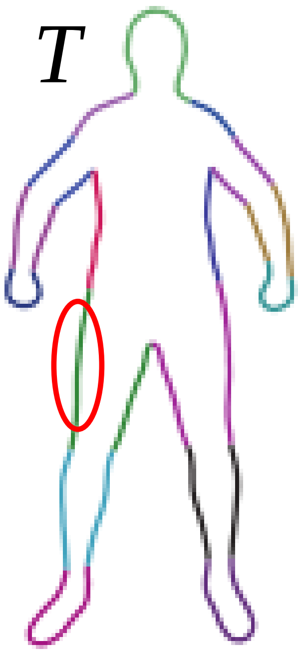


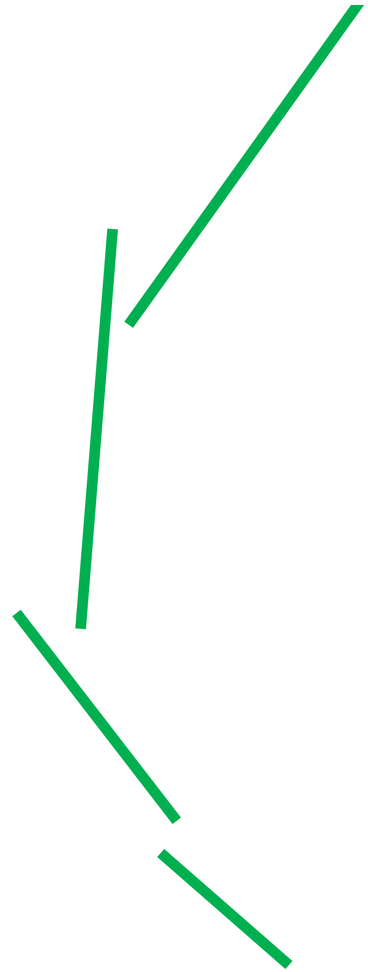
T
$D(\Theta)\,E\,T$

$$D(\Theta) E\, T$$

$$\min_{C} \|D(\Theta) E\, T - E\, C\|^2$$

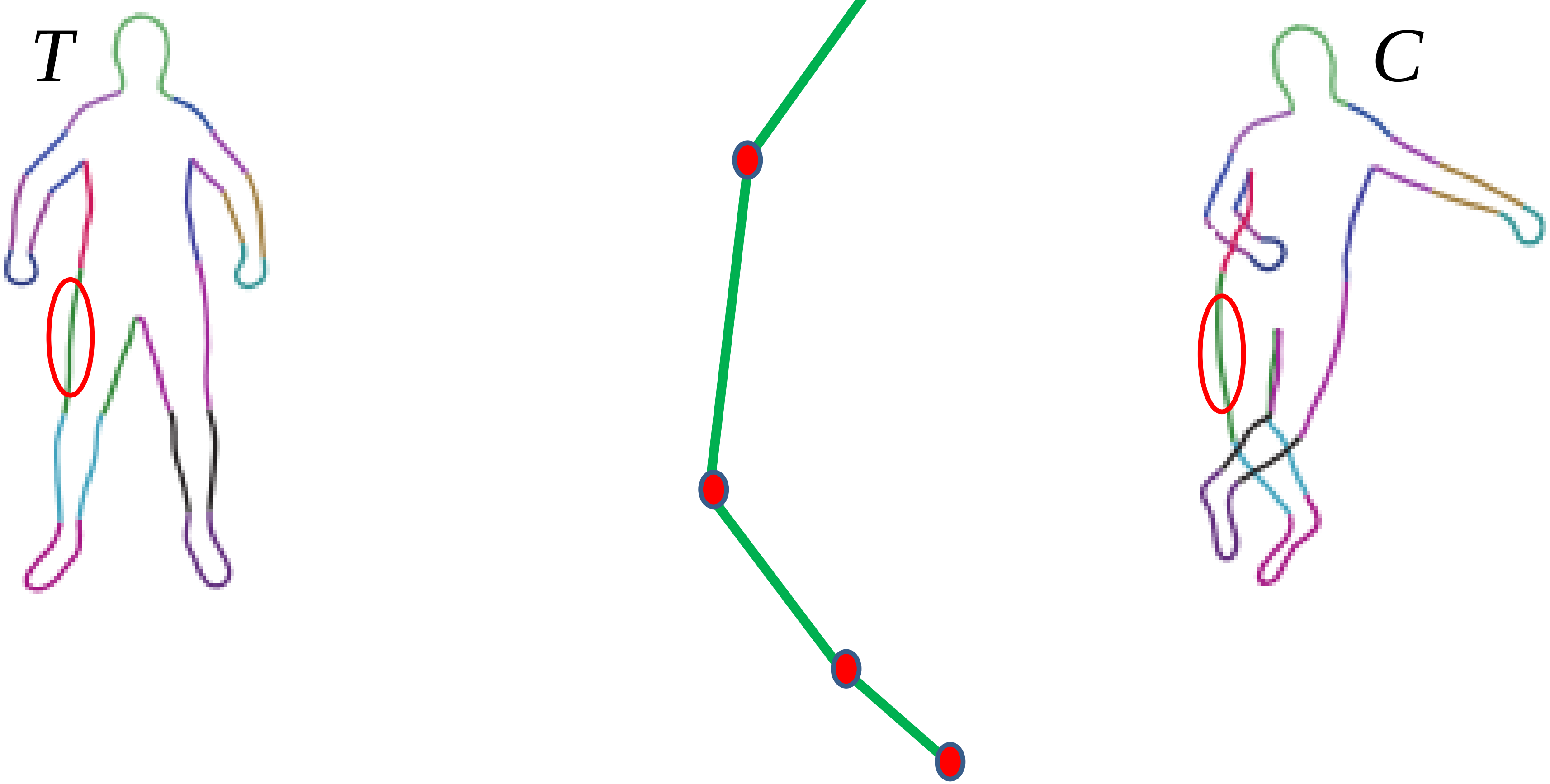
T



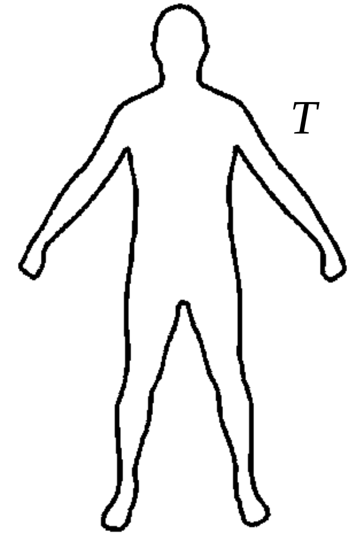


$$C = E^{\dagger} D(\Theta) E\, T$$

$$\min_{C} \left\| D(\Theta) E\, T - E\, C \right\|^2$$



Factorization



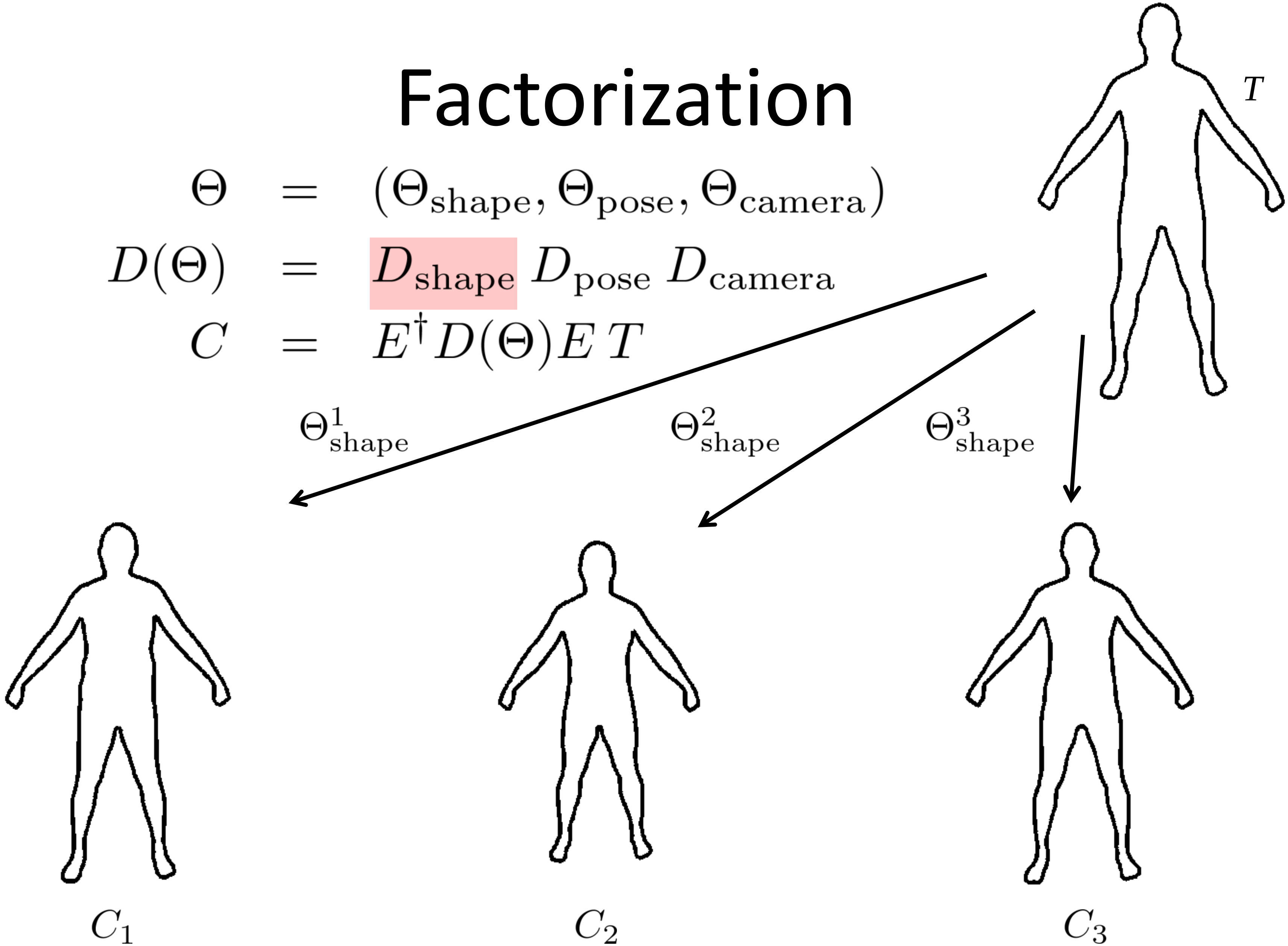
$$
\begin{aligned}
\boldsymbol{\Theta} &= (\boldsymbol{\Theta}_{\text{shape}}, \boldsymbol{\Theta}_{\text{pose}}, \boldsymbol{\Theta}_{\text{camera}}) \\
D(\boldsymbol{\Theta}) &= D_{\text{shape}}\, D_{\text{pose}}\, D_{\text{camera}} \\
C &= E^{\dagger} D(\boldsymbol{\Theta}) E\, T
\end{aligned}
$$

