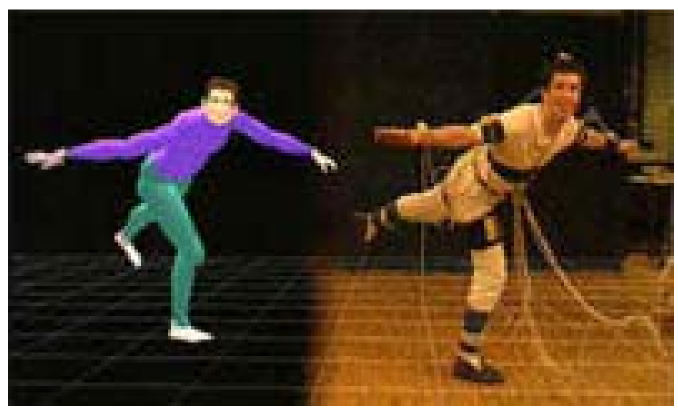
Human body modeling

Hyewon SEO
DR-CNRS@MLMS
PhD@MIRALab, Univ. Geneva

Scope of human modeling modeling

Intelligence

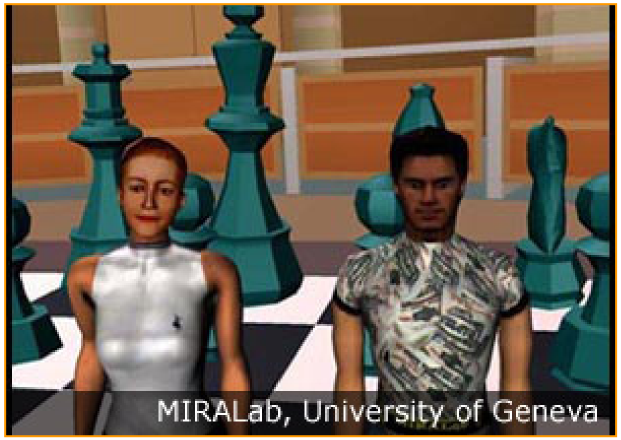
Behavior



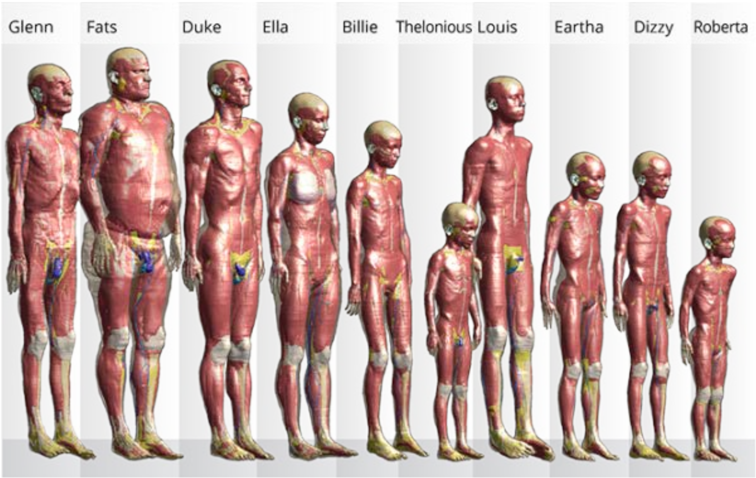
Shape & Appearance

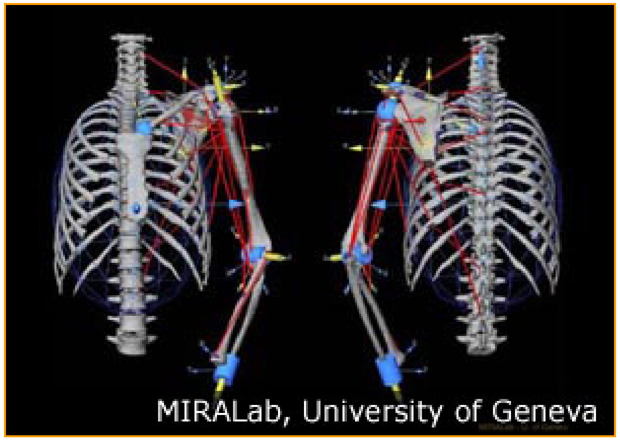
Examples of human models (Applications)

- Ergonomic studies



- Avatars in VE

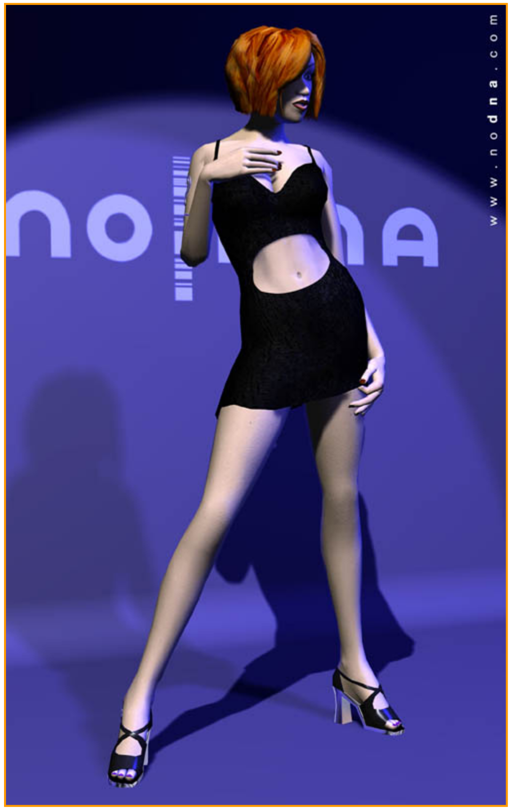




- Medical tools

Examples of human models (Applications)

- Films
- Games
- E-Commerce



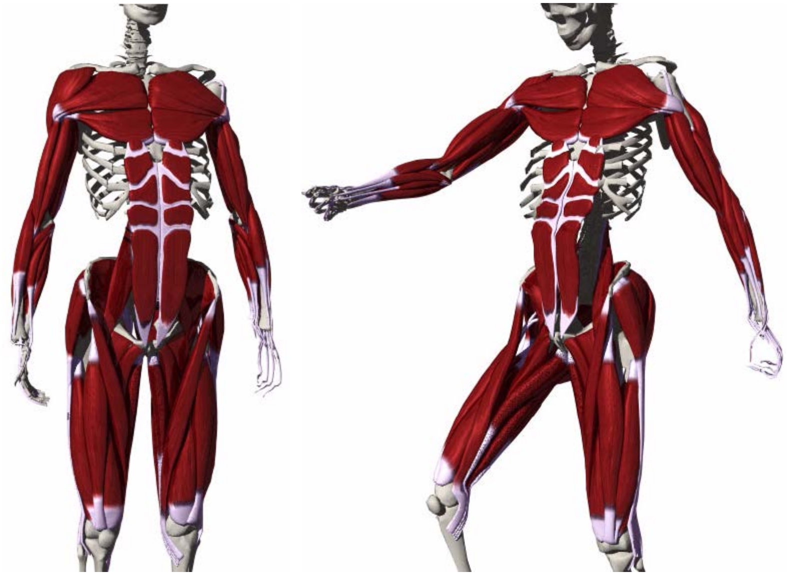
Human body modeling I

- **How to:**
  - build geometry to represent a specific human body?
- **Methods:**
  - interactive design
  - anatomical models
  - reconstruction techniques

c.f. Anatomical models

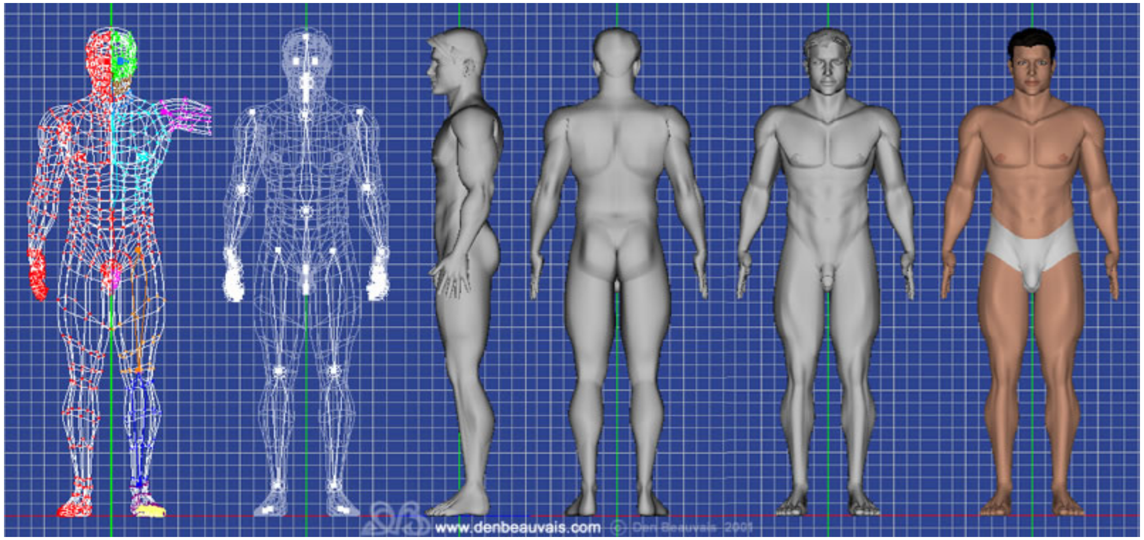
- Physics-based simulation methods, i.e. FEM
- Accurate simulation of muscles, bones and fat tissues

[Aubel et al 02]

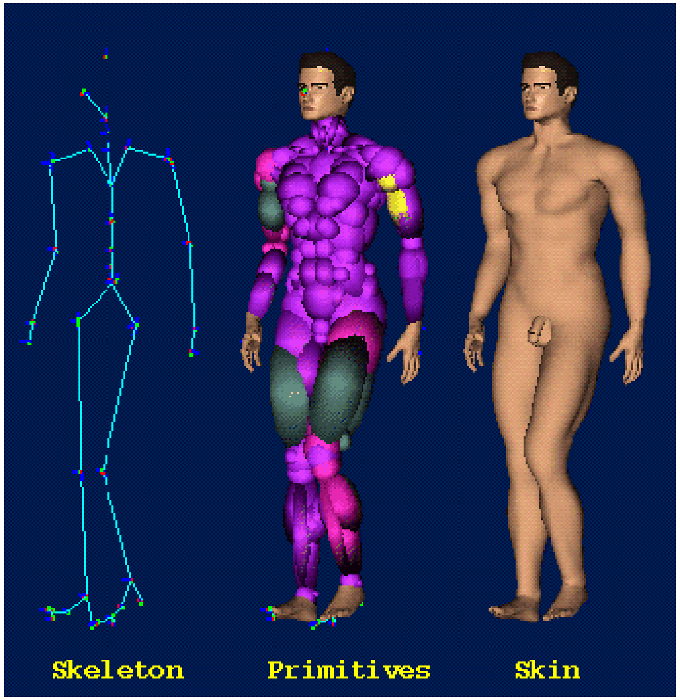


Interactive modeling

- **Surface**

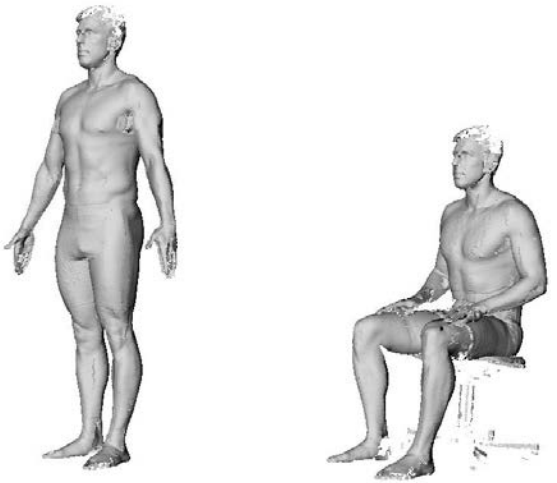


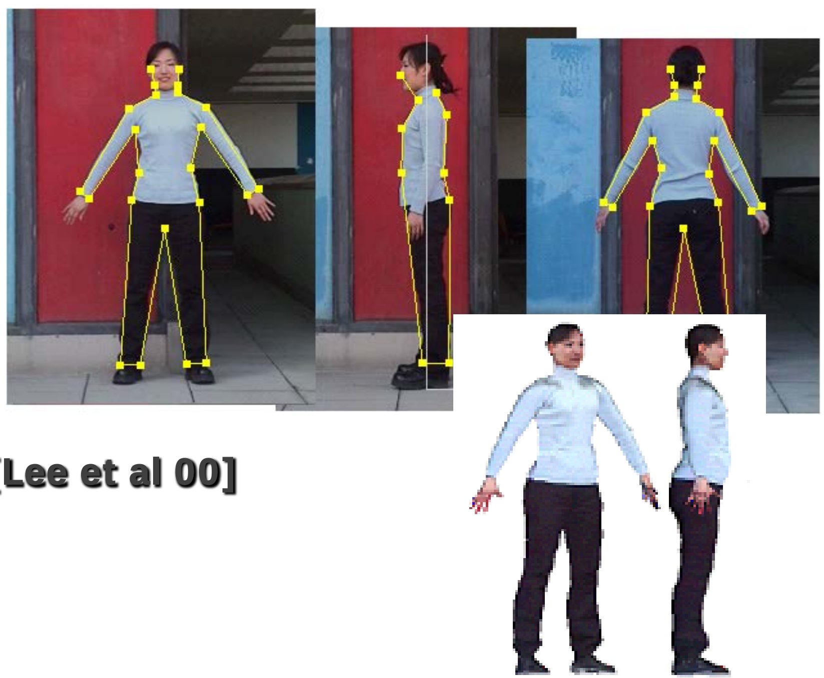
- **Multi-layered**



Reconstructive modeling

- **Capture shapes in real world**
  - **Photos, 3D scans, videos, etc.**





[Lee et al 00]

[Sand et al 03]

Reconstructive modeling

- **Acquisition of precise, realistic appearance**

static

[Lee et al, Eurographics 2000]

posed

CIVILIAN AMERICAN & EUROPEAN
SURFACE ANTHROPOMETRY RESOURCE

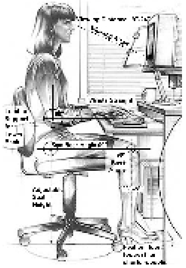
[CAESAR 2002]

dynamic

[de Aguiar et al, SIGGRAPH 2008]

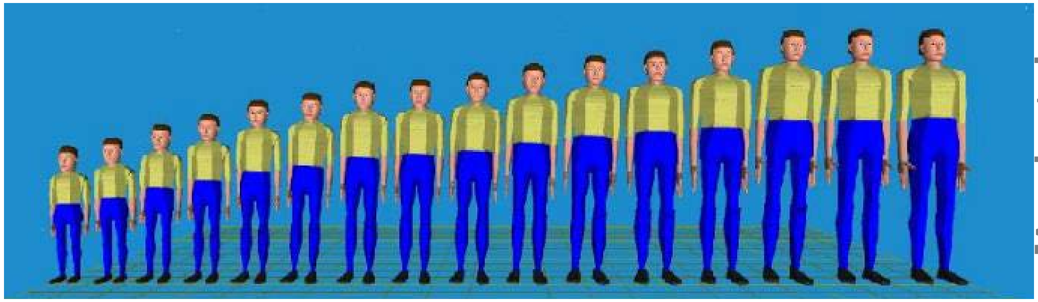
Human shape modeling II

- **How to:**
  - systematically obtain a variety of body shapes?
- **Methods:**
  - Segment scale
  - Variational modeling
  - Example-based
  - Data-driven
- **Trade-offs between**
  - controllability
  - quality of shape
  - time costs



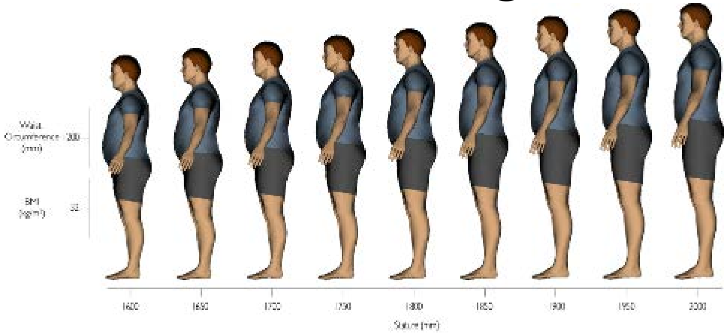
Geometric methods

- **Segment-wise scaling**



[Azuola et al, Proc. IMAGE 1994]

- **Variational modeling**



[Poirson et al, J. Industrial Ergonomics 2014]

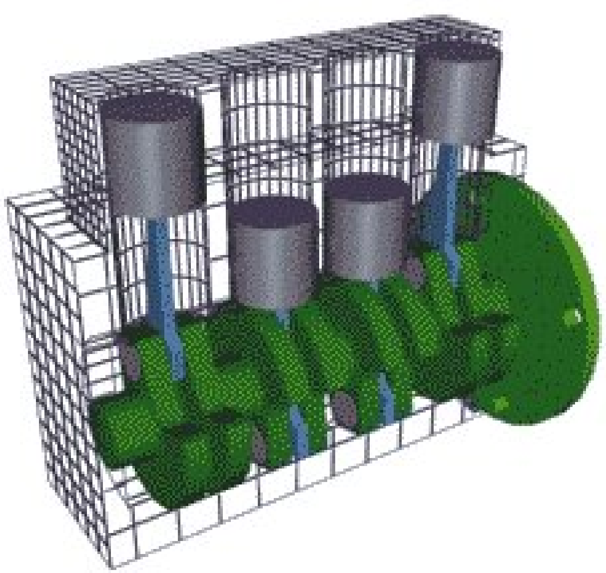
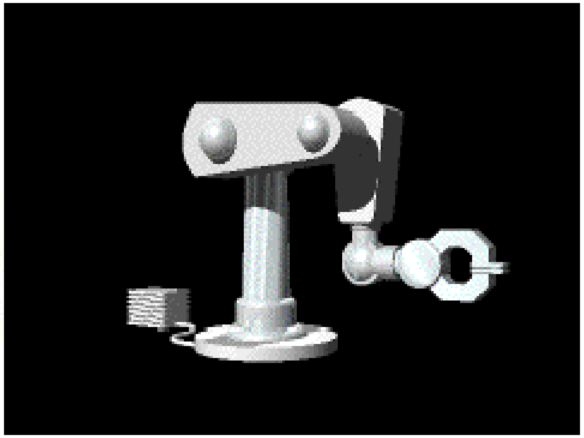
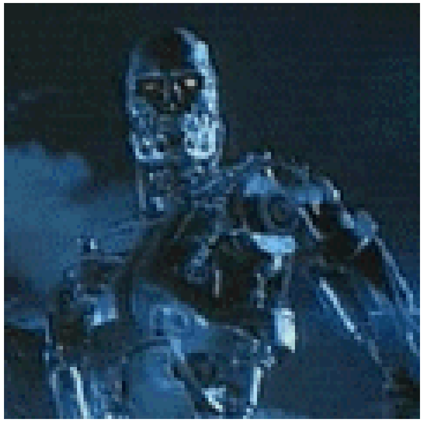
- ✓ Deformation energy as objective function
- ✓ Antropometric measurements as constraints

