3D Human Modeling Reading Report

Caoliwen Wang

August 20, 2023

Abstract

Based on various reports from the forefront of computer graphics courses, the author has chosen "3D Human Modeling" as the reading direction. This document serves as a reading report on this topic. First, it introduces the core ideas and some specific algorithms used in famous parametric human models in historical order, discussing their breakthroughs and limitations. The SCAPE, SMPL, and STAR models are described in detail. Next, the report introduces the relatively new implicit function methods, which are widely used in 3D human reconstruction, showcasing the PIFu algorithm and listing related works. Following that, it briefly discusses RGB-D fusion methods, citing some classic works. Finally, a brief summary is provided based on the author's learning and course content.
Keywords: 3D human modeling, implicit function methods, fusion methods

Contents

Remark. In this document, red superscripts represent footnotes, and black superscripts represent references. Detailed references can be found at the end of the document.

1 Introduction

1.1 Why 3D Human Modeling is Needed

3D human modeling plays an indispensable role in various real-world fields.

In the medical field, 3D human modeling is widely used to create highly accurate anatomical models, which are crucial for doctors to better understand human anatomy, plan surgical procedures, diagnose diseases, and plan treatments. For example, doctors can use 3D human models to simulate surgeries, predict surgical outcomes, and meticulously plan surgical paths, thereby reducing surgical risks. Additionally, 3D human models are used in medical education to help students gain a deeper understanding of human anatomy through visualization, fostering more professionally skilled medical personnel.

In the entertainment and media industry, 3D human modeling also has great creative potential. In film, television, and animation production, producers can use 3D human models to create more vivid characters that interact and move in a realistic and natural way, providing a more engaging viewing experience for audiences.

Virtual reality and game development are another significant application area. Through 3D human modeling, game developers can create stunning virtual worlds, allowing players to fully immerse themselves in the game. Players can control their virtual characters, interact with the game world, and experience various scenarios and challenges. Moreover, 3D human modeling can be used in virtual tour applications to guide users through landmarks, art museums, and other places, offering a more immersive visual and auditory experience.

This technology provides a more realistic and interactive visual experience, enriching innovation across various fields and creating more colorful real and virtual worlds. Through 3D human modeling, we can map from reality to virtuality to gain deeper insights into "how humans interact with each other," and map from virtuality to reality to apply the technology in robotics to facilitate human life.

1.2 Challenges in 3D Human Modeling

Complex Geometry and Topological Changes. The shape and posture of the human body are highly complex, involving the coordination of multiple joints, muscles, and tissues. Capturing and representing such complex geometry and topological changes is a challenge.

Data Acquisition and Processing. Obtaining high-quality human data requires costly scanning devices or sensors, and processing large volumes of data to build accurate models. Data processing may involve issues such as noise, missing data, and data alignment.

Posture and Motion Variations. The human body can move in various postures and actions, such as walking, running, and jumping. Building models that can accurately capture these variations requires considering the impact of different postures and motions on the model.

Detail and Realism. In some applications, such as medical imaging and film production, highly realistic human models are needed. To capture fine details, such as skin folds and

muscle deformations, issues related to detail representation and realism need to be addressed.

Computational Efficiency and Real-time Performance. In real-time applications, such as virtual reality and gaming, generating and rendering realistic human models within a short time is required. Therefore, computational efficiency and real-time performance of the models are a challenge.

Soft Tissue Simulation. In animation and virtual reality, simulating changes and deformations of human soft tissues is also a challenge. Soft tissue simulation needs to consider variations in muscles, fat, and their responses to posture and motion.

Diversity and Personalization. There are differences in human bodies across different populations, such as gender, age, and body type. Constructing models that can adapt to different human characteristics and personalized needs is a challenge.

Thus, the difficulties encountered in 3D human modeling are interdisciplinary and multifaceted, requiring comprehensive consideration of geometry, physics, computation, and perception to achieve accurate, realistic, and real-time human models.

1.3 A Brief History of "Virtual Humans"

This subsection discusses some previous works related to 3D human modeling, mostly enumerating previous works.

In 1882, two-dimensional images were used to record human motion trajectories, known as "the earliest exploration."

In 1973, key points and human skeletal mechanisms were introduced, constructing multiple sets of key points for kinematic analysis.

In 1973, Nevatia & Binford proposed the "generalized cylinders" algorithm, collecting a batch of toy-based shape prototypes and successfully fitting range data to generate reconstruction results. Similar tasks today only require laser rangefinders or radars, but such instruments were not available at that time, making it a remarkable algorithm.

In 1978, Marr & Nishihara improved upon the previously mentioned algorithm and proposed a general, composite structure algorithm capable of 3D reconstruction to approximate human representation. This involved learning from prototype models with skeletal structures and completing reconstructions.

In 1978, Marr & Nishihara introduced the dynamic joint tree, as shown in Figure 1. By defining the root node's position and angle (referred to as "shape prototype" in the text) and extending leaf nodes from the root node, the entire dynamic joint tree is defined by calculating the relative positions and angular relationships between leaf nodes and parent nodes.

In 1983, David Hogg used computer programming software to extract image edges through edge detection algorithms. Further analysis using predefined 3D human models revealed the surface structure of the human body. This was the first application of manually defined 3D

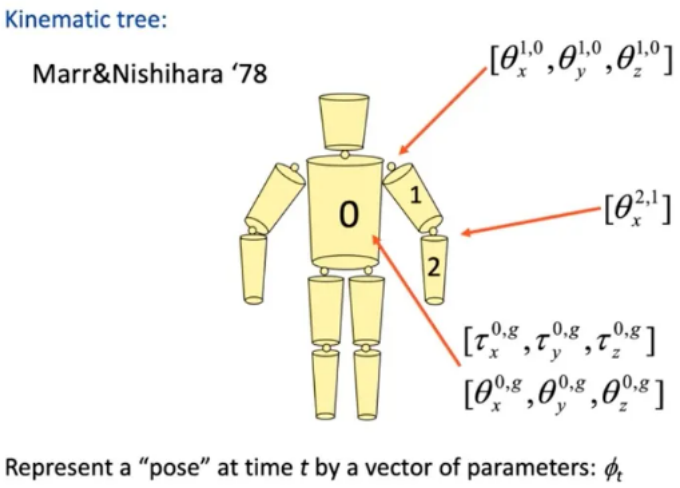


Figure 1: Dynamic Joint Tree

parametric human models. The author mathematically defined key human body parts and set spatial priors to compute possible values for key parameters such as length, width, and orientation.

In 1994, Rohr guided virtual humans to perform similar actions by recognizing human motion in image sequences.

In 1997, S. Wachter & H.-H. Nagel used generative models to track human movement in monocular image sequences.

In 1991, Alex Pentland explored optimization algorithms for virtual human reconstruction under non-rigid conditions.

In 1993, D. Gavrila used posture prototypes to capture differences in body postures across genders, modeling them separately and using multiple cameras for tracking to model virtual humans in motion sequences.

In 1998, Bregler & Malik studied 3D body shapes and projected them into 2D to represent human 2D motion states.

Additionally, there have been more refined studies, as follows:

In 1996, Baumberg and Hogg attempted to model 2D posture parameters. They represented 2D postures using contours and lines formed by points above, and improved the accuracy of 2D shapes by refining the contours. This modeling approach is known as "Eigen-shapes."

In 1999, Blanz & Vetter attempted 3D facial reconstruction. They used Principal Component Analysis (PCA) to reconstruct facial shape and appearance spaces, and optimized methods to calculate necessary parameters for 3D facial reconstruction, including shape, ambient lighting, and pose.

From 1999 to 2001, Cyberware used the Cyberware scanner to generate the CAESAR dataset, which includes data from 2,000 males and females from Europe and the United States. This dataset references 1990s US census data and considers key factors such as age, weight, and height, making it authoritative.

In 2003, Allen et al. attempted to register baseline human templates to the CAESAR human scan models to obtain a gender-independent universal human topology in the same reference coordinate system. After registration, various sub-templates that meet the requirements can be generated by interpolating and deforming the ground truth template. However, the generated animations appeared less realistic due to the absence of pose in the training.

For models that are more modern and of significant value, detailed introductions will be provided later in this document.

1.4 Introduction

In this short-term course, Professor Tao Yu gave a report titled "Human Performance Capture: From Deep Fusion to Deep Implicit Functions," which introduced the history, modeling process, optimization methods, and existing works of 3D human body modeling. Professor Dongdong Weng gave a report titled "Acquisition and Driving of High-Fidelity Digital Humans," providing more information on 3D facial modeling. On the other hand, I felt that 3D human body modeling is ubiquitous in life, including entertainment means such as movies and games. Therefore, I chose "3D Human Body Modeling" as the direction, read related works, and formed this reading note.

This article first introduces the most basic and core issue of 3D human body modeling, namely the parametric human model. I list important works in parametric human models: although SCAPE was published in 2005, a long time ago, its method of decomposing body shape changes into pose changes and body shape differences is the foundation of more modern parametric models; SMPL is one of the most widely used models, some works in the third and fourth parts still apply this model; the STAR model is the optimal version of the SMPL model now. In addition to a relatively complete discussion of the algorithms proposed by these three models, I also list more important works.

In the third part, this article introduces the implicit function method, which was first proposed by PIFu in 2019, and many researchers have since conducted more in-depth exploration on it, including Professor Yebin Liu's team at Tsinghua University, which has also made many outstanding contributions in this regard. This method is relatively new, focusing on human body reconstruction, and was mentioned in Professor Tao Yu's report, so I made some discussion in this article.

The fourth part of this article briefly introduces related works on fusion methods based on RGB-D input. I found that the author mentioned this method in 3.2.3Function4D, and by consulting the data, I know that this method combines color images (RGB images) and depth information (D images), thereby providing richer and more accurate data, supporting various applications and tasks, and has significance in computer vision, computer graphics, and related fields. Professor Tao Yu also mentioned topics related to Fusion, so I made some discussion

in this article.

Finally, I recorded a few shallow discoveries during the reading.

2 Parametric Human Models

This part introduces the core ideas and some specific algorithms used in famous parametric human models in chronological order, and discusses the breakthroughs and limitations of the algorithms.

2.1 SCAPE

2.1.1 Introduction

The SCAPE method (Shape Completion and Animation for People) is a data-driven approach that constructs a unified model for both pose and body shape. This method generates dense full-body meshes and captures details of muscle deformation in various poses by learning two different body change models: one for explaining pose changes and one for body shape differences. The pose model's dataset comes from dense 3D scans of a single person in multiple poses. The pose model decomposes body deformations into rigid and non-rigid components, with the rigid component described by a low-degree-of-freedom skeletal structure and the non-rigid component capturing remaining deformations, such as muscle bends. In this model, body part deformations depend only on adjacent joints, keeping the dimensions relatively low and enabling automatic learning from limited training data.

The body shape model simulates body shape differences between different individuals. The dataset is obtained from a set of 3D scans of different people in different poses. Body shape differences are represented through Principal Component Analysis (PCA), constructing a low-dimensional body deformation subspace. Importantly, the body shape model is not affected by pose-induced deformations as these deformations are considered separately. Together, these two parts form a unified framework of human shape variability, generating complete surface meshes including the joint angles of the body skeleton and feature coefficients describing body shape.

This model is applied in two important graphics tasks. Firstly, it is used for partial view completion. In graphics-related applications, a complete surface model is often required for rendering and animation processing, but it is difficult to obtain a complete surface model of a person. Even in full-body scanners like Cyberware, surface data is still incomplete due to potential occlusions by clothing, etc., in a static state. Most human scans have large missing regions on the surface model. With this model, the best-fitting human shape for the observed partial data can be found, and the complete 3D mesh can be predicted using the model. Since the model also considers non-rigid pose differences, muscle deformations related to specific poses can be predicted even for unobserved parts of the body.

Secondly, the model is used to generate complete 3D animations based on marked motion data. Existing motion capture systems typically provide sparse measurements of a few surface points. By inputting a sequence of extremely sparse data marked at limited locations on the

body (usually between 50 and 60), this model can predict a complete 3D human shape for each frame that matches the observed marker positions. Applying this technique to motion capture data sequences can generate 3D animations of the full human body, thus constructing high-quality, realistic muscle deformation animations for characters with only a single range scan data.

This work was completed in 2005, but the proposed method is classic and of epoch-making significance. This part references the paper by the SCAPE developers[1].

2.1.2 Data Acquisition and Mesh Preprocessing

The SCAPE model is data-driven, and all shape information is obtained from a series of scans. The basic process of data acquisition and mesh preprocessing is shown in Figure 2.

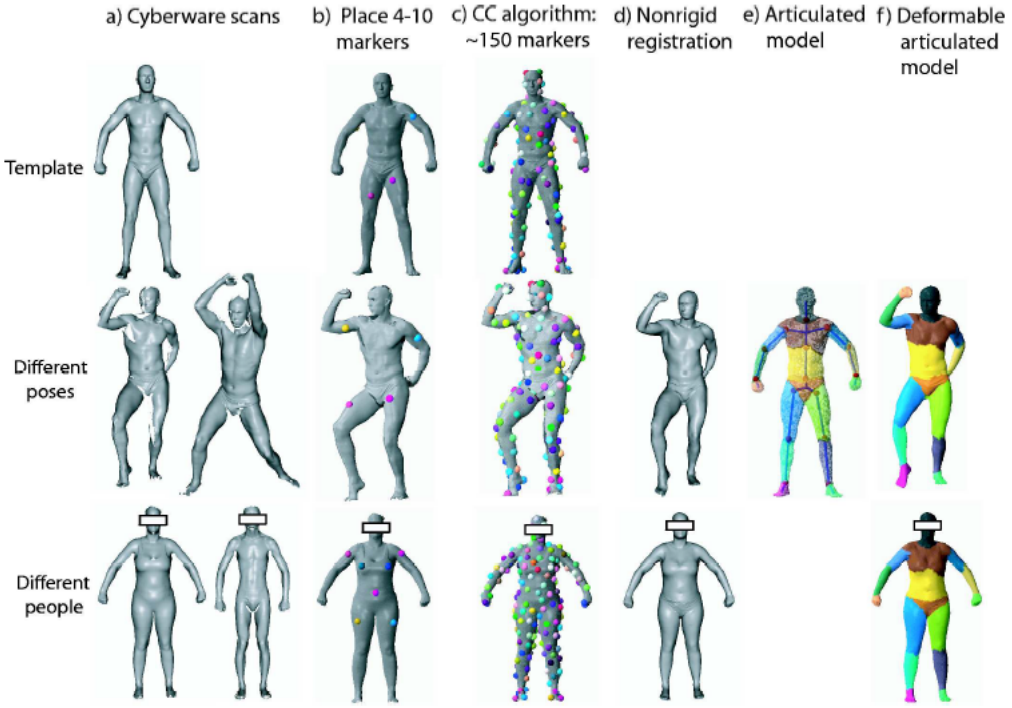


Figure 2: The mesh processing pipeline used to generate our training set. (a) We acquired two data sets spanning the shape variability due to different human poses and different physiques. (b) We select a few markers by hand mapping the template mesh and each of the range scans. (c) We apply the Correlated Correspondence algorithm, which computes numerous additional markers. (d) We use the markers as input to a non-rigid registration algorithm, producing fully registered meshes. (e) We apply a skeleton reconstruction algorithm to recover an articulated skeleton from the registered meshes. (f) We learn the space of deformations due to pose and physique.

Figure 2: Basic process of SCAPE data acquisition and mesh preprocessing

(a) Acquire two datasets covering shape changes caused by different body poses and different body shapes.

(b) Manually map template mesh and scan some landmark points for each range.

(c) Apply related correspondence algorithms to compute a large number of additional landmark points.

(d) Use these landmark points as input for a non-rigid registration algorithm to generate fully registered meshes.

(e) Apply a skeleton reconstruction algorithm to recover a joint skeleton from the registered meshes.

(f) Learn the deformation space caused by poses and body shapes.

Below are some detailed introductions.

Scanning. Surface data was acquired using the Cyberware WBX whole-body scanner. The scanner captured range scans from four directions simultaneously, resulting in models containing approximately 200K points. The four scans were merged into a complete whole-body instance mesh[2], and the instance was sampled to include about 50,000 triangles[3].

Following this process, two datasets can be obtained: a pose dataset containing scan data of 70 poses of specific individuals, and a body shape dataset containing scan data of 37 different individuals in similar (but not identical) poses. The developers also added eight publicly available models from the CAESAR dataset to the personal dataset[4]. One mesh from the pose dataset was selected as the template mesh, and all other meshes were referred to as instance meshes. The template mesh served as a reference for all other scans, and the developers used the algorithm proposed by Davis et al.[5] to fill holes in the template mesh. The template mesh and some instance meshes are shown in Figure 2(a). In some of the images, the head regions have been smoothed to conceal the identity of the scanned objects.