Factorization

$$\Theta = (\Theta_{\text{shape}}, \Theta_{\text{pose}}, \Theta_{\text{camera}})$$

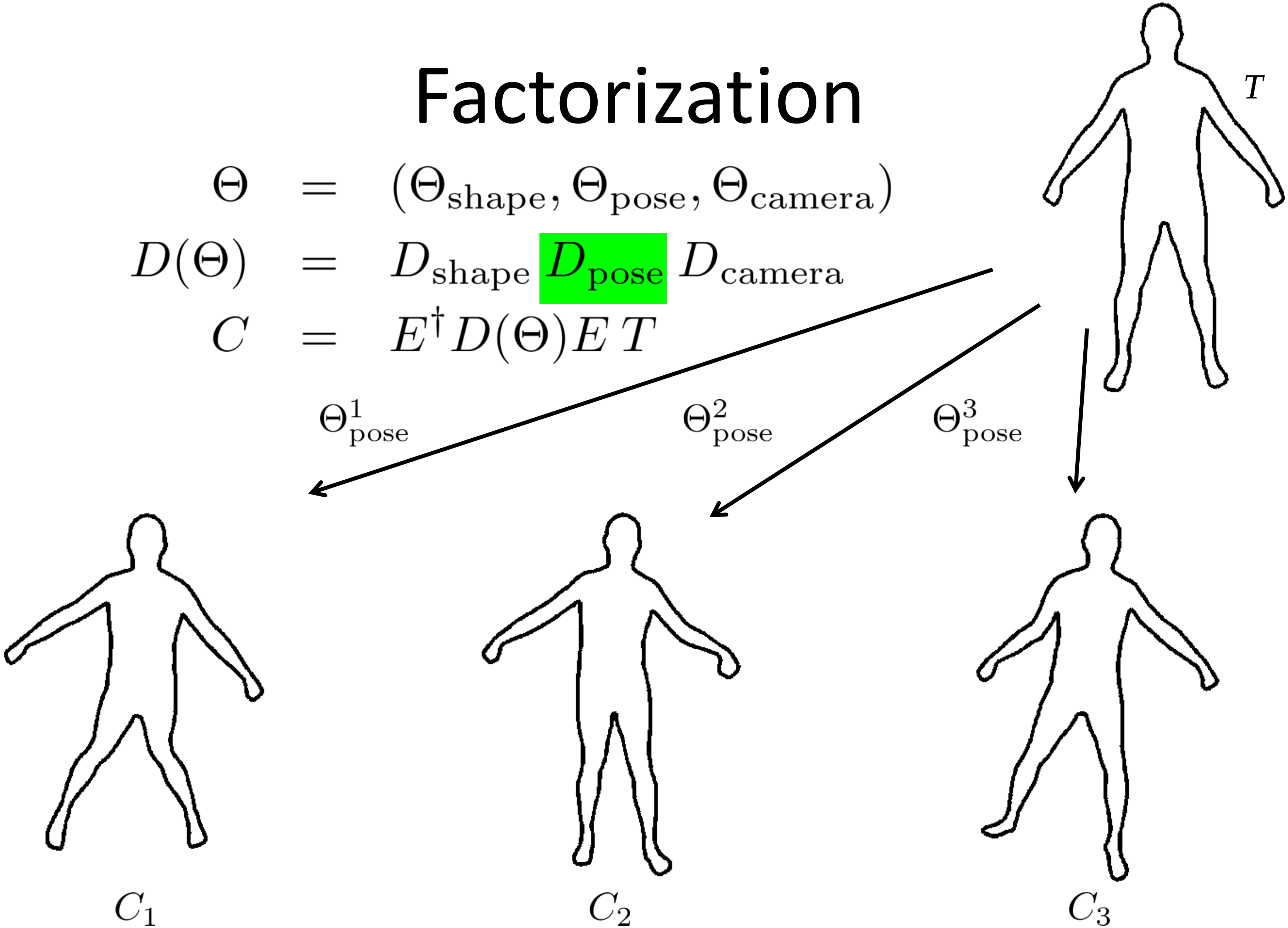
$$D(\Theta) = D_{\text{shape}}\, D_{\text{pose}}\, D_{\text{camera}}$$

$$C = E^{\dagger} D(\Theta) E\, T$$



Factorization

$$\begin{aligned}
\Theta &= (\Theta_{\text{shape}}, \Theta_{\text{pose}}, \Theta_{\text{camera}}) \\
D(\Theta) &= D_{\text{shape}}\, D_{\text{pose}}\, D_{\text{camera}} \\
C &= E^{\dagger} D(\Theta) E\, T
\end{aligned}$$

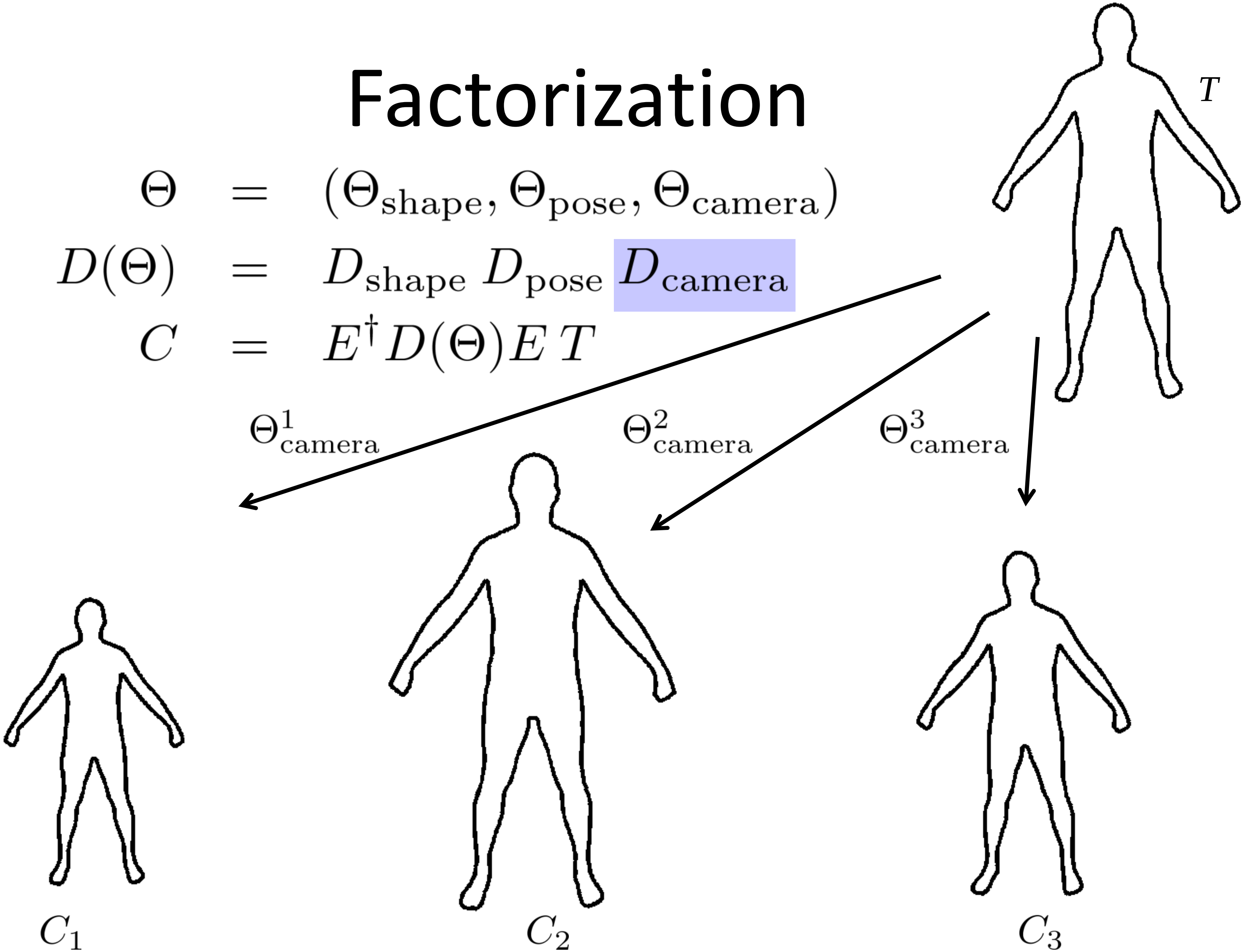


Factorization

$$\Theta = (\Theta_{\text{shape}}, \Theta_{\text{pose}}, \Theta_{\text{camera}})$$

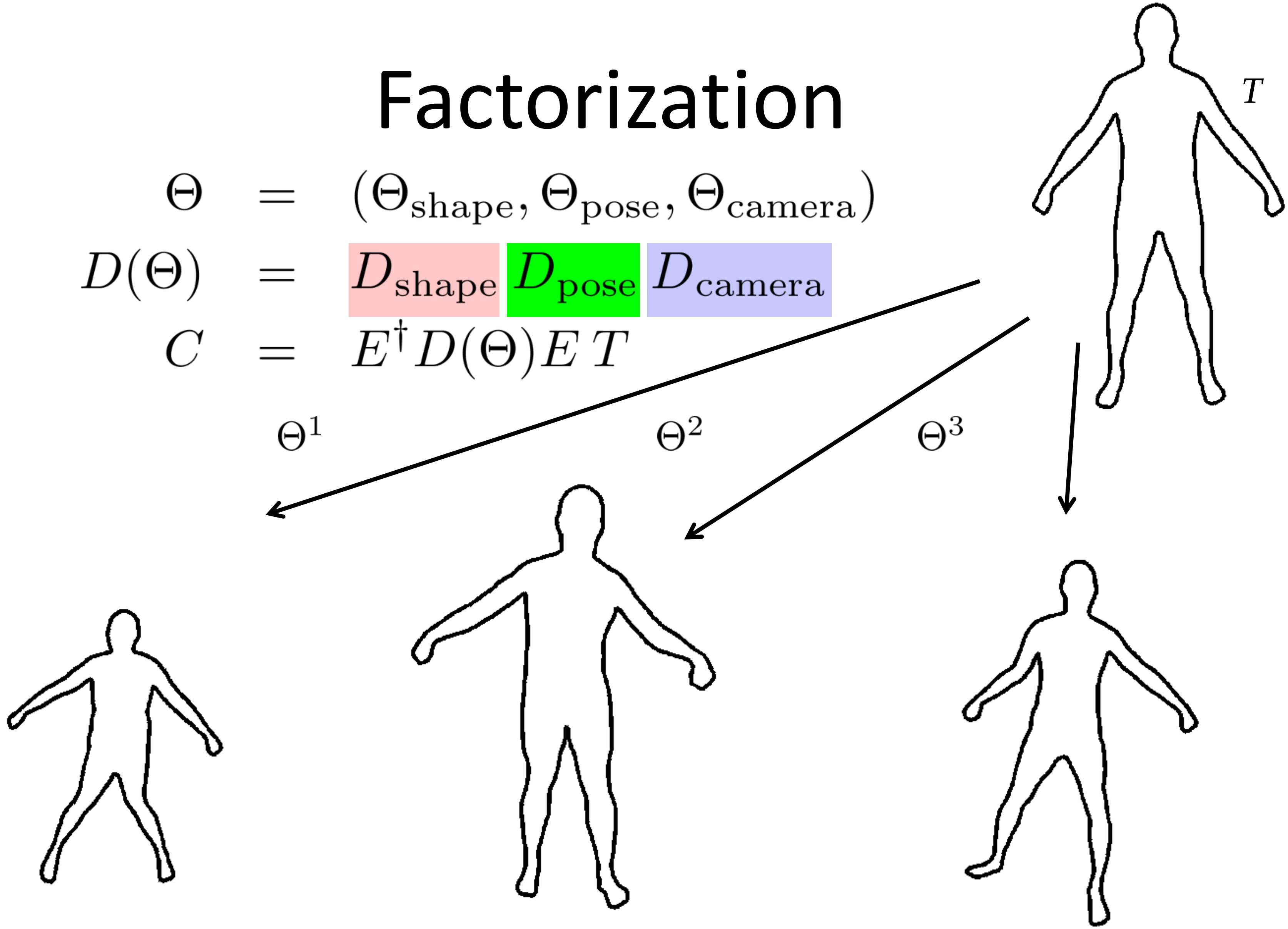
$$D(\Theta) = D_{\text{shape}}\, D_{\text{pose}}\, D_{\text{camera}}$$