Hierarchical modeling – FK

Kinematics

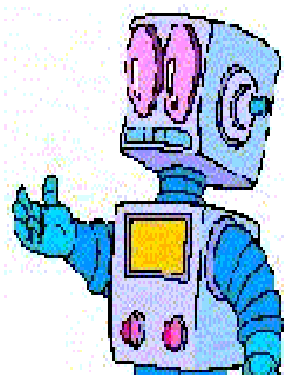
- **Study of object movement**
  - **Irrespective of forces**

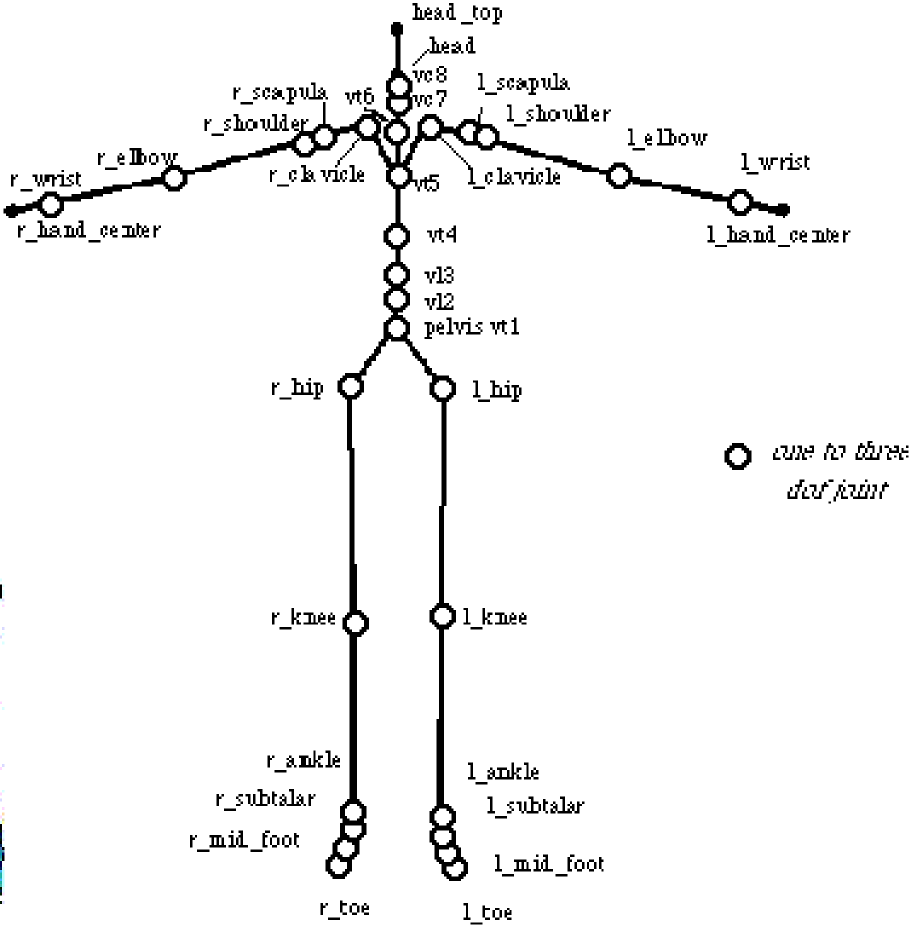
- **c.f. Dynamics**
  - **Compute underlying forces**

Hierarchical Kinematic Modeling

- **A sequence of objects (segments) connected in a chain of joints**
  - **Articulation of characters**
  - **Plants**
  - **Solar system,…**



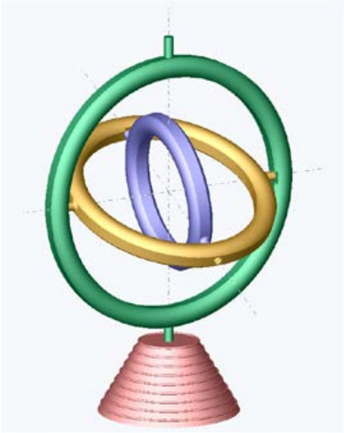


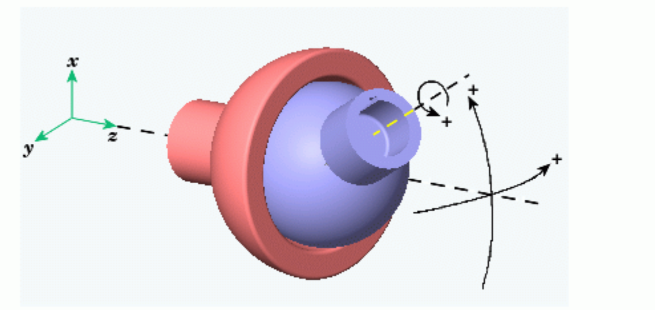
Terminology

- **Hierarchical object -** a sequence of objects connected in a chain of *joints*
- **Segments –** displayable objects forming the connection between the joints
- **End effector -** free end of the chain
- **DoF (Degree of Freedom):** Number of independent displacements/rotations that specify the configuration of the object
- **Local coordinate systems (LCS):** Coord. systems where each segment is respectively defined

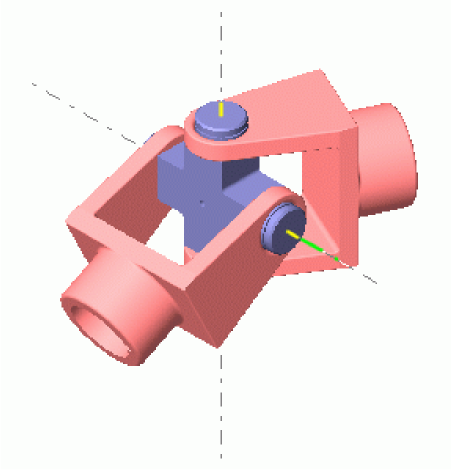
Joint types

- 3 DOF joints
  - Gimbal
  - Spherical

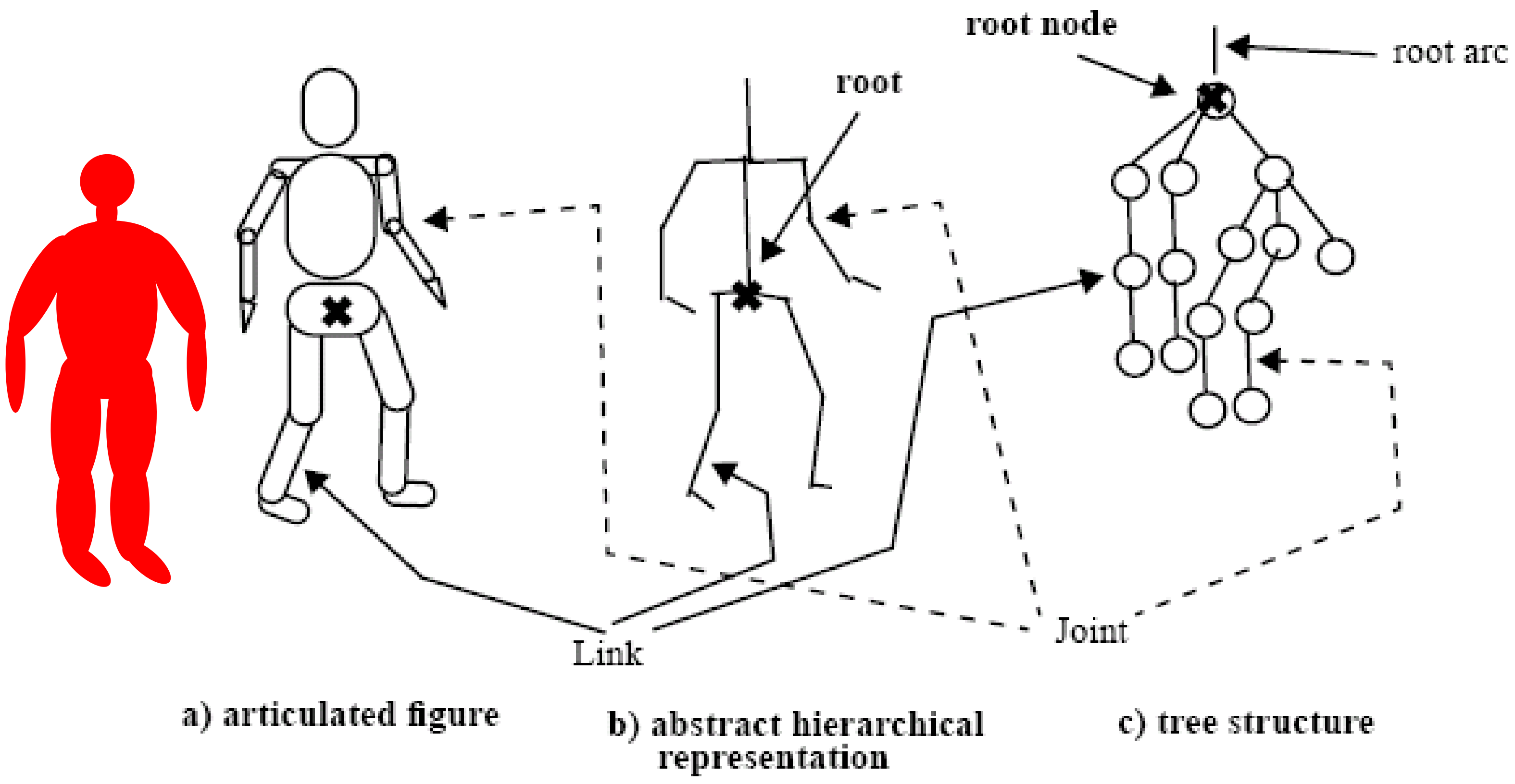




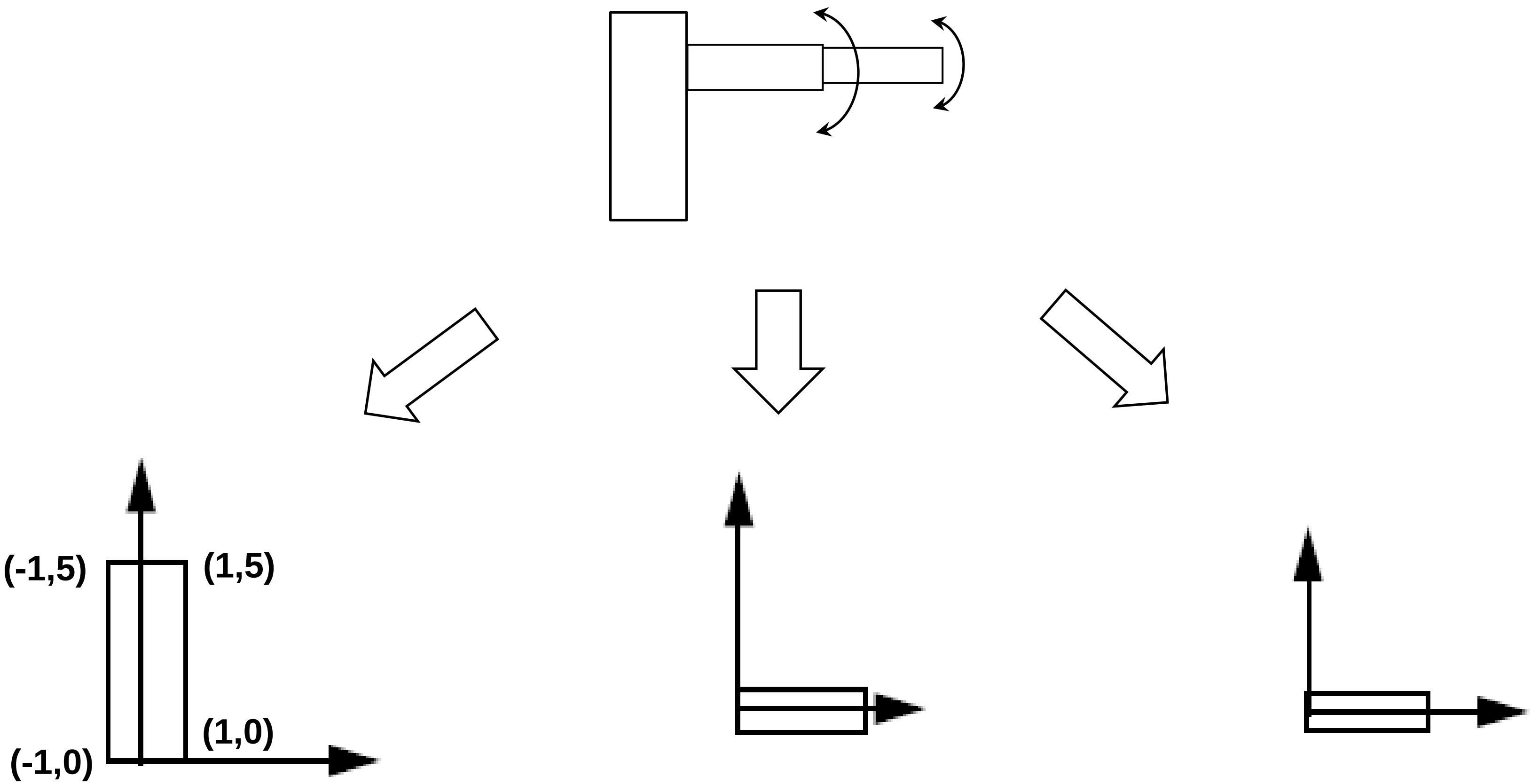
- 2 DOF joints
  - Universal



Representing Articulated Figures



A Simple (3 link) Example

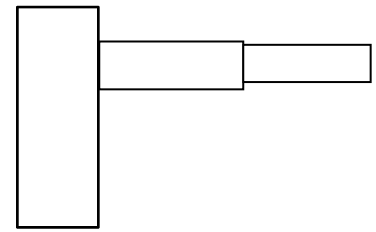


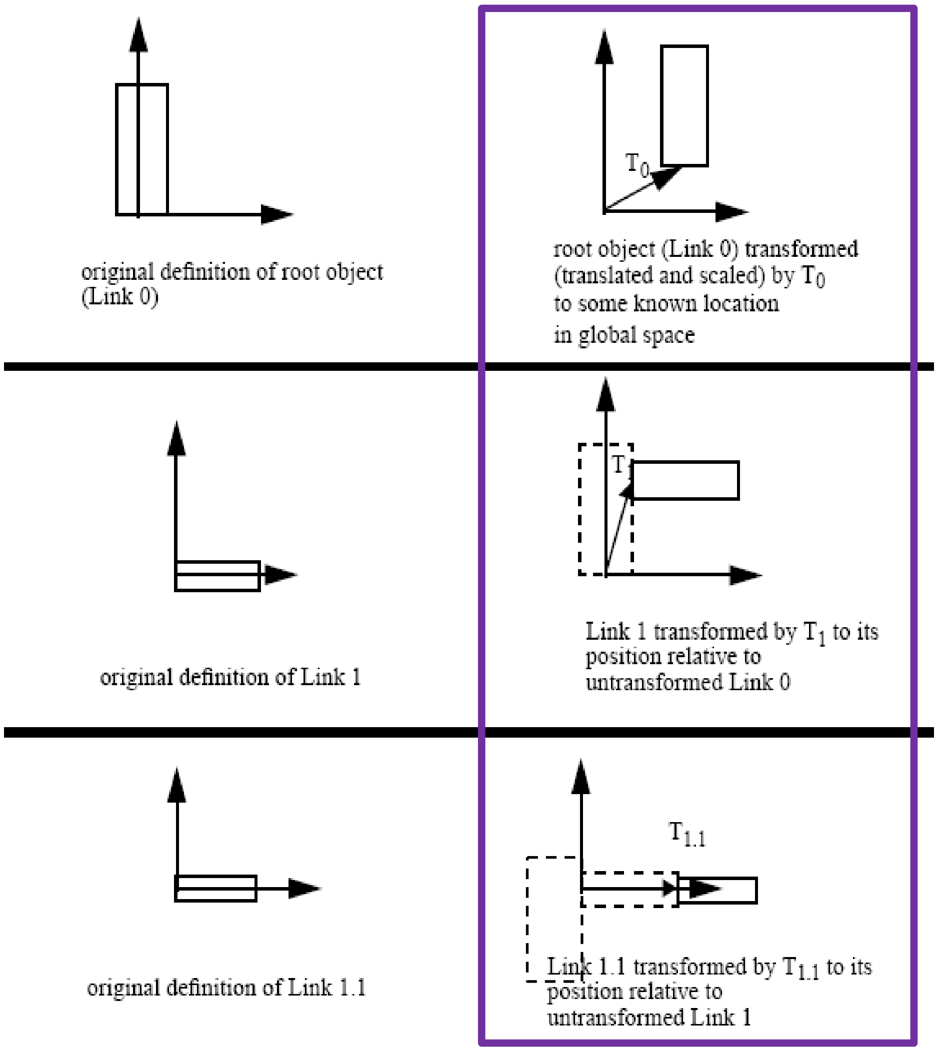
original definition of root object (Link 0)