Correspondence Algorithms. The next step in data acquisition was to establish correspondences between the template mesh and each instance mesh. Non-rigid registration algorithms at that time required a set of corresponding landmark points between each instance mesh and the template mesh. The developers used an algorithm called the Correlated Correspondence (CC) algorithm to obtain these landmarks[6]. The CC algorithm consistently embeds each instance mesh into the transformed template mesh and matches visually similar surface regions. To break the symmetry of the scans, 4-10 landmarks were manually placed in each pair of scans to initialize the CC algorithm. The result of the algorithm is approximately 140-200 (near) correspondences between the two surfaces, as shown in Figure 2(c).

Non-rigid Registration. Given a set of landmarks between two meshes, the goal is to align the meshes closely while aligning the landmarks. The developers applied a standard algorithm[7] to register the template mesh with all meshes in the dataset, resulting in a set of meshes with the same topological structure that closely approximates the surfaces in the original Cyberware scans. Figure 2(d) shows some of these meshes.

Joint Skeleton Recovery. The model uses a low-degree-of-freedom skeleton to simulate joint movements. The developers automatically constructed the skeleton for the template mesh using only the meshes from the dataset and then applied Anguelov's algorithm[8]. This algorithm leverages the fact that vertices on the same joint of the skeleton are spatially contiguous and exhibit similar motions across different scans. Based on the pose dataset, the algorithm automatically constructed a skeleton with 18 parts, with the groin and chest areas divided into two symmetrical parts, resulting in a non-tree structure for the skeleton. For convenience in pose editing, the developers combined these two parts into one, resulting in a skeleton with 16 tree-like joints.

Data Format. The obtained dataset consists of a model mesh X and a set of instance meshes $Y = Y^1, \cdots, Y^N$. The model mesh $X = \{V_X, P_X\}$ has a set of vertices $V_X = x_1, \cdots, x_M$ and a set of triangles $P_X = p_1, \cdots, p_P$. Instance meshes come in two types: scans of the same person in different poses and scans of different people in approximately the same poses (still multiple). Through preprocessing, it is assumed that each instance mesh has the same set of points and triangles as the model mesh. Let $Y^i = y_1^i, \cdots, y_M^i$ be the set of points in the instance mesh Y^i, and let the absolute rotation set of the rigid part of each mesh Y^i be R^i, where R_l^i is the rotation of joint l in the instance mesh i. The data acquisition and preprocessing process provides us with this type of data.

Pose Deformation.Sections 2.1.3 and 2.1.4 represent the core results of the model.

Deformation Process. For each mesh Y_i in the dataset containing different poses of the human body, modeling is performed. The pose deformation model targets each triangle p_k in the template. Considering the non-rigid and rigid components of the deformation, two affine triangle transformations are used. Let the vertices of triangle p_k be $x_{k,1}$, $x_{k,2}$, $x_{k,3}$. Translating point $x_{k,1}$ to the global origin results in a local coordinate system for the triangle, and based on this coordinate system, deformations are applied to the edges of the triangle $\hat{v_{k,j}} = x_{k,j} - x_{k,1}, j = 2, 3$.

First, a 3×3 linear transformation matrix Q_k^i is applied to the triangle. This matrix corresponds to the non-rigid component of the deformation induced by each triangle p_k and each pose Y_i. Then, the deformed polygon is rotated once, i.e., rotated by the rotation R_l^i of the rigid part of the joint skeleton, and the same rotation is applied to all triangles belonging to that part. Let $l[k]$ be the body part associated with triangle p_k, then:

$$v_{k,j}^i = R_{l[k]}^i Q_k^i \hat{v_{k,j}}, ; j = 2, 3 \tag{2.1}$$

A key feature of this model is that it combines deformation elements of the rigid skeleton with elements allowing arbitrary local deformation, which is essential for modeling muscle deformation.

Learning Pose Deformation Model. The following demonstrates how to model pose-induced deformations using a set of matrices Q_k^i, where p_k is a triangle in the template mesh. The goal is to predict these deformations from joint rotation matrices R_{l_1} and R_{l_2} that can be represented as a set of relative joint rotations, where the relative joint rotation matrix can be simply represented as $R_{l_1}^T R_{l_2}$.

Joint Rotation. Joint rotation is typically represented by torque. Suppose M is an arbitrary

3×3 rotation matrix, and m_{ij} is its element in the i-th row and j-th column. The torque of the joint angle is a three-dimensional vector, which can be computed using the following formula:

$$t = \frac{||\theta||}{2 \sin ||\theta||} \begin{bmatrix} m_{32} - m_{23} \\ m_{13} - m_{31} \\ m_{21} - m_{12} \end{bmatrix}. \tag{2.2}$$

$$\text{where } \theta = \cos^{-1}\left(\frac{\text{tr}(M) - 1}{2}\right). \tag{2.3}$$

The direction of the torque vector represents the rotation axis, and the magnitude of the torque represents the rotation angle.

Each triangle p_k learns a regression function that predicts the transformation matrix Q_k^i based on the torques of its two nearest joints, represented as $\Delta r_{l[k]}^i = (\Delta r_{l[k],1}^i, \Delta r_{l[k],2}^i)$. Assuming that the matrix Q_k^i can only be predicted by these two joints greatly reduces the dimensionality of the learning problem.

Each joint rotation requires three parameters to represent, so $\Delta r_{l[k]}^i$ has a total of six parameters. When adding a constant bias term, we associate a 7×1 regression vector $a_{k,lm}$ with each matrix value Q, defined as:

$$q_{k,lm}^i = a_{k,lm}^T \cdot \begin{bmatrix} \Delta r_{l[k]}^i \\ 1 \end{bmatrix}, \; l, m = 1, 2, 3. \tag{2.4}$$

Thus, for each triangle p_k, we need to fit 9×7 entries $a_k = (a_{k,lm} : \; l, m = 1, 2, 3)$.

The goal now is to learn these parameters $a_{k,lm}$. If we know the transformation matrix Q_k^i and the rigid part rotation R^i for each instance mesh Y_i, it is straightforward to solve for the regression values by minimizing a quadratic error function, computed separately for each triangle k and matrix value $q_{k,lm}$:

$$\arg\min_{a_{k,lm}} \sum_i \left([\Delta r^i, 1] a_{k,lm} - Q_{k,lm}^i\right)^2. \tag{2.5}$$

In practice, the model size and computation can be reduced by identifying joints with only one or two degrees of freedom. In some cases, allowing these joints to have three degrees of freedom can lead to overfitting. Principal Component Analysis (PCA) is performed on the observed angles of Δr^i to remove rotation axes with eigenvalues less than 0.1. The corresponding entries in the vector $a_{k,lm}$ are not estimated.

As mentioned earlier, the rigid part rotations are computed in the preprocessing step. Unfortunately, the transformation matrix Q_k^i for a single triangle is not known. Following the method proposed by Sumner[9], a smoothing constraint is introduced that favors similar deformations in adjacent polygons belonging to the same rigid part. Specifically, for each mesh Y^i, solve the following equation to obtain the correct set of linear transformations:

$$\arg\min_{Q_1^i, \cdots, Q_P^i} \sum_k \sum_{j=2,3} ||R_{l[k]}^i Q_k^i \hat{v_{k,j}} - v_{k,j}^i||^2 + w_s \sum_{k_1, k_2} I(l_{k_1} = l_{k_2}) \cdot ||Q_{k_1}^i - Q_{k_2}^i||^2. \tag{2.6}$$

where $w_s = 0.001\rho$, ρ is the resolution of the model mesh X, and $I(\cdot)$ is the indicator function. This equation can be solved for each rigid part and each row of the Q matrix.

For the given estimated Q matrices, we can solve for the (at most) 9×7 regression parameters a_k for each triangle p_k.

Application to Dataset. This method is applied to a dataset of 70 poses to learn the SCAPE pose deformation model. Figure 3 shows an example. These examples do not correspond to the meshes in the training dataset; they are entirely new poses synthesized based on joint rotation vectors R, using equation (2.4) to define the Q matrices and generating the meshes using equation (**??**).

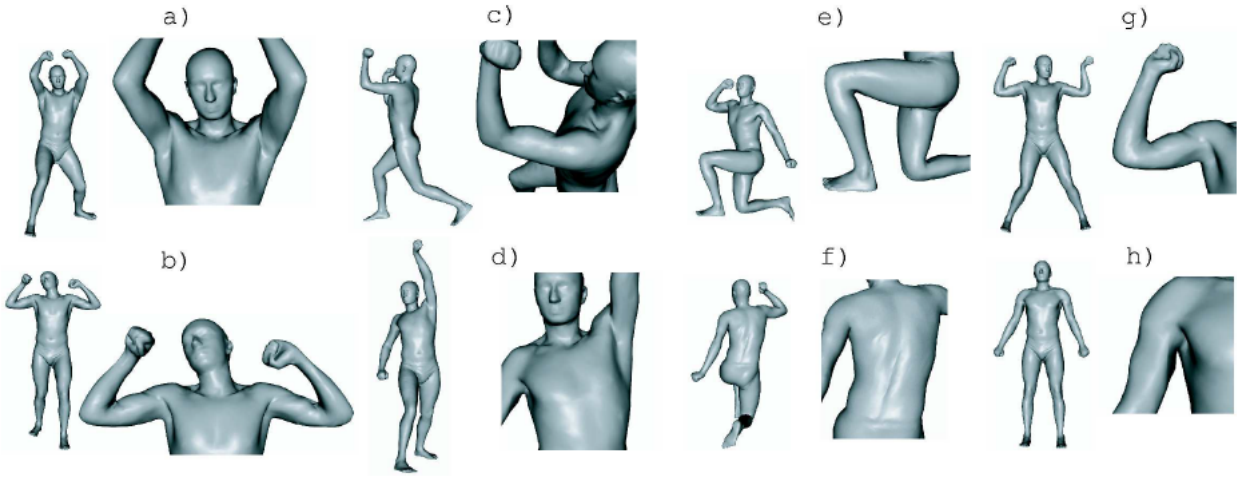


Figure 3: Example of capturing muscle deformation in SCAPE pose models

The model captures shoulder deformation, biceps bulging, and spine twisting well, and performs reasonably well in handling elbow and knee joints. Example (g) shows minor smoothing of the elbow in some poses. Due to the hole filling in the template mesh, a defect appears in the armpit area.

For a given matrix, generating each mesh takes about 1 second, which is 1.5 orders of magnitude slower than real-time, indicating the potential for using this type of model in real-time animation synthesis or caching motion sequences.

2.1.3 Body Shape Deformation

The SCAPE model also encodes body shape differences among individuals, assuming that the scan data in the training set Y^i corresponds to different individuals.

Deformation Process. Body shape deformation is modeled independently of pose variations by introducing a new set of linear transformation matrices S_k^i, corresponding to each instance mesh i and each triangle p_k. Assume that the triangle p_k observed in the instance mesh i is obtained by first applying the pose deformation matrix Q_k^i, then the body shape deformation

matrix S_k^i, and finally the rotation associated with the respective joint $R_{l[k]}^i$. The sequential application of transformation matrices preserves the correct scaling of deformation. We obtain the following extension of equation (2.1):

$$v_{k,j}^i = R_{l[k]}^i S_k^i Q_k^i \hat{v_{k,j}}. \tag{2.7}$$

Thus, the body shape deformation associated with each subject i can be modeled as a set of matrices $S^i = \{S_k^i : k = 1, \cdots, P\}$.

Learning the Body Shape Deformation Model. To capture the body shape deformation space, different matrices S^i are viewed as part of a low-dimensional subspace. For each instance mesh, we create a vector of size $9 \times N$ containing the parameters of the matrix S^i. Assuming these vectors are generated by a simple linear subspace, they can be estimated using PCA:

$$S^i = \overline{U\beta^i + \mu}. \tag{2.8}$$

where $U\beta^i + \mu$ is the PCA reconstruction from the vector form of the $9 \times N$ matrix coefficients, and $\overline{U\beta^i + \mu}$ represents this vector as a set of matrices. PCA is suitable for modeling matrix entries, as the deformations caused by body shape are consistent and not too strong; even shapes differing from the mean by three standard deviations still appear very human-like, as shown in Figure 4.

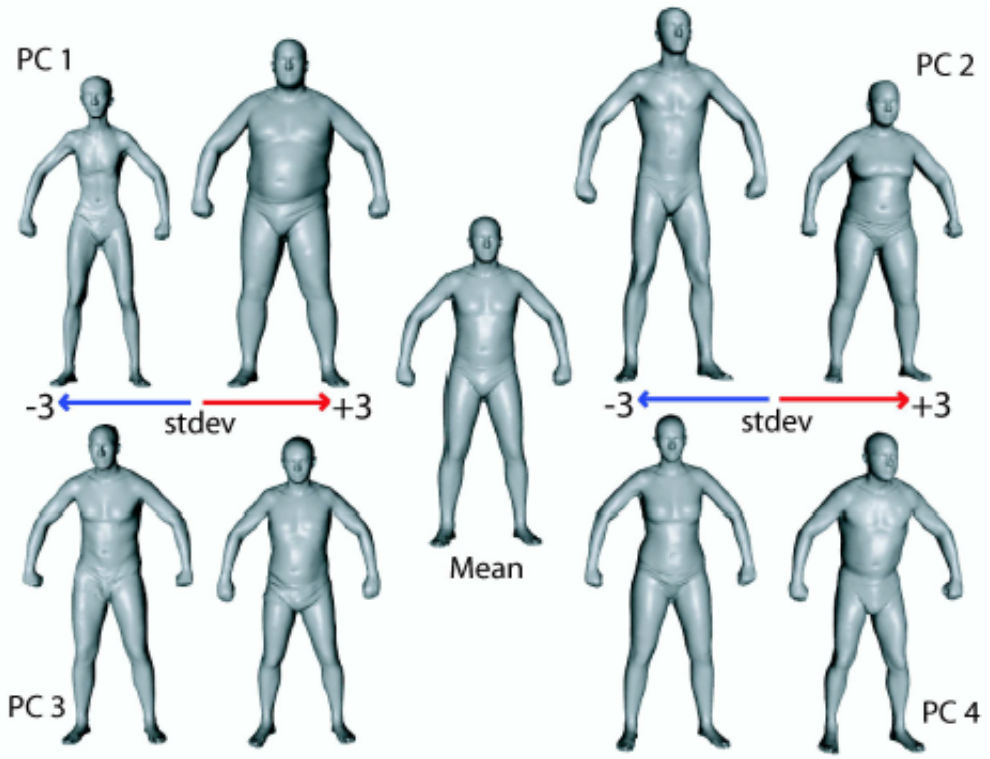


Figure 4: The first four principal components in the body shape deformation space

If the affine matrices S_k^i are known for each i and k, PCA parameters U, μ, and mesh-specific coefficients β_i can be solved. Similar to the pose deformation case, the body shape transformation matrices S_k^i are unknown and need to be estimated. To solve for S_k^i:

$$\arg\min_{S^i} \sum_k \sum_{j=2,3} \left\| R_k^i S_k^i Q_k^i \hat{v_{k,j}} - v_{k,j}^i \right\|^2 + w_s \sum_{k_1,k_2} \left\| S_{k_1}^i - S_{k_2}^i \right\|^2. \tag{2.9}$$

The data preprocessing phase provides estimates of the joint rotations R^i in each instance mesh, allowing us to compute the joint angles Δr^i. From these angles, the predicted pose deformations Q_k^i can be calculated using the learned pose deformation model, so the only unknown in equation (2.9) is the body shape transformation matrix S_k^i. This equation is quadratic in these unknowns and can be solved using least squares optimization.

Application to Dataset. This method was applied to a dataset of 45 body shape instances to learn the SCAPE body shape deformation model. Figure 4 shows the first four principal components of the body shape deformation space. These components represent reasonable variations in weight and height, gender, abdominal fat and chest muscles, and chest-to-hip fullness.

The PCA space covers a wide range of body shapes. Combined with the pose model, realistic scans of various people in a wide range of poses can now be synthesized. Given a set of rigid part rotations R and body shape parameters β, the joint rotations R determine the joint angles ΔR. For a given triangle p_k, the pose model now defines the deformation matrix Q_k, and the body shape model defines the deformation matrix S_k. Similar to equation (**??**), solve for the vertices Y that minimize the objective function.

Using this method, meshes can be generated for any body shape in any pose within the PCA space. Figure 5 shows some examples of different synthesized scans, demonstrating realistic muscle deformations for very different subjects and a wide range of poses.

2.1.4 Application Example —Motion Capture Animation

This model can be applied to image completion; the algorithm is more complex, and we reference the results of this algorithm to introduce another application.

