Total Capture: 3D Deformation Model for Tracking Faces, Hands, and Bodies

Authors:

Hanbyul Joo

Tomas Simon

Yaser Sheikh

Best Student Paper Award

CVPR 2018

Presented by Anantapadmanaabha Prasannakumar

Problem

- Human interactions convey information.
- Information critical for machine understanding.

Image Source: The Panoptic Studio, CMU Robotics Institute

Problem

Marker-based system:

- Laborious process.
- Uncomfortable to the subject.

Image source: Lecture 3, Human Motion Modeling and Analysis, Prof. Yaser Sheikh Lecture Notes
The Robotics Institute, Carnegie Mellon University

Problem

Marker-based system:

- Doesn't capture occlusions.
- Concentrates on particular body part.

Image source: Lecture 3, Human Motion Modeling and Analysis, Prof. Yaser Sheikh Lecture Notes
The Robotics Institute, Carnegie Mellon University

Problem

Markerless systems:

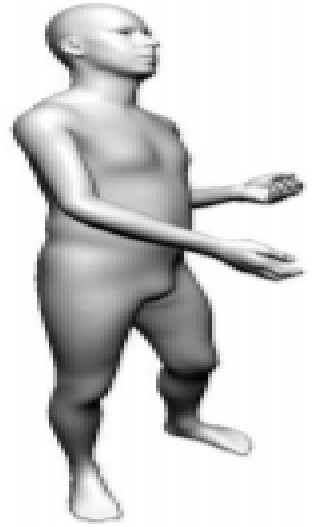
- Use Multi-video stereo camera and a template model.
- Detect keypoints to capture motion.
- Still part specific.

Image source: Markerless Motion Capture of Multiple Characters Using Multiview Image Segmentation, Yebin et al., TPAMI 2013

Proposed Solution

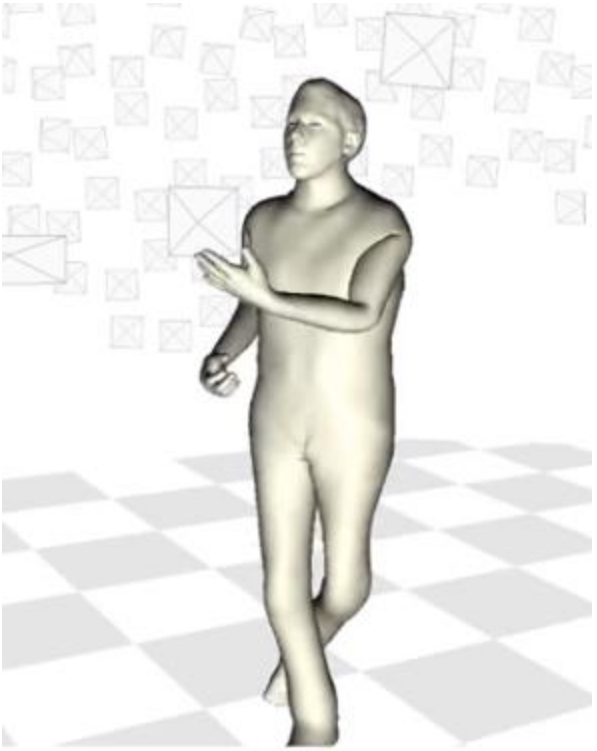
- Create a model for the entire body called Frank.

- Utilize Frank and body keypoints to capture entire human motion.



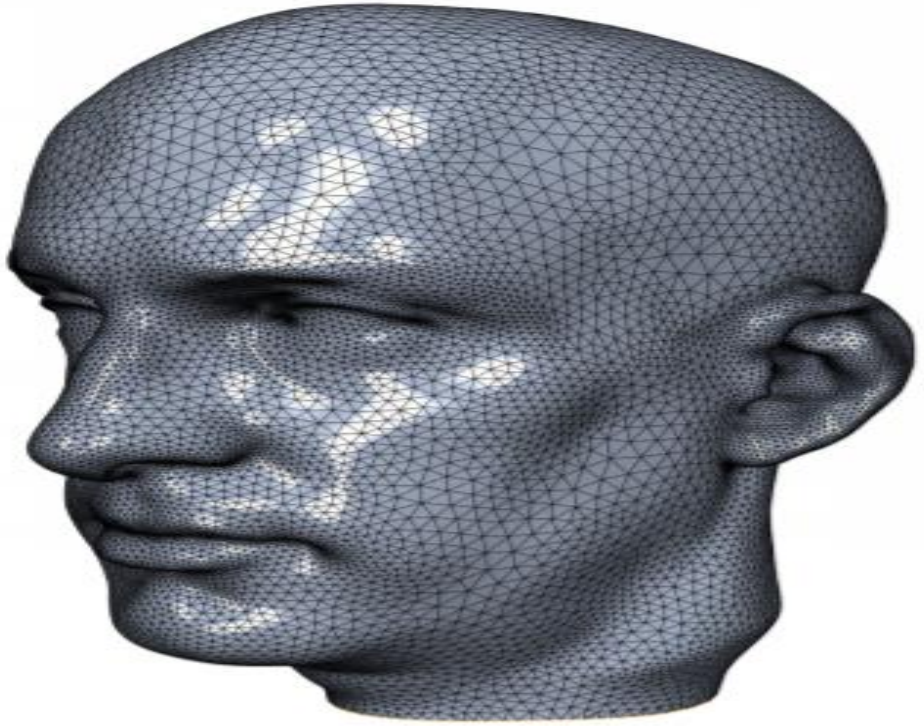
Proposed Solution

- Create a model for the entire body called Frank.
- Utilize Frank and body keypoints to capture entire human motion.
- Create Adam by leveraging Frank.



Building Block

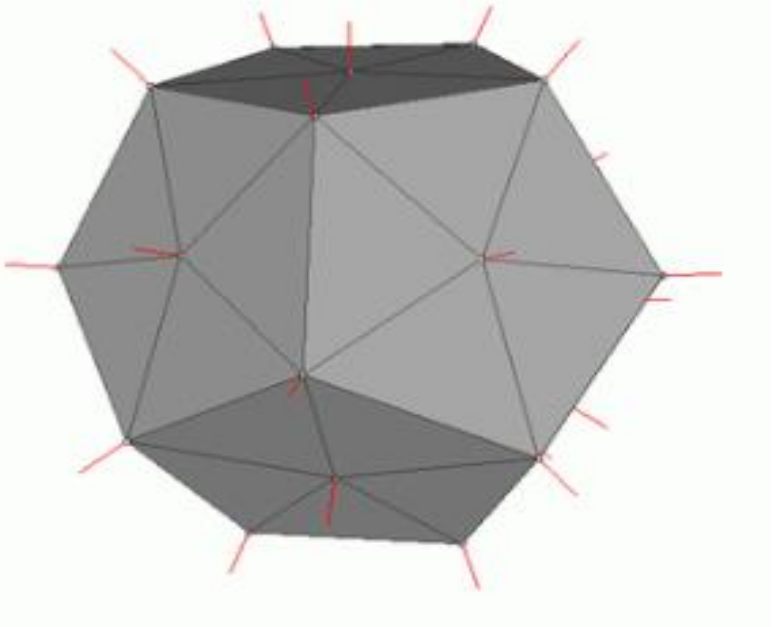
- A mesh is made up of vertices and edges.
- Each vertex has its normal.
- They are orthogonal to each other.



Prof. Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Building Block

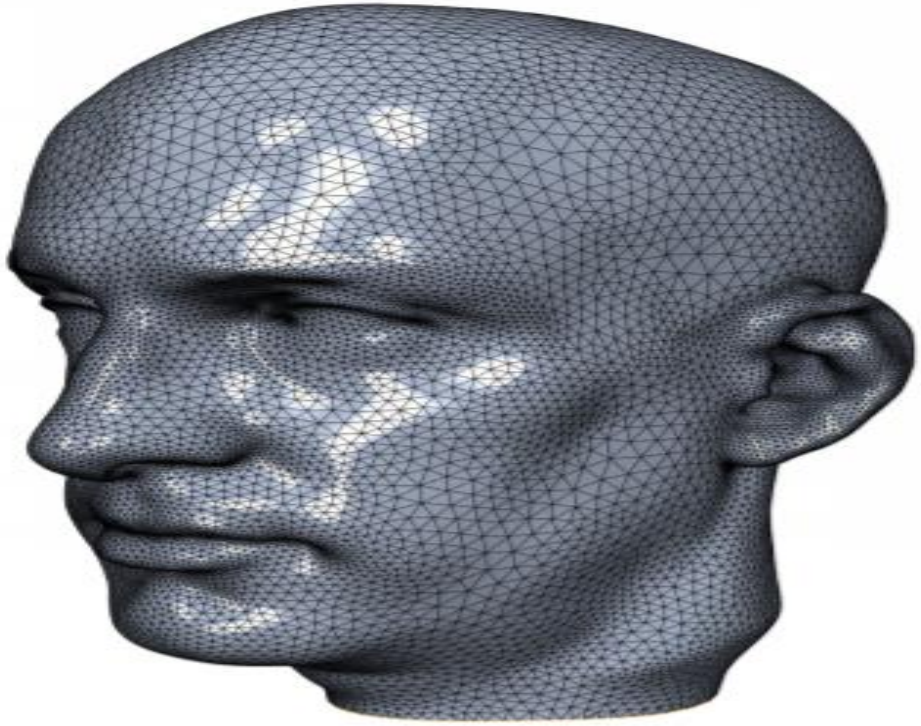
- Orthogonality lost during transformation
- Inverse transformation is necessary.