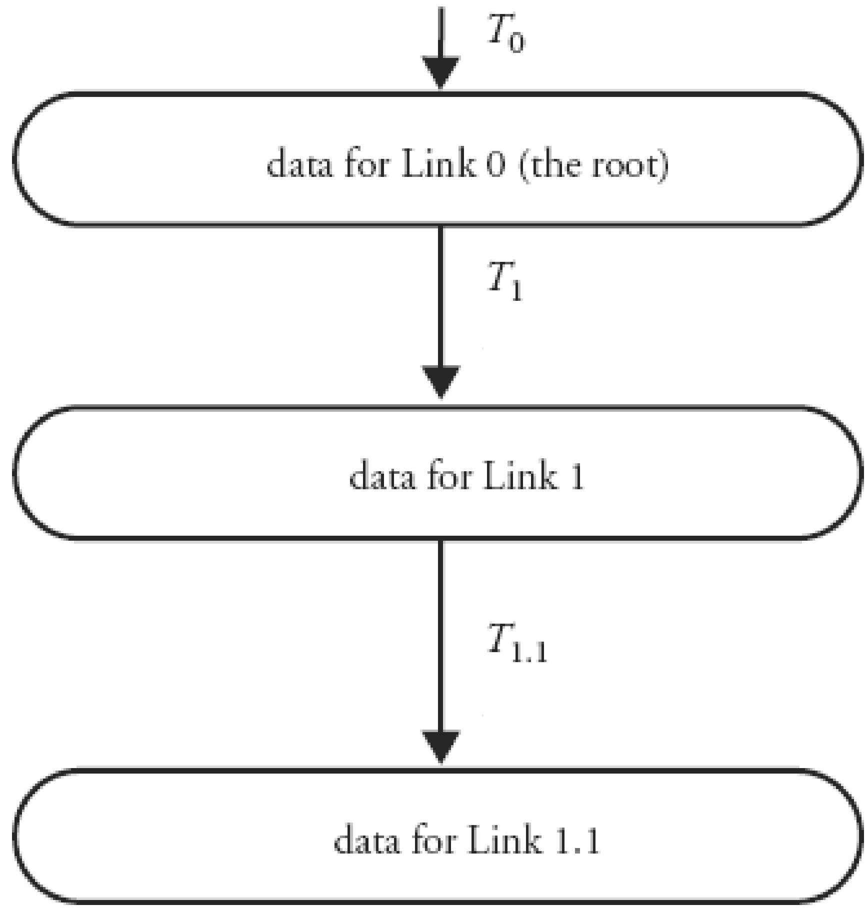
original definition of Link 1

original definition of Link 1.1

A Simple (3 link) Example



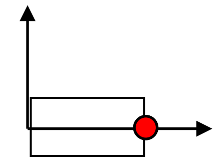




<Tree structure>

Locating a Vertex to WC

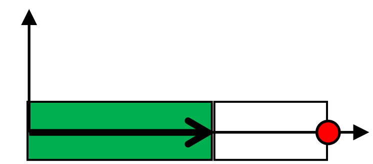
- **Define Link data so its <u>center of rotation</u> is at the origin (optional)**



$V_{1.1}$

- **Position it relative to its parent link in the hierarchy**

$$V^{1}_{1.1} = T_{1.1} \cdot V_{1.1}$$

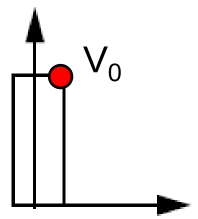


$T_{1.1}$

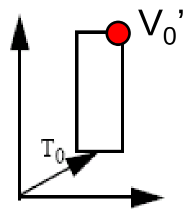
- **Compose all of the transformations up the hierarchy to the root**

$$V^{0}_{1.1} = T_0 \cdot T_1 \cdot T_{1.1} \cdot V_{1.1}$$

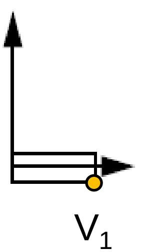
Locating a Vertex to WC



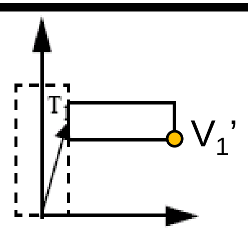
original definition of root object (Link 0)



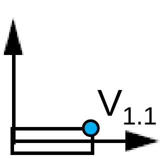
root object (Link 0) transformed (translated and scaled) by T_0 to some known location in global space



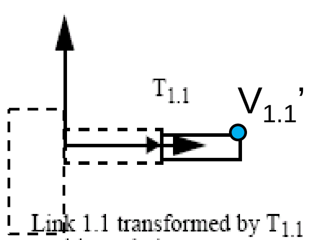
original definition of Link 1



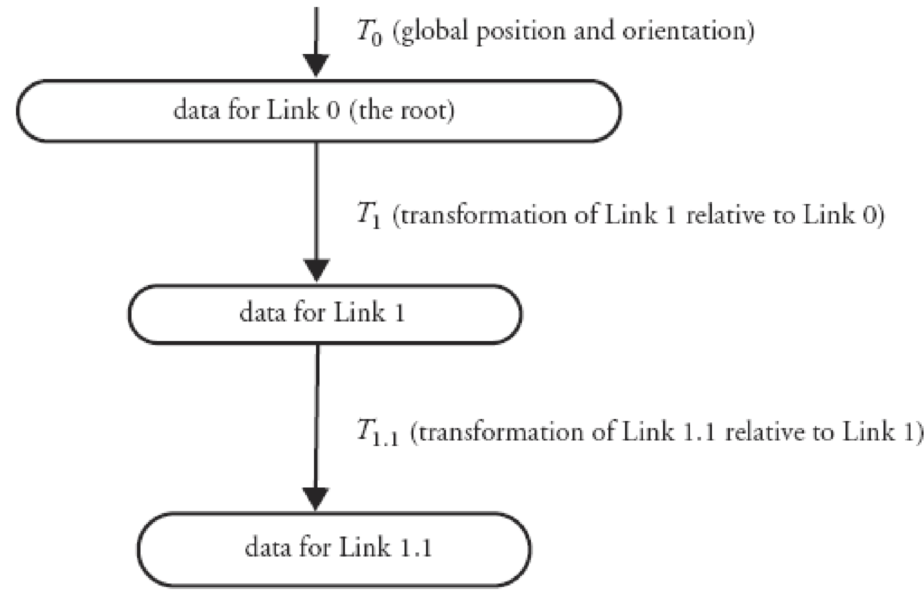
Link 1 transformed by T_1 to its position relative to untransformed Link 0



original definition of Link 1.1



Link 1.1 transformed by $T_{1.1}$ to its position relative to untransformed Link 1



Locating vertices in WC :

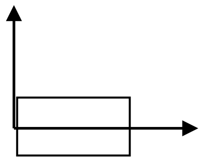
$$V'_0 = T_0 \cdot V_0$$

$$V'_1 = T_0 \cdot T_1 \cdot V_1$$

$$V'_{1.1} = T_0 \cdot T_1 \cdot T_{1.1} \cdot V_{1.1}$$

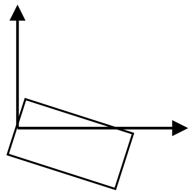
Locating a Vertex to WC (2)

- **Define Link data so its center of rotation is at the origin (optional)**



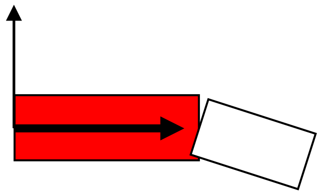
- **Rotate the link**

$$R_{1.1}(\theta_{1.1})$$



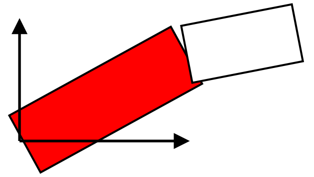
- **Position it relative to its parent link in the hierarchy**

$$T_{1.1}$$



$$T_{1.1} \cdot R_{1.1}(\theta_{1.1})\ \ V_{1.1}$$

- **Compose all of the transformations up the hierarchy to the root**



$$T_0 \cdot T_1 \cdot R_1(\theta_1)\ \ T_{1.1} \cdot R_{1.1}(\theta_{1.1})\ \ V_{1.1}$$

Adding (Variable or Changeable) Rotations

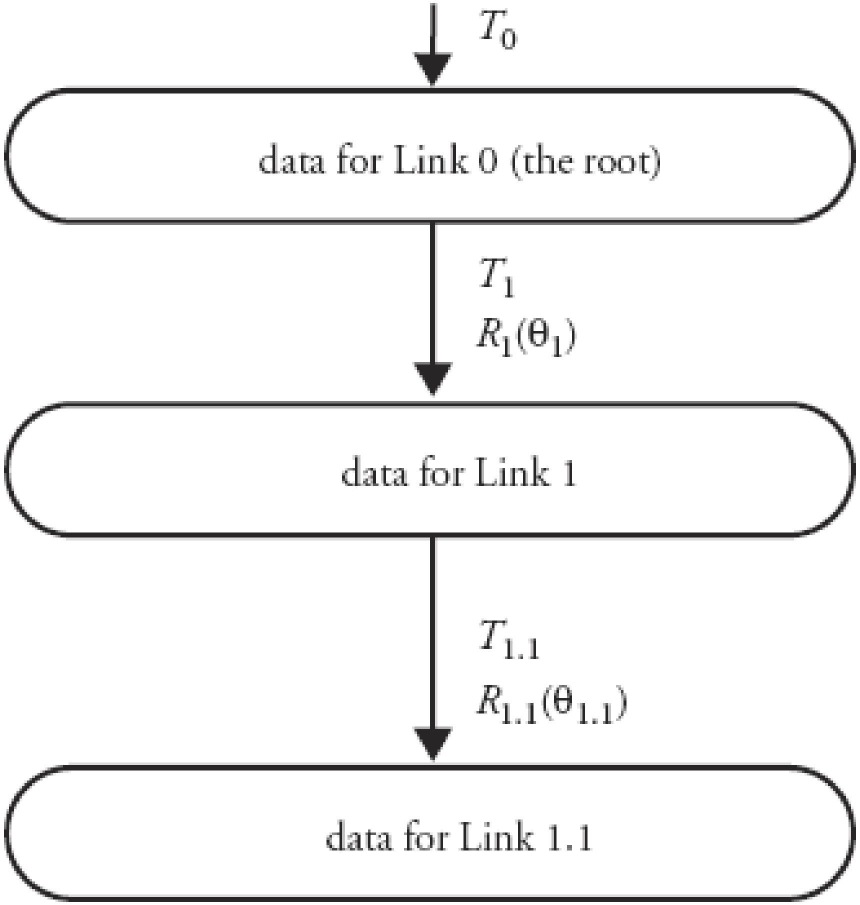
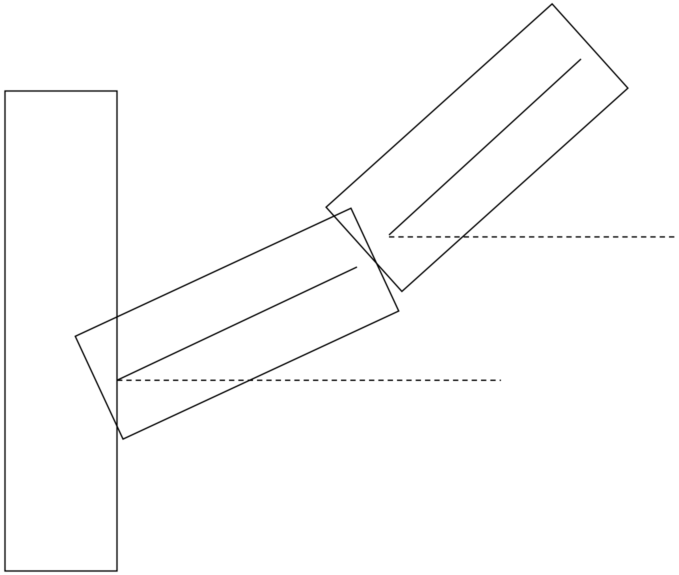
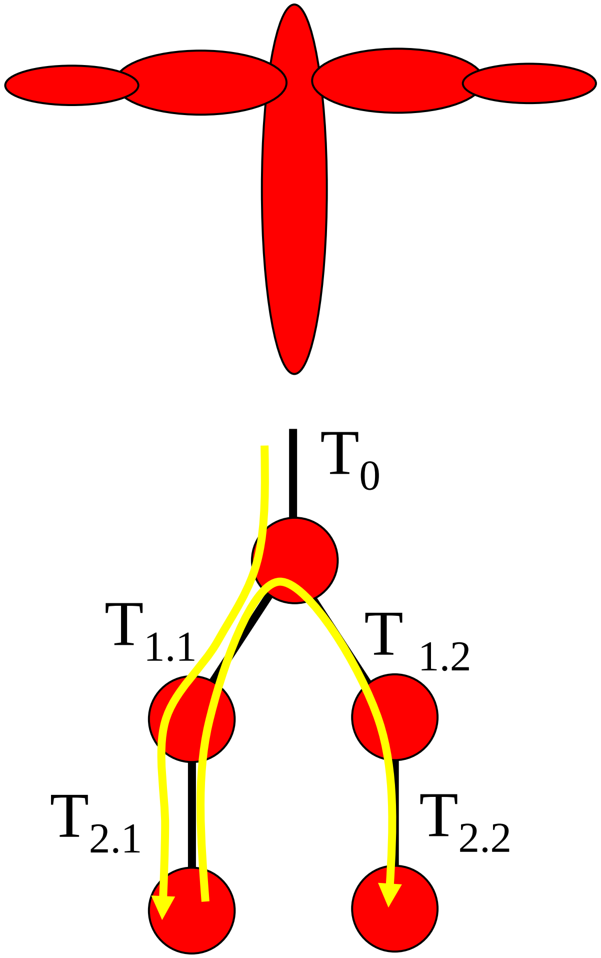