The shape completion framework can be applied to generate animations from captured marker motion sequences. In this case, there is a sequence of frames with some markers, each specifying the 3D positions of these markers. Treat each frame's observed set of markers as the input Z to the image completion algorithm, and use the algorithm to generate a mesh. To generate complete 3D animations, the mesh sequences from different frames can be connected.

It is worth noting that in many motion capture systems, markers protrude from the body, leading to recovered meshes that may contain unrealistic deformations. Therefore, we do not use the completed meshes $Y[Z]$, but rather use the predicted meshes $Y\hat{[Z]}$. Since these meshes are constrained by the body shape encoded by the PCA model, they tend to avoid these unrealistic deformations. This method was applied to two captured motion sequences, both for the same subject S. It is worth noting that the dataset contains only a single scan of

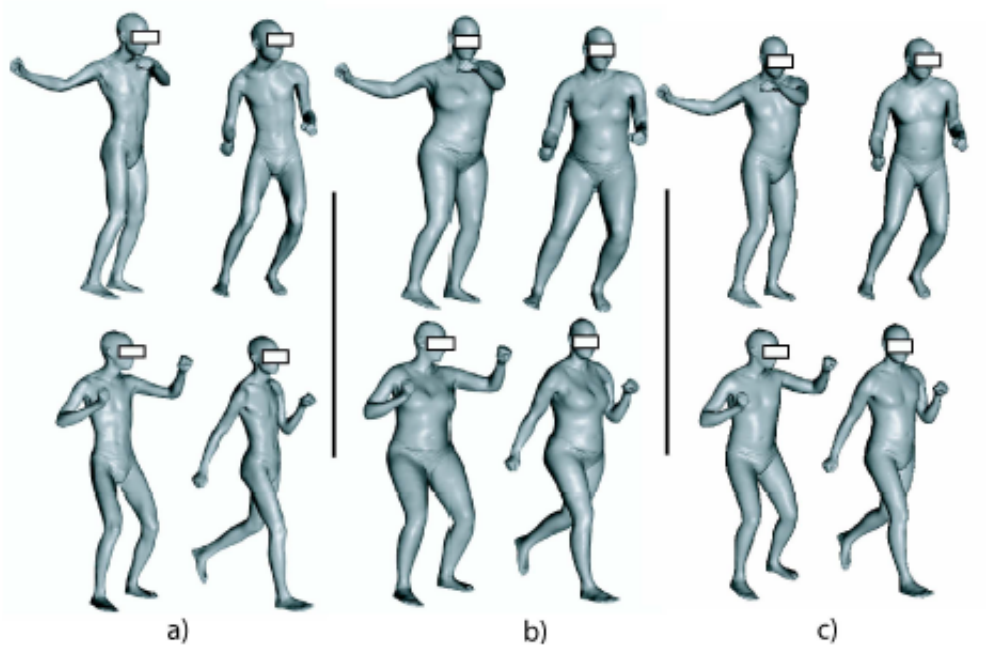


Figure 5: Examples of different synthesized scans

the same subject S, with poses as shown in the third row of Figure 2(a). Each frame in each sequence used 56 markers distributed over the entire body. The subject S was scanned with markers, and the scan results were used to establish correspondences between points on the markers and the subject's surface. The image completion algorithm was then applied to each frame in the sequence. In each frame, the pose R estimated from the previous frame was used as the starting point for optimization. The animation was generated from the sequence of predicted scans $Y[\hat{Z}_f]$. Figure 6 shows some results, with Figure 6(c) demonstrating realistic muscle deformations for subject S, Figure 6(d) showing the transfer of motion to different subjects in the dataset, and Figure 6(e) displaying the replacement of subjects in the motion sequence.

2.1.5 Discussion of Limitations

The SCAPE model decouples the pose deformation model and the body shape deformation model. This design choice greatly simplifies the mathematical expression, improves the model's identifiability from data, and allows for more efficient learning algorithms. However, it also overlooks the strong correlation between body shape and muscle deformation. For example, since everyone uses the same muscle deformation model, we do not capture that more muscled individuals are likely to exhibit greater muscle deformations than others; conversely, muscle deformation might be masked in individuals with higher body fat. Capturing these correlations would require a more expressive model.

The current method requires a series of scans of the same person in different poses to learn pose deformation data. Once this step is completed, scans of different people in different poses can be used to learn body shape deformation data. Currently, there is no method provided

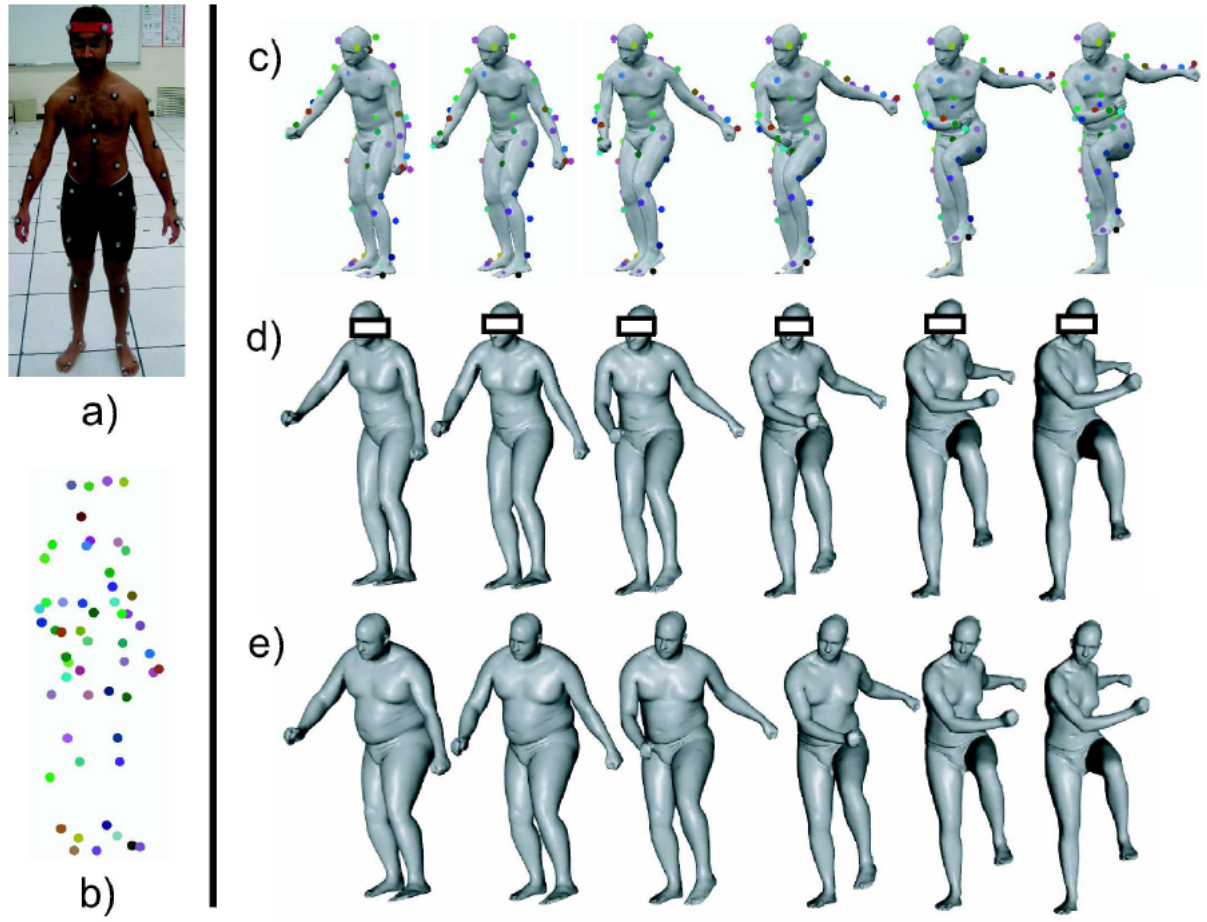


Figure 8: Motion capture animation. (a) Subject wearing motion capture markers (b) Motion capture markers in a single frame (c) An animation of a subject based on a motion capture sequence, with the markers from which the animation was derived superimposed on the meshes. (d) An example of motion transfer to a different subject in our data set. (e) Animation based on motion capture, but where we change the body shape parameters in PCA space as we move through the sequence.

Figure 6: Motion capture animation results demonstration

to learn both types of data from randomly combined data of different people and poses. The assumption about the training set structure is not particularly strict, aiming to simplify data collection and learning processes. One could attempt to learn the model from non-uniform datasets by estimating pose or body shape models while keeping the other model fixed. This process would lead to local minima in the deformation joint space, and the quality of this local minimum depends on the given training data.

In the model, pose deformation is determined by linear regression of adjacent joint angles. This assumption provides surprisingly good animation results and simplifies the shape completeness task. In many partial view completeness instances, a more accurate model may not be needed, as the solutions provided above can deform beyond the SCAPE space to fit observed surfaces. Thus, partial view data can correct some (relatively small) errors caused by the linear regression model assumption. When the SCAPE model is used solely for animation, a more complex model may be required in some cases, and nonlinear regression methods could be considered.

The SCAPE model focuses on representing muscle deformation caused by joint movements, with other deformation factors not encoded, one of which is deformation due to pure muscle activity. Thus, the model cannot distinguish between flexed biceps and relaxed biceps when joint angles are the same. Another factor causing muscle deformation is tissue disturbance due to movement (e.g., fat jiggle), which is also not represented by this model.

Finally, the method is purely data-driven, generating the entire model from a set of scanned data. Only a small set of markers needs to be placed on the scans to serve as a registration starting point, requiring manual intervention. Therefore, this model can be easily applied to other datasets and is capable of generating models suitable for specific types of body shapes or poses. If the data and desired animations are known, a method for fine-tuning model parameters to meet specific requirements would further benefit the approach.

2.2 SMPL

2.2.1 Introduction

The SMPL (Skinned Multi-Person Linear) model can realistically represent a wide range of body types, naturally exhibit motion variations, demonstrate soft tissue dynamics, efficiently generate animations, and is compatible with existing rendering engines. Figure 7 shows some examples of its work.

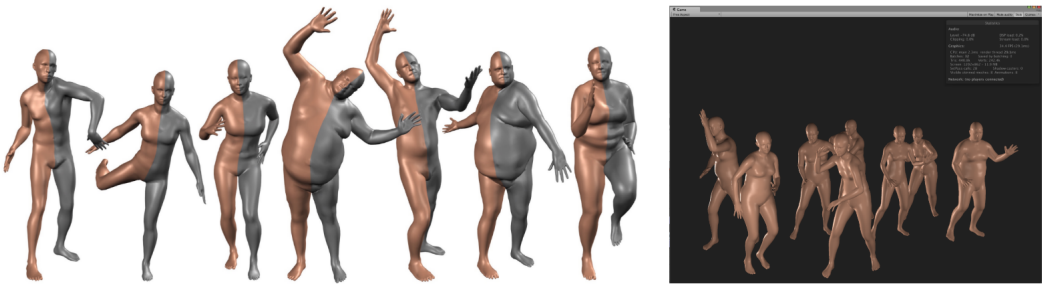


Figure 1: *SMPL is a realistic learned model of human body shape and pose that is compatible with existing rendering engines, allows animator control, and is available for research purposes. (left) SMPL model (orange) fit to ground truth 3D meshes (gray). (right) Unity 5.0 game engine screenshot showing bodies from the CAESAR dataset animated in real time.*

Figure 7: Examples of SMPL's work

Traditional methods model how vertices match with the underlying skeletal structure. LBS (Linear Blend Skinning) models are the most widely used, supporting all game engines and providing high rendering efficiency. However, they produce unrealistic deformations at joints, including "taffy" and "bowtie" effects, as shown in Figure 8. A lot of work is being done to improve such effects[10][11][12]. Much work has also been done on learning highly realistic human models from data[13][14]. These methods capture many body shapes and their non-rigid deformations due to posture, with the most realistic methods based on triangle deformations (discussed in Section 2.1). Despite these efforts, previous models still had many shortcomings, such as lack of realism, incompatibility with existing packages, inability to represent a wide variety of body shapes, incompatibility with standard graphics pipelines, or requiring significant manual labor.

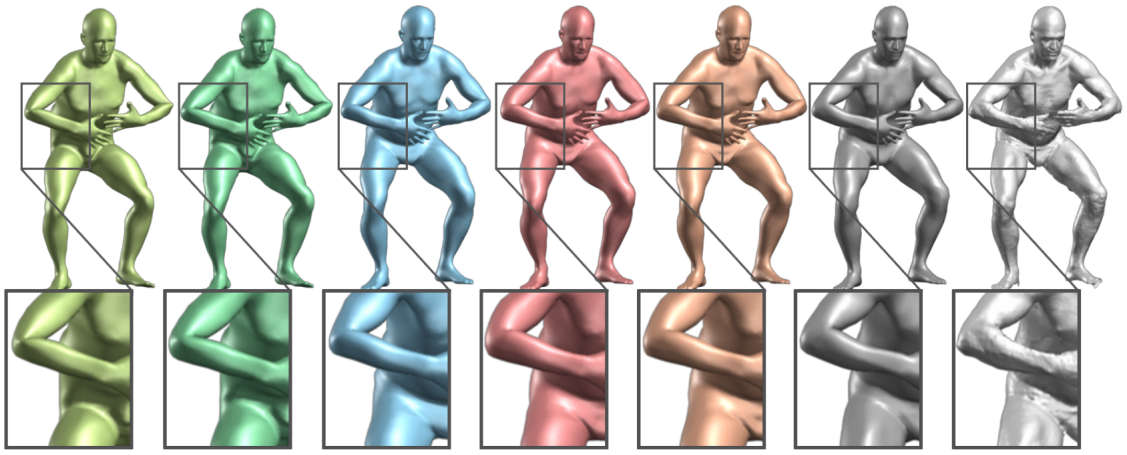


Figure 2: Models compared with ground truth. *This figure defines the color coding used throughout the paper and* **Supplemental Video**. *The far right (light gray) mesh is a 3D scan. Next to it (dark gray) is a registered mesh with the same topology as our model. We ask how well different models can approximate this registration. From left to right: (light green) Linear blend skinning (LBS), (dark green) Dual-quaternion blend skinning (DQBS), (blue) BlendSCAPE, (red) SMPL-LBS, (orange) SMPL-DQBS. The zoomed regions highlight differences between the models at the subject's right elbow and hip. LBS and DQBS produce serious artifacts at the knees, elbows, shoulders and hips. BlendSCAPE and both SMPL models do similarly well at fitting the data.*

Figure 8: Examples of "taffy" and "bowtie" effects

Compared to previous methods (which were developed in 2015 and continue to be updated), a key goal of the developers' work was to make the body model as simple and standardized as possible to ensure its wide usability while maintaining the realism of deformation models learned from data. Specifically, they aimed to learn blend shapes to correct the limitations of standard skinning[1], combining different blend shapes for identity, pose, and soft tissue dynamics with a static template before applying blended skinning transformations. A key component of the SPML method is the use of linear functions of parts of rotation matrix elements to model pose blend shapes. This method, different from previous methods, simplifies training and blending shape animation. Since rotation matrix elements are bounded, the resulting deformations are also bounded, which helps the model generalize better.

The model includes a loss function that penalizes inconsistencies between each vertex of the registered mesh and the model, enabling the model to be trained on data. To understand how people deform with poses, the model uses high-resolution 3D scan data of 1786 different objects in various poses. The developers aligned the template mesh with each scan to create the training set; optimized blend weights, pose-dependent blend shapes, rest poses, and shape-to-joint position regressions to minimize vertex errors on the training set. The joint regressor predicts joint positions as a function of body shape and is crucial for animating realistic pose-related deformations for any body type. All parameters are automatically estimated from aligned scans.

The model learns linear models of male and female body types from the CAESAR dataset, first aligning the template mesh to each scan and normalizing the poses, which is crucial when learning vertex-based shape models. The resulting principal components become the body

[1]Skinning Transformation: A concept in computer graphics and animation used to describe the process of associating the surface of a 3D model with a skeletal structure. This is common in character animation and game development to make the 3D model appear more natural during movement.

shape blend shapes. The developers trained SMPL (Skinned Multi-Person Linear) models in various forms and performed quantitative comparisons with BlendSCAPE models[15] trained with the exact same data. They quantitatively evaluated these models with animations and meshes not used for training, fitting SMPL and BlendSCAPE to these meshes, and comparing vertex errors. They then explored two main variants of SMPL, one using linear blend skinning (LBS) and the other using dual quaternion blend skinning (DQBS). The conclusion was that vertex-based skinning models like SMPL are actually more accurate than deformation-based models like BlendSCAPE, which were trained with the same data.