$$C = E^{\dagger} D(\Theta) E\, T$$



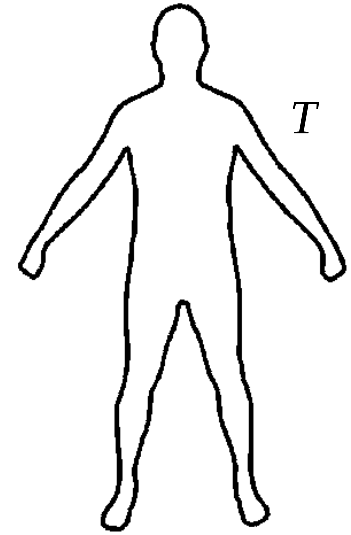
Factorization

$$
\begin{aligned}
\Theta &= (\Theta_{\text{shape}}, \Theta_{\text{pose}}, \Theta_{\text{camera}}) \\
D(\Theta) &= D_{\text{shape}}\, D_{\text{pose}}\, D_{\text{camera}} \\
C &= E^{\dagger} D(\Theta) E\, T
\end{aligned}
$$



Learning

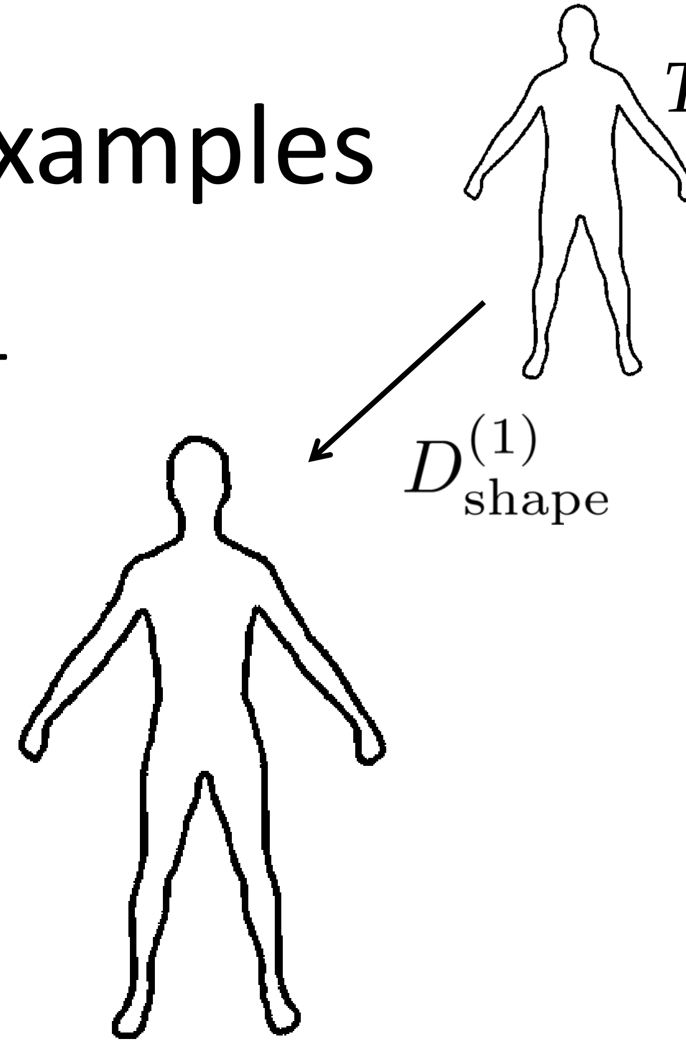
$$\begin{aligned} \Theta &= (\Theta_{\text{shape}}, \Theta_{\text{pose}}, \Theta_{\text{camera}}) \\ D(\Theta) &= D_{\text{shape}} \, D_{\text{pose}} \, D_{\text{camera}} \\ C &= E^{\dagger} D(\Theta) E \, T \end{aligned}$$



- How do we learn this parametric model?
  - Create random shapes and poses from SCAPE
  - Create random cameras
  - Project 3D body to 2D (but keep the body-part segmentation)