Prof. Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Building Block

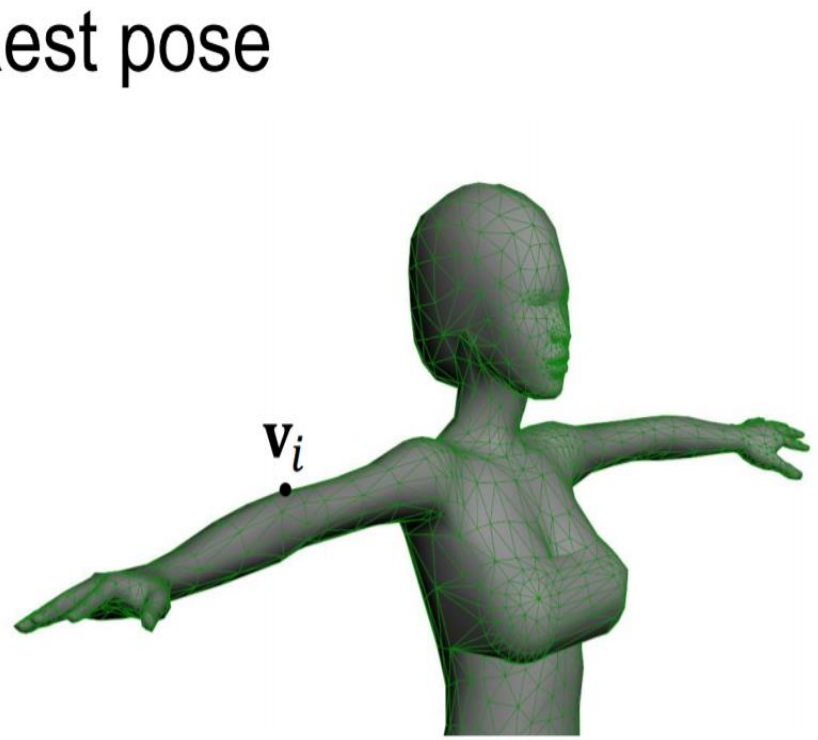
- Triangles formed form the building block of a polygonal mesh.



Prof. Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Skinning (Linear Blend Skinning)

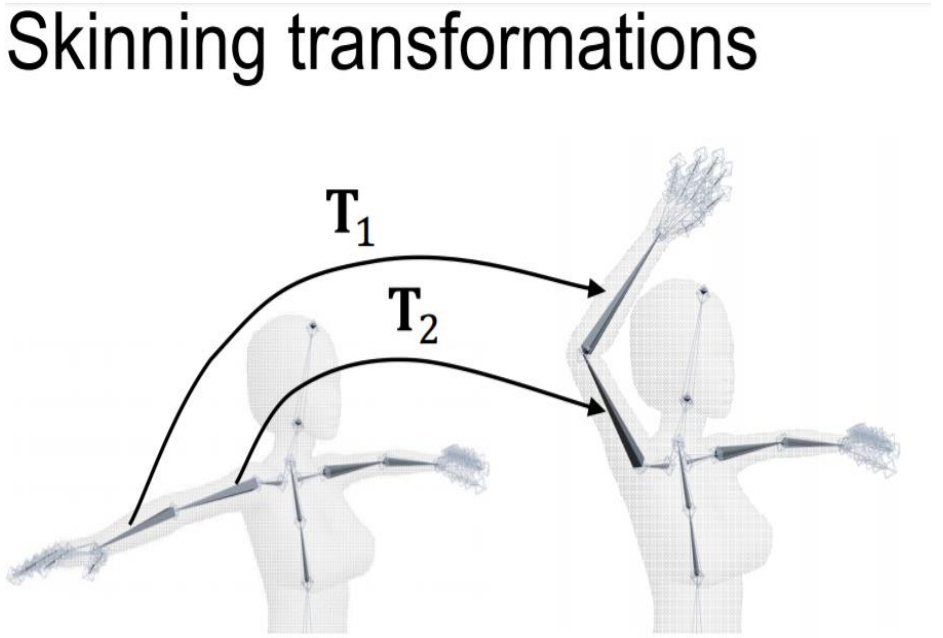
- Skinning is rendering a mesh around the joints.
- Vertices would be in rest pose.



Prof. Ladislav Kavan Course Notes, University of Utah
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org

Skinning (Linear Blend Skinning)

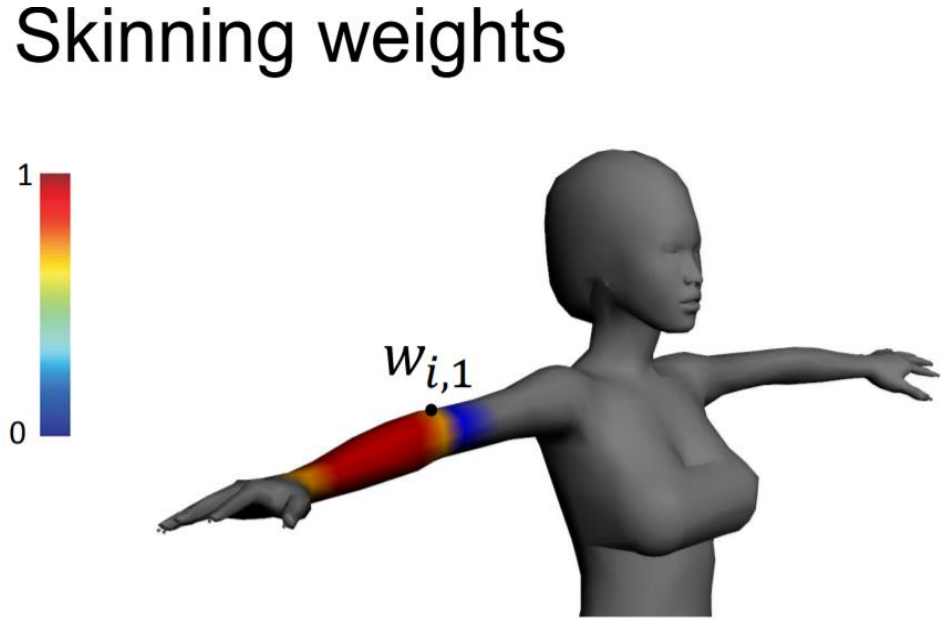
- Apply transformation to vertices to bind them to joints.
- If the joint position changes, the transformation matrix changes.



Prof. Ladislav Kavan Course Notes, University of Utah
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org

Skinning (Linear Blend Skinning)

- Associate each vertex to the joints.
- Transformation of the vertices depends on weights.



Prof. Ladislav Kavan Course Notes, University of Utah
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org

Body Model

- Utilize modified SMPL model.
- Represented as: $\mathbf{V}^B = M^B(\boldsymbol{\theta}^B, \boldsymbol{\phi}^B, \boldsymbol{t}^B)$
- Where $\mathbf{V}^B = \{\mathbf{v}_i^B\}_{i=1}^N$ $\quad v_i^B \in \mathbb{R}^3$
- N = 6890 vertices.
- These vertices are in rest pose.

M^B *resembles body model*

θ^B *is the pose parameter*

ϕ^B *is the shape parameter*

t^B *is the global translation parameter*

Body Model Transformation

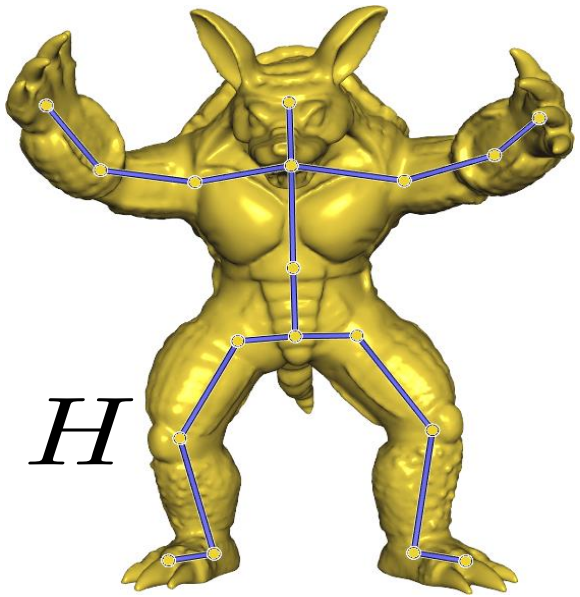
- Linear blend skinning applied to each joint.
- Posed mesh vertices given by:

$$\mathbf{v}_i^B = \mathbf{I}_{3\times 4} \cdot \sum_{j=1}^{J^B} w_{i,j}^B \mathbf{T}_j^B \begin{pmatrix} \mathbf{v}_i^{B0} + \sum_{k=1}^{K_b} \mathbf{b}_i^k \phi_k^B \\ 1 \end{pmatrix}$$