The developers extended the SMPL model to capture soft tissue dynamics by adapting the Dyna model[16], resulting in the Dynamic-SMPL, or DMPL model. DMPL was trained on the same 4D mesh dataset used by Dyna. However, DMPL is based on vertex deformations rather than triangle deformations. The developers calculated vertex errors between SMPL and Dyna training meshes in a static pose, used PCA for dimensionality reduction, and generated dynamic blend shapes. They then trained a soft tissue model similar to Dyna's approach, based on partial angular velocity, acceleration, and dynamic deformation history. Since soft tissue dynamics largely depend on body type, DMPL was trained using bodies with different body mass indices, learning a body-type-dependent dynamic deformation model. In standard rendering engines, soft tissue dynamics can be animated simply by calculating dynamic linear blend shape coefficients for a range of poses. When comparing animations of Dyna and DMPL side by side, DMPL appears more realistic. This extension of SMPL demonstrates the versatility of additive blend shape methods, how deformations depend on body type, and how this method provides a scalable basis for modeling body types.

With standard rendering engines, the animation speed of the SMPL model on a CPU is far from real-time. Thus, SMPL addresses a longstanding issue in the field; it allows animators to access a realistically learned model. The design of the SMPL base template considers animation production; it features a low polygon count, simple vertex topology, clear quadrilateral structure, standard binding skeleton, and reasonable facial and hand details. On the other hand, SMPL can be represented as Autodesk Filmbox (FBX) files, which can be imported into animation systems and operated in Maya, Blender, Unreal Engine, and Unity.

This section references the paper by the SMPL developers[17].

2.2.2 Model Construction

The SMPL model is illustrated in Figure 9. Similar to SCAPE, the SMPL model decomposes body shape into shape components related to body type and non-rigid poses. Unlike SCAPE, SMPL uses a vertex-based skinning approach and modified blend shapes. A single blend shape is represented as an offset vector connecting vertices. Starting with a created mesh with N=6890 vertices and K=23 joints, the mesh has the same topology, spatial resolution, clear quadrilateral structure, part segmentation, initial blend weights, and skeletal structure for both male and female bodies. The part segmentation and initial blend weights are shown in Figure 10.

The standard SMPL model contains three main parts:

- **Base Template:** This is a static body model template for male and female bodies. It

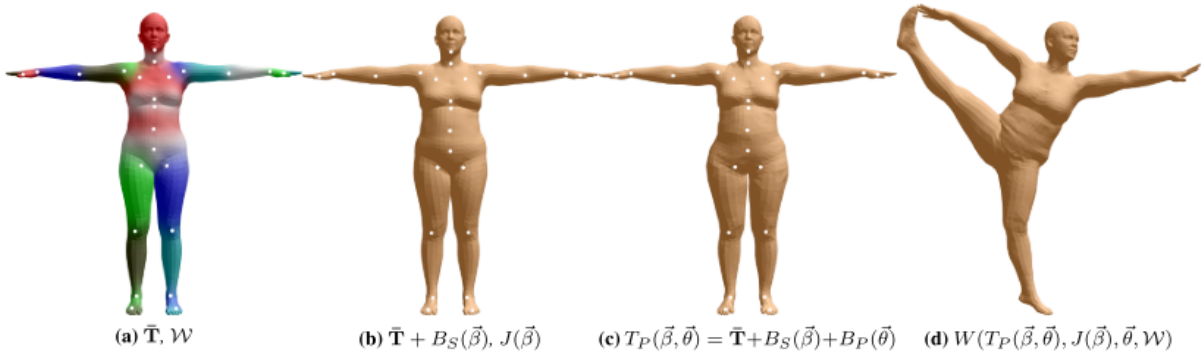


Figure 3: SMPL model. *(a) Template mesh with blend weights indicated by color and joints shown in white. (b) With identity-driven blendshape contribution only; vertex and joint locations are linear in shape vector $\vec{\beta}$. (c) With the addition of of pose blend shapes in preparation for the split pose; note the expansion of the hips. (d) Deformed vertices reposed by dual quaternion skinning for the split pose.*

Figure 9: SMPL Model Example

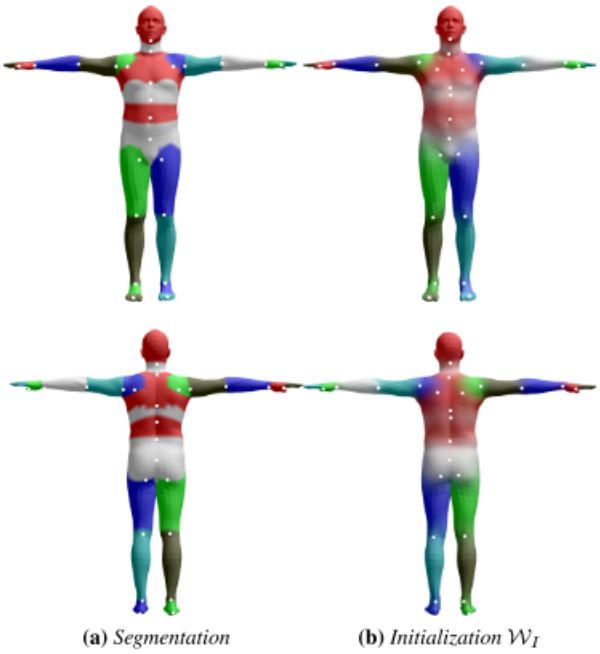


Figure 6: Initialization of joints and blend weights. *Discrete part segmentation in (a) is diffused to obtain initial blend weights, $\mathcal{W}_I$, in (b). Initial joint centers are shown as white dots.*

Figure 10: Part Segmentation and Initial Blend Weights

consists of a low polygon count, simple vertex topology, clear quadrilateral structure, standard binding skeleton, and reasonable facial and hand details. It provides a basis for modeling body shape and pose. The standard SMPL model uses this base template to represent different body types.

- **Blend Shapes:** SMPL uses blend shapes to correct the limitations of standard skinning

models. Blend shapes are learned from data to address issues like "taffy" and "bowtie" effects. Each blend shape represents a deformation offset relative to the base template.

- **Pose-Dependent Deformations:** SMPL models pose-dependent deformations by combining blend shapes with static templates before applying blended skinning transformations. The model learns pose-specific deformations from data and uses rotation matrix elements to model these deformations.

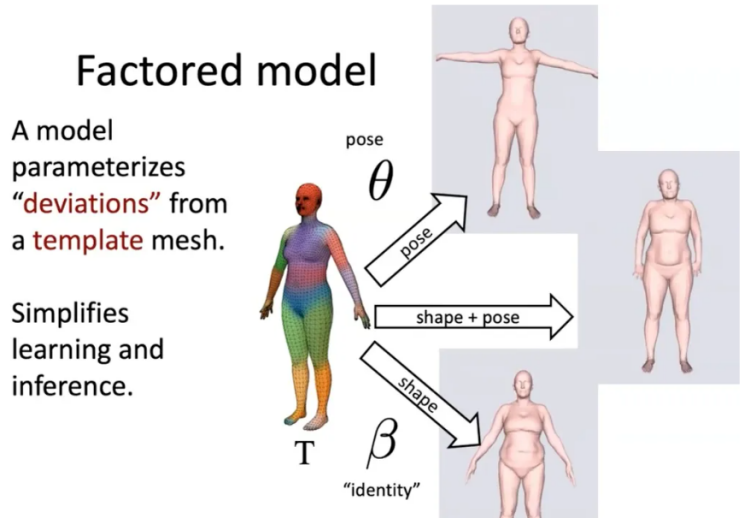


Figure 11: Conceptual Introduction to SMPL Basic Algorithm

However, this does not solve all the problems. The results from scanning devices are point clouds with unordered points, so parametric models cannot be used directly. To enable the SMPL model to work with point cloud results, it is necessary to register based on a common human template. The goal is to align (also known as register) the scanned point clouds so that they are ordered and correspond to the model. The alignment process of point clouds and the model is shown in Figure 12.

There are many challenges in the alignment step, such as:

1. The generated scanning point clouds are generally high resolution, but the corresponding template is often not, due to computational resource limitations, leading to mismatches.

2. Smooth areas have less edge information, and points on smooth areas may match with adjacent regions, causing errors.

3. Pose diversity, contact between adjacent areas, and missing data in the point cloud all affect the accuracy of alignment.

To address these issues, a collaborative alignment algorithm[18] was introduced. This algorithm improves alignment accuracy by learning a precise prediction model and using better registration results to enhance prediction accuracy. To enhance model stability, the algorithm also searches for global poses and uses prior poses in model optimization.

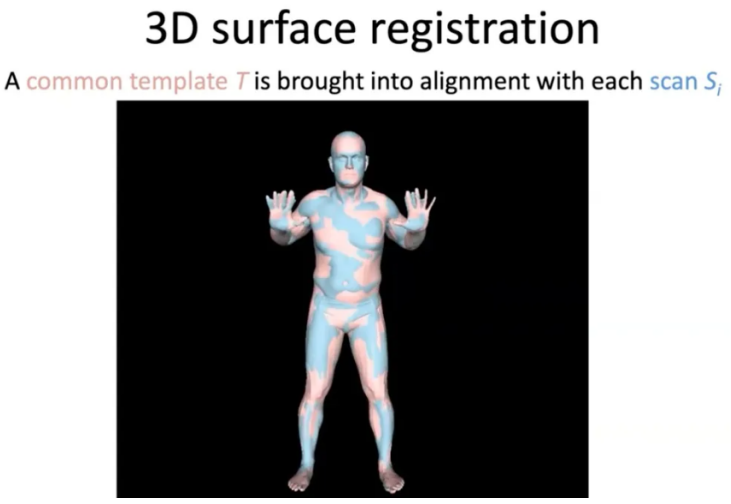


Figure 12: Alignment Process of Point Clouds (Blue) with Model (Red)

After solving the alignment issue, we consider dimensionality reduction for the registered template. The template has 21,000 points, which is too high for feature dimensions, leading to insufficient computational resources and poor model learning. Therefore, it is necessary to reduce the dimensionality of the ground truth. It should be noted that pose normalization has already been done, so pose effects on shape do not need to be considered again. Dimensionality reduction is performed using Principal Component Analysis (PCA). First, normalize the shape (subtract the shape mean after vectorizing each vertex) to place the shape in Euclidean space, then apply PCA for dimensionality reduction. Typically, feature dimensions between 10 and 300 are used; larger dimensions retain more details. For SMPL models, a value of 10 is generally considered, as it retains about 93% of the information, which is sufficient for standard precision modeling. The information content for different values is shown in Figure 13.

After shape modeling, we further discuss standard skinning. Standard skinning is a common modeling technique used in animation, which uses static vertex positions, joint positions, and blend weights[2] to generate the skinning of the human body. The definition of parameters is shown in Figure 14.

Linear Blend Skinning (LBS) assumes that the process of transforming body poses from static poses is a combination of vertex changes influenced by the pose. Given static vertices, after applying pose adjustments, the vertices transform into the body shape for the given pose, simplifying the model to a linear combination of specific vertices. Although linear blend skinning is simple and easy to understand, it has obvious problems, such as failing to maintain volume conservation, leading to deformations. For example, using LBS on vertices near the elbow can result in elbow collapse (Candy Wrapper effect). This effect occurs due to insufficient vertex constraints from direct pose transformations. The solution is to learn blend shapes. Blend

[2]Blend weight: The W matrix, usually describes the influence of multiple bones on vertices in a mesh model, specifying the degree of influence of each bone on the vertices and thus determining the deformation of the vertices in the animation. Blend weights are typically represented as a set of weight values, each corresponding to an associated bone, and these weights are used to compute the final position of vertices under bone transformations. By adjusting blend weights, animators can precisely control each bone's effect on the model, achieving more natural and smooth animation effects.

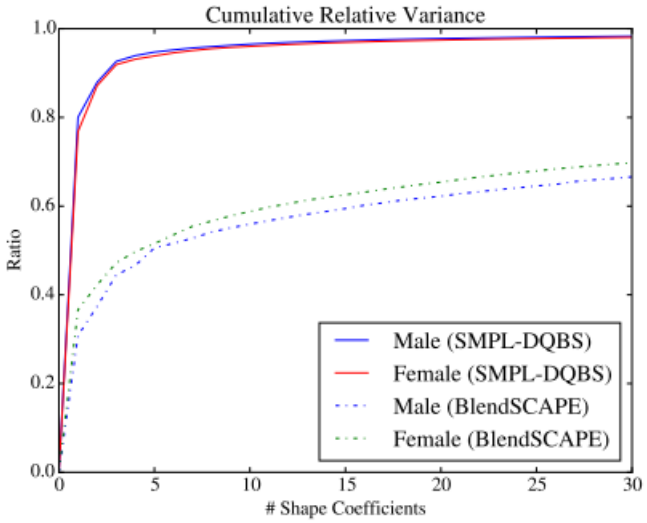


Figure 9: *Cumulative relative variance of the CAESAR dataset explained as a function of the number of shape coefficients. For SMPL the variance is in vertex locations, while for BlendSCAPE it is in triangle deformations.*

Figure 13: Information Content for Different Feature Dimensions of SMPL Model

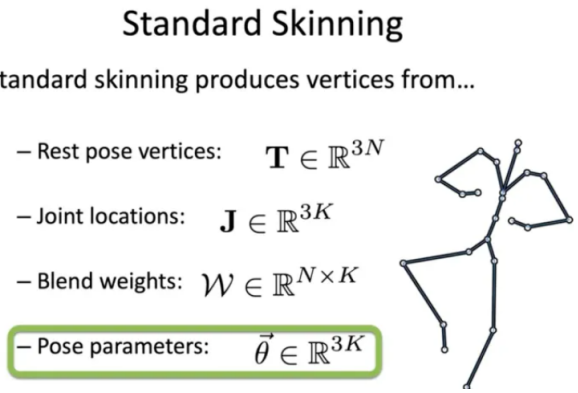


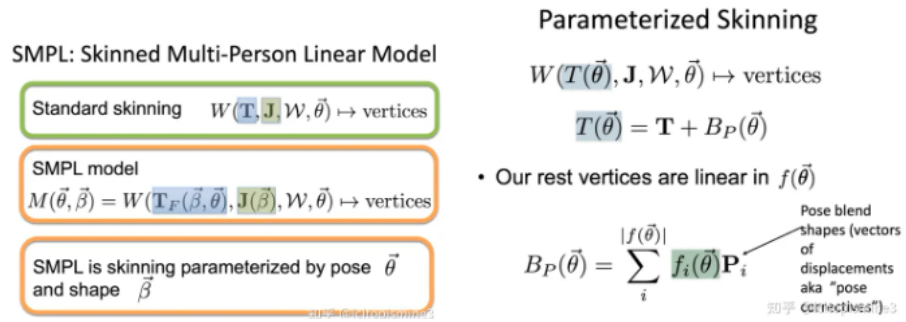
Figure 14: Definition of Parameters in SMPL Model

shapes are displacement vectors of vertices under static poses, ensuring volume conservation. Based on this, the core algorithm of the SMPL model was proposed, as shown in Figure 15.

In Figure 15, human body modeling is represented in terms of principal vertices, determined by static pose vertices T, pose-influenced vertex displacement $B_p(\bar{\theta})$, and shape-influenced vertex displacement. First, we discuss pose-influenced vertex displacement. Pose-influenced vertex displacement is determined by the pose vector (to be learned) and the pose weight vector. The pose weights are optimized using the following operations:

1. Pose Correction: The edges of the registered template after pose projection should match those of the average human pose template. Optimization is performed with this as the target:

$$\arg\min_{\bar{\theta}} \sum_{e} ||W_e(\hat{T_\mu}^P + B_P(\bar{\theta}; \mathscr{P}), \hat{T_\mu}^P, \bar{\theta}, \mathscr{W}) - V_{j,e}^S||^2. \tag{2.10}$$



(a) Differences between SMPL Modeling and Standard Skinning

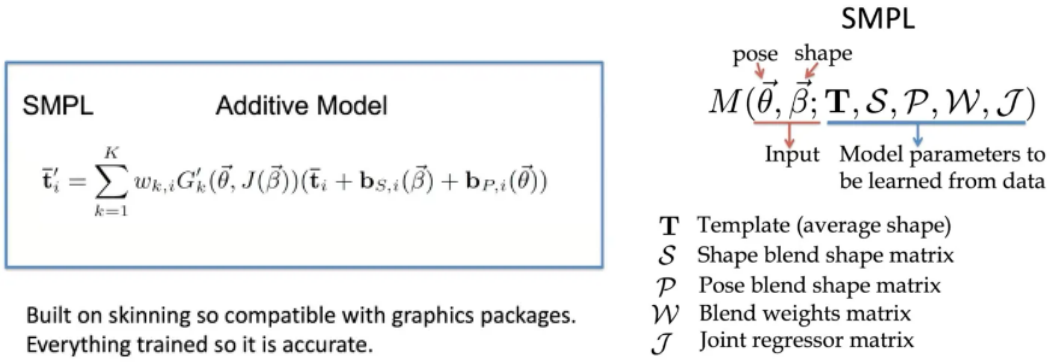
(b) Parametric Modeling Formulas

Figure 15: Core Algorithm of SMPL

2. Fixing the pose, correct the average human pose template: The vertices of the registered template after pose projection should match those of the average human pose template. Optimization is performed with this as the target:

$$\hat{T}_j^{\ S} = \arg\min_{\hat{T}} ||W(\hat{T} + B_P(\bar{\theta}_j; \mathscr{P}), \mathscr{J}\hat{T}, \bar{\theta}_j, \mathscr{W}) - V_j^S||^2. \tag{2.11}$$

Additionally, the influence of vertices must be considered. Controlling only the pose influence allows for applying different actions without distortion, but does not account for shape deformation. Key point modeling must also be considered. Key point modeling can be expressed as a weight vector composed of the static pose vertices T multiplied by the joint weight function. After incorporating the above factors affecting the body shape, the final SMPL additive formula can be obtained, as shown in Figure 16.