Shape training examples

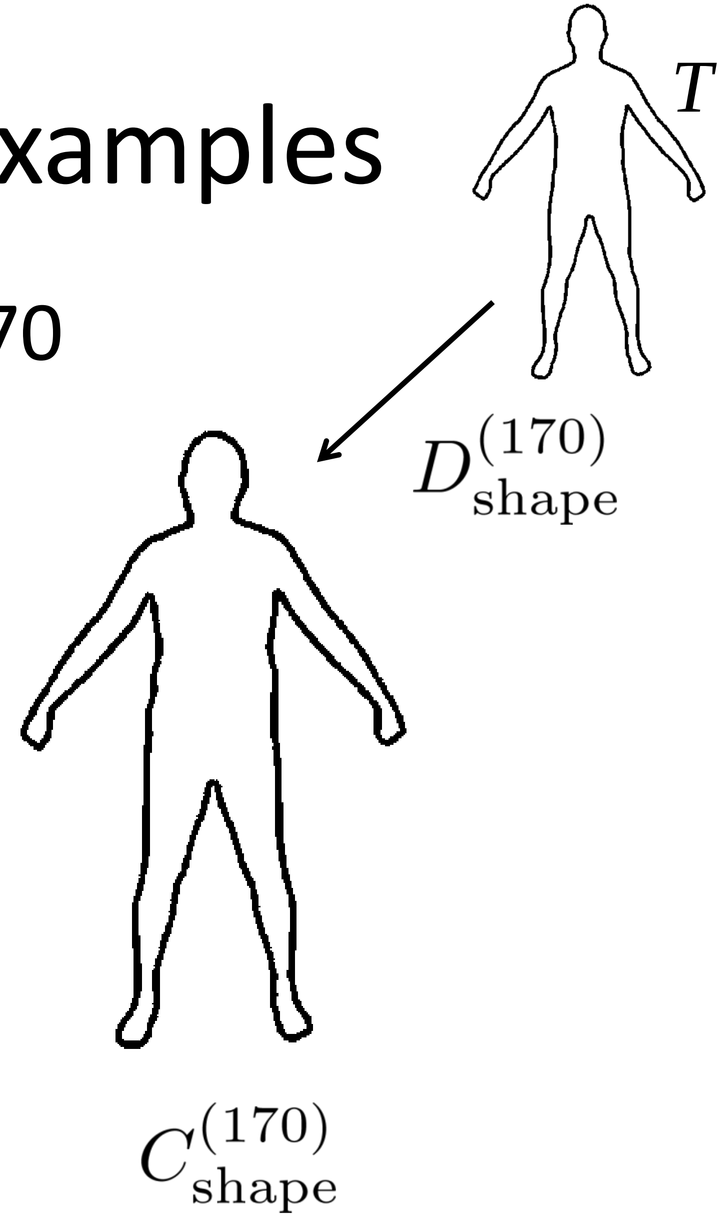
Example #1



$C^{(1)}_{\text{shape}}$

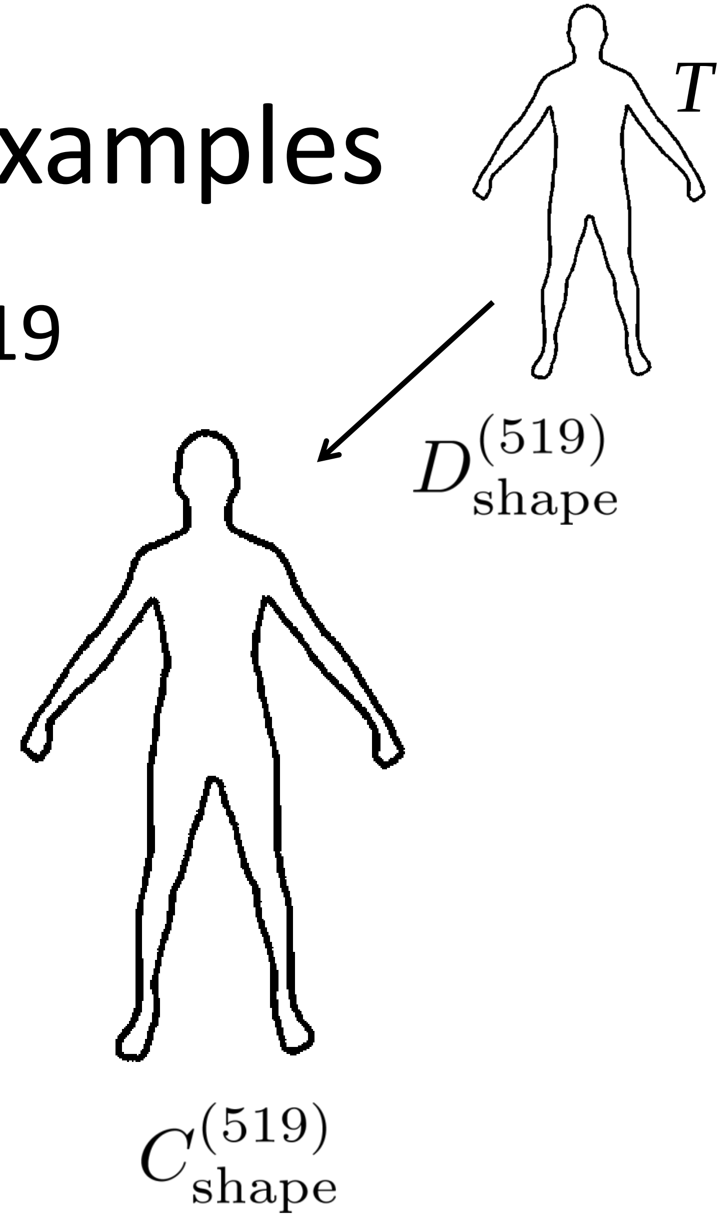
Shape training examples

Example #170



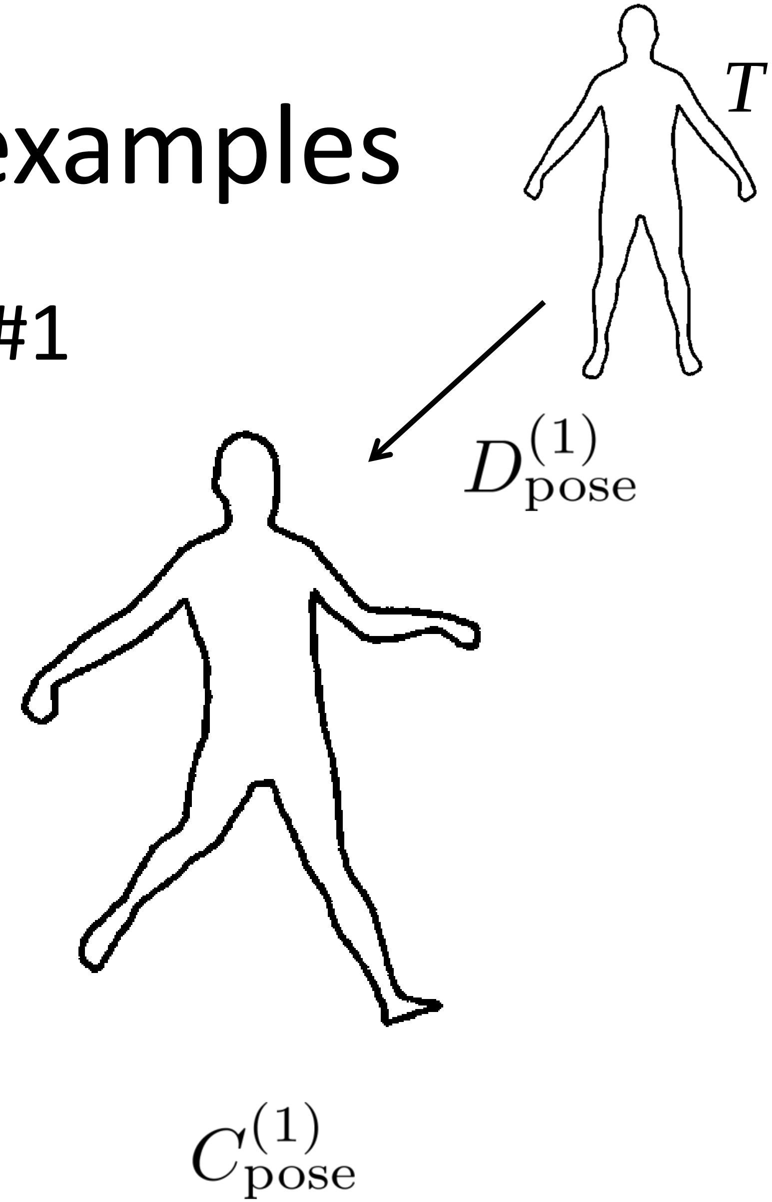
Shape training examples

Example #519



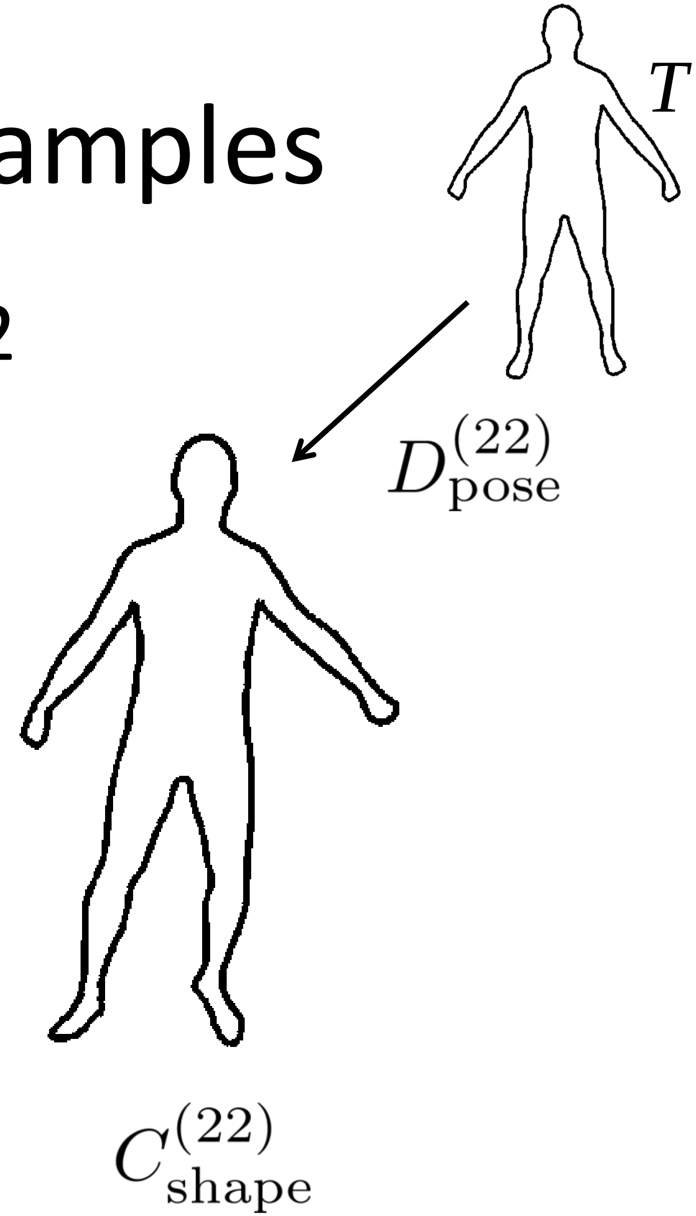
Pose training examples

Example #1



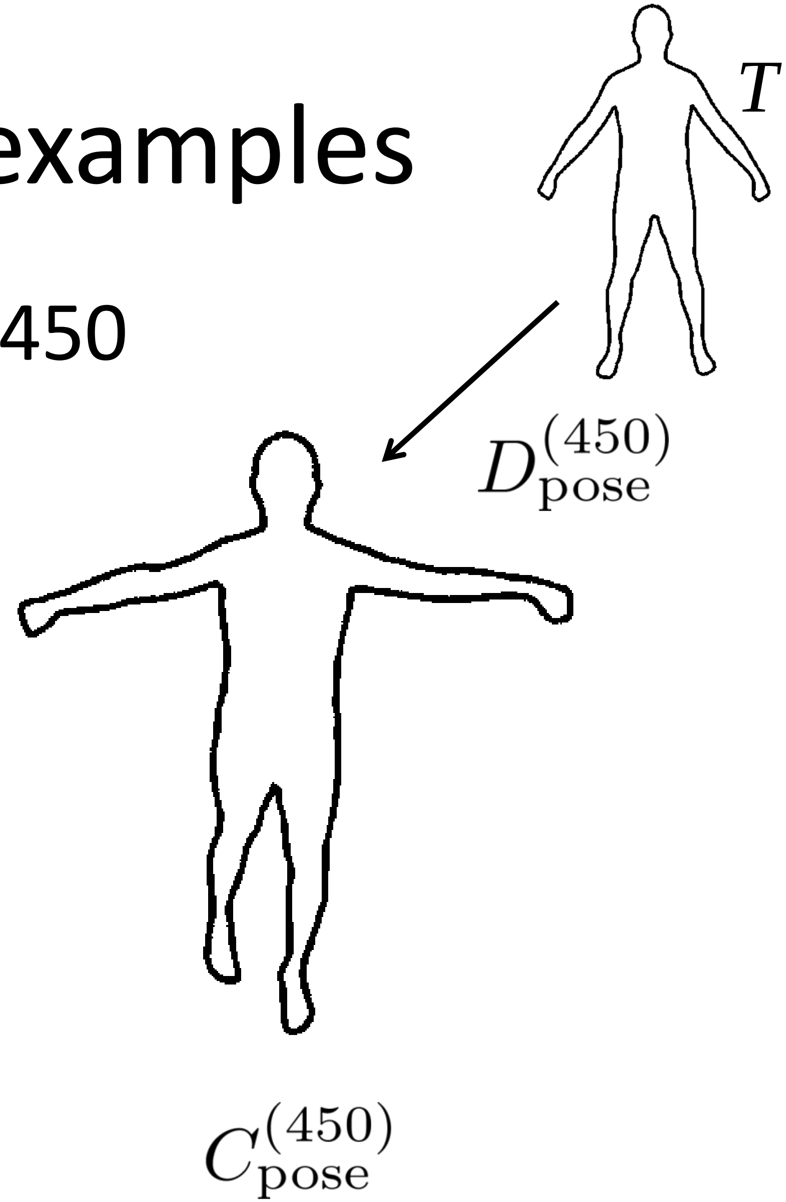
Pose training examples

Example #22



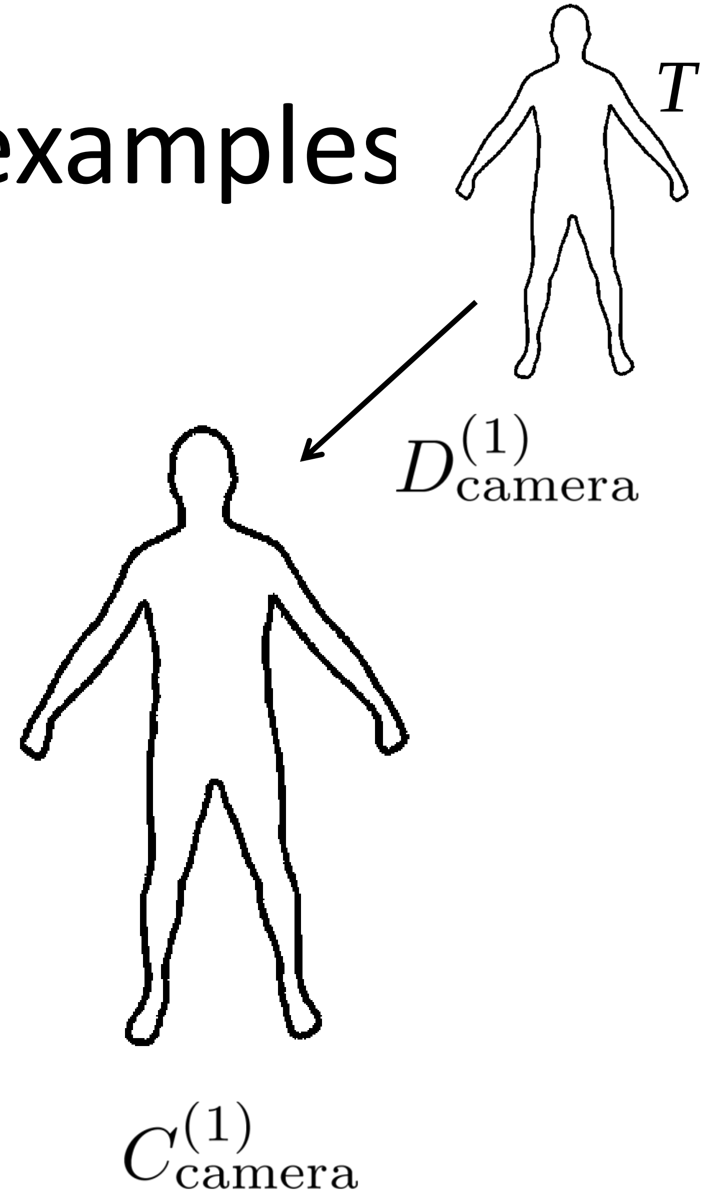
Pose training examples

Example #450



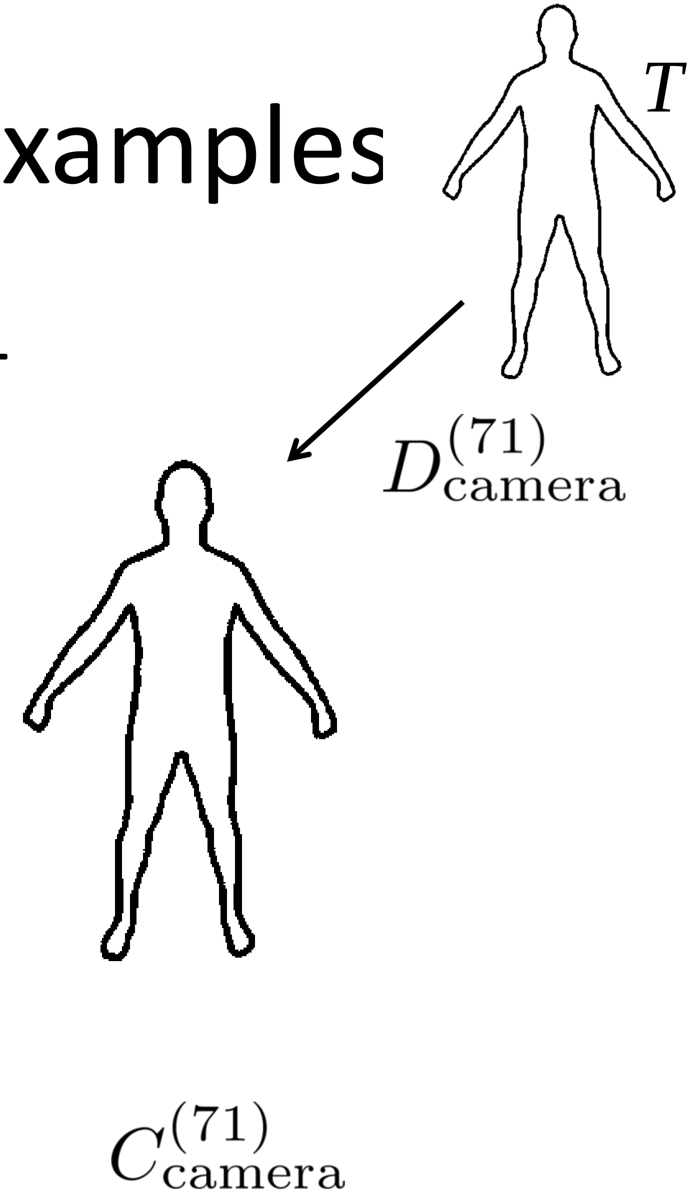
Camera training examples

Example #1



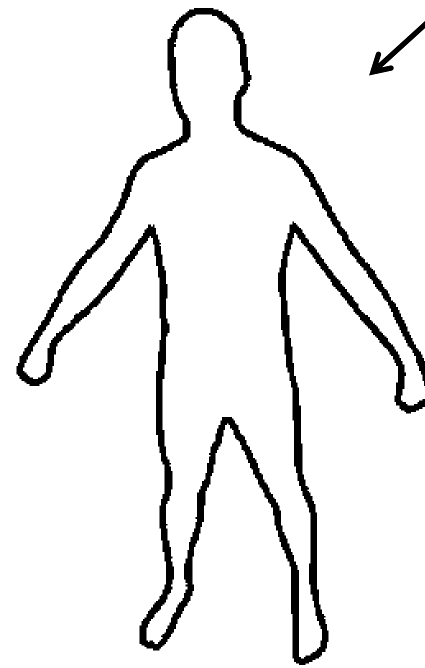
Camera training examples

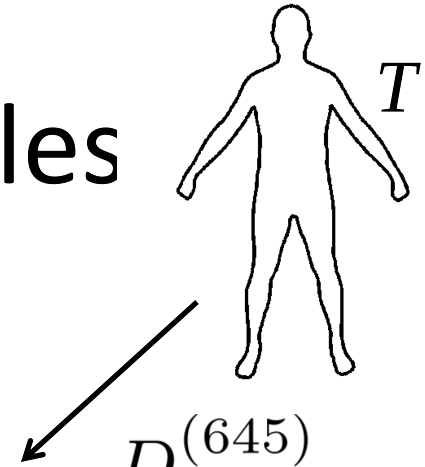
Example #71



Camera training examples

Example #645





Training set

$$D_{\text{shape}}^{(j)}$$

The deformation of the j^{th} contour person

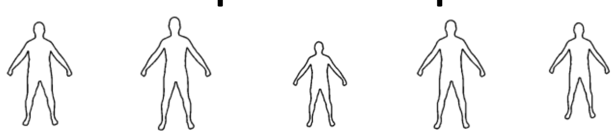
$$D_i^{2\times 2} = S_i \begin{pmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{pmatrix}$$

$$s_i = \log(S_i)$$

$$A_{\text{shape}} = \begin{pmatrix} \cdots & s_1^{(j)} & \cdots \\ \cdots & \theta_1^{(j)} & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & s_n^{(j)} & \cdots \\ \cdots & \theta_n^{(j)} & \cdots \end{pmatrix}$$