Figure 4.8 Hierarchy showing joint rotations



Tree Representation



M = I

M = T0

M = T0*T1.1

M = T0*T1.1*T2.1

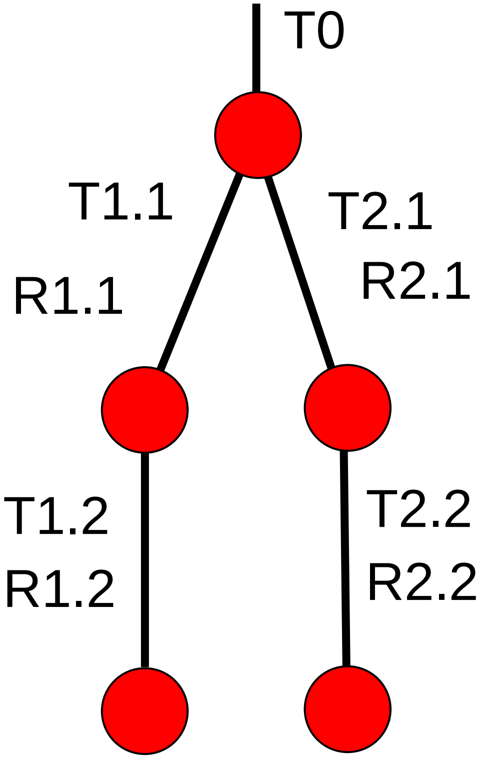
M = T0*T1.1

M = T0

M = T0*T1.2

M = T0*T1.2*T2.2

With (Variable, Changeable) Rotations



M = I

M = T0

M = T0*T1.1*R1.1

M = T0*T1.1*R1.1*T1.2*R1.2

M = T0*T1.1*R1.1

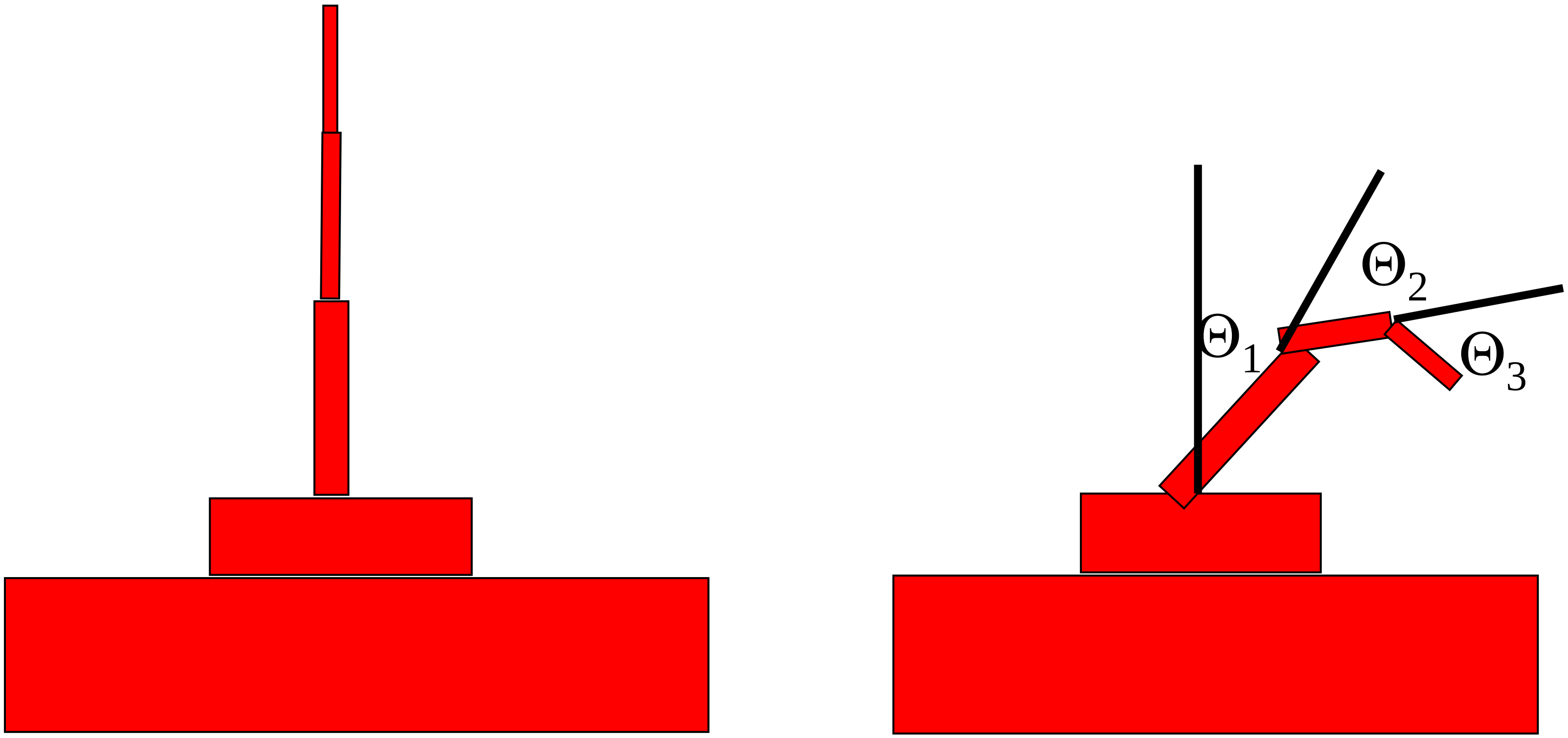
M = T0

M = T0*T2.1*R2.1

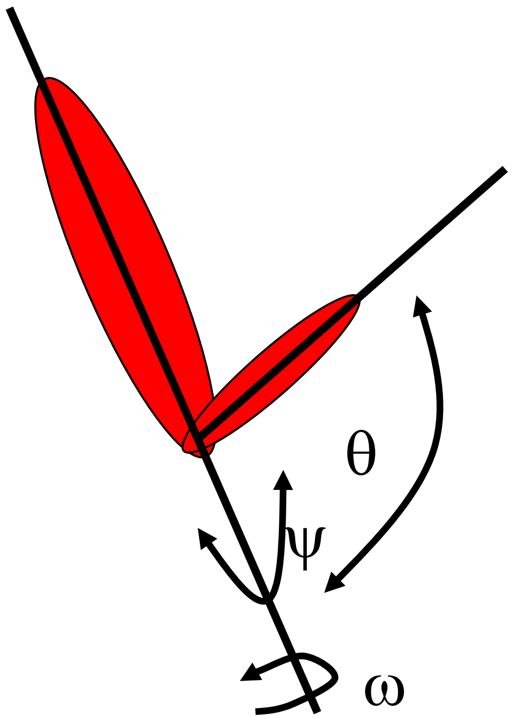
M = T0*T2.1*T2.2*R2.2

Specifying a pose

- **Pose Vector - a complete set of joint parameters (DoFs)**



Specifying a pose

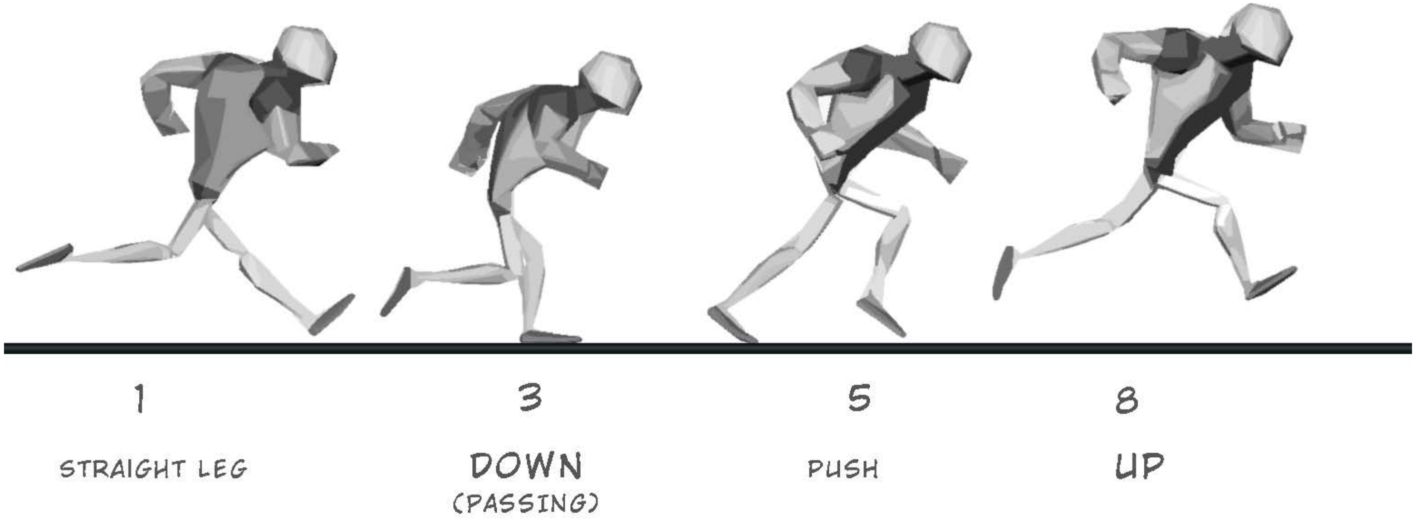


Multiple DoF in case of complex joint

Use
Axis-angle or quaternion

Forward Kinematics for Animation

- Define Joint-Link Hierarchy
- Define a sequence of keyposes with corresponding times
- For given time t:
  - Use keyposes to interpolate pose at time t
  - Traverse tree hierarchy using the pose vector above

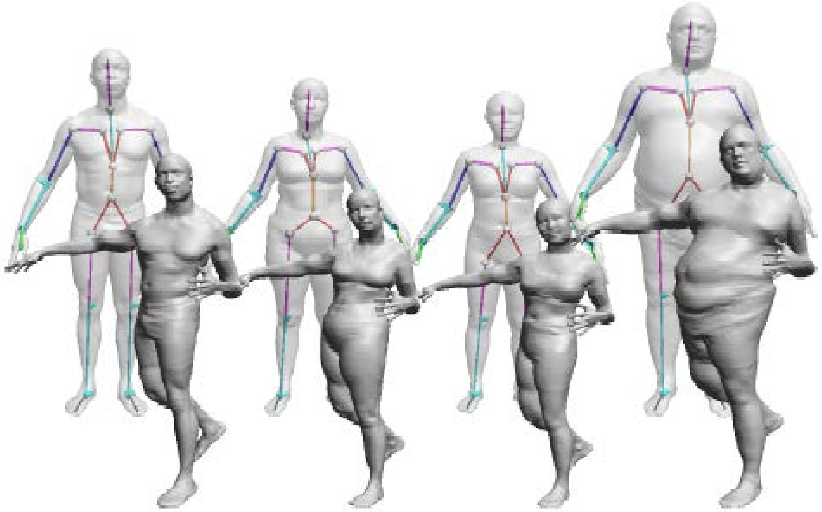


Skeletal-driven Skin Deformation

Skeleton-Driven Deformation (SDD)

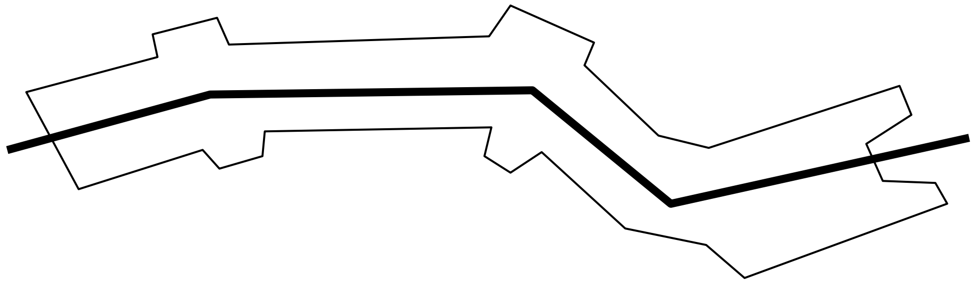
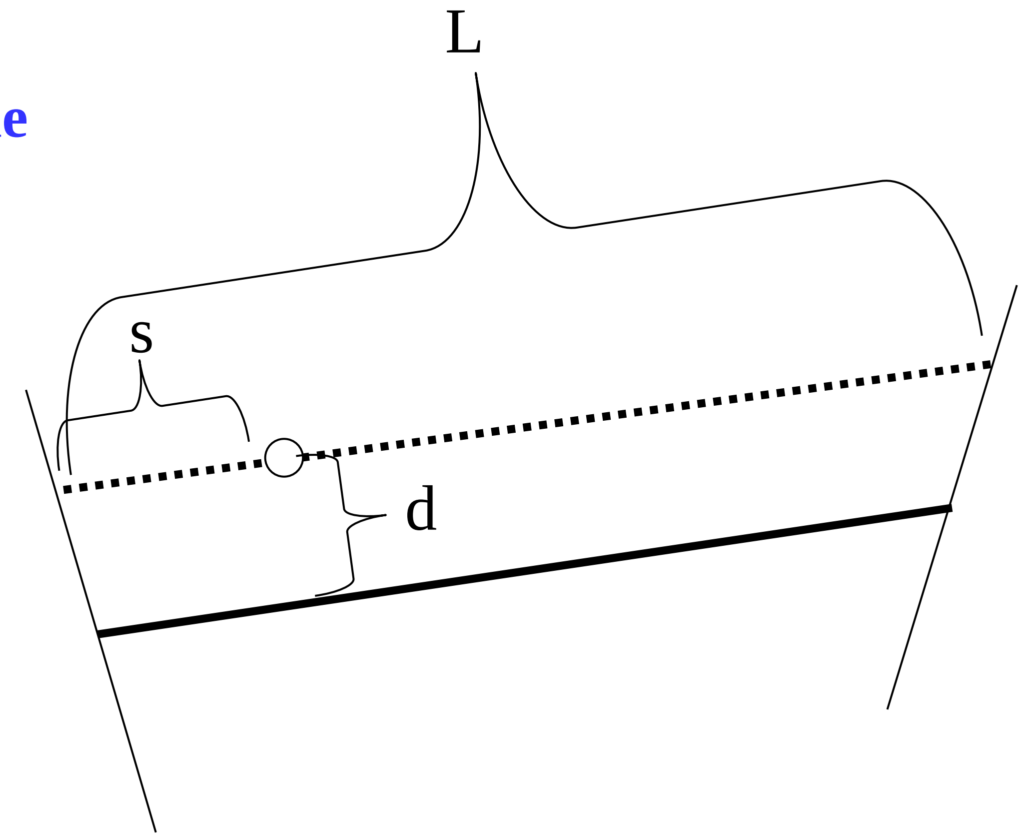
- The skin is deformed using the skeleton
- Apply motion to the bones -> deform the skin accordingly

▪ Has many names
  - Skeletal subspace deformation
  - Linear Blended Skin (LBS)
  - Skinning
  - Enveloping
  - . . .

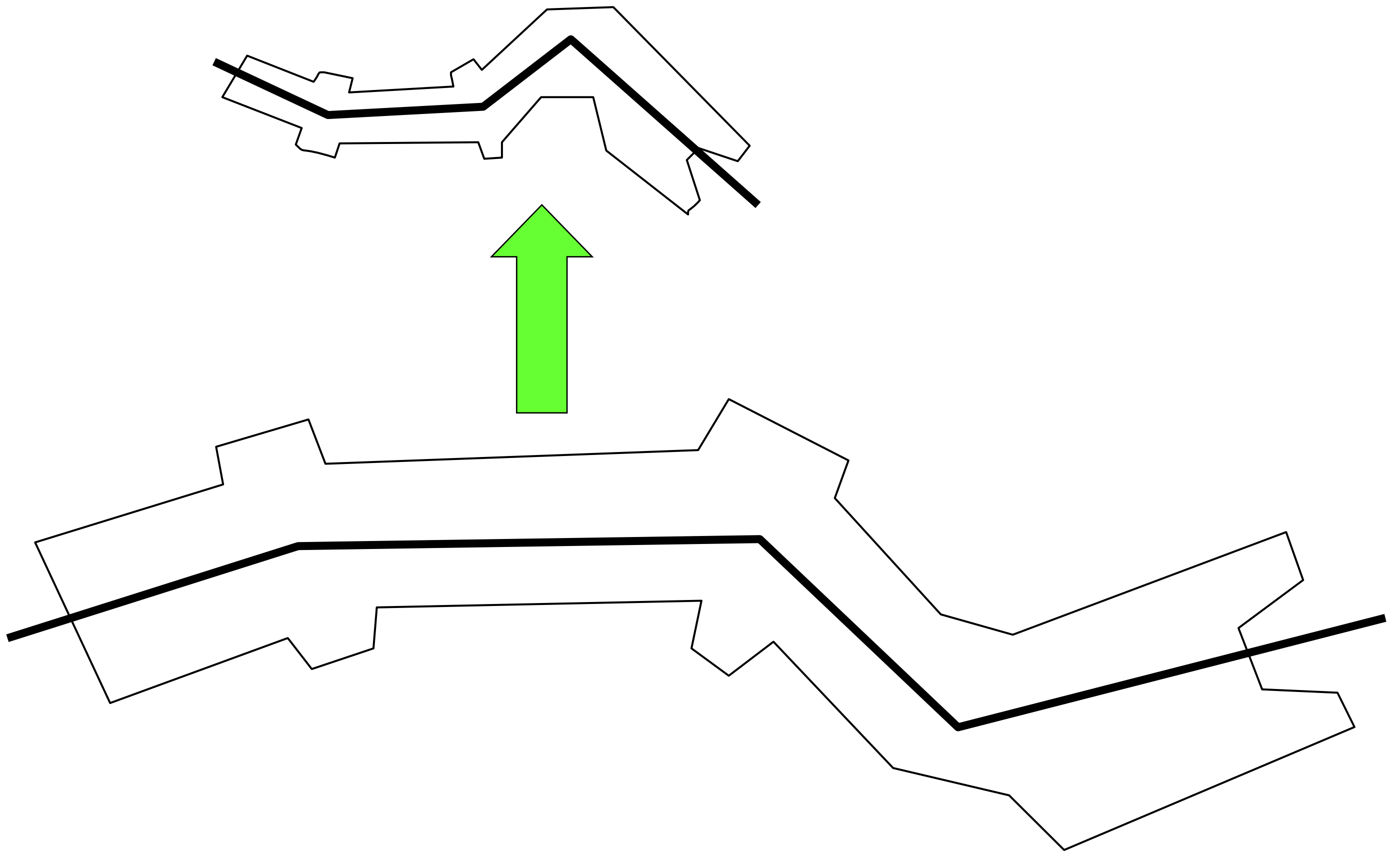


SDD in 2D

- Get object
- Draw polyline
- Map vertices to polyline
- Warp polyline
- Reposition vertices

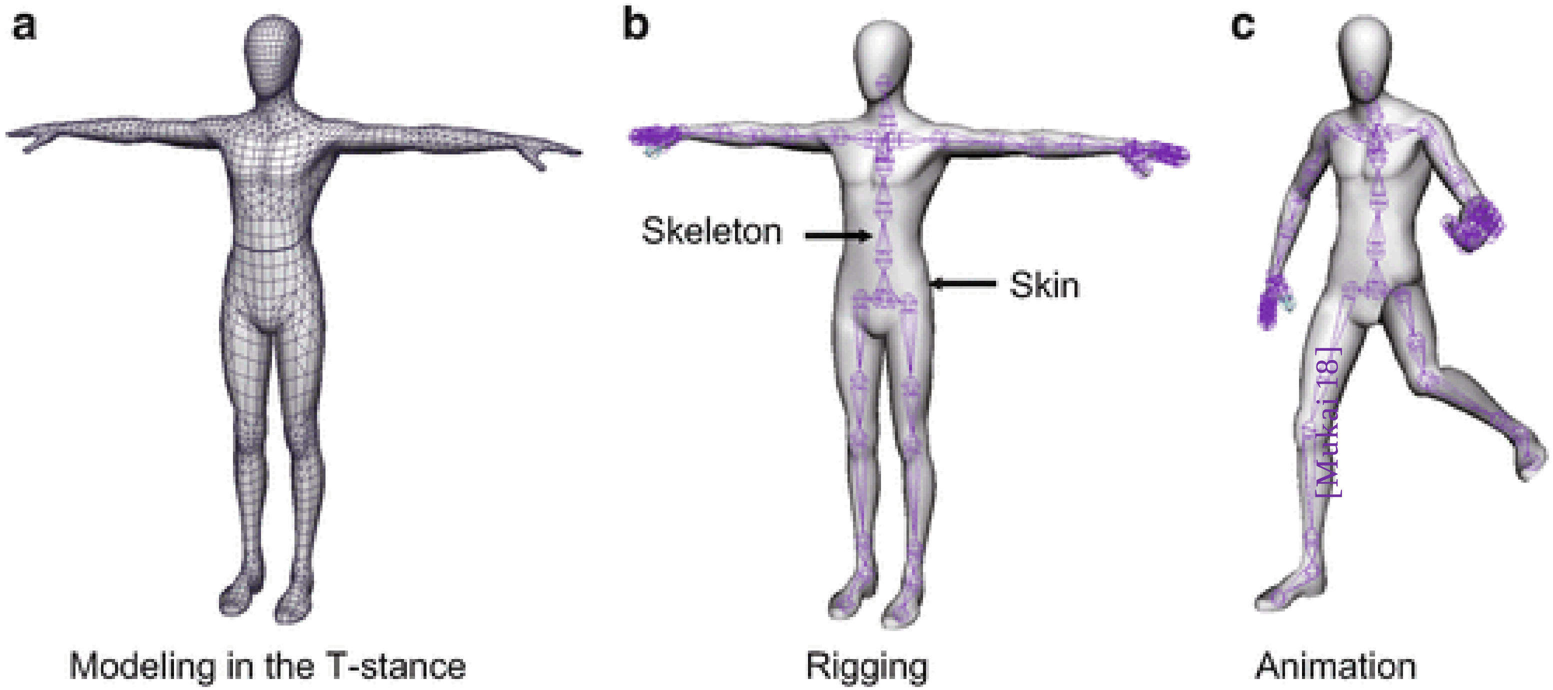



SDD in 2D



Linear Blend Skinning

$$\mathcal{W}(\boldsymbol{\theta}; M, \boldsymbol{J}, \boldsymbol{W})$$



Get the skin surface M.

Define the skeleton $\boldsymbol{J}$.

Map vertices to the skeleton: $\boldsymbol{W}$

Apply rotations $\boldsymbol{\theta}$ to the skeleton.

Reposition vertices ($\mathcal{W}$).

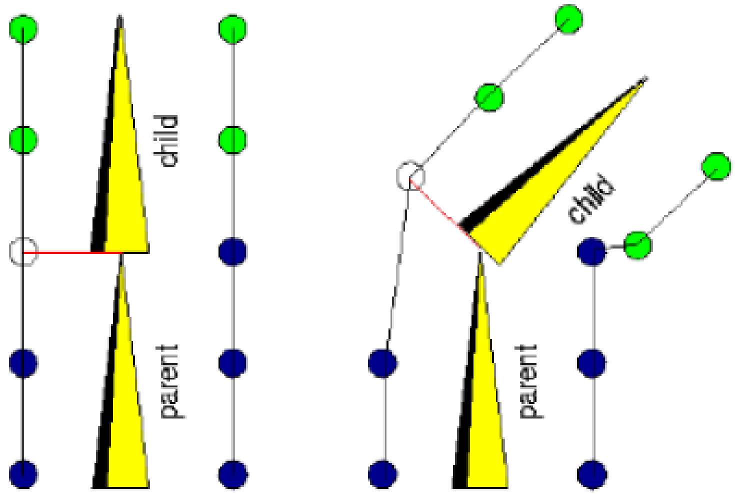
[MTT91] Magnenat-Thalmann N., Thalmann D., "Human Body Deformations Using Joint-dependent Local Operators and Finite-Element Theory", Making Them Move, N.Badler, B.A.Barsky, D.Zeltzer, eds, Morgan Kaufmann, San Mateo, California, pp.243-262, 1991.

