SMPLify

3D Human Shape & Pose from 2D Images

Allen Kim
CS 519, Fall 2017
Cal Poly Pomona
Shape & Pose Estimation

Shape
physical characteristics of a body (height, weight, short, tall, bone structure); pose-invariant 3D surface

Pose
articulated posture of the limbs;
Difficult Problem

Single, monocular 2D image

Ambiguous task—multiple 3D configurations can lead to the same 2D projection

Methods must deal with these ambiguities

Typical way—use prior knowledge about bodies

- **SHAPE PRIOR**: used to set anthropomorphic limits (e.g., bone lengths)
- **POSE PRIOR**: used to favor plausible poses, rule out impossible poses
SMPLify
Skinned Multi-Person Linear model

**SMPLify**

- first **fully automatic** method for 3D shape & pose from 2D joints
- **interpenetration** term that is differentiable with respect to shape and pose
- novel objective function
SMPLify Method

2-step process

1. Use DeepCut CNN to estimate 2D joints.
   Bottom-up estimation

2. Fit a 3D model to the 2D joints.
   Top-down verification using a generative model.
SMPL
Skinned Multi-Person Linear model

Artifacts at elbows and hips
vertex-based skin

- corrective blend shapes
- 6890 vertices
- 23 joints (white dots)
add to $\bar{T}$ to create a body shape

represented as vertex offsets from the mean template mesh

joint locations are a function of body shape and can be affected by spots on the surface

$\bar{T} + B_s(\beta), J(\beta)$
add to $\bar{T}$ while still in $\theta^*$

pose-dependent deformations

notice the expansion of her right hip

$$T_p(\tilde{\beta}, \tilde{\theta}) = \bar{T} + B_s(\tilde{\beta}) + B_p(\tilde{\theta})$$
Posed model

\[ M(\bar{\beta}, \bar{\theta}) = W(T_{\beta}(\bar{\beta}, \bar{\theta}), J(\bar{\beta}), \bar{\omega}) \]
SMPL Model Summary

\[
M(\vec{\beta}, \vec{\theta}) = W \left( T_P(\vec{\beta}, \vec{\theta}), J(\vec{\beta}), \vec{\theta}, \vec{\omega} \right)
\]

\[
T_P(\vec{\beta}, \vec{\theta}) = \bar{T} + B_S(\vec{\beta}) + B_P(\vec{\theta})
\]

**Model generation**

- \(M = \) SMPL function
- \(W = \) Skinning function
- \(B_P = \) Pose blendshapes function
- \(B_S = \) Shape blendshapes function
- \(J = \) Joint regressor

**Model input params**

- \(\beta = \) Shape params
- \(\theta = \) Pose params
- \(\omega = \) Scaled axis of rotation: 3 pose params related to a joint
- \(\theta^* = \) Zero or rest pose

**Model learned params**

- \(S = \) Shape blendshapes
- \(P = \) Pose blendshapes
- \(\omega = \) Blendweights
- \(J = \) Joint regressor matrix
- \(\bar{T} = \) Mean shape of template
SMPL
Principal Shape Components

Principal Shape Component 1
Principal Shape Component 2
Principal Shape Component 3
Joints are a sparse linear combination of surface vertices (equivalently, joints are a function of shape coefficients).

To put joints into poses, apply a global rigid transformation.

Posed 3D joint denoted as $R_{\theta}(J(\beta)_{i})$
After optimization, we end up with a sparse set of vertices and weights for how much they influence each joint.
Previous methods produce many poses that are impossible due to interpenetration.

Calculating interpenetration is typically expensive for complex, non-convex objects (human body).
**Use proxy geometries.** Approximate body as a set of cylindrical capsules with radius and axis length.

**NOVEL METHOD:** learn a linear regressor from body shape coefficients ($\beta$) to capsules’ radii & axis lengths.
Objective Function

\[
E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) + \lambda_\theta E_\theta(\theta) + \lambda_a E_a(\theta) + \lambda_{sp} E_{sp}(\theta; \beta) + \lambda_\beta E_\beta(\beta)
\]

Goal is to minimize this function

\(\lambda_\beta, \lambda_a, \lambda_\theta, \lambda_{sp}\) are scalar weights
Objective Function

\[
E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) + \lambda_\theta E_\theta(\theta) + \lambda_a E_a(\theta) + \lambda_{sp} E_{sp}(\theta; \beta) + \lambda_\beta E_\beta(\beta)
\]

Joint-based data term

\[
= \sum_{\text{joint } i} w_i \rho\left(\Pi_K(R_\theta(J(\beta)_i)) - J_{est,i}\right)
\]

target 2D joint

DeepCut CNN

How well do the projected joints match?
How well do the projected joints match?