$n = \#\text{line segments}$

Shape examples



Principal Component Analysis

- PCA on A_{shape}:

$$\begin{aligned} \theta_i &= \bar{\theta}_i + \sum \beta_k \theta_i^k \\ s_i &= \bar{s}_i + \sum \beta_k s_i^k \\ \Theta_{\text{shape}} &= (\beta_1, \ldots, \beta_K) \end{aligned}$$

k stands for the k^{th} eigenvector

Principal Component Analysis

- PCA on A_{shape}:

$$\begin{aligned} \theta_i &= \bar{\theta}_i + \sum \beta_k \theta_i^k \\ s_i &= \bar{s}_i + \sum \beta_k s_i^k \\ \Theta_{\text{shape}} &= (\beta_1, \ldots, \beta_K) \end{aligned}$$

  k stands for the k^{th} eigenvector

- This defines

$$D_{\text{shape}} = D_{\text{shape}}(\Theta_{\text{shape}})$$

- We do the same for A_{camera} and this defines

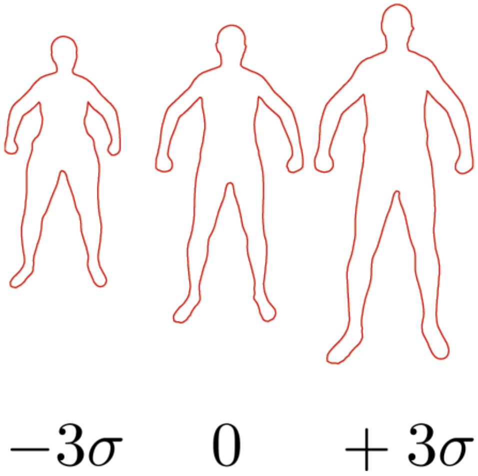
$$D_{\text{camera}} = D_{\text{camera}}(\Theta_{\text{camera}})$$

Principal Component Analysis

- Why use angle and log-scale? Interpretation: PCA in a Lie-algebra [Vaillant at al. '04]

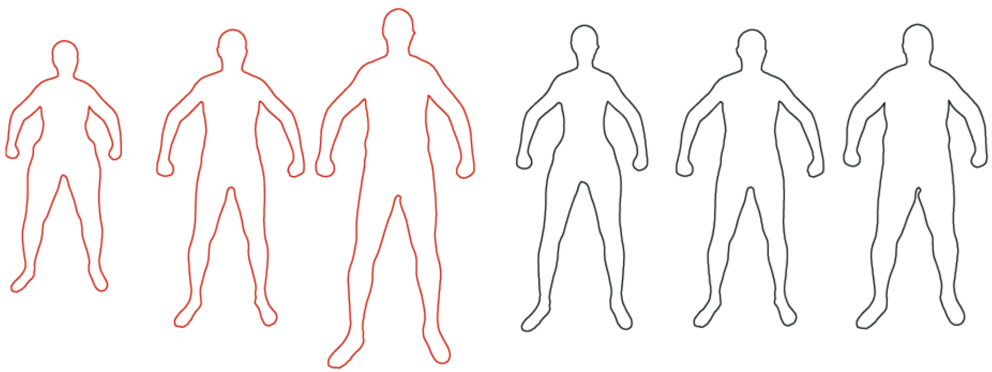
$$S_i \begin{pmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{pmatrix} = \exp\left(\begin{pmatrix} s_i & 0 \\ 0 & s_i \end{pmatrix} + \begin{pmatrix} 0 & -\theta_i \\ \theta_i & 0 \end{pmatrix}\right)$$