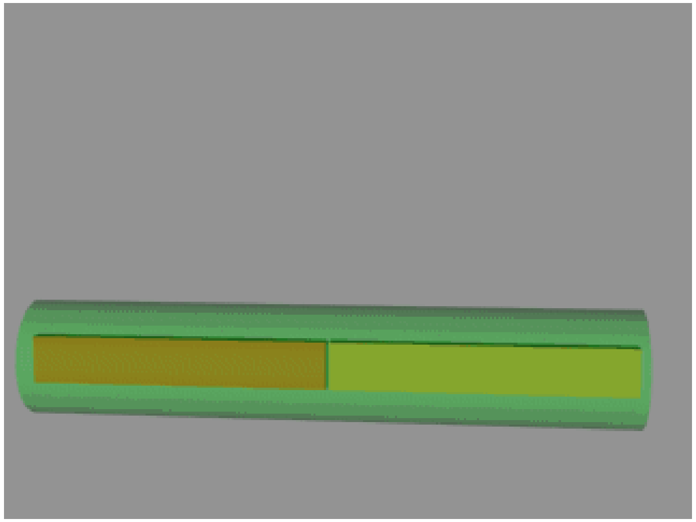
SDD, Rigid Parts

- For models composed of rigid parts such as robots.
- Each vertex is attached to a single joint.
- Every vertex is transformed by exactly one matrix

$$\mathbf{v}' = \mathbf{W} \cdot \mathbf{v}$$

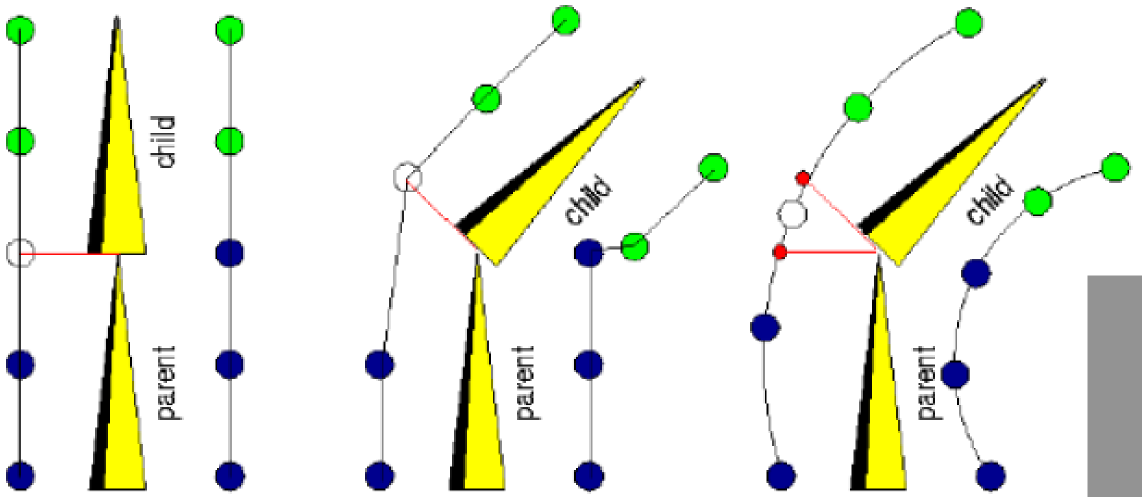
where v is defined in joint's local coordinate system

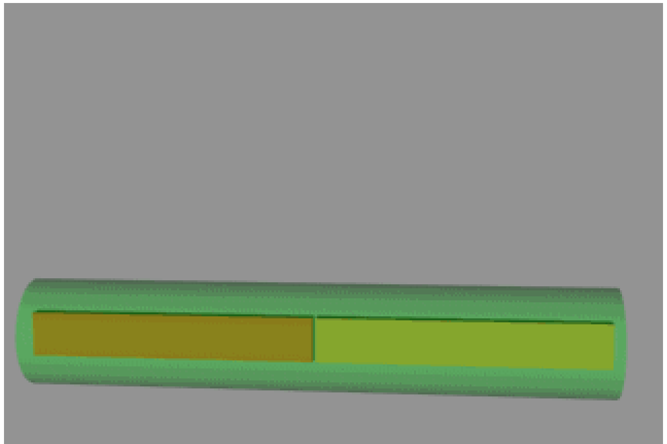




SDD, Smooth Skin Algorithm

- **A vertex can be attached to more than one joint with adjustable weights**
- **Weights define the contribution of the joints on the vertices.**
- **Mainly used for video games**





http://www.flipcode.com

SDD, Smooth Skin Algorithm

- **The deformed vertex position is a weighted sum:**

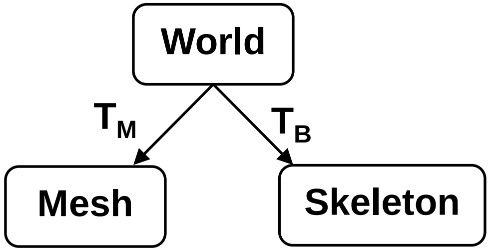
$$\mathbf{v}' = w_1(\mathbf{M}_1 \cdot \mathbf{v}) + w_2(\mathbf{M}_2 \cdot \mathbf{v}) + \ldots w_N(\mathbf{M}_N \cdot \mathbf{v}) \quad \text{with} \sum w_i = 1$$

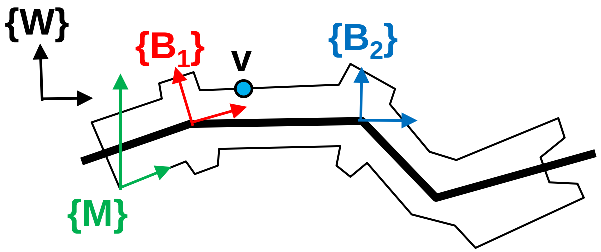
$$\mathbf{v}' = \sum w_i(\mathbf{M}_i \cdot \mathbf{v})$$

- **Note that the blending is made in one common space: world!**

- **Initially, we have $\mathbf{v}^M$: coord. in the mesh CS**

SDD procedure

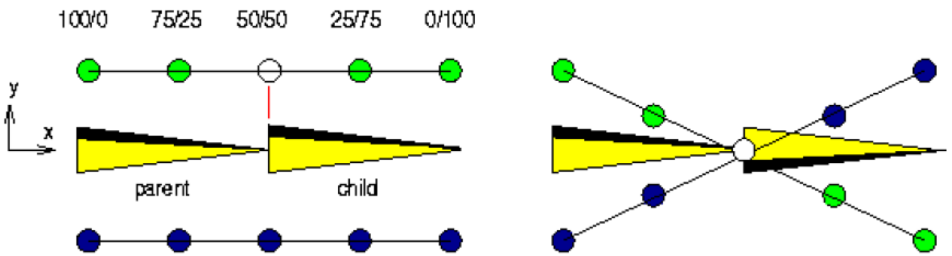


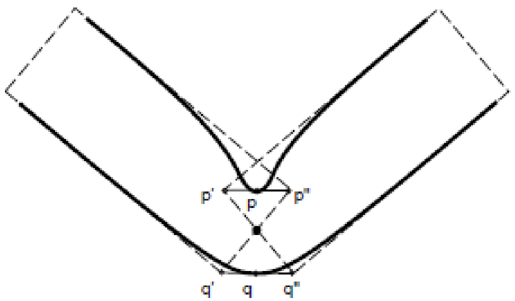


- **Get the skin surface: v_M is given**
- **Define the skeleton**
- **Map vertices to the skeleton:**
  - **Determine the influencing bones & associated** w_i & **v_i (i=1,…) in the LC of bone *i's*.**
  - **We need to bring v from $MC(v_M)$** (to **WC** and) **to $BC(v_i)$**
- **Change the pose of (apply rotations to) the skeleton: set M_i**
- **Reposition vertices: $BC(v_i)$->$WC(v_W)$**

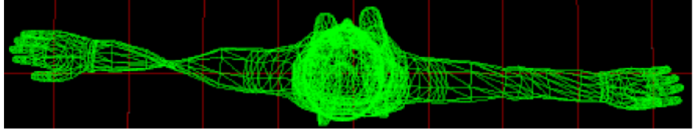
Pros and cons of SDD

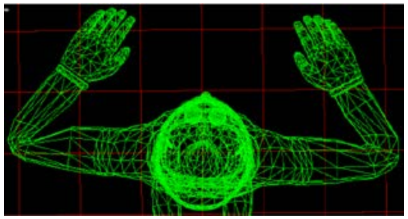
- Pros: Simple and fast
- Cons:
  - Leads to unnatural deformations to certain poses





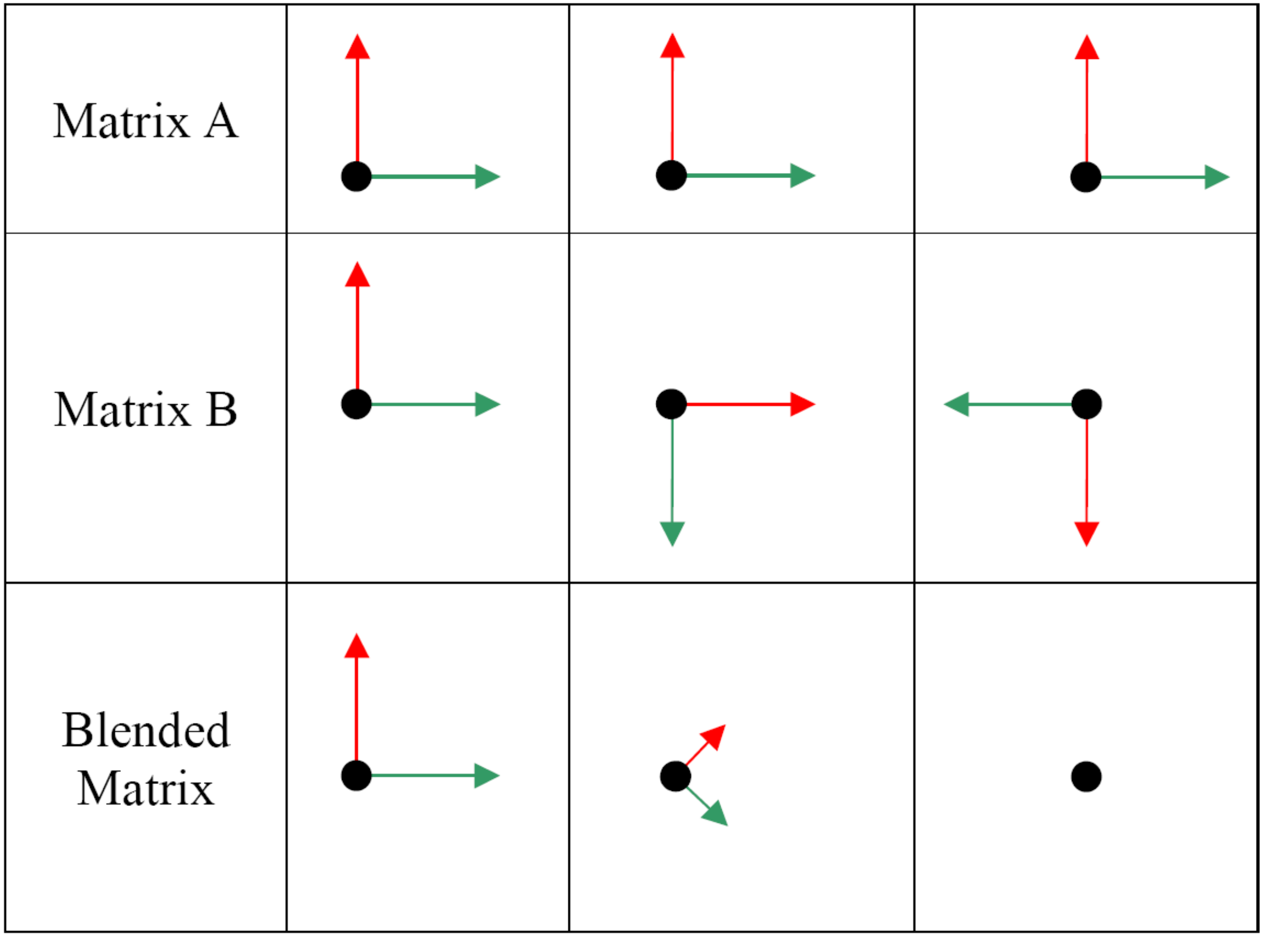
  - Difficult to get specific control
  - Unintuitive and difficult to edit
- Still, used in many 3D animation packages



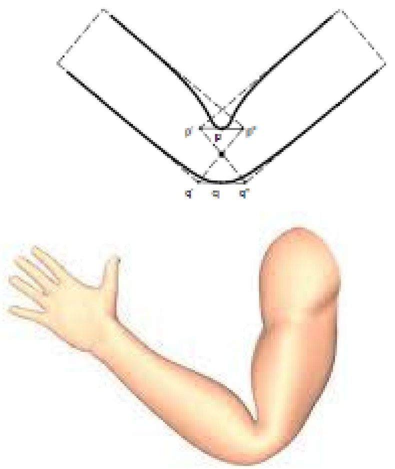


Limitations of SDD

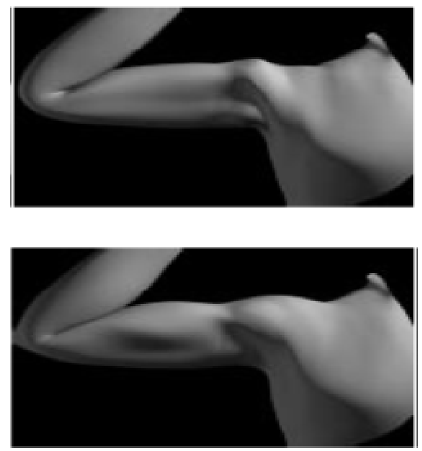
Linear combination of matrices:



Limitations of SDD



[Yang & Zhang 06]



[Lewis et al 06]

[Romero et al 20]

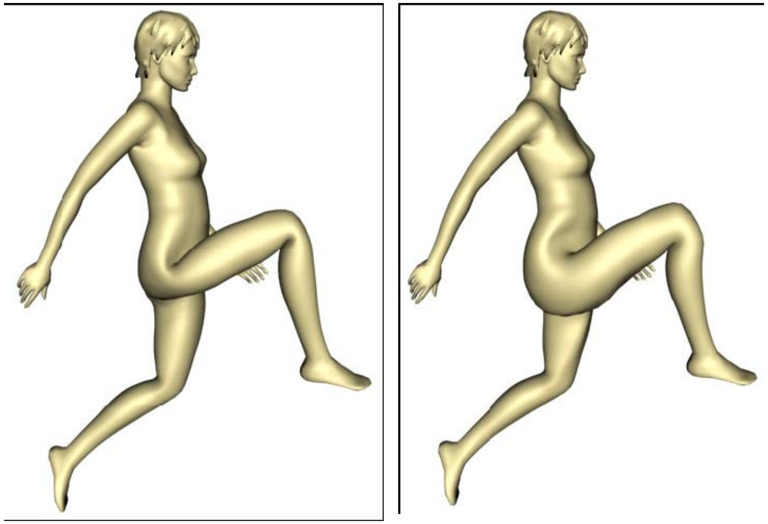
Unnatural deformations at certain poses

Impossible to express nonlinear deformation i.e. muscle bulging

Impossible to simulate skin dynamics i.e. jiggle effect

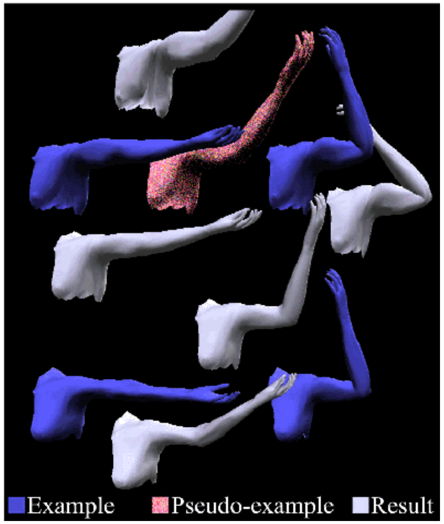
Several solutions

Geometric



[Magnenat-Thalmann et al. 04]

Example-based

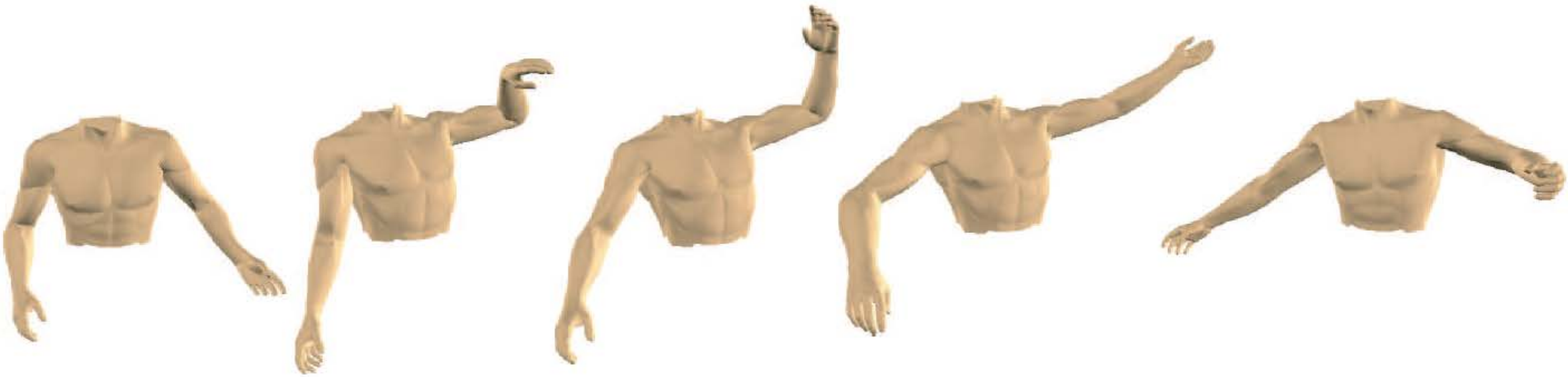


[Sloan et al. 01]

Physics-based

[Ziva Dynamics]

Improvements of SDD – pseudo joints



- **Users sculpt desired “*example*” shapes on several poses**
- **Pseudo joints and their influence weights are computed s.t. when linear blending is applied including pseudo joints, the result shapes are as close as possible to the examples**

Improvements of SDD – better matrix blending

- **Blending transformation matrices is a problematic operation**
  - **Especially for the rotation matrices**

Can’t interpolate rotation matrices

$$\begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

90° z-axis

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

-90° z-axis

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

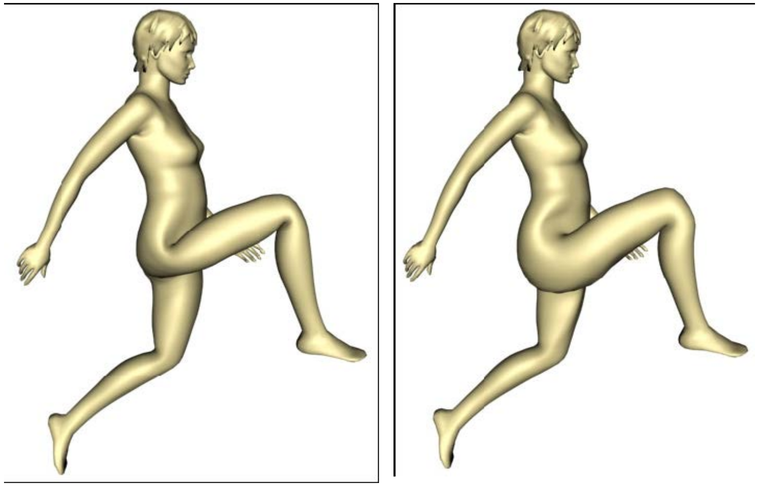
??