(a) SMPL Additive Formula

(b) SMPL Key Parameters

Figure 16: SMPL Additive Formula and Key Parameters

2.2.3 Model Training

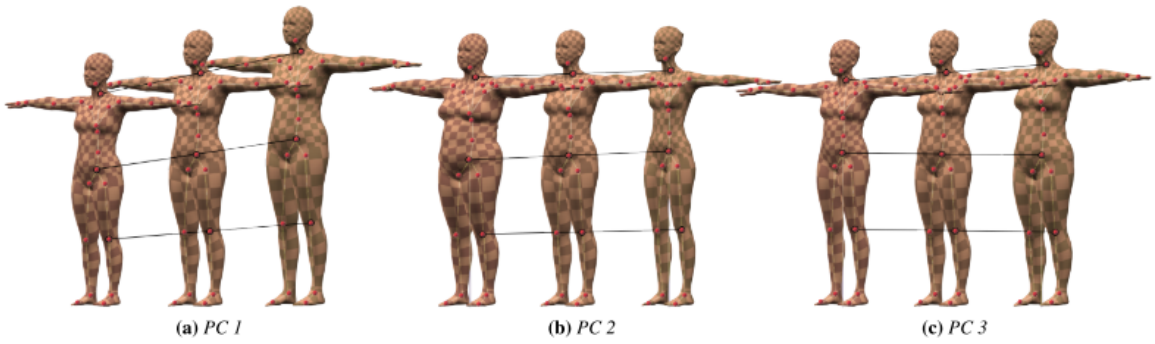
Figure 17 illustrates the training methods for each parameter. Figure 18, sourced from the original paper, shows the shaping of mixed shapes.

Training SMPL

Minimize Euclidean *surface* reconstruction error on the training set.

$\mathbf{T}\ \mathcal{S}$	Trained using multi-shape dataset (mean and principal components)
$\mathcal{P}\ \mathcal{W}\ \mathcal{J}$	Trained using multi-pose dataset
$\mathcal{P}$	Regularized towards zero
$\mathcal{J}$	Regularized towards predicting part boundary centers. Sparse.
$\mathcal{W}$	Regularized towards LBS initial weights

Figure 17: Training Methods for SMPL Parameters



(a) *PC 1* (b) *PC 2* (c) *PC 3*

Figure 8: Shape blend shapes. *The first three shape principal components of body shape are shown. PC1 and PC2 vary from -2 to +2 standard deviations from left to right, while PC3 varies from -5 to +5 standard deviations to make the shape variation more visible. Joint locations (red dots) vary as a function of body shape and are predicted using the learned regressor,* $\mathcal{J}$.

Figure 18: Shaping of Mixed Shapes in SMPL

2.2.4 Model Evaluation

The developers evaluated the SMPL model from five dimensions: quantitative evaluation, sparsity, visual quality, runtime, and compatibility with rendering engines, demonstrating its superiority over the SCAPE model. The model evaluation link provided in the paper is: http://smpl.is.tue.mpg.de.

2.2.5 Introduction to DMPL

Although the SMPL model can model static soft tissue deformation through poses, it cannot simulate dynamic deformations due to body movement and collisions with the ground. Considering 4D registration data that includes soft tissue dynamics, the pose and personalized template shape of the SMPL model are optimized. Displacements between the SMPL model and the observed mesh correspond to dynamic soft tissue movement. To model these, a set of new additional mixed shapes, termed dynamic mixed shapes, is introduced. These additional displacements are related to the body's and limbs' velocities and accelerations, and not to poses. Following the approach of Dyna[19], the same training data is used, and these ideas are applied to an additional vertex-based model, resulting in the DMPL model, which produces more realistic soft tissue dynamics compared to Dyna.

2.2.6 Discussion of Limitations

The pose-related offsets in SMPL are independent of body shape. SMPL works well without this limitation, but modeling unrealistic characters with significantly different body part scales, or including human spaces like infants and adults, might not be effective with traditional methods. This limitation can be addressed by training a more general function. Since dynamic mixed shape coefficients do depend on body shape, pose mixed shapes should similarly be adaptable. This will not significantly complicate the model or runtime behavior, but may require more training data.

The SMPL model is solely a function of joint angles and shape parameters: it does not simulate breathing, facial movements, muscle tension, or any changes unrelated to skeletal joint angles and overall shape. If appropriate decomposed data is available, these aspects might be learned as additional mixed shapes (such as DMPL)[20].

2.3 STAR

The STAR model is an upgraded version of the SMPL model, proposed in 2020, which can directly replace the SMPL model. This section refers to the paper by the STAR developers[21].

The paper first points out some drawbacks of the SMPL model, supplementing section 2.2.6:

SMPL enhances traditional linear blend skinning (LBS) by learning pose-related correction offsets from 3D scans. Specifically, SMPL uses pose-corrected mixed shape functions $\mathscr{P}(\theta)$: $R^{|\theta|} \rightarrow R^{3N}$, where N is the number of mesh vertices. The function $\mathscr{P}$ predicts correction offsets for each mesh vertex to ensure that the output mesh looks realistic when posed. It can be viewed as a fully connected layer (FC) that associates correction offsets for each mesh vertex with elements of rotation matrices for all body joints. This dense mixed shape formula has several drawbacks. Firstly, it significantly increases the number of model parameters, exceeding 4.2 million, making SMPL prone to overfitting during training. Even with extensive regularization, the model may learn spurious correlations in the training set, as shown in Figure 19(a), where moving one elbow causes another elbow to bulge. This is problematic for graphics, model fitting, and deep learning, as dense formulas lead to dense spurious gradients propagating through the model, and mesh surface loss will backpropagate false gradients to geographically distant joints. Existing pose-corrected mixed shape formulas limit the model's compactness and visual realism.

SMPL distinguishes between shape changes caused by body shape and those caused by pose: this is advantageous as it produces a simple model with additive shape functions; however, it is a weakness as it fails to capture the correlation between body shape and how soft tissues deform with pose.

To address Issue 1 mentioned, the developers created a new compact human model called STAR (Sparse Trained Articulated Regressor). Compared to SMPL, the STAR model is more accurate and features sparse and spatially localized blend shapes, meaning that a joint affects only a group of sparse vertices geographically close to it. The original SMPL paper also acknowledged this issue and proposed a model called SMPL-LBS-Sparse, which restricts

(a) (b)

Fig. 2: **SMPL Limitations**: Examples of some SMPL limitations. Heat maps illustrate the magnitude of the pose-corrective offsets. Fig. 2a highlights the spurious long-range correlations learned by the SMPL pose corrective blend shapes. Bending one elbow results in a visible bulge in the other elbow. Fig. 2b shows two subjects registrations (show in blue) with two different body shapes (High BMI) and (Low BMI). While both are in the same pose, the corrective offsets are different since body deformation are influenced by both body pose and body shape. The SMPL pose corrective offsets are the same regardless of body shape.

Figure 19: Limitations of SMPL as highlighted in the STAR paper

pose-corrected blend shapes so that a vertex is influenced only by the joint with the highest skinning weight. However, SMPL-LBS-Sparse has lower accuracy compared to SMPL.

To address Issue 2 mentioned, the developers extended the existing pose correction formula by regressing correction terms using body pose θ and shape β. Here, the second principal component of the body shape space, which is highly correlated with body mass index (BMI), is used.

The core consideration of the STAR model is that the influence of body joints should be inferred from training data. The main challenge is to devise a model and training objective to learn meaningful joint support regions that are sparse and spatially localized, as shown in Figure 20. To achieve this, the developers formalized a differentiable threshold function based on a rectified linear unit (ReLU) operator, which learns to predict the activation of irrelevant vertices in the model as zero. The output activations are used to mask the joint blend shape regressor output, affecting only vertices with non-zero activations. This results in a sparse pose-related deformation model.

The STAR model further enhances model compactness. SMPL uses Rodrigues' representation for joint angles[3], applying a separate pose correction regressor for each matrix element, with nine regressors per joint. The developers switched to quaternion representation, requiring only four numbers per joint without losing performance. Combined with sparsity, this means that STAR's parameters are only 20% of those of SMPL. The developers assessed STAR's performance by training it on different datasets: STAR was found to be more accurate than SMPL when trained on the same data, preserving test data accuracy better.

[3]Rodrigues' representation: A mathematical method for representing rotations. It represents three-dimensional rotation as a combination of an axis vector and a rotation angle. This representation facilitates rotation operations and is widely used in computer graphics, computer vision, and robotics.

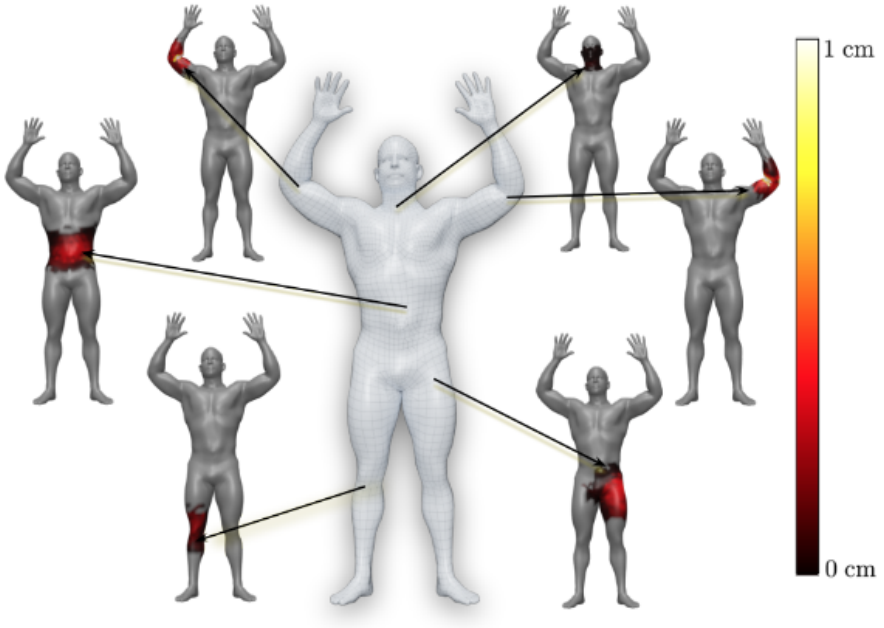


Fig. 1: **Sparse Local Pose Correctives:** STAR factorizes pose-dependent deformation into a set of sparse and spatially local pose-corrective blendshape functions, where each joint influences only a sparse subset of mesh vertices. The white mesh is STAR fit to a 3D scan of a professional body builder. The arrows point to joints in the STAR kinematic tree and the corresponding predicted corrective offset for the joint. The heat map encodes the magnitude of the corrective offsets. The joints have no influence on the gray mesh vertices.

Figure 20: STAR Sparse Local Pose Correction

Improvements in datasets are also addressed. The SMPL shape space was trained using the CAESAR database, which contains 1,700 male and 2,107 female subjects. However, the body data in the CAESAR database is distributed according to the 1990 US population distribution and does not reflect current global body statistics. The SMPL trained on CAESAR cannot capture recent and more diverse variations in the SizeUSA dataset, which includes 10,000 subjects (2,845 males and 6,436 females). To address these issues, the developers combined CAESAR and SizeUSA scan data to train the STAR model, allowing STAR to generalize better to unseen body shapes.

Figure 21 demonstrates STAR's superiority over SMPL in various applications.

2.4 Summary of Other Works

This section primarily refers to the abstracts of related model papers. Among them, Video Avatar uses optimization-based methods, while HMR, Octopus, GHUM, GHUML, and HPS use learning-based methods.

1. **Computer vision:** We propose a compact model that is 80% smaller than SMPL. We achieve compactness in two ways: First, we formalize sparse corrective blend shapes and learn the set of vertices influenced by each joint. Second, we use quaternion features for offset regression. While STAR is more compact than SMPL, it generalizes better on held-out test data.
2. **Graphics:** Non-local deformations make animation difficult because changing the pose of one body part affects other parts. Our local model fixes this problem with SMPL.
3. **Health:** Realistic avatars are important in health research. We increase realism by conditioning the pose corrective blend shapes on body shape. Bodies with different BMI produce different pose corrective blend shapes.
4. **Clothing Industry:** Accurate body shape matters for clothing. We use the largest training set to date to learn body shape and show that previous models were insufficient to capture the diversity of human shape.

Figure 21: STAR's Superiority over SMPL in Applications

2.4.1 SMPL Series

The SMPL series includes several models based on the SMPL model, such as "SMPL-X" and "SMPL+H." Differences between these models are shown in Figure 22[22].

Model	Keypoints	v2v error	Joint error
"SMPL"	Body	57.6	63.5
"SMPL"	Body+Hands+Face	64.5	71.7
"SMPL+H"	Body+Hands	54.2	63.9
SMPL-X	Body+Hands+Face	52.9	62.6

Table 1: Quantitative comparison of "SMPL", "SMPL+H" and SMPL-X, as described in Section 4.2, fitted with SMPLify-X on the EHF dataset. We report the mean vertex-to-vertex (v2v) and the standard mean 3D body (only) joint error in mm. The table shows that richer modeling power results in lower errors.

Figure 22: SMPL Family and Differences

2.4.2 HMR

This model was proposed and published in 2018. HMR (Human Mesh Recovery) is an end-to-end framework that can reconstruct a complete 3D human mesh from a single RGB image. Unlike most current methods that compute 2D or 3D joint positions, this model generates a richer and more useful mesh representation parameterized by shape and 3D joint angles. The main objective is to minimize the projection loss of keypoints, allowing the model to be trained on images captured in the wild. However, standalone projection loss is highly unconstrained. To address this, the work introduces an adversarial network that uses a large 3D human mesh database to determine whether the human parameters are realistic.[23]

2.4.3 Video Avatar

This model was proposed and published in 2018. It addresses how to obtain accurate 3D human models and textures from a monocular video of a person in motion. Based on parameterized human models, the developers propose a robust processing pipeline that enables 5mm precision 3D model fitting for clothed individuals. The main contribution is a non-rigid deformation corresponding to dynamic human contours, resulting in a visual shell in a common reference coordinate system, thus enabling surface reconstruction. This allows efficient estimation of consistent 3D shapes, textures, and embedded animation skeletons based on a large number of frames.[24]

2.4.4 Octopus

This model was proposed and published in 2019. Octopus is a learning-based model that can infer personalized 3D shapes of people from a small number of frames (1-8 frames) of a monocular video. The individuals in the video are moving, and the reconstruction accuracy is 4 to 5 millimeters, which is several orders of magnitude faster than previous methods. By using semantic segmentation images, the Octopus model can reconstruct 3D shapes, including SMPL parameters, clothing, and hair, in less than 10 seconds. The model achieves rapid and accurate predictions based on two key design choices: first, learning to encode the person's image into a pose-invariant latent code by predicting shapes in a canonical T-pose space; second, predicting with bottom-up and top-down flows (one flow per view) based on feedforward predictions, which are usually fast but not always aligned with the input images, allowing information to flow bidirectionally.[25]

2.4.5 GHUM and GHUML

This model was proposed and published in 2020. The developers proposed a statistical, articulated 3D human shape modeling pipeline embedded in a fully trainable, modular deep learning framework. This framework uses high-resolution complete 3D body scan data that captures human shapes in various poses, as well as close-ups of head and facial expressions, and hand joint motions. All model parameters are trained together in a single, consistent learning loop, including a variational autoencoder-based nonlinear shape space, pose space deformation correction, bone joint center predictor, and mixed skinning functions. These models are trained with all 3D dynamic scan data simultaneously to capture correlations and ensure consistency across components. The model supports facial expression analysis and body (including detailed hand) shape and pose estimation. The developers provide two fully trained generic human models with different resolutions: the medium-resolution GHUM model, with 10,168 vertices; and the low-resolution GHUML model, with 3,194 vertices.[26]

2.4.6 Dyna

This model was proposed and published in 2021. To make digital full-body virtual characters look like real people, they need to exhibit soft tissue deformations similar to those of real humans. However, methods for soft tissue physical simulation lack realism, are computationally expensive, or are difficult to adjust. Learning soft tissue motion from instances is limited by the lack of dense, high-resolution training data. To address this problem, Dyna uses a 4D capture system and a method to accurately register time-varying 3D scans to a template mesh.