J^B = 21 joints

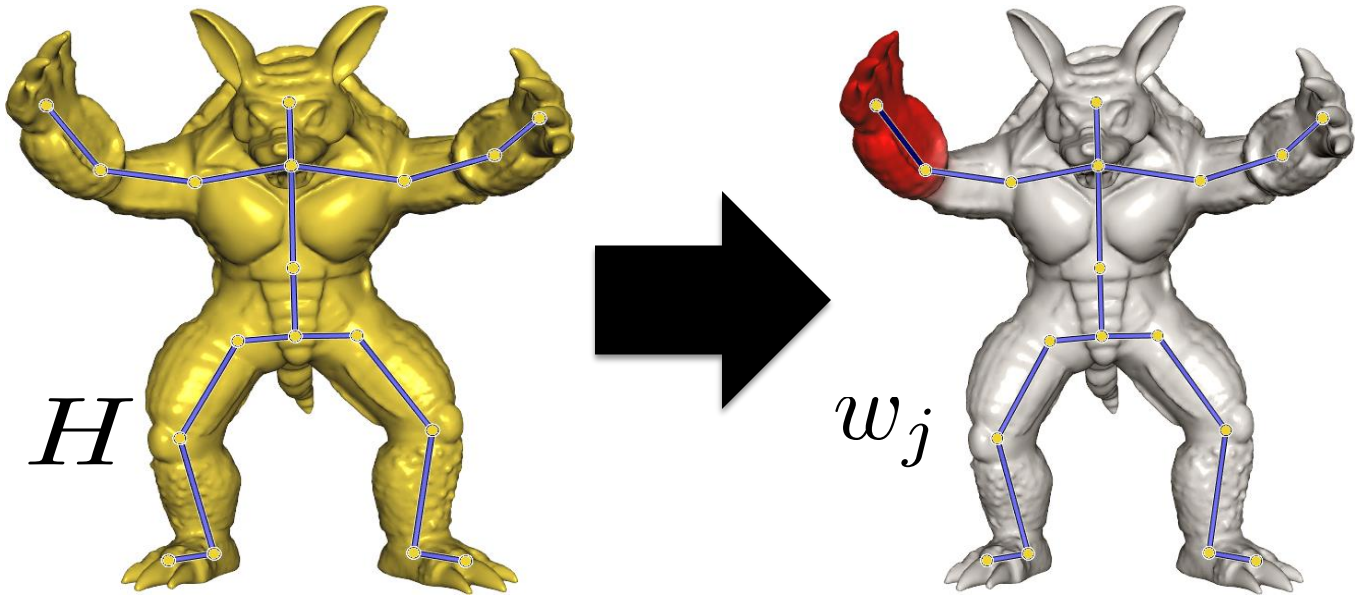
- $I_{3\times4}$ is the Identity Matrix
- W being the weights.
- T_j^B is the transformation matrix.
- $T_j^B \in$ SE(3)
- Sum of all weigths sum to 1.

- v_i^{B0} is the mean shape i-th vertex.
- K_b = 10, number of identity body shape coefficients.
- b_i^k is the vertex of the k-th blendshape.
- ϕ_k^B is the k-th shape coefficient.



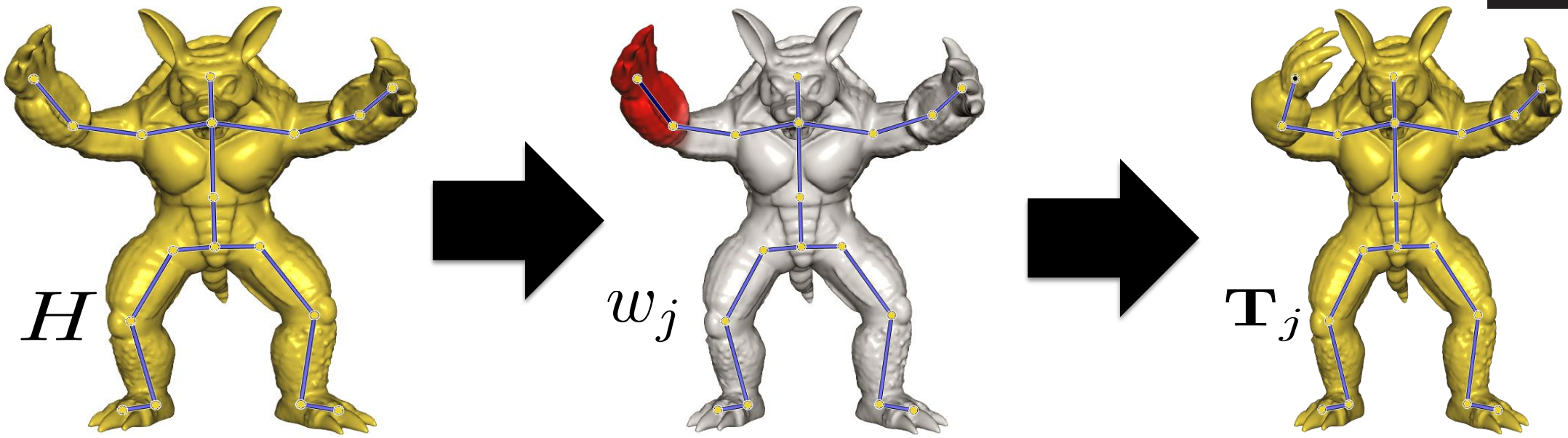
$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

Prof. Alex Jacabson Course Notes, University of Toronto
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org



$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

Prof. Alex Jacabson Course Notes, University of Toronto
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org

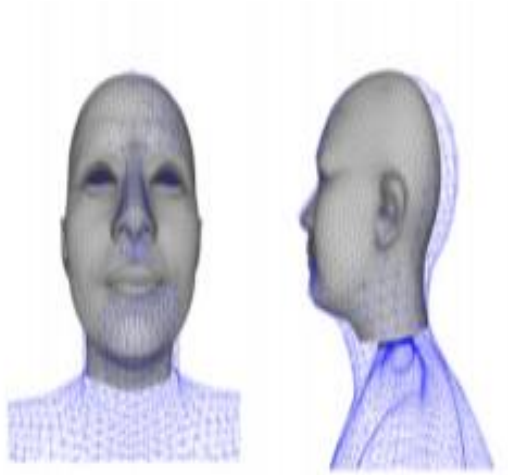


$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

Prof. Alex Jacabson Course Notes, University of Toronto
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org

Face Model

- Align the facial mesh [1] with the body model.
- The face model is given by: $\mathbf{V}^F = M^F(\theta^F, \phi^F, \mathbf{T}^F)$
- Where $\mathbf{V}^F = \{\mathbf{v}_i^F\}_{i=1}^F$
- $N^F = 11510$

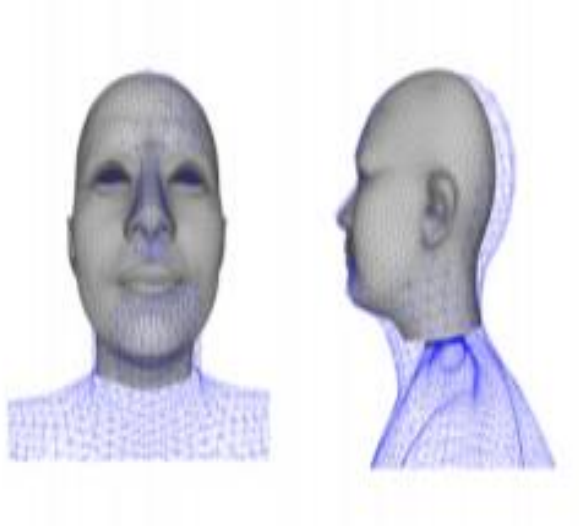


[1] FaceWarehouse: a 3D Facial Expression Database for Visual Computing, Cao et al., 2014

Face model

- Rest pose vertices:

$$\hat{\mathbf{v}}_i^F = \mathbf{v}_i^{F0} + \sum_{k=1}^{K_f} \mathbf{f}_i^k \phi_k^F + \sum_{s=1}^{K_e} \mathbf{e}_i^s \theta_s^F$$

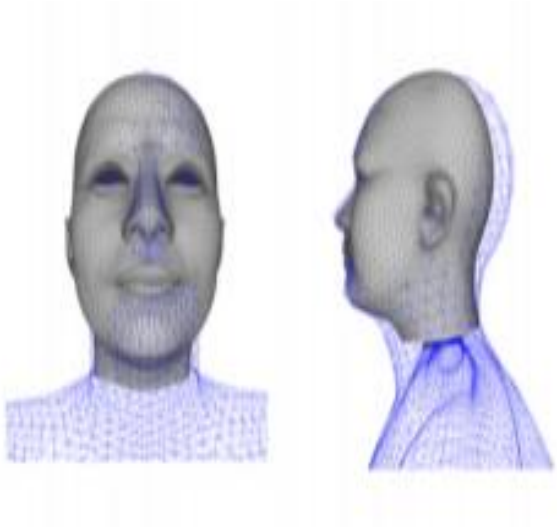


- V_i^{F0} is the mean shape.
- k_f = 150, is the identity blendshape.
- f_i^k is the vertex at k-th blendshape.
- ϕ_k^F is the shape coefficient.
- k_e = 200, is the expression blendshape.
- e_i^S is the vertex at s-th expression blendshape.
- θ_s^F is the pose blendshape parameter.

Face Model

- Each transformed vertex given by:

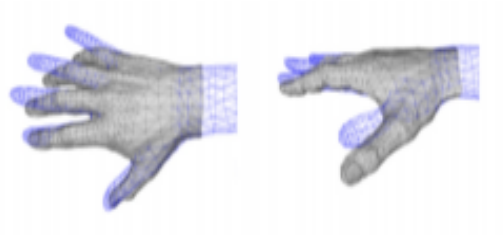
$$\mathbf{v}_i^F = \mathbf{I}_{3\times 4} \cdot \mathbf{T}_{j=F}^{B} \cdot \mathbf{\Gamma}^F \begin{pmatrix} \hat{\mathbf{v}}_i^F \\ 1 \end{pmatrix}$$



- $I_{3\times4}$: Truncated Identity Matrix.
- $T_i^B \in$ SE(3) denotes the transformation matrix.
- Γ^F is the alignment matrix.