Objective Function

\[ E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) + \lambda_\theta E_\theta(\theta) + \lambda_\alpha E_\alpha(\theta) + \lambda_\beta E_\beta(\theta) \]

Joint-based data term

\[ = \sum_{\text{joint } i} w_i \rho \left( \Pi_K(R_\theta(J(\beta)_i) - J_{est,i}) \right) \]

estimated 3D joint based on model’s body shape

target 2D joint
DeepCut CNN
Objective Function

\[ E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) + \lambda_{\theta} E_{\theta}(\theta) + \lambda_{a} E_{a}(\theta) + \lambda_{sp} E_{sp}(\theta; \beta) + \lambda_{\beta} E_{\beta}(\beta) \]

Joint-based data term

\[ = \sum_{\text{joint } i} w_i \rho \left( \Pi_K(R_\theta(J(\beta)_i)) - J_{est,i} \right) \]

Posed 3D joint—global rigid body transformation induced by pose \( \theta \)

estimated 3D joint based on model’s body shape

target 2D joint DeepCut CNN

How well do the projected joints match?
Objective Function

\[ E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) \]

+ \lambda_\theta E_\theta(\theta)
+ \lambda_a E_a(\theta)
+ \lambda_{sp} E_{sp}(\theta; \beta)
+ \lambda_\beta E_\beta(\beta)

Joint-based data term

\[
= \sum_{\text{joint } i} w_i \rho \left( \Pi_K \left( R_\theta (J(\beta)_i) \right) - J_{est, i} \right)
\]

3D-to-2D projection using perspective camera \( K \)

estimated 3D joint based on model’s body shape

Posed 3D joint—global rigid body transformation induced by pose \( \theta \)

target 2D joint DeepCut CNN

How well do the projected joints match?
Objective Function

$$E(\beta, \theta) = E_J(\beta, \theta; K, J_{\text{est}})$$

Joint-based data term

$$= \sum_{\text{joint } i} w_i \rho \left( \Pi_K (R_{\theta} (J(\beta)_i) - J_{\text{est}, i}) \right)$$

- **3D-to-2D projection** using perspective camera $K$
- **estimated 3D joint** based on model’s body shape
- **Posed 3D joint**—global rigid body transformation induced by pose $\theta$
- **target 2D joint** DeepCut CNN

How well do the projected joints match?
Objective Function

\[ E(\beta, \theta) = E_J(\beta, \theta; K, J_{\text{est}}) \]
\[ + \lambda_\theta E_\theta(\theta) \]
\[ + \lambda_a E_a(\theta) \]
\[ + \lambda_{sp} E_{sp}(\theta; \beta) \]
\[ + \lambda_\beta E_\beta(\beta) \]

**Pose prior:** Mixture of Gaussians

\[ E_\theta(\theta) \equiv -\log \sum_j \left( g_j \mathcal{N}(\theta; \mu_{\theta,j}, \Sigma_{\theta,j}) \right) \]
\[ \approx \min_j \left( -\log(e g_j \mathcal{N}(\theta; \mu_{\theta,j}, \Sigma_{\theta,j})) \right) \]

This prior was trained by fitting a mixture of Gaussians to ~1m poses spanning 100 subjects obtained by fitting SMPL to the CMU marker dataset.
**Objective Function**

\[
E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) + \lambda_\theta E_\theta(\theta) + \lambda_a E_a(\theta) + \lambda_{sp} E_{sp}(\theta; \beta) + \lambda_\beta E_\beta(\beta)
\]

**Pose prior:** Elbows and Knees

\[
E_a(\theta) = \sum_i \exp(\theta_i)
\]

Penalizes unnatural bending

Positive bends are unnatural. Punished heavily by exponential.
Objective Function

\[ E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) \]
\[ + \lambda_\theta E_\theta(\theta) \]
\[ + \lambda_a E_a(\theta) \]
\[ + \lambda_{sp} E_{sp}(\theta; \beta) \]
\[ + \lambda_\beta E_\beta(\beta) \]

**Pose prior: Interpenetration**

\[ E_{sp}(\theta; \beta) = \sum_i \sum_{j \in I(i)} \exp\left( \frac{\| C_i^e(\theta, \beta) - C_j(\theta, \beta) \|^2}{\sigma_i^2(\beta) + \sigma_j^2(\beta)} \right) \]

Centers of spheres along capsule axis
Objective Function

\[
E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) + \lambda_\theta E_\theta(\theta) + \lambda_\alpha E_\alpha(\theta) + \lambda_{sp} E_{sp}(\theta, \beta) + \lambda_\beta E_\beta(\beta)
\]

**Pose prior:** Interpenetration

\[
E_{sp}(\theta, \beta) = \sum_i \sum_{j \in I(i)} \exp \left( \frac{\| C_i(\theta, \beta) - C_j(\theta, \beta) \|^2}{\sigma_i^2(\beta) + \sigma_j^2(\beta)} \right)
\]

Centers of spheres along capsule axis

\[
\sigma(\beta) = \frac{r(\beta)}{3}
\]

Relate error to the *intersection volume* between incompatible capsules
Objective Function

\[
E(\beta, \theta) = E_J(\beta, \theta; K, J_{est}) + \lambda_{\theta}E_{\theta}(\theta) + \lambda_{\alpha}E_{\alpha}(\theta) + \lambda_{sp}E_{sp}(\theta; \beta) + \lambda_{\beta}E_{\beta}(\beta)
\]

Shape prior

\[
E_{\beta}(\beta) = \beta^T \Sigma^{-1}_{\beta} \beta
\]

diagonal matrix with \textbf{squared singular values} from PCA on the shapes in SMPL training set
Example Output
Example Output
Example Output