Improvements of SDD – better matrix blending

- **Quaternion interpolation for rotation**
  - **Decompose the matrix to translation and rotation**
  - **Interpolate them separately, with rotation represented by *quaternions***

- **Matrix exponent***
  - **Carry out matrix linear blending in the exponent space**

$$\bigoplus_i w_i \cdot M_i = e^{\sum_i w_i \log(M_i)}$$



* M. Alexa, "Linear Combination of Transformations", SIGGRAPH 2002 Conference Proceedings, Annual Conference Series, published by ACM SIGGRAPH Addison Wesley, 21(3), pp. 380-387, July 2002

Improvements of SDD – example based methods

- Users sculpt desired "*example*" shapes on several poses
- Displacements of each vertex compared to SDD are computed and saved
- Displacements for "inbetween" poses are computed via weighted blending of example shapes

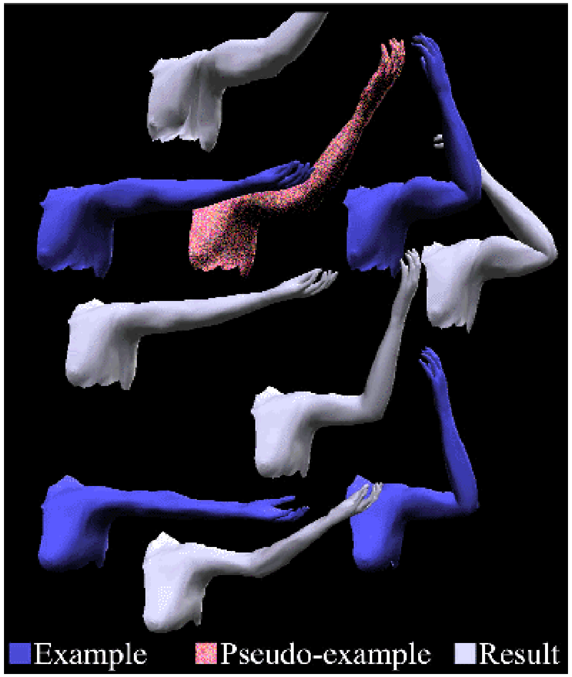
[Lewis et al, 00]

Lewis J.P., Cordner M., Fong N., "Pose Space Deformations: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation", Proc. SIGGRAPH 2000, Addison-Wesley, pp.165-172, 2000.

Pose space deformation: example-based method

- Nonlinear skin deformation component as scattered data interpolation
  - Examples
    - $X_i = \{\boxed{x_i}, \boxed{\theta_i}\}$ vertex position that comprises skin surface; pose of the character
  - PS-Deformation
    - Compute $g(\theta; \{\theta_i\})$
      - ✓ scattered data interpolation
    - Add $g(\theta)$ to $SDD(\theta)$

      nonlinear linear
- Pros vs. cons
  - Realistic
  - Data dependent, preprocessing



FK example: Motion capture

Controversy

Digital art:
Hand-Animated

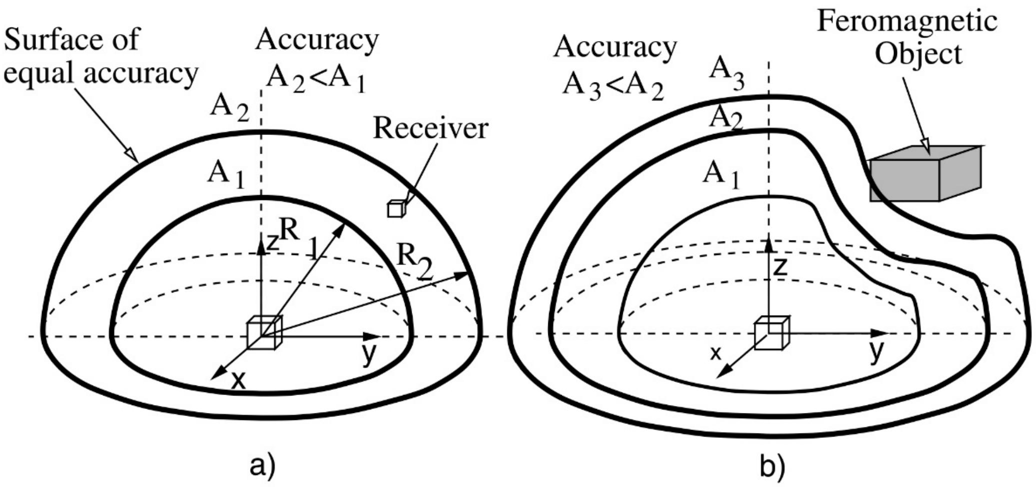
Performance-Capture:
Mocap / Rotoscoping

Pixar

Electronic Arts

Motion capture – magnetic systems

- **Sensitive to metal**
- **Low frequency (60Hz)**



Motion capture – Mechanical Systems

- Any environment
- Measures joint angles
- Restricts the motion

Motion capture – Optical systems

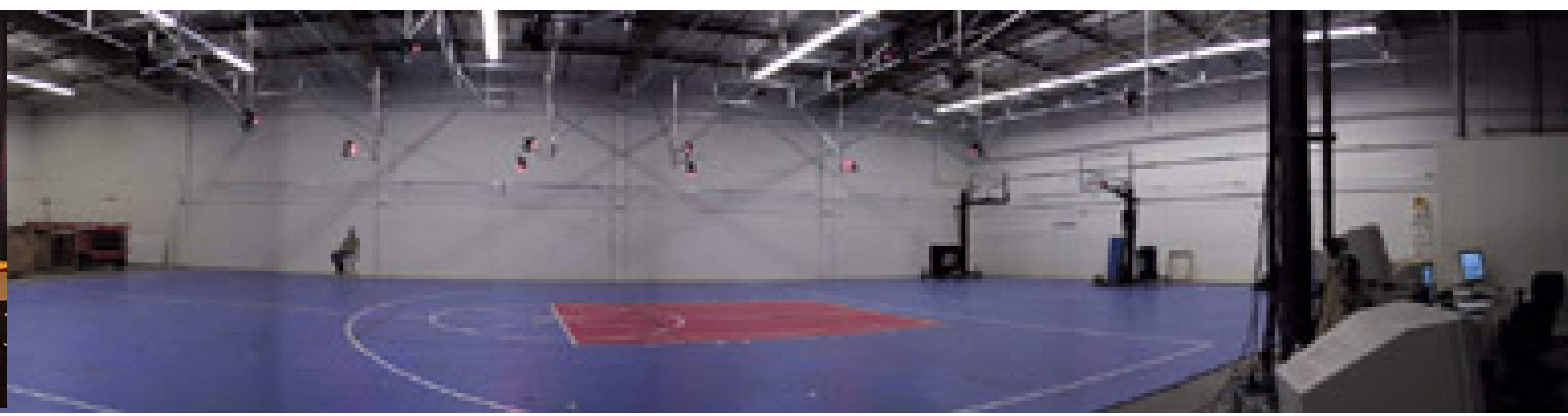
- Place markers on the actor
- Cameras can determine marker positions

- 8 or more cameras
- Restricted volume
- High Frequency (240Hz)
- Occlusions

Optical motion capture : system setup

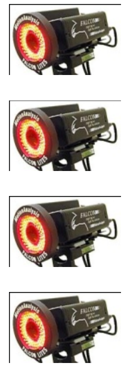






Optical motion capture : Standard Pipeline

1



Multi-View High-Speed Recording

2

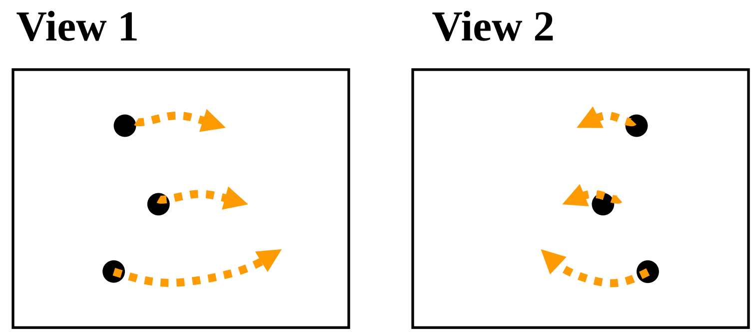
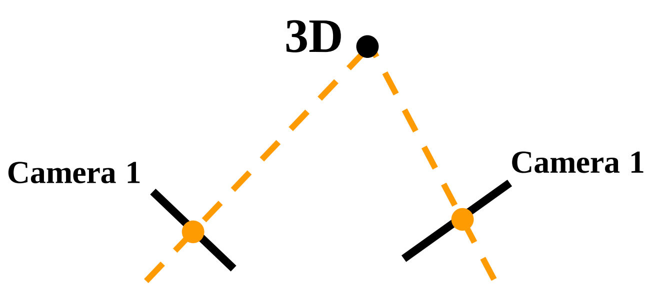


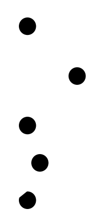
Image Feature Tracking

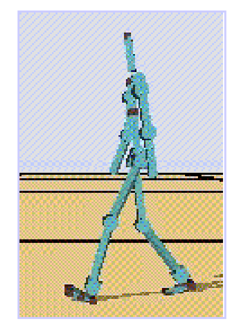
3



2D -> 3D Reconstruction

4





Kinematic Model Fitting

Optical motion capture process

1. Find the skeleton dimensions and exact marker positions on the body

2. Perform a motion trial

3. Compute marker positions from camera images

4. Identify and uniquely label markers

5. Calculate joint angles from maker paths

Process: User’s perspective

Calibration

Measure subject's bones.

Measure marker offsets.

Build skeleton model.

Capture

Capture desired motions.

Capture extra motions.

Generate 3D points.

Reconstruction

Connect marker paths.

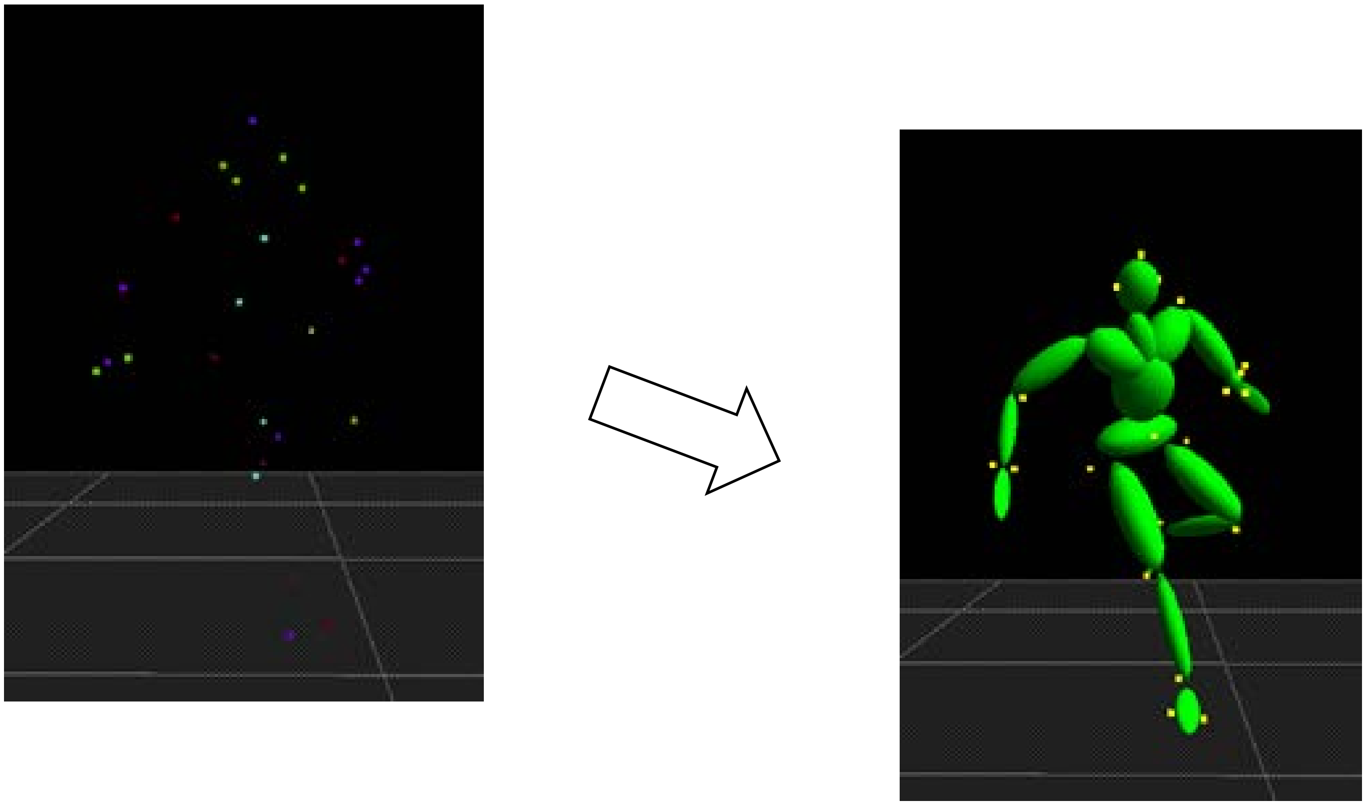
Label markers.

Refine skeleton model.

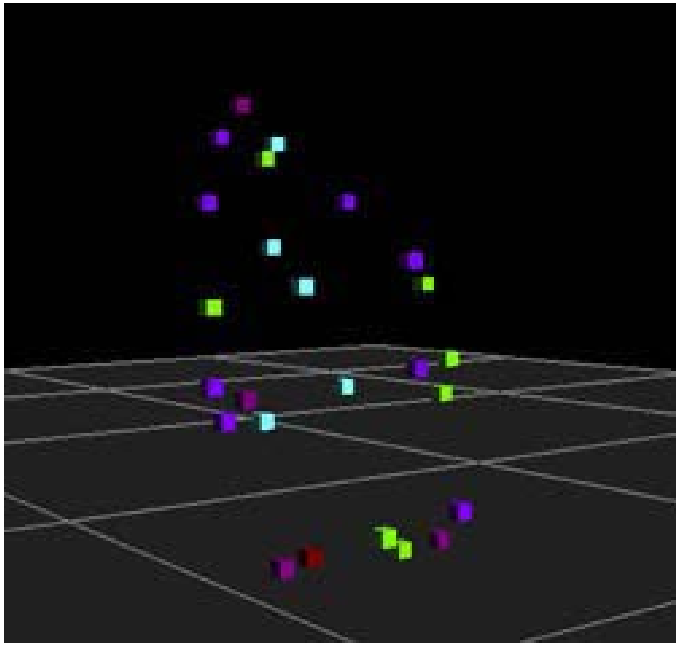
Produce joint angles w/ IK.

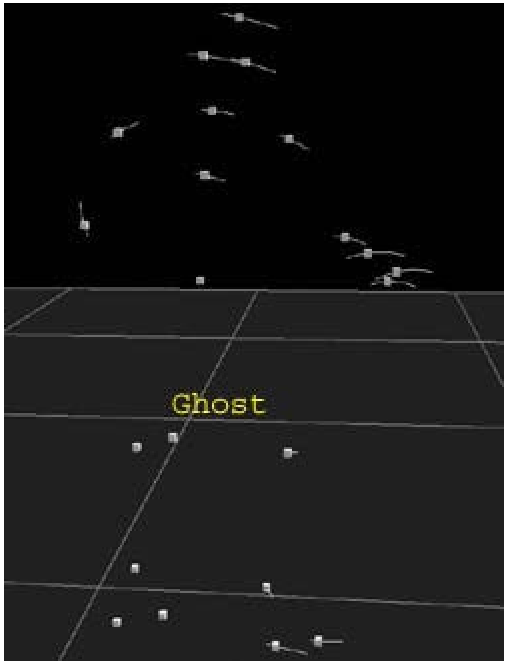
Marker Identification

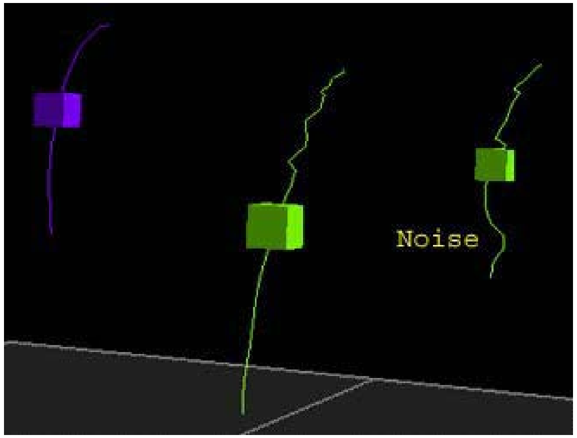
- **At each frame, motion capture gives us a set of points**



Marker Identification







- **Making sense of raw data...**

Optical motion capture process

1. Find the skeleton dimensions and exact marker positions on the body

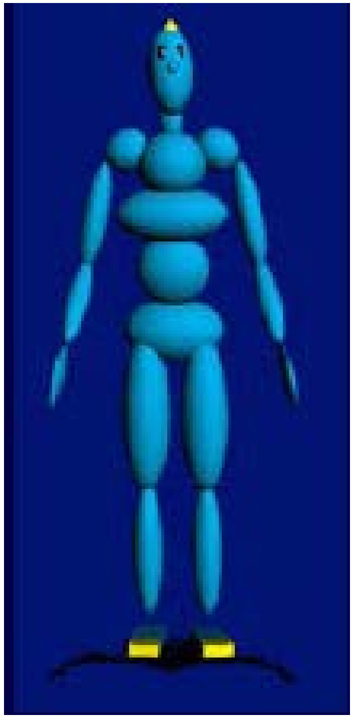
2. Perform a motion trial

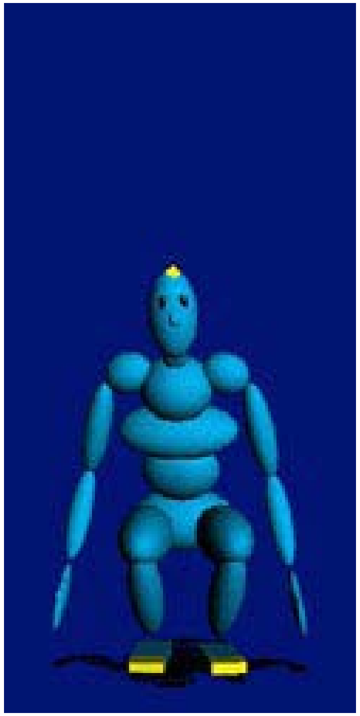
3. Compute marker positions from camera images