Eigen-shapes



Principal component #1

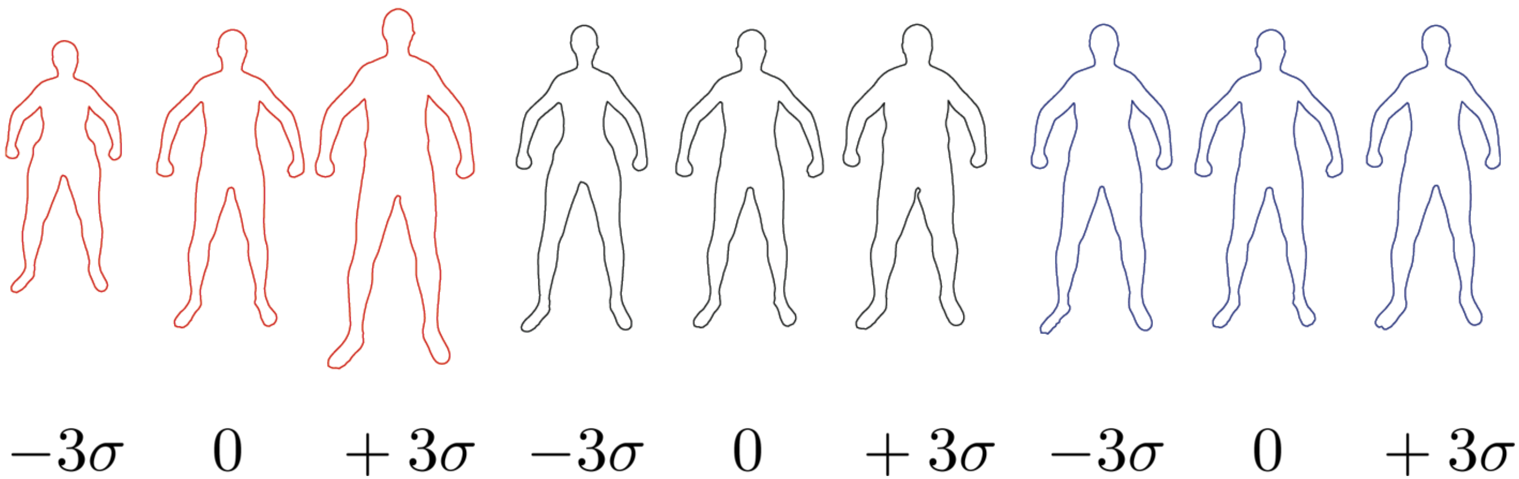
Eigen-shapes



-3σ 0 $+3\sigma$ -3σ 0 $+3\sigma$

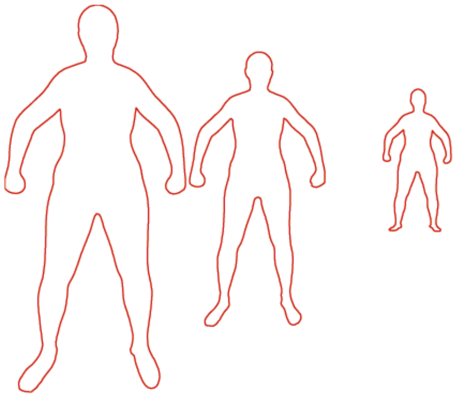
Principal component #1 Principal component #2

Eigen-shapes



Principal component #1 Principal component #2 Principal component #3

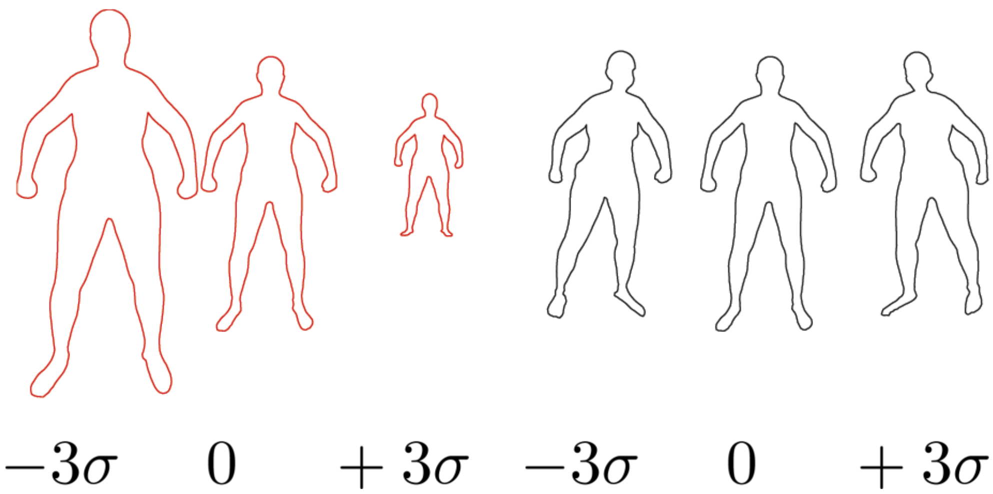
Eigen-cameras



$-3\sigma \quad 0 \quad +3\sigma$

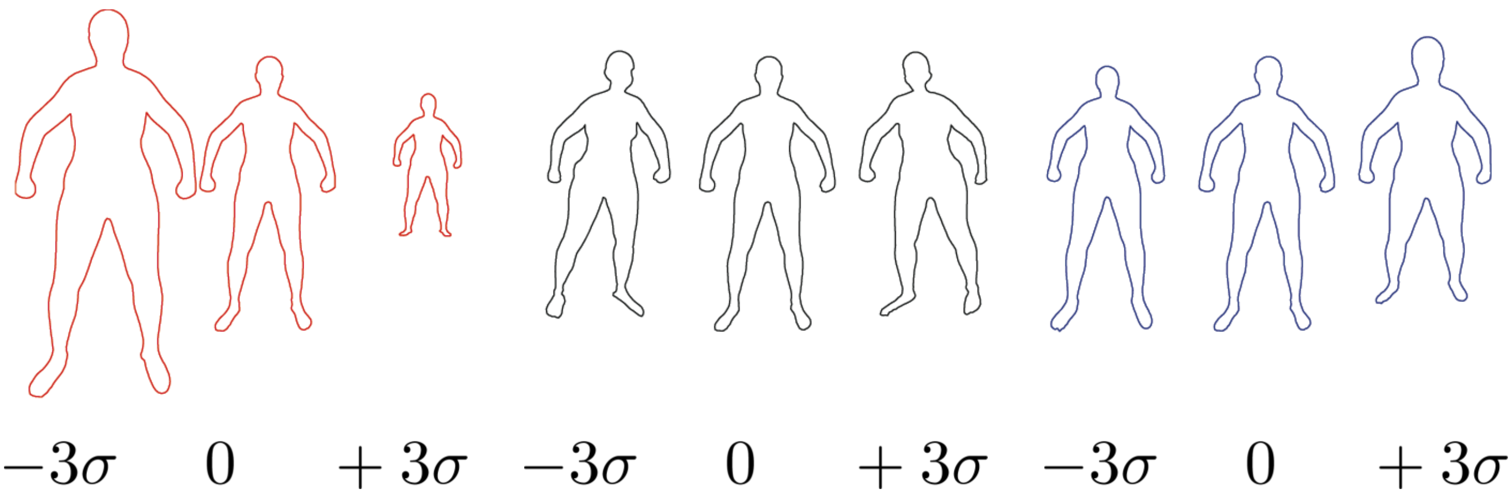
Principal component #1

Eigen-cameras



Principal component #1   Principal component #2

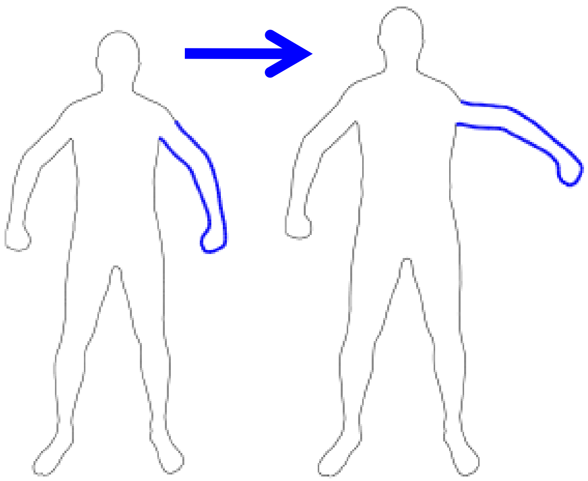
Eigen-cameras



Principal component #1 Principal component #2 Principal component #3

Pose: rigid + non-rigid deformations

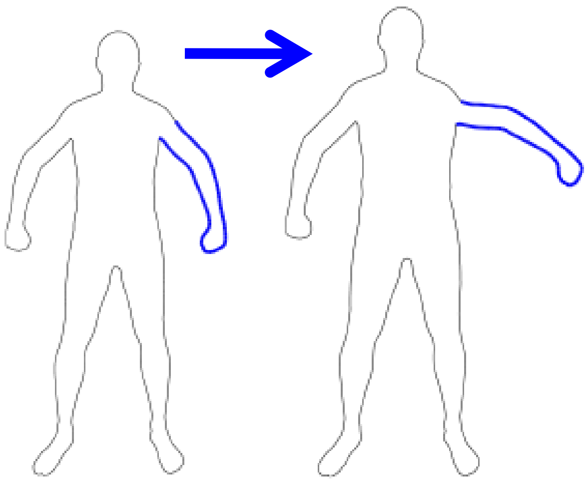
(PS meets SCAPE)



$$D_{\text{pose}} = D_{\text{pose}}(\Theta_{\text{pose}})$$

Pose: rigid + non-rigid deformations

(PS meets SCAPE)

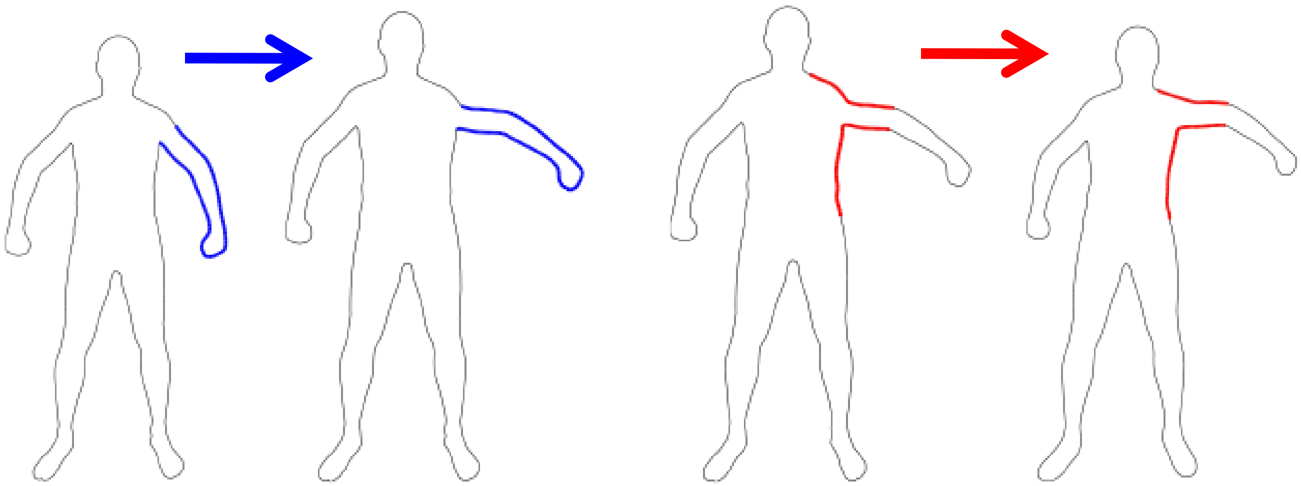


$$\forall i \in \text{arm} \qquad \theta_i = \theta_{\text{RIGID}}$$

$$s_i = s_{\text{RIGID}}$$

Pose: rigid + non-rigid deformations

(PS meets SCAPE)