By analyzing over 40,000 scans from ten subjects, the soft tissue motion causing deformations in the mesh triangles relative to the base 3D body model is computed, and a low-dimensional linear subspace approximating this soft tissue deformation is learned.

Dyna associates these linear coefficients of body surface deformation with body pose variations and learns a second-order autoregressive model to predict soft tissue deformation based on previous deformations, body velocity and acceleration, and limb angular velocity and acceleration. Dyna also simulates how deformations change with body mass index (BMI), producing different deformation effects for different body shapes, thereby realistically representing soft tissue dynamics for previously unseen subjects and motions. Finally, the developers provide tools for animators to vary BMI to produce different effects, selectively control the position and intensity of soft tissue movements, and apply the model to new, stylized characters.[27]

2.4.7 HPS

This model was proposed and published in 2021. HPS (Human Pose and Location Estimation) jointly estimates the complete 3D human pose and location of a subject in large 3D scenes using only wearable sensors. The developers use IMU[4] data to obtain an approximate 3D body pose, and use a head-mounted camera for self-localization to determine the subject's position in the 3D scene; then, approximate body pose, camera position and orientation, and 3D scene are jointly optimized to obtain the final pose and location estimates.[28]

3 Pixel-Aligned Implicit Function Methods

This section introduces some works on 3D human reconstruction using implicit functions, specifically focusing on the pioneering PIFU model and its specific algorithms, as well as a series of derived works.

3.1 PIFu

3.1.1 Introduction

If digitizing an entire 3D object were as simple as taking a photo, there would be no need for complex 3D scanning devices, multi-view stereo algorithms, or cumbersome capture processes that require moving sensors to different positions. For certain types of objects, such as faces, human bodies, or known man-made objects, it is already possible to infer relatively accurate 3D surfaces from images using parametric models, data-driven techniques, or deep neural networks. Advances in 3D deep learning around 2019 showed that general shapes can be inferred from very few images or even a single input. However, due to inefficient model representations, the resolution and accuracy of results are often limited, even for specific-domain modeling tasks.

To address this challenging issue, a novel pixel-aligned implicit function PIFU (Pixel-Aligned Implicit Function) representation was proposed, which infers the textured surface of dressed

[4]IMU: IMU stands for Inertial Measurement Unit, a device that integrates multiple inertial sensors to measure an object's acceleration, angular velocity, and direction. IMUs typically include accelerometers, gyroscopes, and magnetometers.

3D human bodies from a single or multiple input images. Although most successful 2D image processing deep learning methods (such as semantic segmentation[29], 2D joint detection[30], etc.) utilize "fully convolutional" network architectures to maintain spatial alignment between the image and the output, this is challenging in 3D. Although voxel representations[5] can be applied in a fully convolutional manner, their inherently memory-intensive nature limits their ability to produce finely detailed surfaces. Global representation-based inference techniques[31][32] are more memory-efficient but cannot ensure that details from input images are preserved. Implicit function-based methods[33][33][34][35] rely on the global context of images to infer overall shapes, which may not be accurately aligned with the input images. PIFU aligns local features with the global context of the entire object at the pixel level using fully convolutional methods, avoiding the high memory usage associated with voxel representations, which is particularly important for 3D human reconstruction from single or multiple images.

3.1.2 Detailed Explanation

Specifically, the developers trained an encoder to learn independent feature vectors for each pixel in the image. Given this per-pixel feature vector and a specified z-depth on the camera ray originating from that pixel, the model learns an implicit function that can classify whether a 3D point corresponding to that z-depth is inside or outside the surface. The key aspect is that the feature vectors spatially align the global 3D surface shape with the pixels, enabling the model to retain local details from the input image while inferring reasonable details in unseen regions.

An end-to-end and unified digitization method can directly predict high-resolution 3D shapes of people with complex hairstyles and wearing arbitrary clothing. Despite the presence of large unseen areas, especially in the case of single-view input, this method can generate models similar to those obtained from multi-view stereo or other 3D scanning techniques. As shown in Figure 25, the algorithm can handle various complex garments such as dresses, scarves, and even high heels, while capturing high-frequency details like wrinkles that match the input image at the pixel level.

By simply regressing the RGB values of each query point along the rays using an implicit function, PIFu naturally extends to inferring the color of each vertex. The developers' digitization framework also generates complete textures for the surface while predicting reasonable appearance details in unseen regions. With additional multi-view stereo constraints, PIFu can naturally extend to handling multiple input images, which is often necessary in practical human capture environments. Since a complete textured mesh can already be generated from a single input image, adding more views further improves the results by providing additional information for unseen areas.

This work was completed by the USC Lihao team and was accepted as an Oral presentation at ICCV 2019. This section references the developers' paper[36].

[5]Voxel Representation: A representation that divides 3D space into cubic grids (voxels) to represent an object's 3D shape and structure. Similar to pixels in 2D images, voxels are elements in 3D space that can contain information about the object's properties (such as color, material) and spatial position.

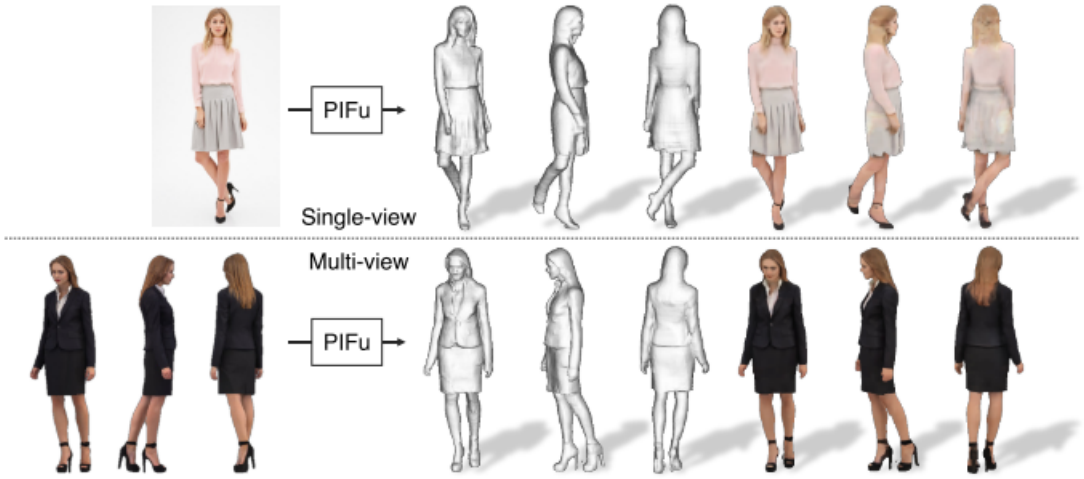


Figure 1: **Pixel-aligned Implicit function (PIFu):** We present pixel-aligned implicit function (PIFu), which allows recovery of high-resolution 3D textured surfaces of clothed humans from a single input image (top row). Our approach can digitize intricate variations in clothing, such as wrinkled skirts and high-heels, including complex hairstyles. The shape and textures can be fully recovered including largely unseen regions such as the back of the subject. PIFu can also be naturally extended to multi-view input images (bottom row).

Figure 23: PIFu Algorithm

3.1.3 Pixel-Aligned Implicit Function

An Implicit Function is a function used to represent the surface of an object, typically in the form $F(X) = 0$, where X represents any point on the surface. The relationship between F and 0 determines the position of the point relative to the surface: equality indicates the point is on the surface, while greater or lesser values may indicate that the point is inside or outside the surface. The Pixel-Aligned Implicit Function proposed in the PIFu method incorporates pixels from 2D images, hence the name Pixel-Aligned. Its function form is as follows:

$$f(F(x), Z(X)) = s. \tag{3.1}$$

where $s \in \mathbb{R}$, $x = \pi(X)$ represents the projection position of the 3D point X in the 2D image, and $Z(X)$ represents the depth value of X in the camera coordinate system of this 2D image. $F(X) = g(I(x))$ represents the feature vector learned from the depth of the 2D image at x, with $g(\cdot)$ consisting of a fully convolutional network.

The function of the Pixel-Aligned Implicit Function is to determine whether a given 3D point X_i is on the surface of an object. First, the 3D point is projected according to camera parameters[6] to obtain the 2D point position x_i and the depth d_i under the camera. At the same time, the image feature vector v_{x_i} at this 2D point location is found. PIFu outputs $f(v_{x_i}, d_i)$ to indicate whether the point is on the object's surface.

The effectiveness of the Pixel-Aligned Implicit Function hinges on the pixel-aligned image

[6]Camera Parameter Projection: The process of projecting a point in the 3D world onto a 2D image plane, mapping the position of the 3D object onto the 2D image. Camera parameters include a set of camera-related information, such as internal parameters (e.g., focal length, principal point) and external parameters (e.g., camera position and orientation). These parameters determine the camera's perspective and imaging method.

feature vectors, allowing the learned $f(\cdot)$ to preserve some details presented in the image, such as wrinkles in clothing, within the reconstructed model. Additionally, the continuous nature of the Pixel-Aligned Implicit Function allows for the reconstruction of geometric information with any topology in a memory-efficient manner.

The Pixel-Aligned Implicit Function also has good extensibility. For example, without changing the function input information, the output can be altered as follows:

$$f(F(x), Z(X)) = rgb. \tag{3.2}$$

where $rgb \in \mathbb{R}^3$, allowing for color prediction. Figure 26 provides an overview of the PIFu framework.

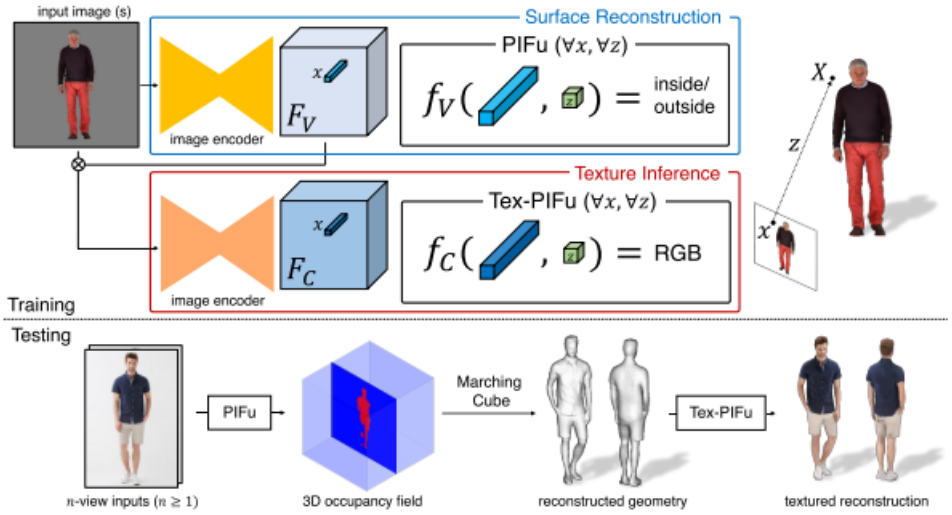


Figure 2: **Overview of our clothed human digitization pipeline:** Given an input image, a pixel-aligned implicit function (PIFu) predicts the continuous inside/outside probability field of a clothed human. Similarly, PIFu for texture inference (Tex-PIFu) infers an RGB value at given 3D positions of the surface geometry with arbitrary topology.

Figure 24: Overview of the PIFu Framework

3.1.4 Single-View Surface Reconstruction

Due to the complex topology of the human body, it is difficult to find an Implicit Function that can be expressed with a formula. We only have some discrete observation points, such as a point inside the body and another point outside the body. Therefore, in PIFu, the GT Implicit Function is expressed as follows:

$$f_v^*(X) = \begin{cases} 1, & \text{if } X \text{ is inside the mesh surface.} \\ 0, & \text{otherwise.} \end{cases} \tag{3.3}$$

Specifically, the developers trained an encoder to learn an independent feature vector for each pixel in the image. Given this per-pixel feature vector and the specified z-depth on the camera ray originating from that pixel, they learned an implicit function that can classify whether a

3D point corresponding to that z-depth is inside or outside the surface. The key is that the feature vectors spatially align the global 3D surface shape with the pixels, allowing the model to preserve local details from the input image while inferring reasonable details in unseen regions.

The end-to-end and unified digitization approach can directly predict high-resolution 3D shapes of people with complex hairstyles and any clothing. Despite the presence of large unseen areas, especially in single-view input cases, the method can generate complete models similar to those obtained from multi-view stereo or other 3D scanning techniques. As shown in Figure 25, the algorithm can handle various complex clothing, such as dresses, scarves, and even high heels, while capturing high-frequency details like wrinkles that match the input image at the pixel level.

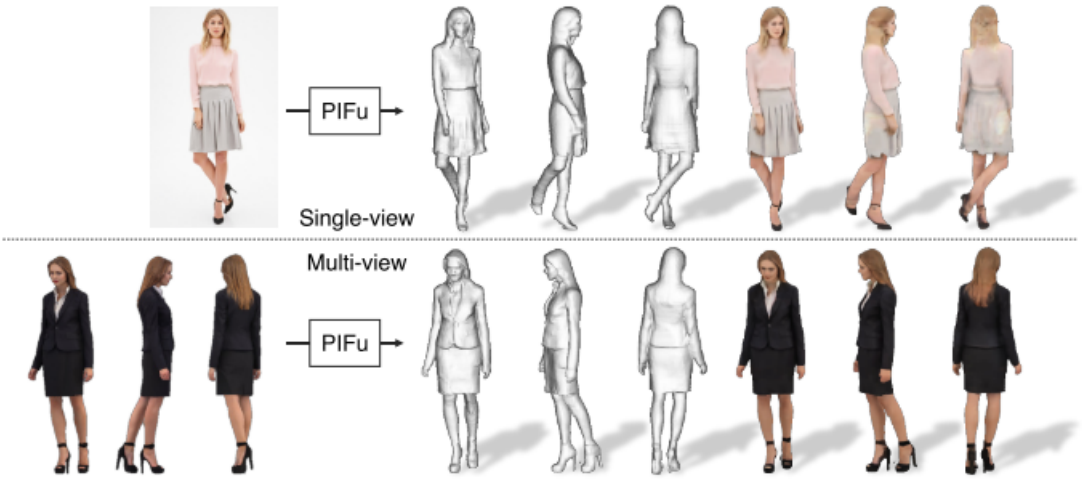


Figure 1: **Pixel-aligned Implicit function (PIFu):** We present pixel-aligned implicit function (PIFu), which allows recovery of high-resolution 3D textured surfaces of clothed humans from a single input image (top row). Our approach can digitize intricate variations in clothing, such as wrinkled skirts and high-heels, including complex hairstyles. The shape and textures can be fully recovered including largely unseen regions such as the back of the subject. PIFu can also be naturally extended to multi-view input images (bottom row).

Figure 25: PIFu Algorithm

By simply using the implicit function to regress the RGB values of each query point along the ray, PIFu can naturally extend to inferring the color of each vertex. The digitization framework developed by the authors also generates complete surface textures while predicting reasonable appearance details in unseen areas. With additional multi-view stereo constraints, PIFu can also naturally extend to handling multiple input images, which is often necessary in practical human capture environments. Since complete textured meshes can already be generated from a single input image, adding more views will only further improve the results by providing additional information for unseen regions.

This work was completed by the USC Lihao team and was accepted as an oral presentation at ICCV 2019. This section references the developers' paper[37].

3.1.5 Pixel-Aligned Implicit Function

An Implicit Function is a function used to express the surface of an object in the form: $F(X) = 0$, where X represents any point on the surface. The relation between F and 0 determines the position of the point relative to the surface: equality indicates the point is on the surface, while greater or lesser values could indicate the point is inside or outside the surface. The Pixel-Aligned Implicit Function proposed in the PIFu method introduces pixels from the 2D image, hence the term "Pixel-Aligned." Its function form is as follows:

$$f(F(x), Z(X)) = s. \tag{3.4}$$

where $s \in \mathbb{R}$, $x = \pi(X)$ represents the projection position of the 3D point X on the 2D image, and $Z(X)$ represents the depth value of X in the 2D image's camera coordinate system. $F(X) = g(I(x))$ denotes the deep learning feature vector at x in the 2D image, with $g(\cdot)$ composed of a fully convolutional network.