4. Identify and uniquely label markers

5. <u>Calculate joint angles from maker paths</u>

We will be introduced Inverse kinematics soon…





1. Create a handle on body
   - position or orientation

2. Pull on the handle

3. IK figures out how joint angles should change

Mocap data

- **.csm contains:**

```
$firstframe 1
$lastframe 3
$spinelinks 3
$rate 60
$order
C7 CLAV LANK LBHD LBWT LELB LFHD LFIN LFWT LKNE LSHO LTOE LUPA LWRA LWRB RANK
RBHD RBWT RELB RFHD RFIN RFWT RKNE RSHO RTHI RTOE RWRA RWRB STRN T10
$points
1 980.6 -2365.8 1541.3 1030.6 -2239.9 1492.1 967.3 -2427.5 181.8 936.2 -
2330.1 1672.9 939.4 -2359.1 1109.3 782.9 -2263.3 1175.7 959.0 -2196.9 1753.8
762.3 -2135.1 862.6 949.3 -2187.2 1082.1 934.2 -2343.2 583.1 870.4 -2246.7
1525.4 1031.7 -2339.7 83.1 814.6 -2218.7 1318.8 799.4 -2132.3 993.7 729.2 -
2230.2 966.5 1110.1 -2060.1 197.3 1066.8 -2351.6 1670.5 1114.0 -2391.4 1116.4
1290.5 -2574.6 1396.5 1105.7 -2231.2 1740.1 1217.5 -2369.8 1149.5 1159.8 -
2233.6 1093.8 1138.8 -2119.2 605.9 1136.6 -2342.8 1564.9 1081.3 -2126.7 775.0
1065.2 -1944.5 112.2 1276.8 -2398.1 1271.2 1294.0 -2490.1 1183.6 1044.2 -
2199.9 1395.9 988.1 -2404.3 1417.4
.
.
3 979.6 -2351.7 1539.3 1029.2 -2224.5 1487.1 966.1 -2426.1 181.6 936.4 -
2313.5 1669.3 942.7 -2343.0 1105.1 784.3 -2236.2 1171.7 959.4 -2180.8 1750.6
773.8 -2090.6 864.4 949.3 -2168.6 1078.7 934.2 -2334.3 582.5 871.3 -2235.2
1521.8 1031.3 …
```

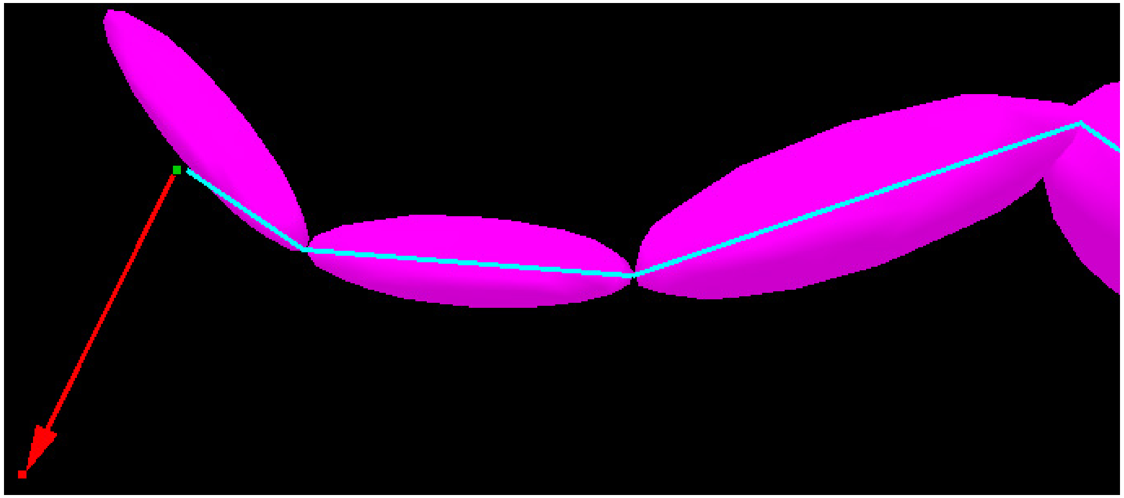
Mocap data

- **.htr contains:**

```
[SegmentNames&Hierarchy]
#CHILD      PARENT
LOWERTORSO         GLOBAL
UPPERTORSO         LOWERTORSO
NECK        UPPERTORSO
.
.
[BasePosition]
#SegmentName       Tx, Ty, Tz, Rx, Ry, Rz, BoneLength
LOWERTORSO         0.00     0.00     0.00     0.00     0.00     0.00   200.00
UPPERTORSO         0.00   200.00     0.00    -1.38     0.00     0.35   286.95
NECK               0.00   286.95     0.00     2.90    -0.08     3.20   101.66
.
.
[LOWERTORSO]
            1   263.72   816.20  -2874.77    18.03    -7.70   -10.34     1.00
            2   264.42   812.41  -2740.34    19.81   -13.46   -11.93     1.00
[UPPERTORSO]
            1     0.00     0.00     0.00     8.33   -17.38     8.59     1.00
            2     0.00     0.00     0.00     8.71    -6.14     8.64     1.00
[NECK]
            1     0.00     0.00     0.00    -2.14     0.13    -0.01     1.00
            2     0.00     0.00     0.00    -4.40     0.27    -0.02     1.00
.
.
```

Inverse Kinematics

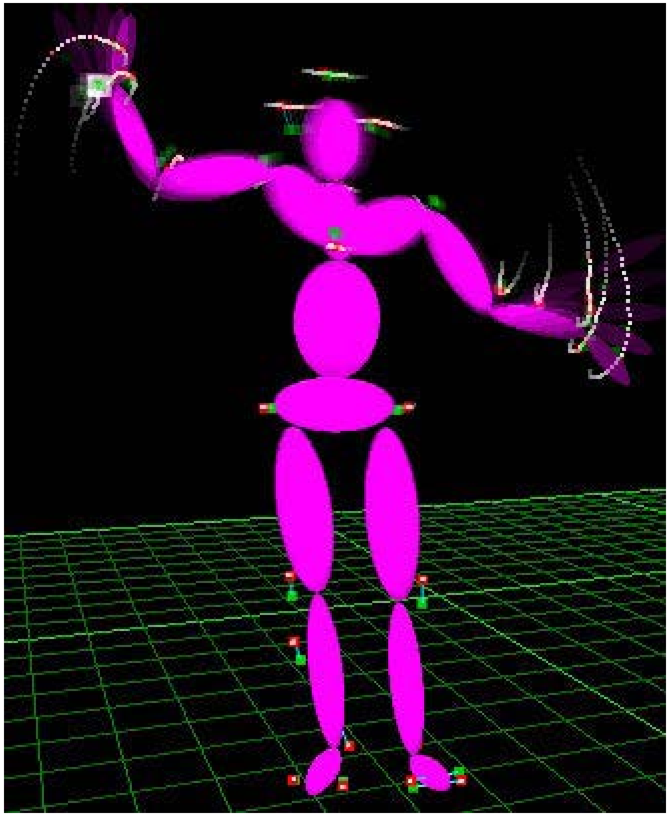
IK problem Definition



- ***Inputs:***
  - **An articulated skeleton with handles.**
  - **Desired positions for handles.**

- ***Outputs:***
  - **Joint angles that move handles to desired positions.**

Inverse Kinematics (cont'd)



- We are solving IK on a complex model (~50 DOFs and 30 handles).
- Motion capture data often contains missing markers.
- Many different formulations for IK problem, would like to use one that is best for motion capture data.

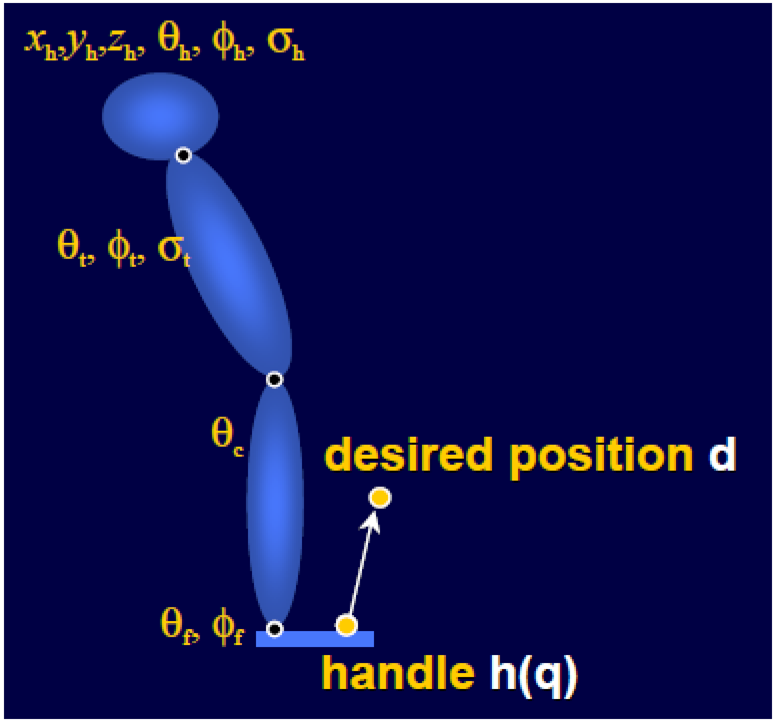
More Formally

- **Let:**
  - **q actor *state vector***
  - **$C(q)$ constraint functions that pull handles**

- **Then:**
  - **solve for q such that $C(q) = 0$**

What's a Constraint?

- **q=[** $x_h, y_h, z_h, \theta_h, \varphi_h, \sigma_h, \theta_t, \varphi_t, \sigma_t, \theta_c, \theta_f, \varphi_f$ **]**



- Can be rich, complicated
- But most common is very simple:
- **Position constraint** just sets difference of two vectors to zero:

  C(q) = h(q) - d = 0

Data-driven human modeling

3D image capture technology - Why?

- Systematic observation of human bodies.
  - Example: CAESAR (Civilian American and European Sur Anthropometry Resource) project

  - 3D whole body scanners (c.f. tape measure)
    - extract body measurements
    - 3D Anthropometric database

  - Advantages:
    - Accuracy
    - Speed
    - Comfort

Data-driven modeling

[Allen et al,03]

- **Scan *examples***
  - **Captured geometry of real people provides the best available resource**
    - **variation, commonality**
    - **Realistic estimation**
- **Predetermined topology**
  - **Vector representation**
    - **At a desired level**
  - **Reuse of the skinning data**
  - **Easily handle scan bodies of different postures**
    - **joint center estimation**
- **Interpolation**
  - **continuous transformation field**

3D Scanned Data

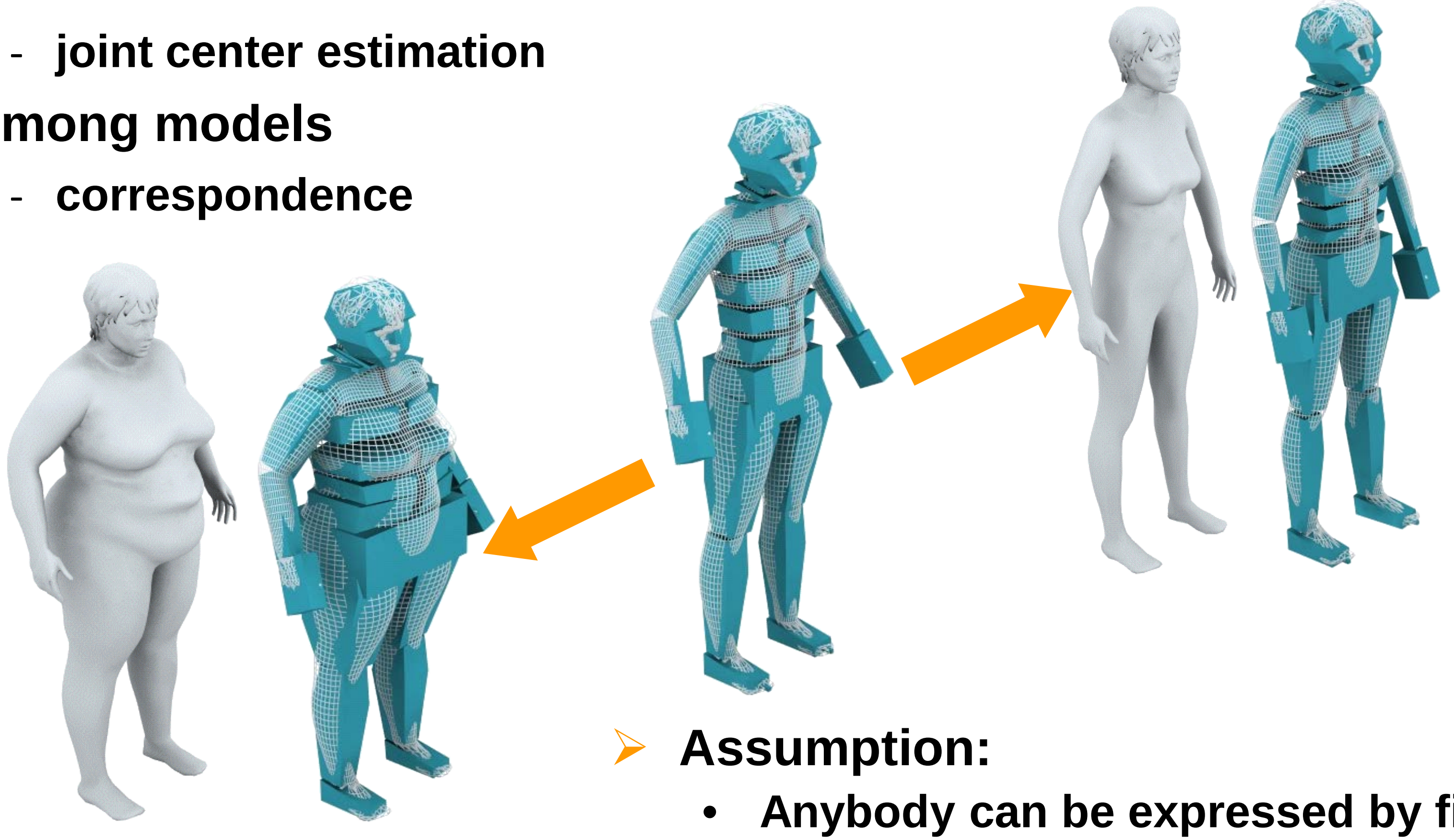
- 100 subjects (European adults)
  - Techmath,AG range scanner
  - Erect posture with arms and legs apart, lightly clothed
  - Without faces, no texture
- Additional processing using commercial packages
  - One single mesh with no holes and no open edges
  - Moderate complexity (# of triangles: <=75,000)

Template Model

- **Skeleton**
  - **H-Anim standard**
  - **LoA 2, 33 joints excluding hands and feet**
- **Template mesh**
  - **Grid structure**
    - **Bezier patches**
  - **Two levels of detail**
    - **861 and 3401 vertices**
- **Skinning setup**
  - **Using 'BonesPro'** **[www.digimation.com]**

Preprocessing

- Within a model
  - joint center estimation
- Among models
  - correspondence



- Assumption:
  - Anybody can be expressed by finding an appropriate deformation

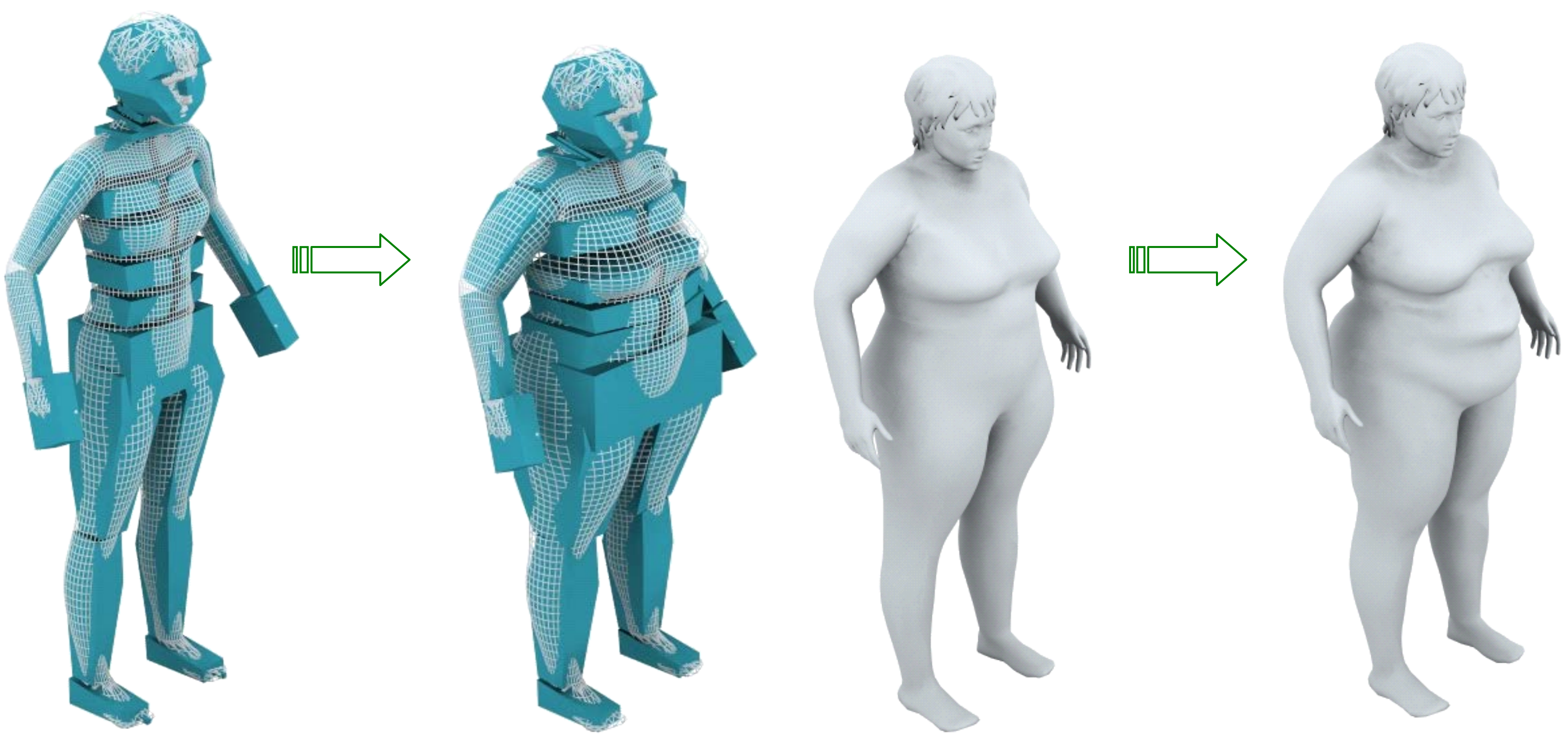
Representation

Joint configuration

J = $([tx_1, ty_1, tz_1, sx_1, sy_1, sz_1], tx_2, \ldots, sz_m) \in R^{6m}$

Displacement map

D = $([dx_1, dy_1, dz_1], dx_2, \ldots, dz_n) \in R^{3n}$



Template model **Skeleton adjustment** **Fine refinement**

PCA for shape parameters

- Remove redundancy
- Interpolate PC space rather than the original vector space
- Principal component analysis
  - One common technique to reduce the data dimensionality
- Given D = $\{x_1,..,x_N\}$, each of dimension d, we want to find the directions $e_1,..,e_M$ (M<<N) s.t.