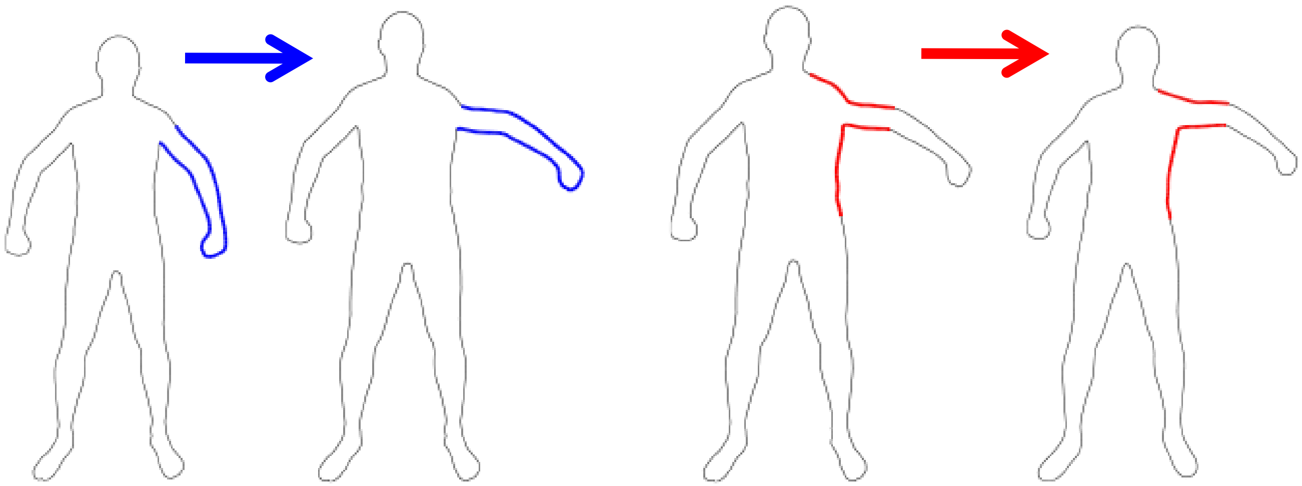
$$\begin{pmatrix} \Delta\theta_i \\ \Delta s_i \end{pmatrix} = H_i(\theta_{\mathrm{RIGID}}, s_{\mathrm{RIGID}}, 1)^T$$

$$\theta_i = \theta_{\mathrm{RIGID}} + \Delta\theta_i$$

$$s_i = s_{\mathrm{RIGID}} + \Delta s_i$$

Pose: rigid + non-rigid deformations

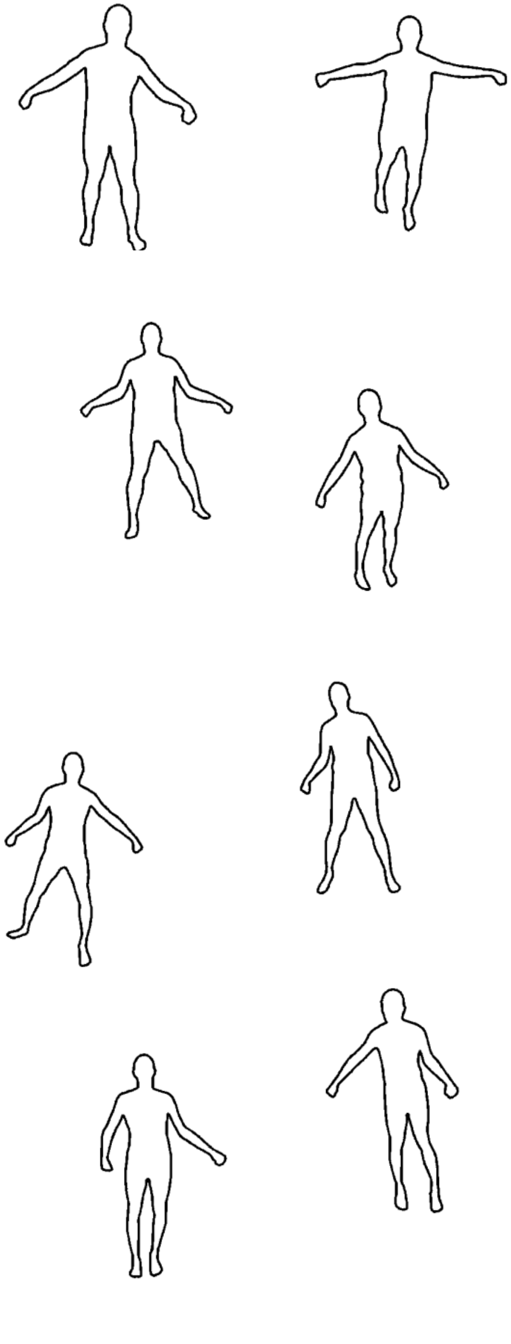
(PS meets SCAPE)

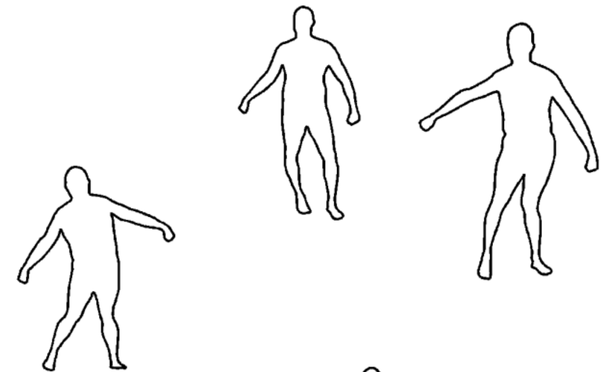


$$\begin{pmatrix} \Delta\theta_i \\ \Delta s_i \end{pmatrix} = H_i(\theta_{\mathrm{RIGID}}, s_{\mathrm{RIGID}}, 1)^T$$

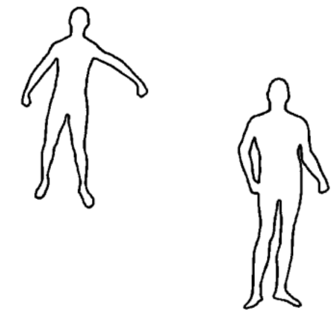
The matrix H_i is learned from examples

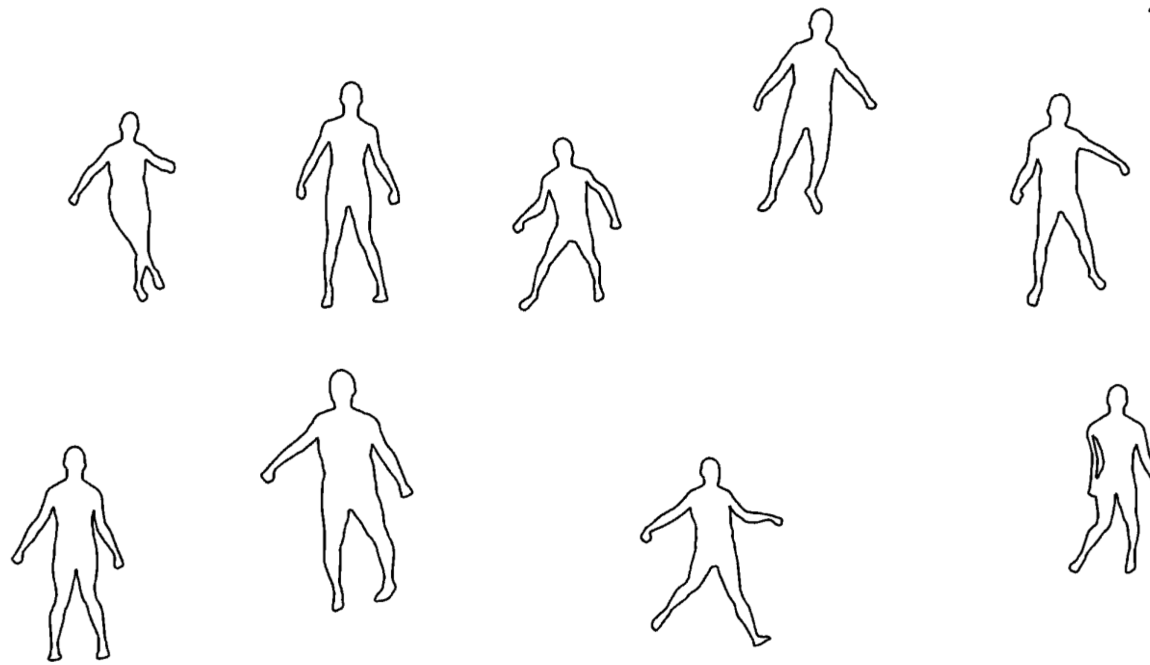
Synthesis





$$\begin{aligned} \Theta &= (\Theta_{\text{shape}}, \Theta_{\text{pose}}, \Theta_{\text{camera}}) \\ C &= E^{\dagger} D(\Theta) E\, T \end{aligned}$$





Fitting CP to silhouettes

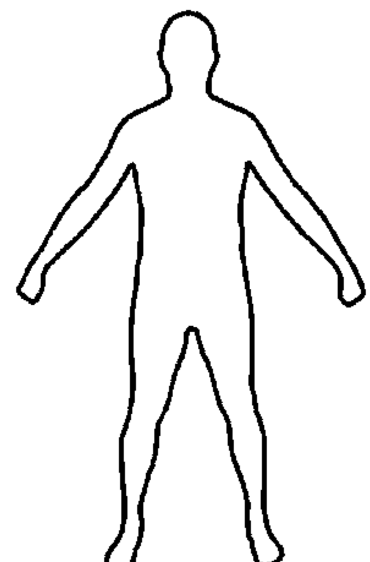
- We minimize a cost function of the form

$$F(\text{mask}, \Theta) = d(\text{mask}, E^{\dagger} D(\Theta) E\, T)$$

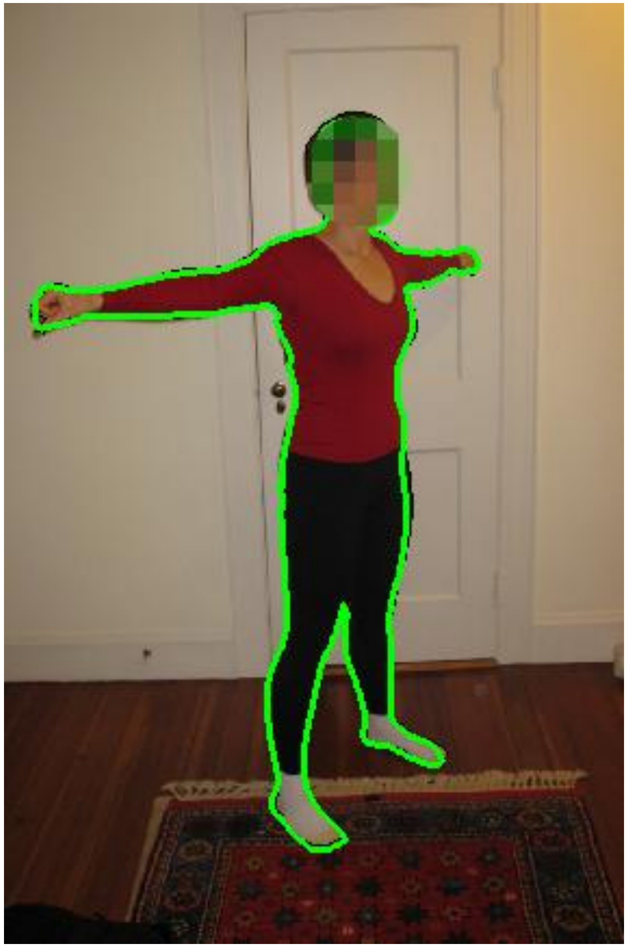
E.g., d can the bi-directional Chamfer distance

mask

T



Selected results



Combine pose/shape estimation with segmentation

- Similar to PoseCut and ObjCut

[Bray et al. ECCV ’06, Pawan et al. PAMI ‘10]