The function of the Pixel-Aligned Implicit Function is to determine whether any 3D point X_i is inside or outside the object's surface. First, project the point based on the camera parameters to obtain its 2D point position x_i and depth d_i in that camera. Then, find the image feature vector v_{x_i} at the 2D point position, and PIFu outputs $f(v_{x_i}, d_i)$ to indicate whether the point is on the object's surface.

The effectiveness of the Pixel-Aligned Implicit Function relies on the pixel-aligned image feature vectors, which allows the learned $f(\cdot)$ to preserve details from the image, such as wrinkles on clothing. Additionally, this continuity inherently allows for reconstructing geometrical information of any topology in a memory-efficient way.

The Pixel-Aligned Implicit Function also has good extensibility. For example, without changing its function input, its output can be modified as follows:

$$f(F(x), Z(X)) = rgb. \tag{3.5}$$

where $rgb \in \mathbb{R}^3$, which can be used to predict color. Figure 26 provides an overview of the PIFu framework.

3.1.6 Single-View Surface Reconstruction

Due to the complex topology of the human body, it is difficult to find an Implicit Function that can be expressed with a formula. We only have discrete observation points, such as some points inside the body and others outside. Therefore, in PIFu, the GT Implicit Function is expressed as follows:

$$f_v^*(X) = \begin{cases} 1, & \text{if } X \text{ is inside the mesh surface.} \\ 0, & \text{otherwise.} \end{cases} \tag{3.6}$$

This is essentially the same as the implicit function mentioned above, effectively converting the implicit function's value into the probability that a 3D point is occupied by the object.

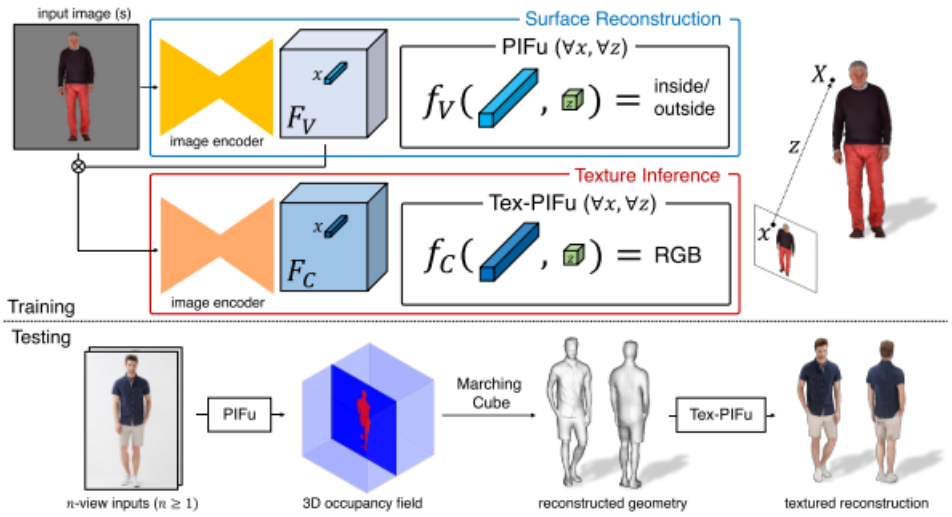


Figure 2: **Overview of our clothed human digitization pipeline:** Given an input image, a pixel-aligned implicit function (PIFu) predicts the continuous inside/outside probability field of a clothed human. Similarly, PIFu for texture inference (Tex-PIFu) infers an RGB value at given 3D positions of the surface geometry with arbitrary topology.

Figure 26: Overview of the PIFu Framework

Values closer to 1 indicate a higher probability of being occupied, while values closer to 0 indicate a higher likelihood of being empty. This also facilitates using the network's Sigmoid output as the output of the Pixel-Aligned Implicit Function. Thus, our goal is to obtain a function $f_v(\cdot)$ that fits the GT Implicit Function $f_v^*(\cdot)$ as closely as possible.

Existing training data consists of m corresponding (2D image, 3D model) pairs. According to the surface reconstruction process of PIFu shown above: for each (2D image, 3D model) pair, the 2D image is first input into an image encoder g consisting of fully convolutional layers to obtain the depth features F_V of the same size as the original 2D image. For the 3D model, n 3D points $\{X_1, X_i, \cdots, X_n\}$ can be sampled, and their corresponding Implicit Function ground truths $\{f_v^*(X_1), f_v^*(X_i), \cdots, f_v^*(X_n)\}$ are known.

Based on this information, the following loss function can be constructed:

$$Loss = \frac{1}{n}\sum_{i=1}^{n} |f_v(F_V(X_i), z(X_i)) - f_v^*(X_i)|^2. \tag{3.7}$$

PIFu uses a Multilayer Perceptron (MLP)[7] to fit the above equation $f_v(\cdot)$. During the gradient descent[8] process, the image encoder g and $f_v(\cdot)$ are jointly optimized.

After training the image encoder g and $f_v(\cdot)$, during inference, the input is an image and the corresponding camera parameters, as well as an approximate range where the person in the

[7]Multilayer Perceptron (MLP): A basic architecture of artificial neural networks. It is a feedforward neural network consisting of multiple layers of neurons, where information flows from the input layer to the output layer through a series of intermediate layers for transformation and processing.

[8]Gradient Descent: An optimization algorithm used to find the local minimum (or maximum) of a function. It is commonly used to adjust model parameters to minimize (or maximize) a loss function.

image is located. The approximate range can be discretized in three dimensions, and inputting all points into the $f_v(\cdot)$ function yields the 3D Occupancy information shown in Figure 26. Running Marching Cubes[38] provides the model of the object.

3.1.7 Spatial Sampling

During training, we need to sample n 3D points $\{X_1, X_i, \cdots, X_n\}$ from the 3D model in the training data. The quantity and method of sampling are crucial for training. The most direct approach is to perform uniform sampling across each dimension of the approximate range of each 3D model, as shown in the left side of Figure 27. However, the PIFu developers found that using this method alone yielded suboptimal results. They needed the network to focus more on information near the object's surface, so the authors sampled around each vertex of the 3D model, introducing jitter in the x, y, and z dimensions, with the jitter distance following a Gaussian distribution $\mathcal{N}(0, \sigma)$, as shown on the right side of Figure 27. PIFu combines these two sampling methods, uniform and Gaussian sampling in a ratio of 1:16, to achieve the best results. Figure 28 shows the impact of different sampling methods on the model's performance.

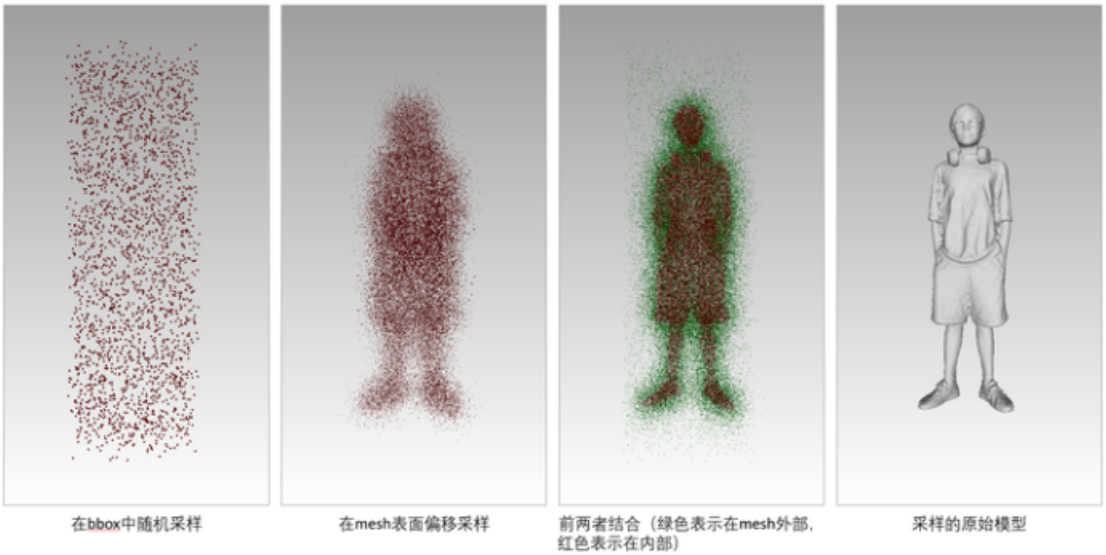


Figure 27: Different Sampling Methods

3.1.8 Human Surface Color Reconstruction

As described in section 3.1.2, by modifying the output of the Pixel-Aligned Implicit Function as in (3.2), it can be used to predict the color of the 3D point X. For this point, the loss function is defined as:

$$Loss = \frac{1}{n}\sum_{i=1}^{n} |f_c(F_C(X_i), z(X_i)) - C(X_i)|. \tag{3.8}$$

$C(X_i)$ represents the color of the 3D point X_i projected onto the 2D image, similar to the surface geometry reconstruction method, optimizing both the image encoder and the implicit

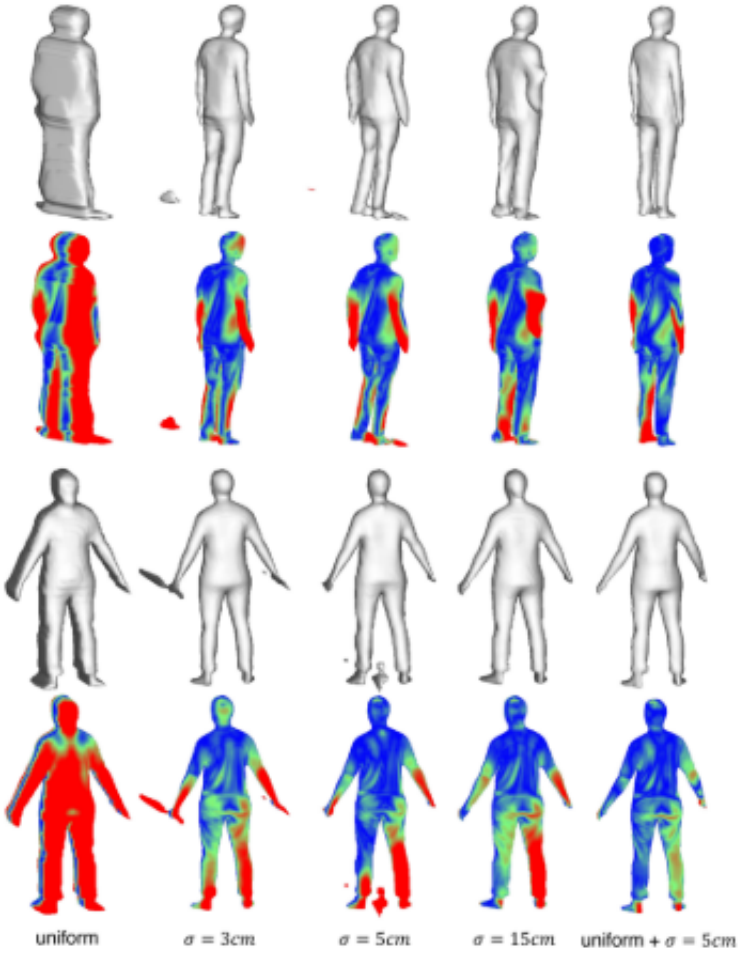


Figure 9: Reconstructed geometry and point to surface error visualization using different sampling methods.

Figure 28: Impact of Different Sampling Methods on Model Performance

function $f_c(\cdot)$. However, the authors found that this approach easily leads to overfitting because it requires predicting the color of surface vertices while also learning the latent 3D information, which is necessary for inferring colors of occluded parts of the body (e.g., back vertices). Therefore, the paper proposes another approach based on the depth features F_V obtained from the previous step. In this case, $f_c(\cdot)$ only needs to focus on the color of vertices and not on latent 3D information, so the loss function becomes:

$$Loss = \frac{1}{n}\sum_{i=1}^{n} |f_c(F_C(X_i, F_V(X_i)), z(X_i)) - C(X_i)|. \tag{3.9}$$

Additionally, the paper introduces an offset $\epsilon \sim \mathcal{N}(0, d)$ for surface vertices, meaning that points within the range of $X_i + \epsilon$ use the color $C(X_i)$. This approach aims to increase training data and enhance the ability of $f_c(\cdot)$ to guess the colors of occluded vertices.

3.1.9 Multi-View Surface Reconstruction

The process of multi-view surface reconstruction is shown in Figure 29.

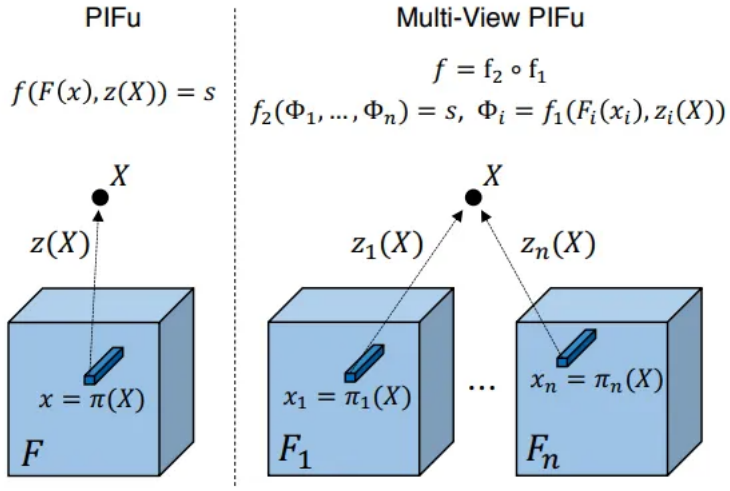


Figure 3: **Multi-view PIFu:** PIFu can be extended to support multi-view inputs by decomposing implicit function f into a feature embedding function f_1 and a multi-view reasoning function f_2. f_1 computes a feature embedding from each view in the 3D world coordinate system, which allows aggregation from arbitrary views. f_2 takes aggregated feature vector to make a more informed 3D surface and texture prediction.

Figure 29: Multi-View Surface Reconstruction Process

For a given 3D point X, features of this point from multiple views can be obtained. These features are then averaged across all views and input into the Pixel-Aligned Implicit Function to determine whether the point is occupied by the model or to get the RGB value of the vertex. The reconstruction results with different numbers of views are shown in Figure 30, indicating that as the number of views increases, the reconstruction quality improves.

3.2 Summary of Other Works

This section primarily refers to abstracts of relevant model papers.

3.2.1 PIFuHD

This model was proposed and published in 2020. PIFuHD is an improved version of PIFu aimed at addressing the problem of PIFu's capability to only represent low-resolution human models. The proposed solution involves first obtaining global depth information and receptive fields through a low-resolution architecture, and then inputting the results from the first stage into a high-resolution architecture to produce finer 3D reconstruction results.[39]

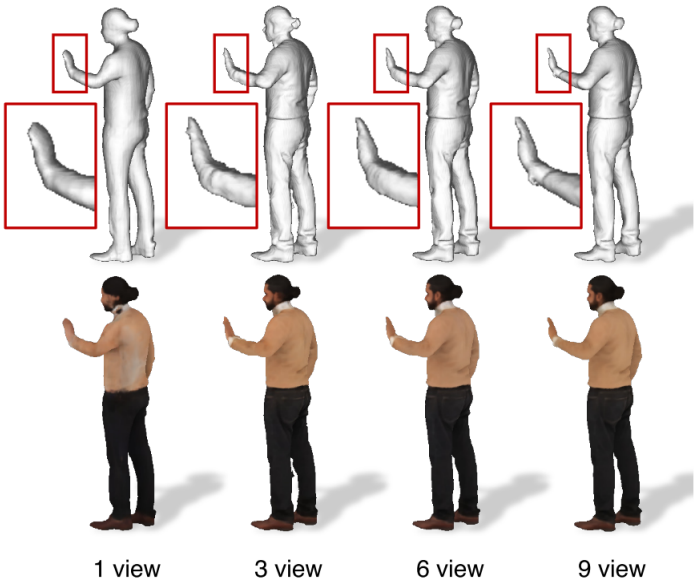


Figure 8: Our surface and texture predictions increasingly improve as more views are added.