Hand Model

- Each vertex is given by:

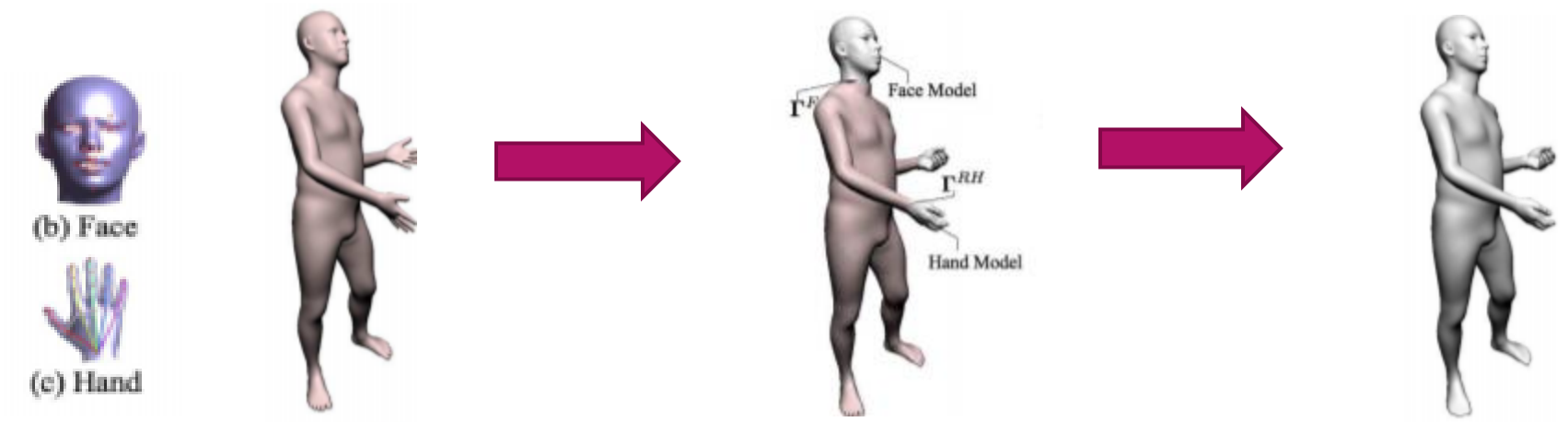


$$\mathbf{v}_i^H = \mathbf{I}_{3\times4} \cdot \mathbf{T}_{j=H}^B \cdot \Gamma^H \cdot \sum_{j=1}^{J} w_{i,j}^H \mathbf{T}_j^H \begin{pmatrix} \mathbf{v}_i^{H0} \\ 1 \end{pmatrix}$$

- $I_{3\times4}$ = Truncated Identity Matrix.
- T_j^B denotes the transformation matrix.

- v_i^{H0} is the mean shape.
- J indicates number of joints in hand, 16 joints.
- W indicates weights in the hand model.
- T_j^H is the matrix transformation.

Creating Frank



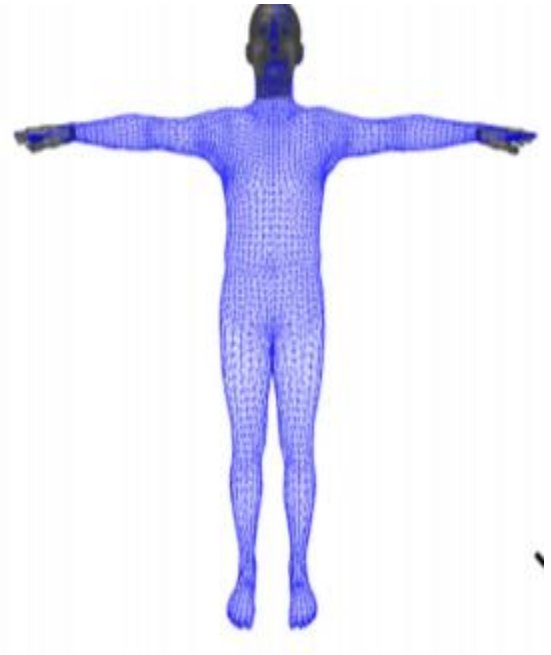
Different templates

Alignment

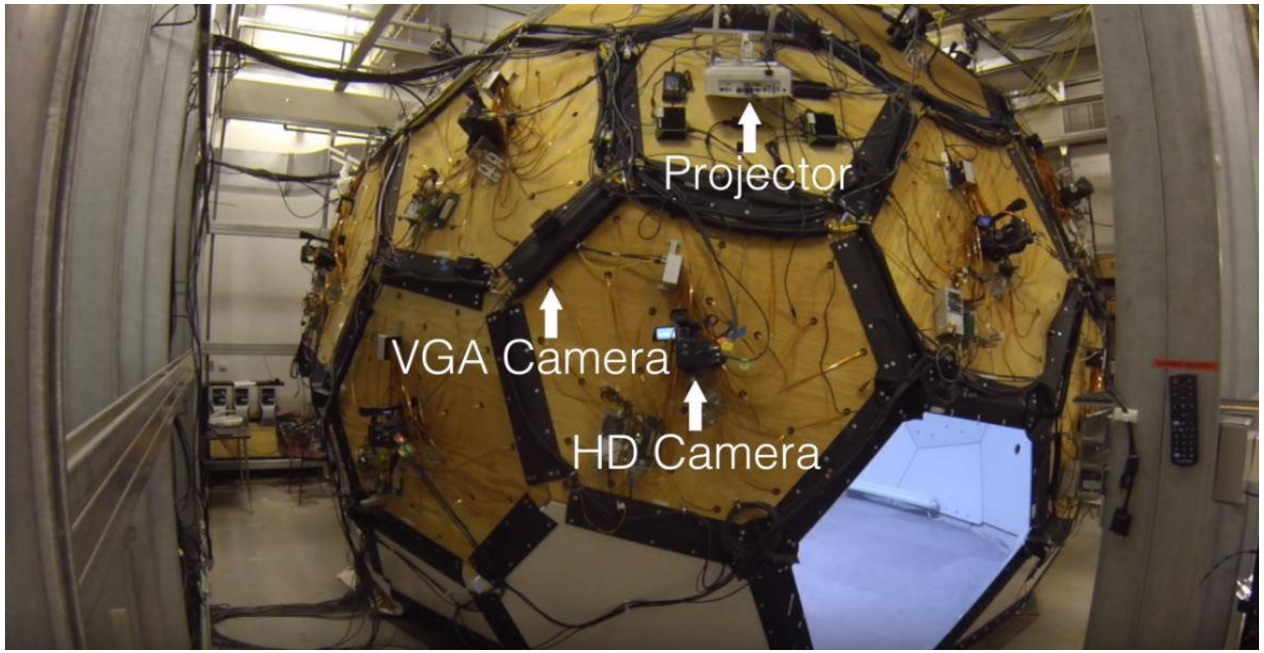
Frank model

Frank Model

- Frank model upon stitching body, face and hands is given by: $\mathbf{V}^U = M^U(\boldsymbol{\theta}^U, \boldsymbol{\phi}^U, \mathbf{t}^U)$
- Where $\mathbf{V}^U \in \mathbb{R}^{N^U \times 3}$
- $N^U = 18540$
- The linear blending matrix: $\mathbf{V}^U = \mathbf{C}\left[(\mathbf{V}^B)^T (\mathbf{V}^F)^T (\mathbf{V}^{LH})^T (\mathbf{V}^{RH})^T\right]$
- Where: $\mathbf{C} \in \mathbb{R}^{N^U \times (N^B + N^F + 2N^H)}$



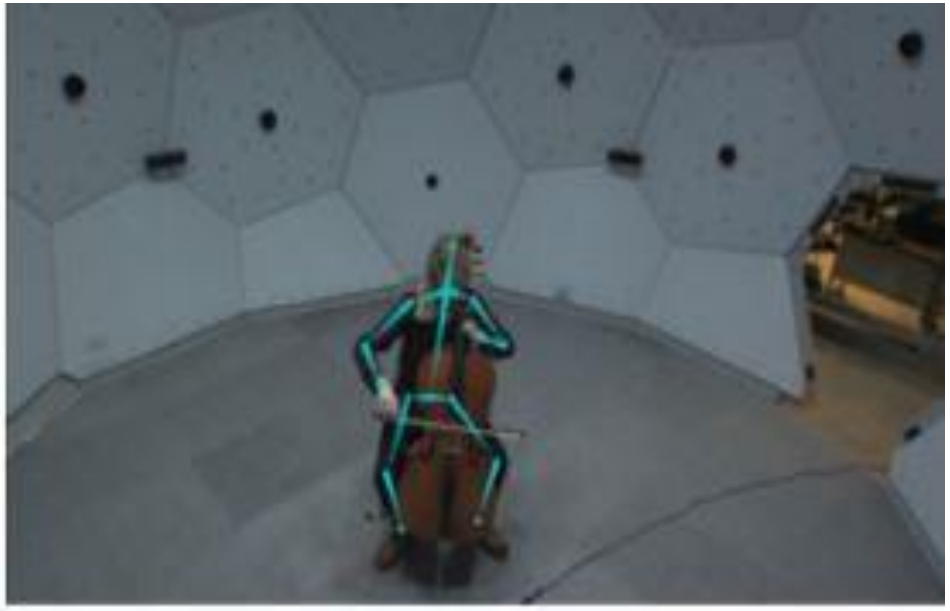
Dataset for Keypoints



- 540 Video Graphics Array Camera.
- 31 High Definition (HD) cameras.
- 10 Kinect cameras (RGB-D sensors).