- We minimize a cost function of the form

$$F(I,\Theta) = F_{\text{region}}(I,\Theta) + F_{\text{edge}}(I,\Theta) + F_{\text{prior}}(\Theta)$$

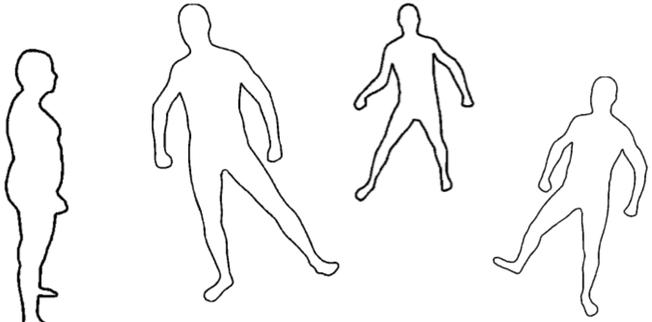
- We used a PS algorithm as an initialization

[Andriluka et al. CVPR ‘09]

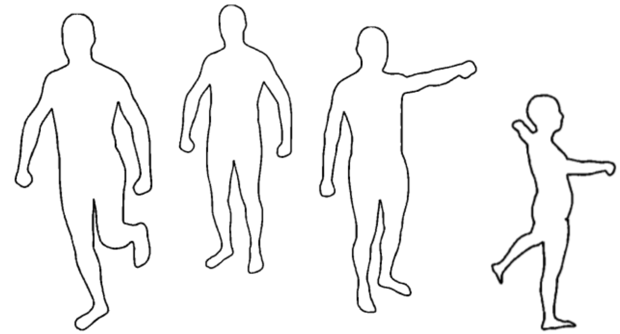
Selected results

PS

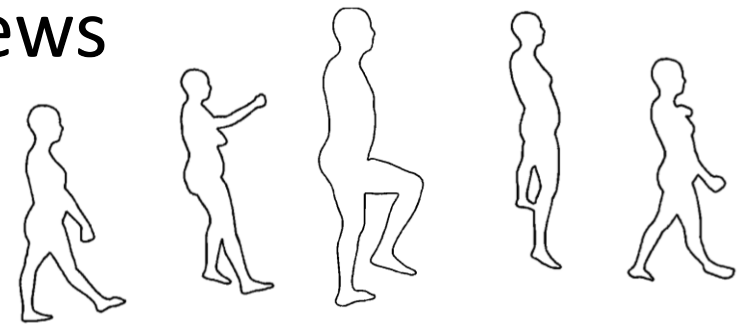
CP

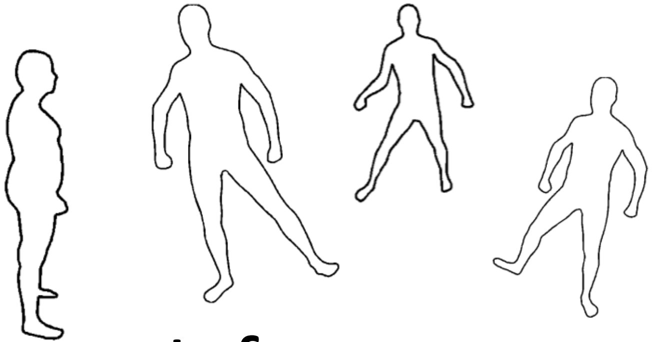


Future work

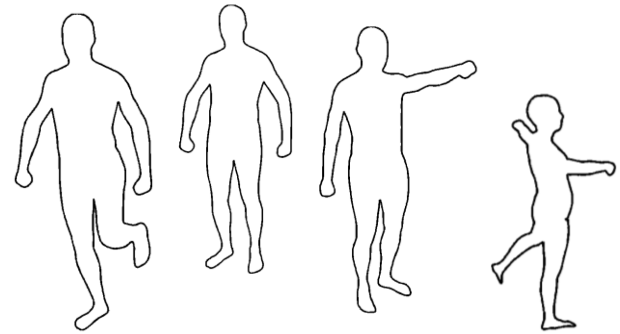


- Inference over discrete views
- Tracking
- Clothing [Guan et al. ECCV ’10]
- Coarse-to-fine inference (PS to CP)

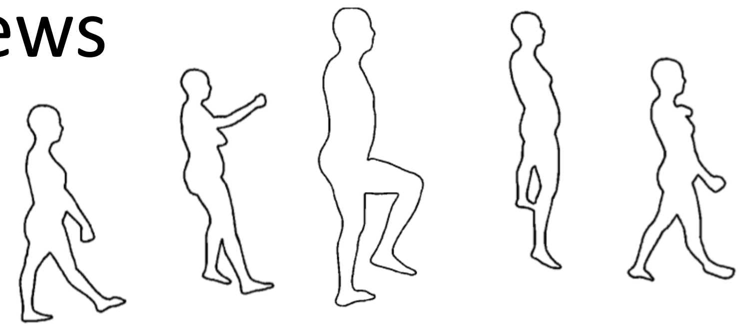


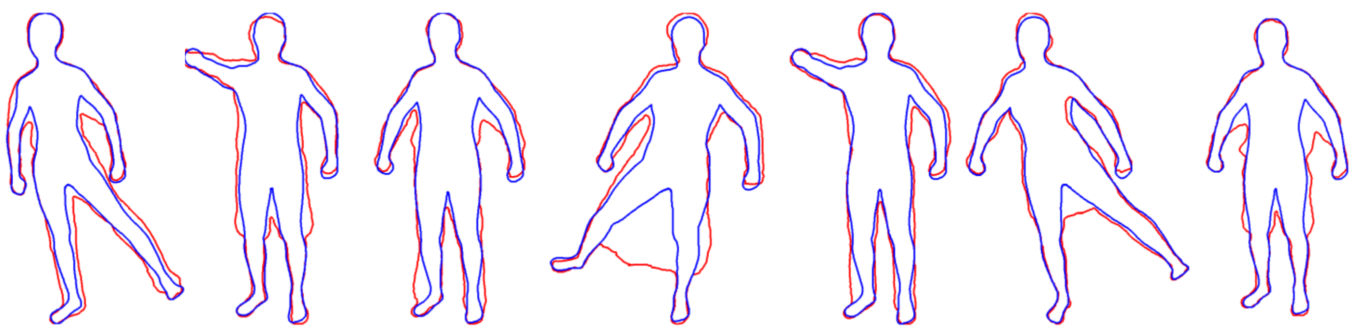


Future work

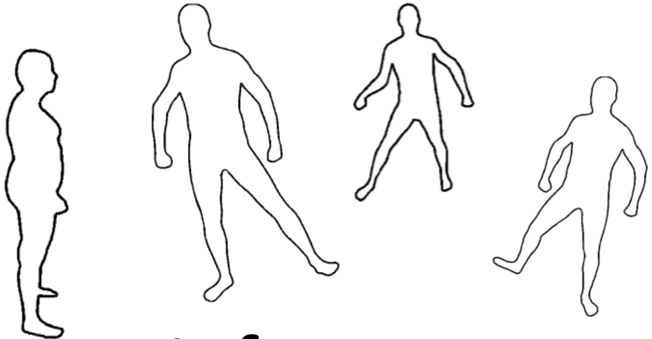


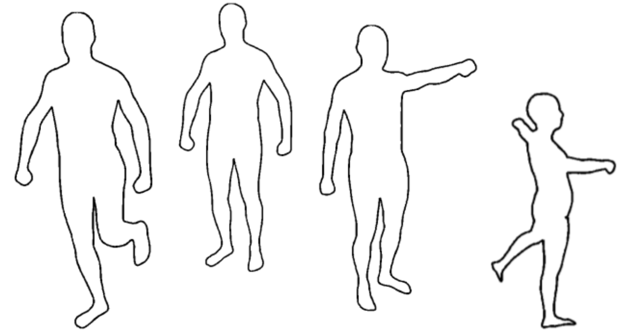
- Inference over discrete views
- Tracking
- Clothing [Guan et al. ECCV ’10]
- Coarse-to-fine inference (PS to CP)



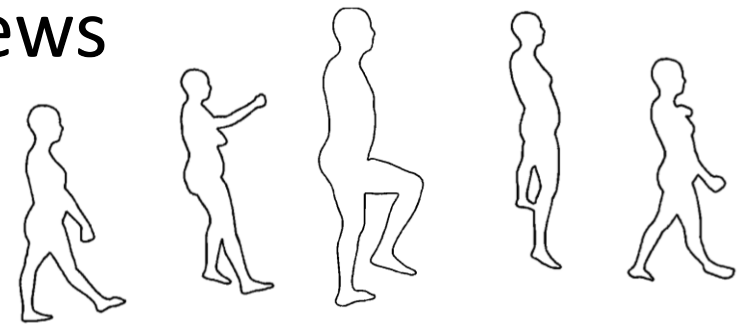


Future work



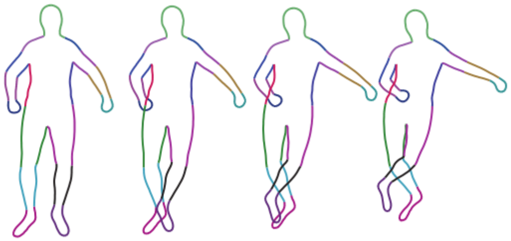


- Inference over discrete views
- Tracking
- Clothing [Guan et al. ECCV ’10]
- Coarse-to-fine inference (PS to CP)



Big question

- PS-like inference using a part-based structure?



Conclusions

- A new 2D part-based generative model of humans
  - Beyond previous deformable template models [Cootes et al. CVIU ‘95, Baumberg & Hogg ECCV ‘94, and many others…]
- Factors shape, pose, and camera deformations
- Has advantages of PS models with the detail of a SCAPE-like model
- Initial application: Human-specific image segmentation

Acknowledgments

- This work was funded in part by
  - NIH EUREKA 1R01NS066311-01 grant
  - NSF IIS-0812364 grant
- We thank D. Hirshberg and A. Balan for their assistance with the code
- We thank Andriluka et al. for their PS implementation