Figure 30: Reconstruction Results with Different Numbers of Views

3.2.2 PaMIR

This model was proposed and published in 2021. The PaMIR model combines the SMPL model with the PIFu model, first regressing to SMPL to obtain 3D features, and then combining 2D features from PIFu, inputting them into the implicit function to finally determine whether the sampled points are inside or outside the human body. This work also optimizes parameter optimization methods and loss functions.[40]

3.2.3 Function4D

This model was proposed and published in 2021. It introduces a method combining temporal volumetric fusion and deep implicit functions for human volumetric capture. To achieve high-quality and temporally consistent reconstruction, dynamic sliding fusion is proposed to integrate adjacent depth observations with topological consistency. Additionally, to generate detailed and complete surfaces, a depth implicit function is proposed for RGBD input, which not only retains geometric details on the depth input but also produces more reasonable texture results.[41]

3.2.4 DMC

This model was proposed and published in 2021. The developers introduced a new method called DeepMultiCap for capturing multi-person performances using sparse multi-view cameras. This method can capture time-varying surface details without using pre-scanned template models. To address the challenge of severe occlusion in closely interacting scenes, the paper combines PIFu with parametric models to robustly reconstruct invisible surface areas and designs an effective attention-aware module for capturing fine-grained geometric details from multi-view images, generating high-fidelity results. In addition to the spatial attention

approach, a novel temporal fusion method is proposed for video input to reduce noise and temporal inconsistencies in moving character reconstruction.[42]

3.2.5 ICON

This model was proposed and published in 2022. Currently, methods for learning realistic and animatable 3D dressed avatars either require 3D scanned data with pose control or carefully controlled 2D images of user poses. ICON aims to learn an avatar using only 2D character images with unrestricted poses. Given a set of images, the method estimates detailed 3D surfaces from each image and then combines these surfaces into an animatable avatar. For the first task, implicit functions are used; however, the current method is not robust to varying human poses, often resulting in 3D surfaces with broken or detached limbs, missing details, or non-human shapes. To address this issue, the developers proposed ICON (Implicit Clothed humans Obtained from Normals), which utilizes local features. ICON has two main modules, both leveraging the SMPL(-X) human model.[43]

4 Fusion Methods Based on RGB-D Input

The advent of inexpensive consumer-grade RGB-D cameras[9] has significantly advanced visual scene reconstruction methods. Computer graphics and computer vision researchers have devoted considerable effort to developing new algorithms that utilize RGB-D cameras to capture comprehensive shape models of static and dynamic scenes, leading to significant advancements across multiple dimensions. This section provides a brief overview of classic works related to fusion methods based on RGB-D input, primarily referring to the developers' papers.

4.1 For General Scenes

4.1.1 KinectFusion

This work, published in 2011, has been cited over 4800 times and is highly valuable. It introduces a system capable of real-time, accurate mapping of complex and arbitrary indoor scenes using only a moving, low-cost depth camera and general-purpose graphics hardware under varying lighting conditions. The developers fused all depth data streamed from the Kinect sensor[10] into a global implicit surface model of the scene. By employing an iterative closest point (ICP) algorithm that refines from a coarse to fine model relative to real-time depth frames and simultaneously tracks these frames, the current sensor position is determined. Compared to frame-by-frame tracking, this work demonstrates the advantage of tracking the

[9] RGB-D cameras: devices that capture both color images (RGB images) and depth information simultaneously. Unlike traditional RGB cameras that capture only color information, RGB-D cameras can obtain the distance or depth information of each pixel from the camera. This depth information is usually presented as grayscale images or pixel depth values, allowing each pixel to have both color and distance information.

[10] The Kinect sensor, developed by Microsoft, is a depth camera device widely used in computer vision, virtual reality, and augmented reality. Originally launched in 2010 for the Xbox gaming console, it quickly gained popularity in other fields due to its advanced depth perception and imaging capabilities. The Kinect sensor has been extensively applied in virtual reality, augmented reality, 3D scanning, gesture recognition, and indoor navigation, providing robust technical support for innovations in these areas. However, since 2017, Microsoft has ceased production of new Kinect hardware and discontinued updates to its software development kit.

continually growing complete surface model, achieving tracking and mapping results with limited drift and high accuracy over time in indoor-sized scenes. The real-time natural scene modeling using only general sensors and GPU hardware marks an exciting step for augmented reality (AR), particularly allowing real-time reconstruction of dense surfaces with a level of detail and stability surpassing solutions provided by passive computer vision.[44]

4.1.2 DynamicFusion

This work, published in 2015 by the same developers of KinectFusion, presents the first dense SLAM system capable of reconstructing non-rigidly deforming scenes in real-time by fusing RGBD scan data captured from general sensors. The DynamicFusion method estimates a dense volumetric 6D motion field while reconstructing scene geometry, deforming the estimated shape into real-time frames. Similar to KinectFusion, the system produces more detailed and complete reconstruction results as more measurement data is fused and updates the model in real-time. The method is suitable for various moving objects and scenes since it does not require templates or prior scene models.[45]

4.2 For Human Scenes

4.2.1 DoubleFusion

Published in 2015, DoubleFusion is a new real-time system that combines volumetric dynamic reconstruction with data-driven template fitting to simultaneously reconstruct detailed geometry, non-rigid motion, and internal human shapes from a single depth camera. A key contribution of this method is the dual-layer representation, consisting of a complete parameterized internal human shape and a gradually fused external surface layer. Predefined node graphs on the human surface parameterize non-rigid deformations near the body, while a free-form dynamic change graph parameterizes the external surface layer farther from the body, enabling more general reconstruction. The developers also proposed a joint motion tracking method based on the dual-layer representation for robust and fast motion tracking performance. Additionally, the internal human shape is optimized online and adapted to the external surface layer. Building on DynamicFusion, this method demonstrates reasonable performance in reconstructing internal human shapes, showing improved rapid motion tracking and loop closure detection performance in more challenging scenes.[46]

4.2.2 Fusion4D

Published in 2016 by Microsoft, Fusion4D introduces a new real-time multi-view performance capture pipeline that generates temporally coherent, high-quality reconstruction results. The algorithm supports incremental reconstruction, improving surface estimation over time while parameterizing non-rigid scene motion. The method is robust to large inter-frame motion and topological changes, capable of reconstructing challenging scenes and demonstrating geometric reconstruction results comparable to offline methods requiring orders of magnitude more processing time and RGBD cameras.[47]

4.2.3 Function4D

The content of this work is introduced in Section 3.2.3.

5 Summary and Outlook

Since the emergence of the SCAPE model, in-depth research on 3D human modeling has been ongoing for nearly 20 years. The work in this area is impressive and widely applied in practical life. It is observed that parameterized human models have laid the foundation for the development of 3D human modeling, with models capable of matching the broadest range of objects. However, accuracy may vary depending on conditions, leading to numerous optimization works based on specific objectives. Professor Yutao classifies existing research in this field into roughly four parts, as shown in Figure 31.

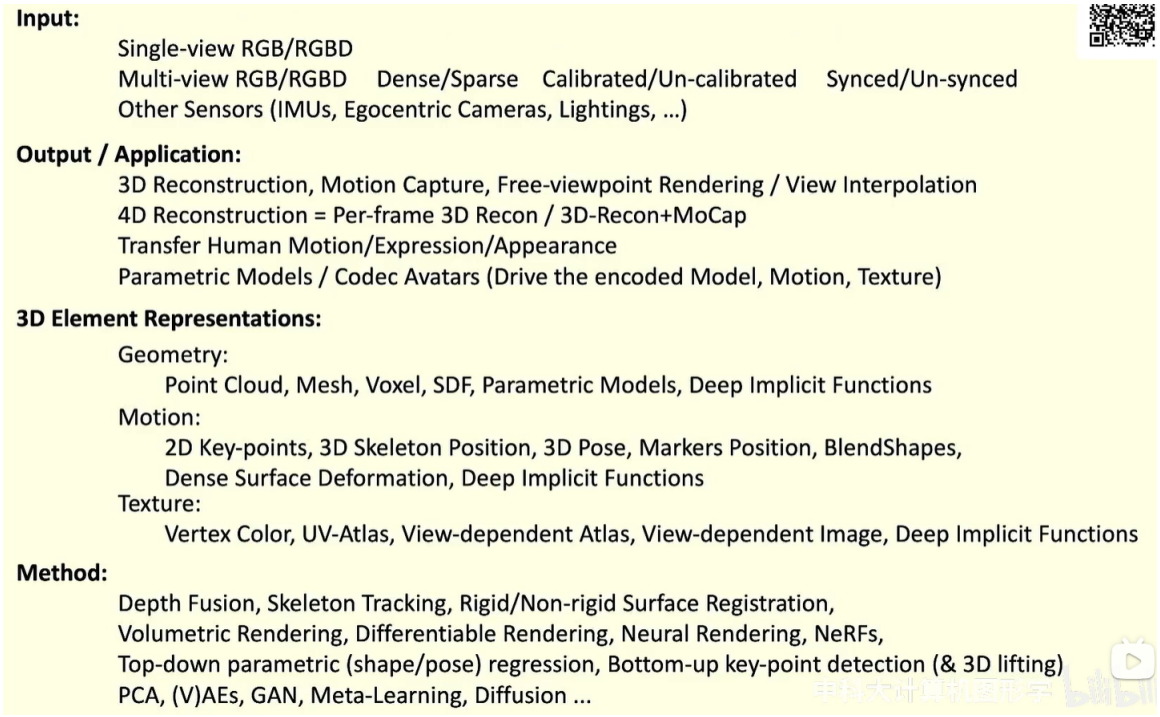


Figure 31: Summary of existing work by Professor Yutao

Based on my readings, I feel that research in 3D human modeling also requires extensive knowledge of machine learning and deep learning. Model establishment typically involves learning and training datasets, with some optimization issues, such as defining loss functions, gradient descent, multilayer perceptrons, and other related knowledge that I have recently studied in deep learning.

Finally, whether it is 3D human modeling or broader research themes, how to transition from papers to real-life applications, how to approach the limits of physical models (i.e., optimization problems), and how to advance from previous work to more contemporary solutions are all worthy of contemplation.

References

[1]Anguelov D, Srinivasan P, Koller D, et al. Scape: shape completion and animation of people[M]//ACM SIGGRAPH 2005 Papers. 2005: 408-416.

[2]Curless B, Levoy M. A volumetric method for building complex models from range images[C]//Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. 1996: 303-312.

[3]Garland M, Heckbert P S. Surface simplification using quadric error metrics[C]//Proceedings of the 24th annual conference on Computer graphics and interactive techniques. 1997: 209-216.

[4]Allen B, Curless B, Popović Z. The space of human body shapes: reconstruction and parameterization from range scans[J]. ACM transactions on graphics (TOG), 2003, 22(3): 587-594.

[5]Davis J, Marschner S R, Garr M, et al. Filling holes in complex surfaces using volumetric diffusion[C]//Proceedings. First international symposium on 3d data processing visualization and transmission. IEEE, 2002: 428-441.

[6]Anguelov D, Srinivasan P, Pang H C, et al. The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces[J]. Advances in neural information processing systems, 2004, 17.

[7]Haehnel D, Thrun S, Burgard W. An extension of the ICP algorithm for modeling nonrigid objects with mobile robots[C]//IJCAI. 2003, 3: 915-920.

[8]Anguelov D, Koller D, Pang H C, et al. Recovering articulated object models from 3d range data[J]. arXiv preprint arXiv:1207.4129, 2012.

[9]Sumner R W, Popović J. Deformation transfer for triangle meshes[J]. ACM Transactions on graphics (TOG), 2004, 23(3): 399-405.

[10]Wang X C, Phillips C. Multi-weight enveloping: least-squares approximation techniques for skin animation[C]//Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation. 2002: 129-138.

[11]Kavan L, Žára J. Spherical blend skinning: a real-time deformation of articulated models[C]//Proceedings of the 2005 symposium on Interactive 3D graphics and games. 2005: 9-16.

[12]Merry B, Marais P, Gain J. Animation space: A truly linear framework for character animation[J]. ACM Transactions on Graphics (TOG), 2006, 25(4): 1400-1423.

[13]Freifeld O, Black M J. Lie bodies: A manifold representation of 3D human shape[C]//Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part I 12. Springer Berlin Heidelberg, 2012: 1-14.

[14]Chen Y, Liu Z, Zhang Z. Tensor-based human body modeling[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2013: 105-112.

[15]Hirshberg D A, Loper M, Rachlin E, et al. Coregistration: Simultaneous alignment and modeling of articulated 3D shape[C]//Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12. Springer Berlin Heidelberg, 2012: 242-255.

[16]Pons-Moll G, Romero J, Mahmood N, et al. Dyna: A model of dynamic human shape in motion[J]. ACM Transactions on Graphics (TOG), 2015, 34(4): 1-14.

[17]Loper M, Mahmood N, Romero J, et al. SMPL: A skinned multi-person linear model[M]//Seminal Graphics Papers: Pushing the Boundaries, Volume 2. 2023: 851-866.

[18]Hirshberg D A, Loper M, Rachlin E, et al. Coregistration: Simultaneous alignment and modeling of articulated 3D shape[C]//Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12. Springer Berlin Heidelberg, 2012: 242-255.

[19]Pons-Moll G, Romero J, Mahmood N, et al. Dyna: A model of dynamic human shape in motion[J]. ACM Transactions on Graphics (TOG), 2015, 34(4): 1-14.

[20]Tsoli A, Mahmood N, Black M J. Breathing life into shape: Capturing, modeling and animating 3D human breathing[J]. ACM Transactions on graphics (TOG), 2014, 33(4): 1-11.

[21]Osman A A A, Bolkart T, Black M J. Star: Sparse trained articulated human body regressor[C]//Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16. Springer International Publishing, 2020: 598-613.

[22]Pavlakos G, Choutas V, Ghorbani N, et al. Expressive body capture: 3d hands, face, and body from a single image[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 10975-10985.

[23]Kanazawa A, Black M J, Jacobs D W, et al. End-to-end recovery of human shape and pose[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 7122-7131.

[24]Alldieck T, Magnor M, Xu W, et al. Video based reconstruction of 3d people models[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 8387-8397.

[25]Alldieck T, Magnor M, Bhatnagar B L, et al. Learning to reconstruct people in clothing from a single RGB camera[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 1175-1186.

[26]Xu H, Bazavan E G, Zanfir A, et al. Ghum & ghuml: Generative 3d human shape and articulated pose models[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 6184-6193.

[27]Palafox P, Božič A, Thies J, et al. Npms: Neural parametric models for 3d deformable shapes[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 12695-12705.

[28]Guzov V, Mir A, Sattler T, et al. Human poseitioning system (hps): 3d human pose estimation and self-localization in large scenes from body-mounted sensors[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 4318-4329.

[29]Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 3431-3440.

[30]Tompson J J, Jain A, LeCun Y, et al. Joint training of a convolutional network and a graphical model for human pose estimation[J]. Advances in neural information processing systems, 2014, 27.

[31]Groueix T, Fisher M, Kim V G, et al. A papier-mâché approach to learning 3d surface generation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 216-224.

[32]Kanazawa A, Black M J, Jacobs D W, et al. End-to-end recovery of human shape and pose[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 7122-7131.

[33]Chen Z, Zhang H. Learning implicit fields for generative shape modeling[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 5939-5948.

[34]Park J J, Florence P, Straub J, et al. Deepsdf: Learning continuous signed distance functions for shape representation[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 165-174.

[35]Mescheder L, Oechsle M, Niemeyer M, et al. Occupancy networks: Learning 3d reconstruction in function space[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 4460-4470.

[36]Saito S, Huang Z, Natsume R, et al. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 2304-2314.

[37]Saito S, Huang Z, Natsume R, et al. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019: 2304-2314.

[38]William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. In ACM SIGGRAPH Computer Graphics, volume 21, pages 163–169. ACM, 1987.

[39]Saito S, Simon T, Saragih J, et al. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 84-93.

[40]Zheng Z, Yu T, Liu Y, et al. Pamir: Parametric model-conditioned implicit representation for image-based human reconstruction[J]. IEEE transactions on pattern analysis and machine intelligence, 2021, 44(6): 3170-3184.

[41]Yu T, Zheng Z, Guo K, et al. Function4d: Real-time human volumetric capture from very sparse consumer rgbd sensors[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 5746-5756.

[42]Zheng Y, Shao R, Zhang Y, et al. Deepmulticap: Performance capture of multiple characters using sparse multiview cameras[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 6239-6249.

[43]Xiu Y, Yang J, Tzionas D, et al. Icon: Implicit clothed humans obtained from normals[C]//2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2022: 13286-13296.

[44]Newcombe R A, Izadi S, Hilliges O, et al. Kinectfusion: Real-time dense surface mapping and tracking[C]//2011 10th IEEE international symposium on mixed and augmented reality. Ieee, 2011: 127-136.

[45]Newcombe R A, Fox D, Seitz S M. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 343-352.

[46]Yu T, Zheng Z, Guo K, et al. Doublefusion: Real-time capture of human performances with inner body shapes from a single depth sensor[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 7287-7296.

[47]Dou M, Khamis S, Degtyarev Y, et al. Fusion4d: Real-time performance capture of challenging scenes[J]. ACM Transactions on Graphics (ToG), 2016, 35(4): 1-13.