Image source: Panoptic Studio, The CMU Robotics Institute

Dataset for Keypoints



- 140 VGA camera to get 3D triangulated keypoints.

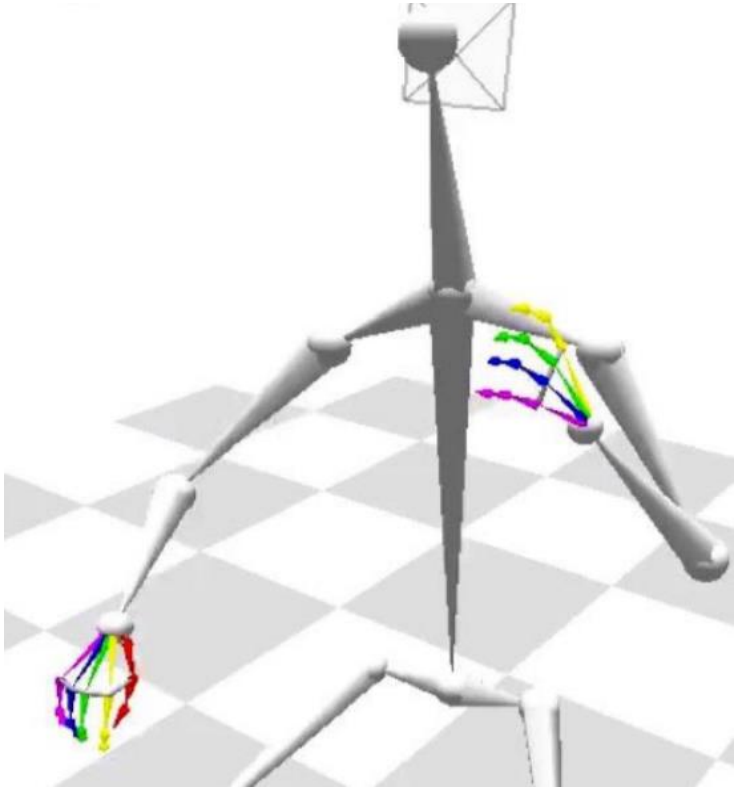
Image source: Panoptic Studio, The CMU Robotics Institute

Dataset for Keypoints

- 480 VGA cameras to detect 3D foot keypoints.

Image source: Panoptic Studio, The CMU Robotics Institute

Dataset for Keypoints



- 31 HD cameras to detect finger triangulated keypoints.

Image source: Panoptic Studio, The CMU Robotics Institute

Dataset for Keypoints

- 10 Kinect cameras to get Multi-View Stereo Pointclouds.

Image source: Panoptic Studio, The CMU Robotics Institute

3D Keypoints and Cloudpoints

- Use state-of-the-art methods [1] to get 2D keypoints.

[1] OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields, Cao et al., CVPR 2018

3D Keypoints and Cloudpoints

- Use state-of-the-art methods [1] to get 2D keypoints.
- Use MVS cameras to get 3D triangulated keypoints

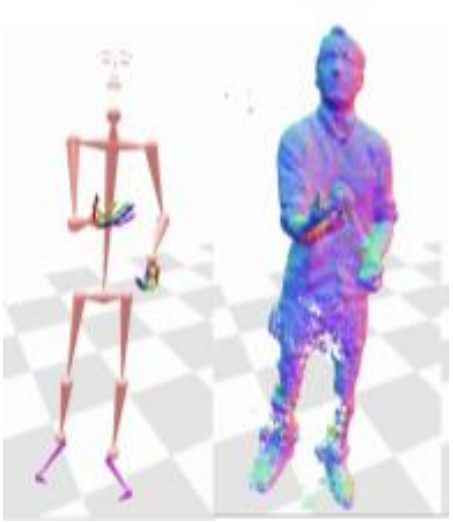
[2] Panoptic Studio: A Massively Multiview System for Social Interaction Capture, hanbyul et al. TPAMI 2017

3D Keypoints and Cloudpoints

- Use state-of-the-art methods [1] to get 2D keypoints.
- Use MVS cameras to get 3D triangulated keypoints
- Use RealityCapture [3] to 3D cloudpoints.

[3] realitycapture.com

Fitting Frank



3D Keypoints and Pointclouds

Fitting Frank to 3D data

Fitting result

Objective Functions

- Each frame is fit independently.
- Levenberg Macquardt for optimization purposes.
- A library called Ceres Solver is used.
- Overall cost function: $E(\boldsymbol{\theta}^U, \boldsymbol{\phi}^U, \boldsymbol{t}^U) = E_{\text{keypoints}} + E_{\text{icp}} + E_{\text{seam}} + E_{\text{prior}}$

Seam Cost

- Frank is composed of part models
- it would lead to discontinuities.
- Penalizing the distance between vertices of part models.

Seam Constraints

- The cost function is given by:

$$E_{\text{seam}} = \sum_{(i,j)\in\mathcal{C}^{LH}} ||\mathbf{B}_i\mathbf{V}^{\boldsymbol{B}} - (\mathbf{v}_j^{LH})^T||^2 + \sum_{(i,j)\in\mathcal{C}^{RH}} ||\mathbf{B}_i\mathbf{V}^{B} - (\mathbf{v}_j^{RH})^T||^2 + \sum_{(i,j)\in\mathcal{C}^{F}} ||\mathbf{B}_i\mathbf{V}^{B} - (\mathbf{v}_j^{F})^T||^2,$$

- $\mathbf{B}_i \in \mathbb{R}^{1\times N^B}$ where $\mathbf{B}_i\mathbf{1}_{N^B} = 1$

- C contains correspondences (i, j) where i denotes the cloud points, and denotes the ring

Anatomical Keypoint Cost

- Find correspondences between 3D keypoints and the mesh model.
- It includes joints in body and hands.
- Joints in face, finger tips and toes present on the surface.

Anatomical Keypoint Cost

❖The cost function: $$E_{\text{keypoints}} = \lambda_{\text{keypoints}} \sum_{i \in \mathcal{D}} \left\| \mathbf{J}_i \mathbf{V} - \mathbf{y}_i^T \right\|^2$$

❖ Where: *D* indicates available keypoints in a frame.

$\mathbf{J} \in \mathbb{R}^{C \times N^U}$ denotes regression matrix, resembling joints.

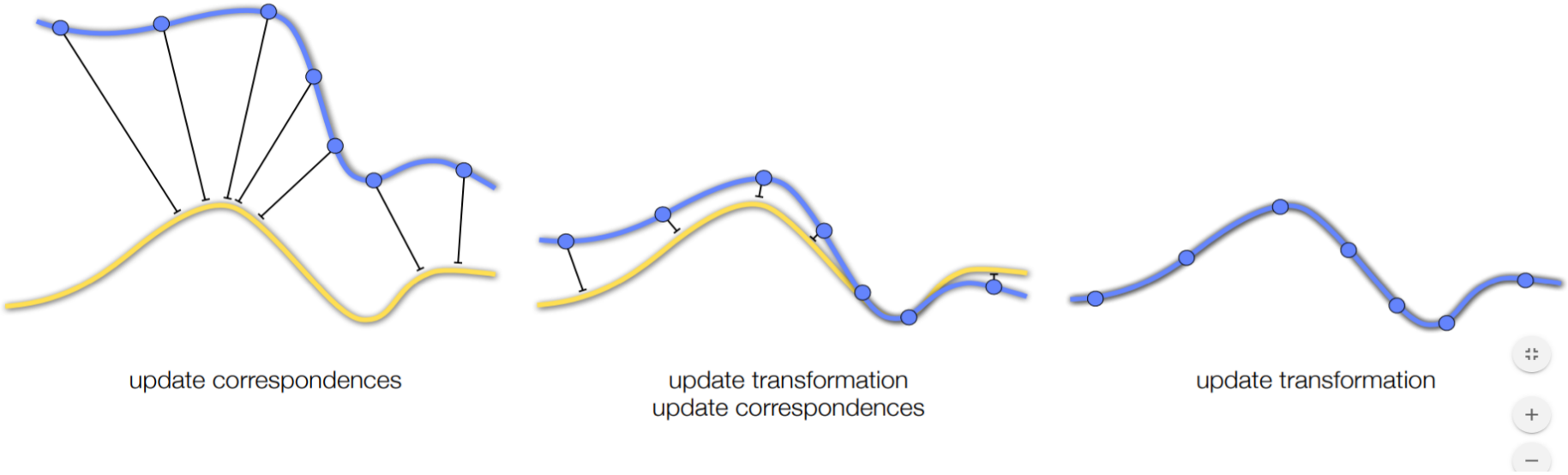
C is the number of correspondences.

N^U is the number of vertices in the mesh.

$y_i \in R^{3\times1}$, indicates 3D detections.