  $x' = m + c_1 \cdot e_1 + c_2 \cdot e_2 + c_M \cdot e_M$ is the best

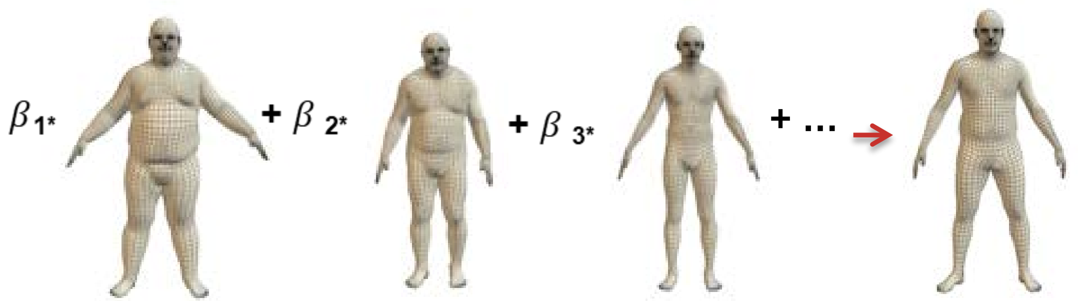
  - The best c_j should be projection x_i to e_j.
  - It is common to use $(c_1,..,c_M)$ to represent x.
- M=30
  - 30 joint interpolators, 30 displacement interpolators

Data-driven methods

- **Generalization of example-based methods**
- **Common strategy: Learn the model from a dataset!**
  - **Subspace, manifold, latent space**
  - **Captures shape (&texture) variations with a set of basis**
  - **morphable-, statistical-, parametric-, linear-model…**

Data-driven body shape modelers

- **Static shape modeler** [SMT03, ASK+05]
  - A new model is generated by a vector of blending weights
  - The solution space becomes constrained, solvable by common optimization techniques



$$M(\vec{\boldsymbol{\beta}}) = \bar{\boldsymbol{T}} + \sum \beta_n \boldsymbol{S_n}$$

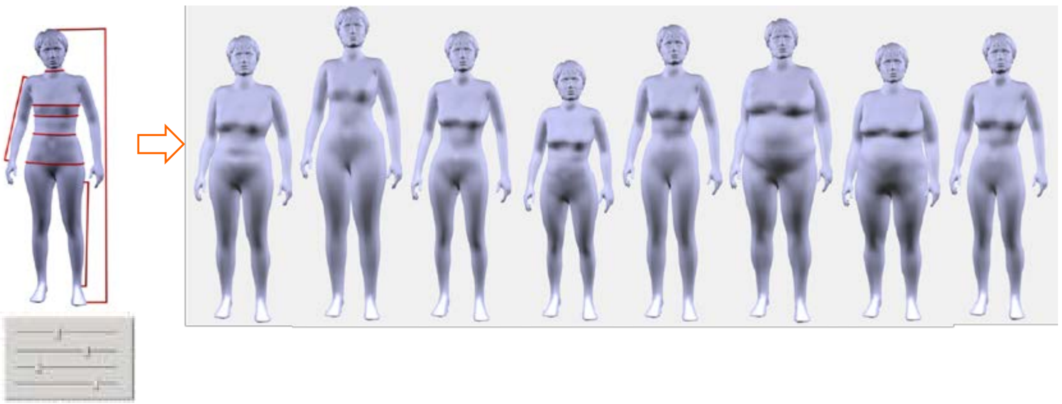
Basis shape vectors

[SMT03] Seo H., and Magnenat-Thalmann N., “An Automatic Modeling of Human Bodies from Sizing Parameters”, ACM SIGGRAPH 2003 Symposium on Interactive 3D Graphics (April), pp.19-26, Monterey, USA, 2003.

[ASK+05] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis J., SCAPE: Shape Completion and Animation of People. ACM Trans. Graph. (Proc. SIGGRAPH 24, 3, 408–416) 2005.

Data-driven body shape modelers

- **Controllability**
  - **Mapping function**



[Seo et al 04, SIGGRAPH I3D 2003]

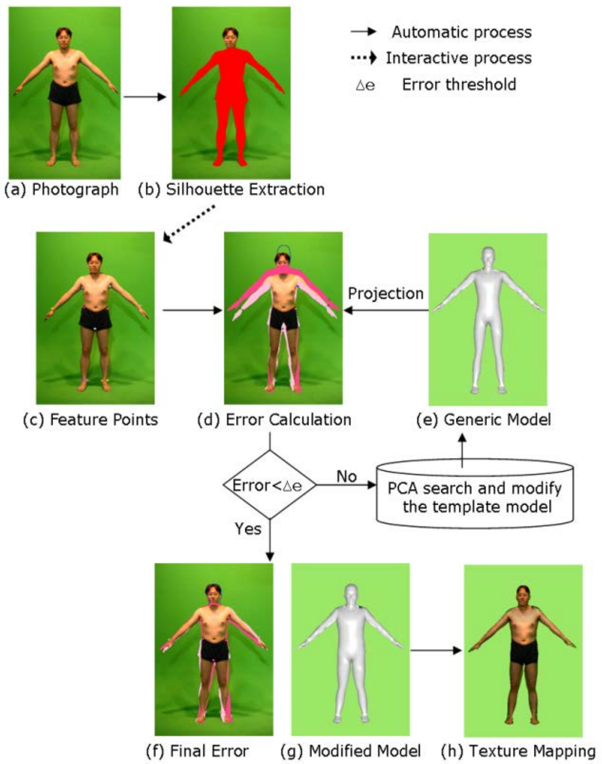
  - Typically, a function is learned from the dataset.
  - It maps attributes to weights (parameters)

$$\mathsf{F}(\mathbf{c}) = \beta$$

Model-based reconstruction

- **Shape recovery by parameter searching**

- How to:
  – Reconstruct 3D body shapes from image input?
- Method: shape by search
  – Find $\gamma=(\gamma_1, \gamma_2, \dots \gamma_{30})$ and $\delta=(\delta_1, \delta_2,\dots, \delta_{30})$ s.t.
  $E(\gamma,\delta)=\alpha \underline{E_d}+(1-\alpha)\underline{E_a}$ is minimized.
  Feature point error   Silhouette error
  – Search -> deform -> project -> error measure
- Contribution
  – Quality shape
  – Robustness: noises or missing views

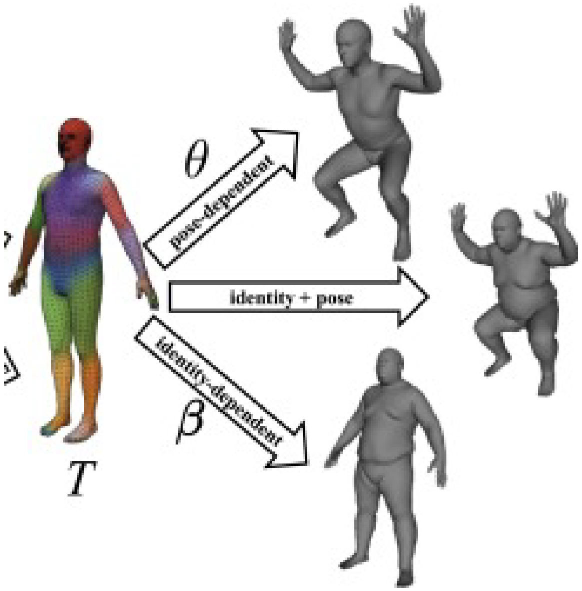


[Seo et al 06, LNCS 2006]

Data-driven human modeling

- **A unifying framework for subject- & pose-dependent shapes**

[HLRB12,LMRP+15]



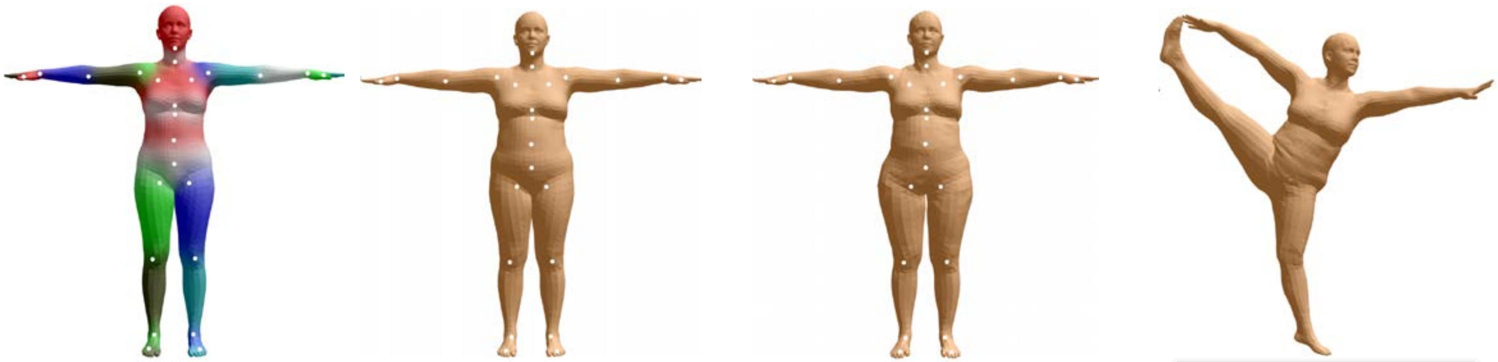
$$M(\boldsymbol{\beta},\boldsymbol{\theta}) = \mathcal{W}(\boldsymbol{\theta},\bar{M}(\boldsymbol{\beta},\boldsymbol{\theta}),J(\boldsymbol{\beta});\,\boldsymbol{W})$$

[HLRB12] D. Hirshberg, M. Loper, E. Rachlin, and M. Black, Coregistration: Simultaneous alignment and modeling of articulated 3D shape. In European Conf. on Computer Vision (ECCV), LNCS 7577, 2012.

[LMRP+15] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. SMPL: A Skinned Multi-Person Linear Model. ACM Trans. Graphics (Proc. SIGGRAPH Asia), 2015.

SMPL: A Skinned Multi-Person Linear Model

[LMRP+15]



(a) $\bar{\mathbf{T}}, \mathcal{W}$ (b) $\bar{\mathbf{T}} + B_S(\vec{\beta}), J(\vec{\beta})$ (c) $T_P(\vec{\beta},\vec{\theta}) = \bar{\mathbf{T}} + B_S(\vec{\beta}) + B_P(\vec{\theta})$ (d) $W(T_P(\vec{\beta},\vec{\theta}), J(\vec{\beta}), \vec{\theta}, \mathcal{W})$

$$M(\boldsymbol{\beta},\boldsymbol{\theta}) = \mathcal{W}(\boldsymbol{\theta}, \boxed{\bar{M}(\boldsymbol{\beta},\boldsymbol{\theta})}, J(\boldsymbol{\beta}), \boldsymbol{W})$$

SDD: linear blend skinning

$$\boxed{\bar{M}(\boldsymbol{\beta},\boldsymbol{\theta})} = \bar{\boldsymbol{T}} + M_S(\boldsymbol{\beta}) + M_P(\boldsymbol{\theta})$$

Template model

Shape blend shape

Pose blend shape

$$M_S(\boldsymbol{\beta}) = \boldsymbol{\mu}_S + \sum_{n=1}^{|\boldsymbol{\beta}|} \beta_n \boldsymbol{s}_n$$

$$M_P(\boldsymbol{\theta}) = \sum_{n=1}^{9K} (R_n(\boldsymbol{\theta}) - R_n(\boldsymbol{\theta}^0))\, \boldsymbol{P}_n$$

SMPL model-based methods

- **Reconstruction & tracking**

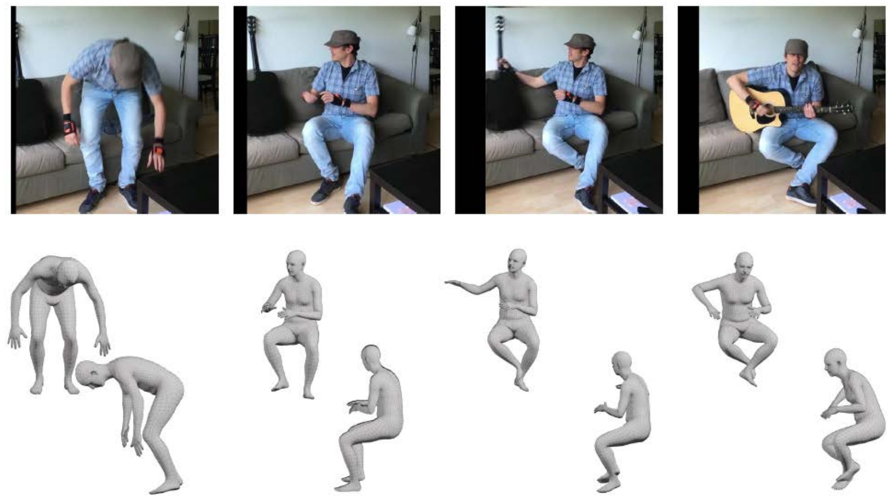
From a single-image

[Bogo et al, ECCV 2016]

$$E(\boldsymbol{\beta}, \boldsymbol{\theta})$$

Find error minimizing model parameters via opti**mization**

From video

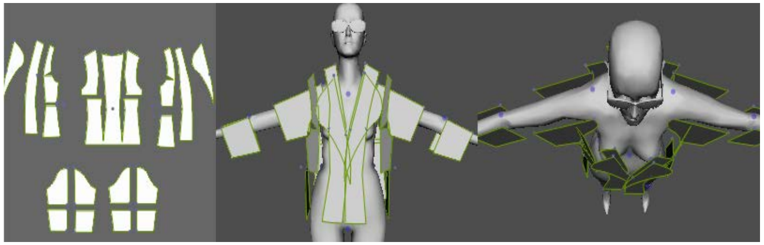


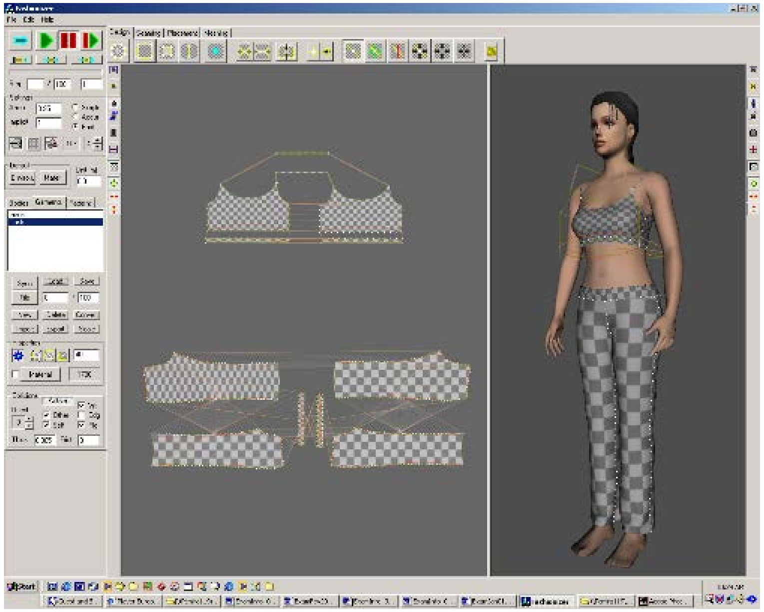
[Kocabas et al, **C**VPR 2020]

Going further: dressing the human model

3D garment design/simulation

- Garment making
  - Automatic garment placement
  - Interactive 2D/3D design

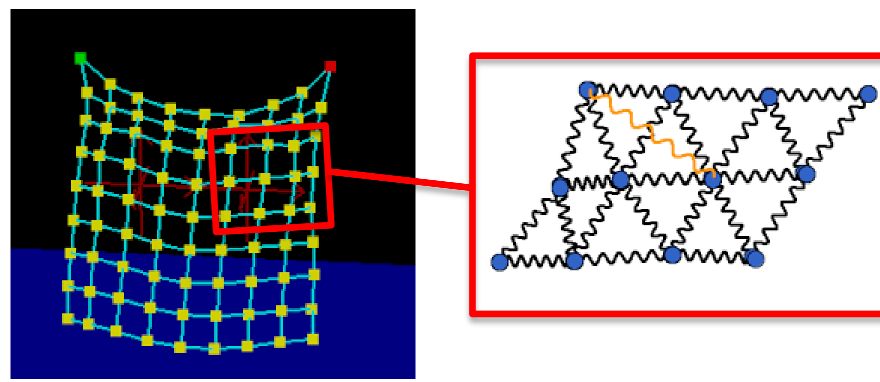




3D garment design/simulation

- **Garment simulation**
  - **Traditionally, the clothes have been modeled by PBS (Physics Based Simulation)**

[Volino et al, **Springer book** 2003]



Reconstruction of clothed humans

- **Learning a cloth model is a real challenge but is likely to be near…**

[Habermann et al '21]

Summary

- **Representation, acquisition of geometric model**
  - Triangle mesh
  - Texture mapping

- **Deformation, shape manipulation for facial & body models**
  - Kinematic model: distance-based function, SDD (LBS)..
  - Reconstruction
  - Example-based: correspondence, scattered data interpolation, ..
  - Data-driven: linear statistical analaysis (PCA),