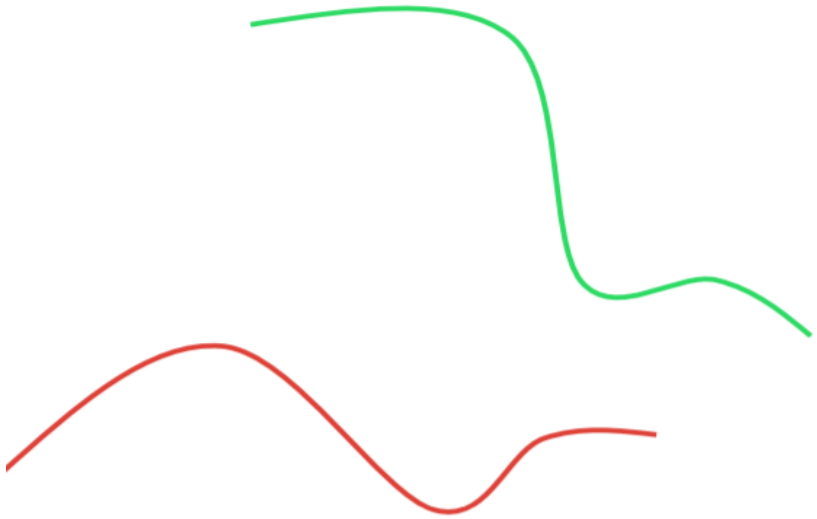
Iterative Closest Point Cost

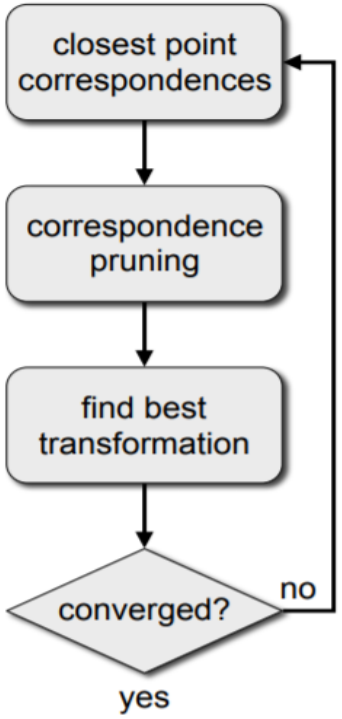
- **Step 1:** optimizing correspondences
- **Step 2:** optimizing transformations



https://lgg.epfl.ch/publications/2015/Htrack_ICP/slides.pdf

Iterative Closest Point Cost

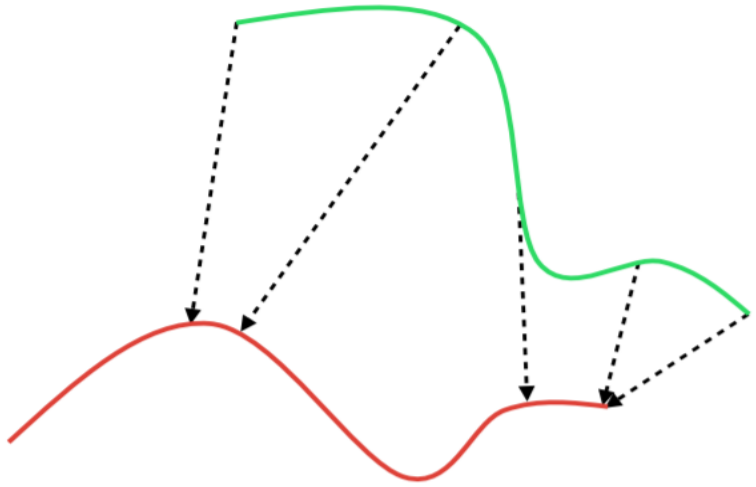


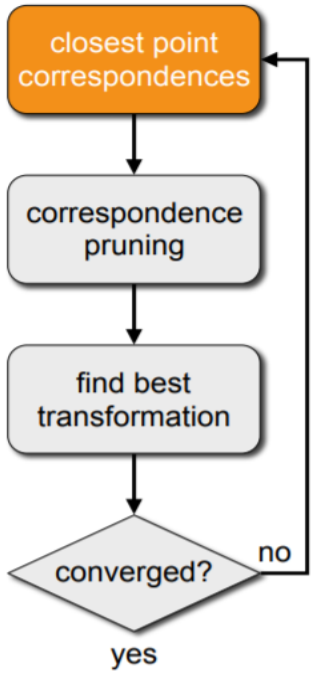


Set of vertices and pointclouds.

Professor Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Iterative Closest Point Cost

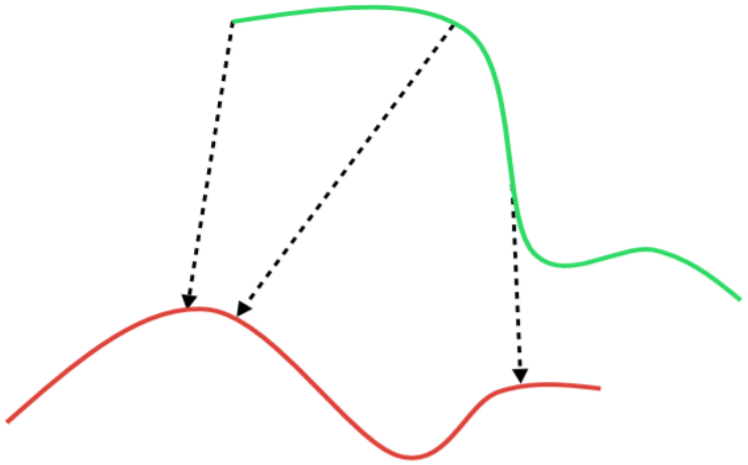


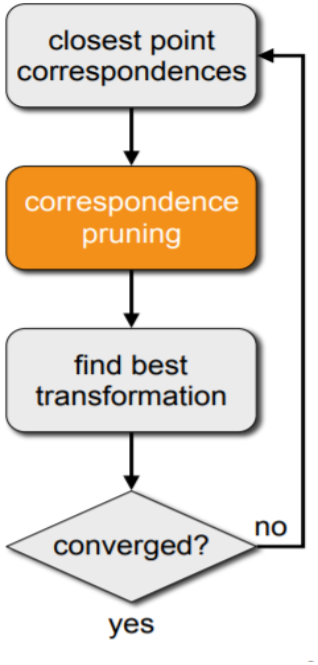


Find correspondences

Professor Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Iterative Closest Point Cost

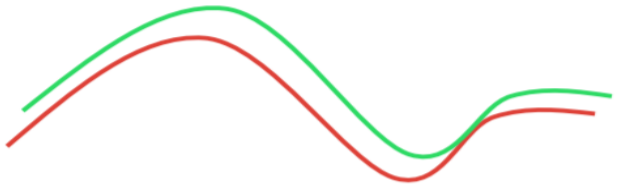


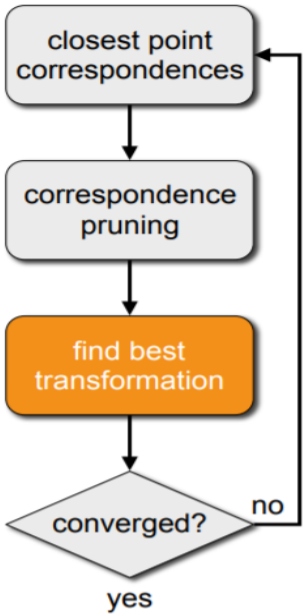


Take out erroneous data.

Professor Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Iterative Closest Point Cost

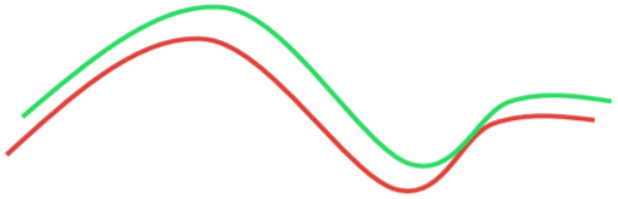


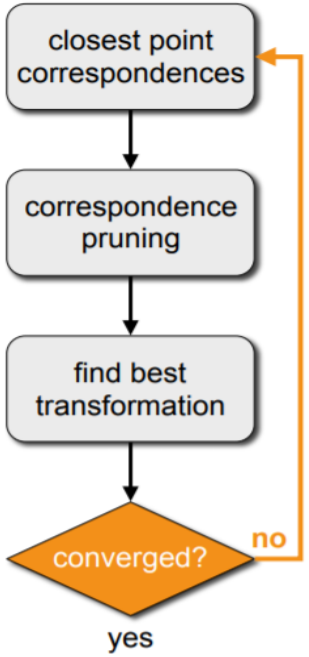


Do necessary transformation.

Professor Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Iterative Closest Point Cost

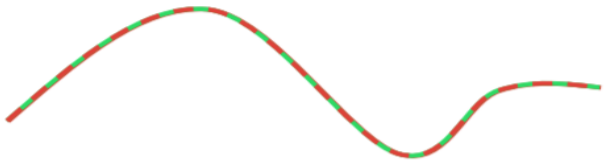


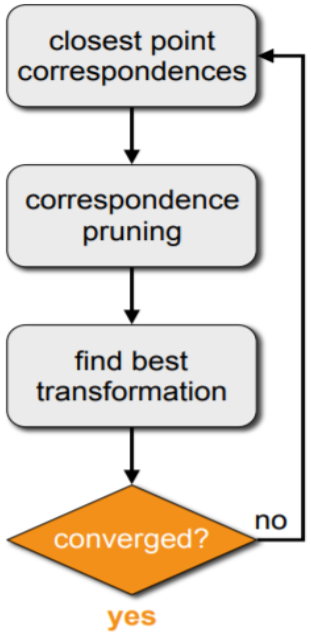


Check for convergence.

Professor Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Iterative Closest Point Cost





Repeat till convergence.

Professor Dr. Mario Botsch's Geometry Processing Lecture Notes
Computer Graphics and Geometry Processing Group, Bielefeld University

Iterative Closest Point Cost

- ICP is used to find the correspondences between the mesh vertices and the 3D Cloud Points.

- Distance along the normal direction is computed using:

$$E_{\text{icp}} = \lambda_{\text{icp}} \sum_{\mathbf{v}_j \in \mathbf{V}^U} \mathbf{n}(\mathbf{x}_{j^*})^T (\mathbf{x}_{j^*} - \mathbf{v}_j)$$

where: x_j is the closest 3D point to the j-th vertex v_j

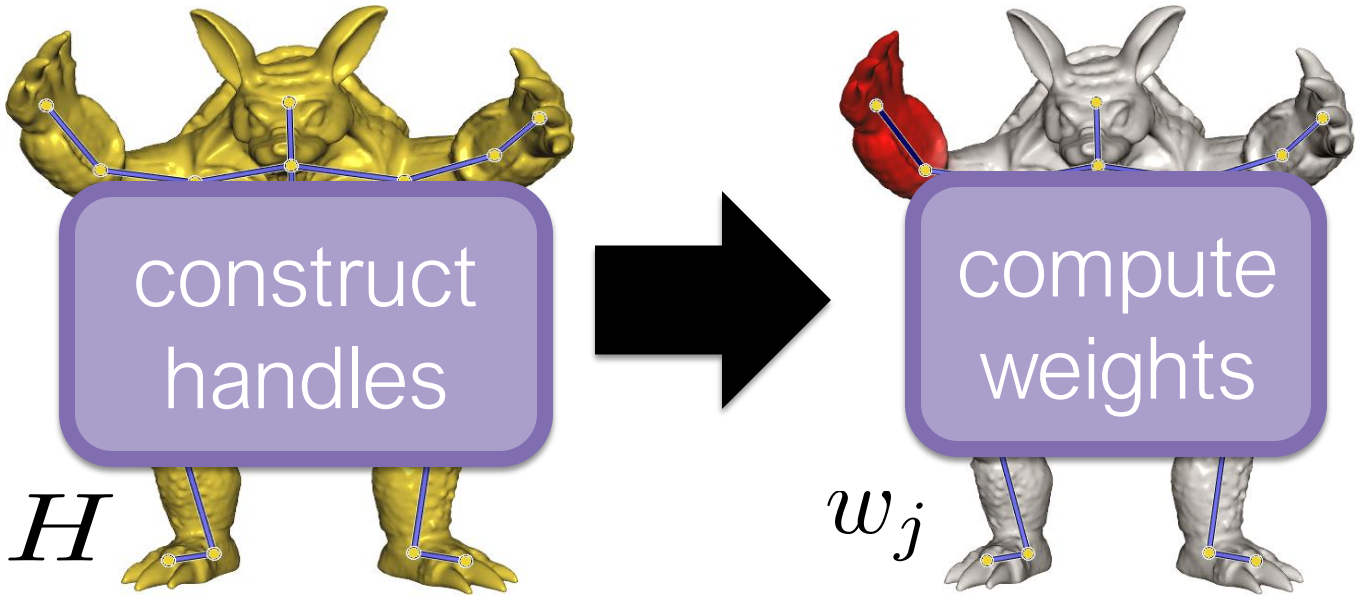
$n(\cdot) \in R^3$, is the point's normal.

λ_{icp} is the relative weight.

N^U is the number of vertices in the mesh.

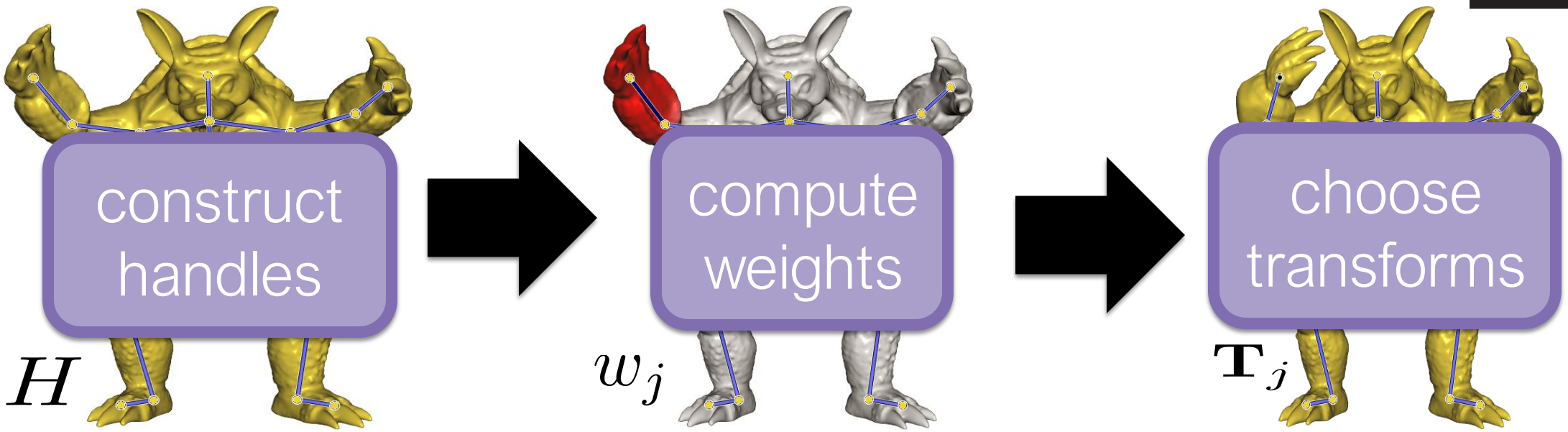
$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

Prof. Alex Jacabson Course Notes, University of Toronto
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org



$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

Prof. Alex Jacabson Course Notes, University of Toronto
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org



$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

Prof. Alex Jacabson Course Notes, University of Toronto
Skinning: Real-Time Shape Deformation, ACM SIGGRAPH 2014 Courses, skinning.org

Prior Cost

- The 3D Point Cloud could be noisy.
- The SMPL and FaceWarehouse models doesn't consider hair and clothing.
- Joint locations of the model are not completely in sync with the 2D keypoints.
- For better fitting of the model to data: $E_{\text{prior}} = E^{F}_{\text{prior}} + E^{B}_{\text{prior}} + E^{H}_{\text{prior}}$

Drawbacks of Frank Model

- Frank model doesn't consider hair and clothing
- Don't have simple parameterization process.

Adam Model

- Leverage Frank model and train it on 70 subjects.
- Consider 5 frames for each subject, resulting in 350 meshes.
- Single joint hierarchy.
- Common parametrization for all parts.

Fitting Clothes and Hair

- Deform the each vertex along its normal,
- Deformed vertex represented as: $\tilde{v}_i = v_i + n(v_i)\delta_i$

where δ_i is the scalar displacement between mesh vertex and the 3D Cloud Point.

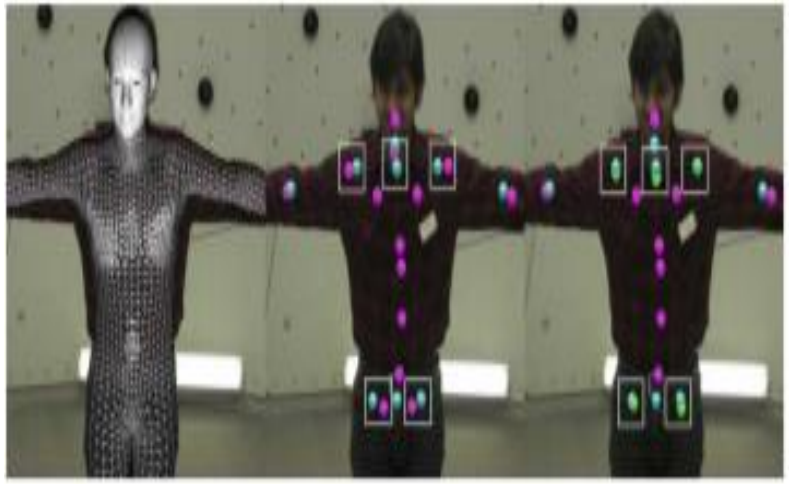
Fitting Clothes and Hair

- Pose the problem as a linear system.
- That is, $\begin{pmatrix} \mathbf{N}^T \\ (\mathbf{WLN})^T \end{pmatrix} \Delta = \begin{pmatrix} (\mathbf{P} - \mathbf{V}^U)^T \\ \mathbf{0} \end{pmatrix}$

Where Δ is the per-vertex displacement.

- P is the corresponding cloud point.
- V^U is the number of vertices in the mesh.
- N is the vertex normal.
- W is the diagonal weight matrix.
- L is the Laplace Beltrami operator to regularize the deformation.

Detection Target Regression



- In order to align the mesh with 3D keypoints properly, the vertices are aligned with 3D keypoints directly.

- The resulting mesh has 61 joints.

Adam model representation

- The model parameterized as: $M^A(\boldsymbol{\theta}^A, \boldsymbol{\phi}^A, \boldsymbol{t}^A) = \mathbf{V}^A$
- $\mathbf{V}^A = \left\{\mathbf{v}_i^A\right\}_{i=1}^{N^A}$ where $N^A = 18540$, is the number of vertices

  where $\hat{\mathbf{v}}_i^A = \mathbf{v}_i^{A0} + \sum_{k=1}^{K_A} \mathbf{s}_i^k \phi_k^A$

$\mathbf{v}_i^{A0}$ is the mean shape.

$K_A = 40$, number of blendshape coefficient.

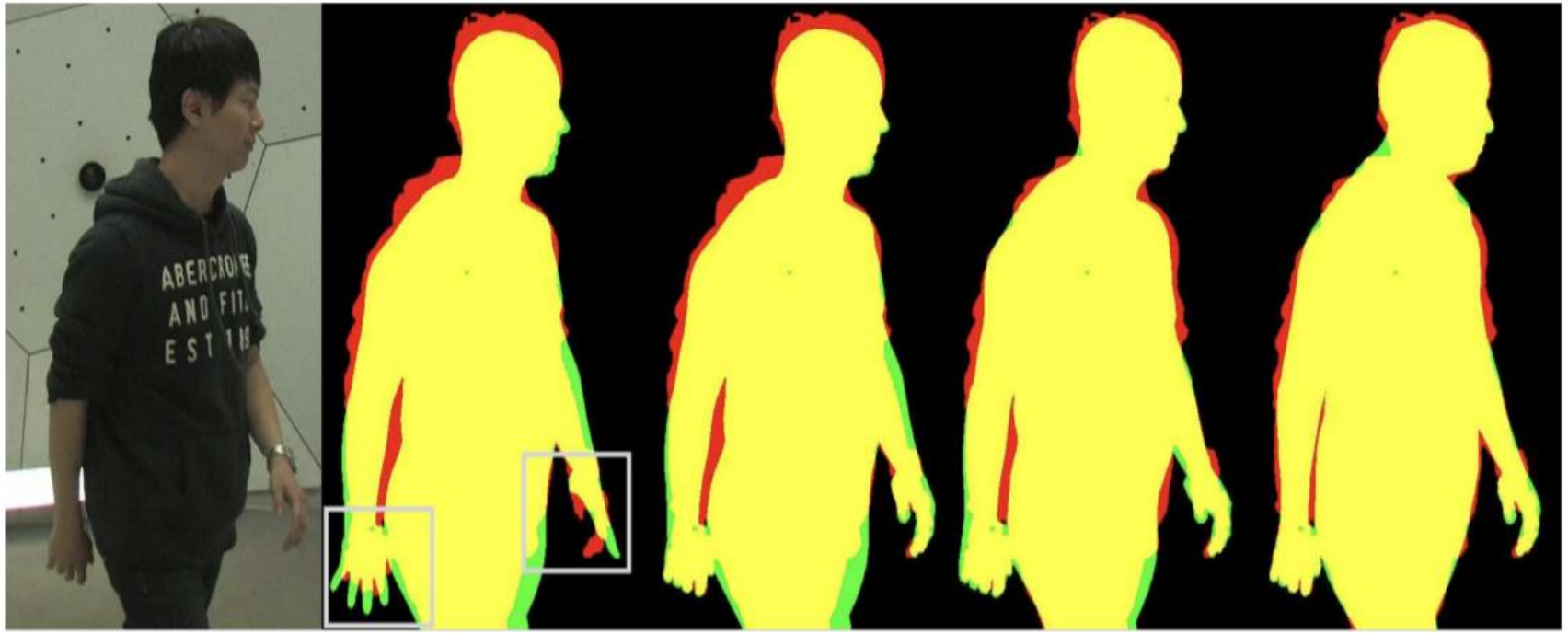
$\mathbf{s}_i^k$ is the vertex at k-th blendshape.

ϕ_k^A is the blendshape coefficient.

Adam Cost Function

- The Cost Function is given by: $E(\boldsymbol{\theta}^A, \boldsymbol{\phi}^A, \boldsymbol{t}^A) = E_{\text{keypoints}} + E_{\text{icp}} + E_{\text{prior}}$
- Does not require seam constraint
- Single set of unified shape and pose parameters for all parts
- Optical flow between frames to avoid jittery video.

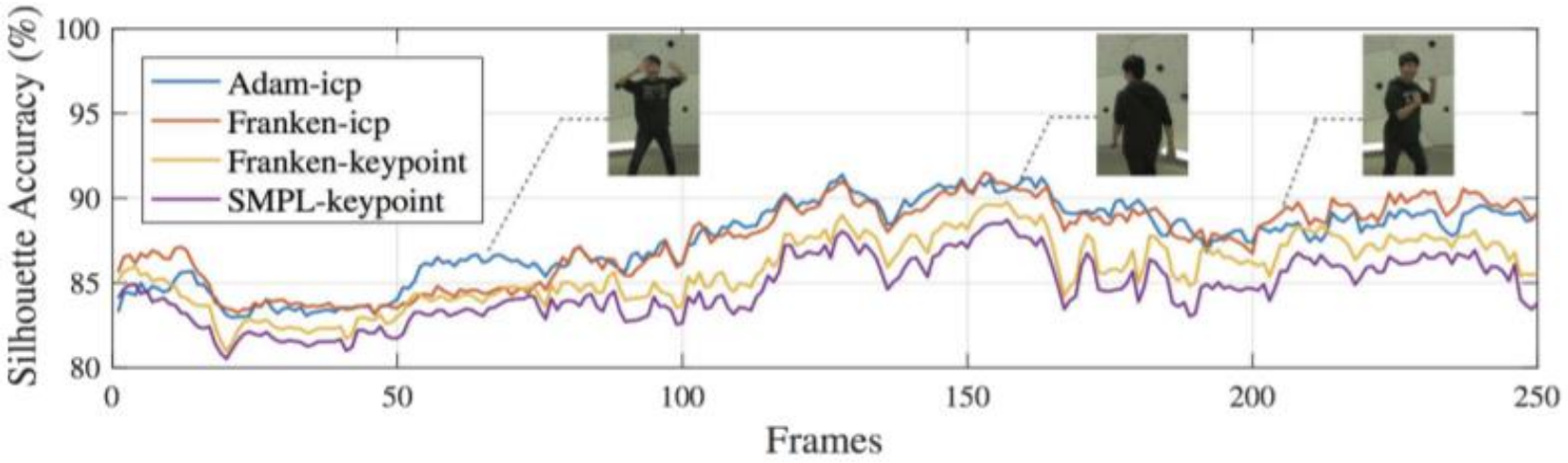
Results



(a) Input image | (b) SMPL keypoints | (c) Frank keypoints | (d) Frank keypoints + ICP | (f) Adam keypoints + ICP

- Red: Ground truth
- Green: Rendered silhouettes
- Yellow: Correctly overlapping areas

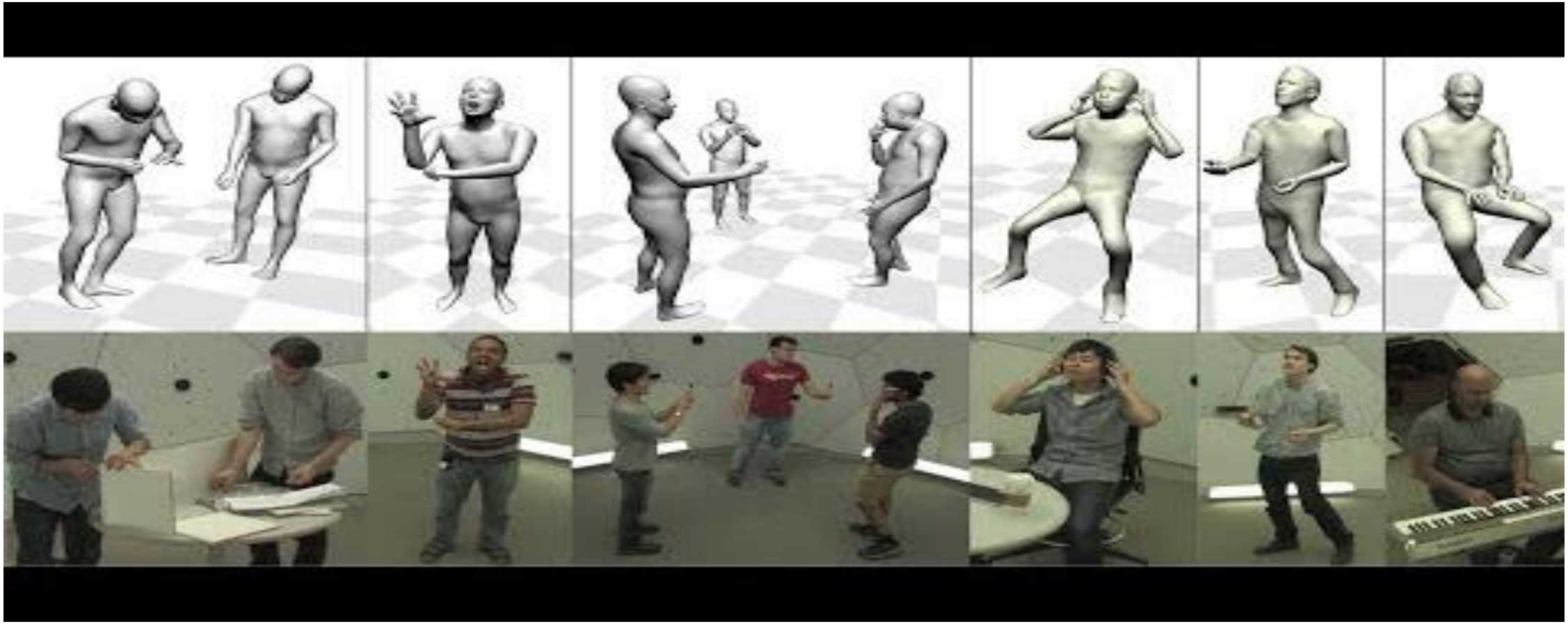
Results



- Silhouette Accuracy

	SMPL[34]	Frank	Frank ICP	Adam ICP
Mean	84.79%	85.91%	87.68%	87.74%
Std.	4.55	4.57	4.53	4.18

Results



Thank You