Computational Intelligence and Neuroscience

Generating Point Cloud from measurements based on Convolutional Neural Network – an application for building 3D human model

Nguyen Mau Tung[1,2], Dang Thanh Vu[3], Tran Thi Minh Kieu[1], and Pham The Bao[3]

[1]University of Science and Technology, School of Textile – Leather and Fashion

[2]Industrial University of Hochiminh city

[3]Sai Gon University, Hochiminh city

ABSTRACT

3D point clouds enclose fundamental appearances of objects. Generating point clouds and meshes is an essential step in constructing computer-based 3D models. This paper introduces a novel method to create the point cloud of 3D objects from crucial measurements. To find the relation between shapes and sizes, we present a method of representing 3D data called slice- structure. A Neural Network-based learning model is then manipulated to be compatible with the data representation. Principal slices are generated by matching the measurements at predetermined heights before the whole point cloud is tuned by Convolutional Neural Network (CNN). We conduct experiments on a dataset of 3D scan on human bodies, which contains 1706 examples. The average error between predicted models and the real ones is 0.0772, and the generated models have good visualization as well.

INDEX TERMS Anthropometry, 3D point cloud, Neural Network, Human body modeling.

I. Introduction

It is well known that 3D models have many roles in various fields such as human animation, garment industry, and simulation. An intriguing characteristic of computer-based 3D models is that the geometry structures of objects can be visualized effectively. With many remarkable impacts on human life, 3D modeling is increasingly becoming an essential technique in the computer-aided design community.

Due to the usage of 3D scanning devices is time-consuming and expensive, more attentions have been paired to the problem of automatically generating 3D models. Traditionally, measurements such as lengths, perimeters, and curvatures have been beneficial for describing the appearance of realistic objects. However, reconstructing computer-based models from these measurements is still a challenge. The common drawback is that the sparse set of measures is inadequate to capture the complexity in shapes of realistic objects.

Current solutions to 3D modeling are required a rich source of inputs such as images or 3D templates [1, 2, 3, 4]. The approaches to build 3D models using only measurements are relatively old and not realistic, particularly in building 3D human models using anthropometries [5, 6, 7]. Reconstructing models based on Neural Networks is adopted prosperously to generate a 3D shape from 2D/2.5D data; however, they request a large amount of training data and high computational cost [8, 9, 10, 11]. We addressed these issues by combining both template fitting and exemplar-based approach along with appropriate data representation. Our proposed framework uses light-weight models, thus achieving a good trade-off between the computational complexity and accuracy.

Our main contribution is to formulate a novel representation for 3D models based on the point cloud that is suited to explore the relationship between the measurements and 3D shapes using Neural Networks. The main idea is to separate an object into independent components and slices. This secession allows us to define a unique Neural Network's architecture for the particular shape instead of working on the whole 3D model. Firstly, we deform initial shapes into the forms of a template at predefined heights, namely primary slices. This step is accomplished using multi-Neural Network models with a hidden layer. In an attempt to reconstruct the entire point cloud of objects, we use Convolutional Neural Networks as multiple filters that analyze the geometric connections among adjacent slices. Taking the 3D human modeling for an application, we demonstrate a two-stage procedure of synthesizing a new human model given anthropometric measurements, Figure 1. This work also provides a novel dataset about 3D scanning of human bodies that would support other studies in the long run.

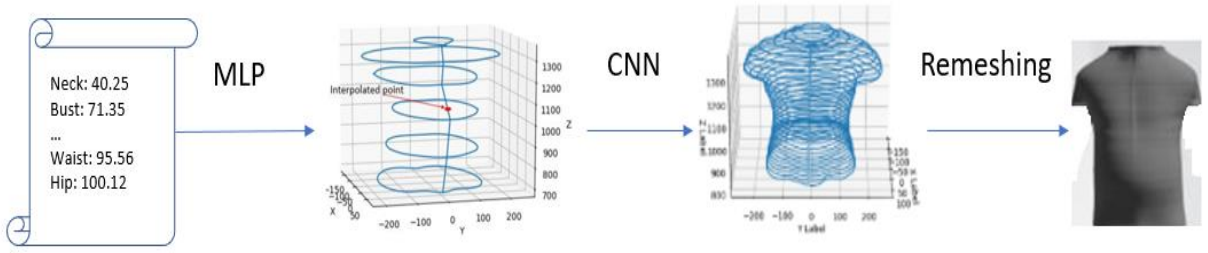


Figure 1. The overall flowchart. The input is a set of measurements required to generate primary slices using Multi-Layer Perceptron (second figure). CNN is adapted to interpolate intermediate slices (third figure), and we use triangular remeshing at the last stage to construct the 3D target (fourth figure).

II. Related works

Preliminary work in this literature focused primarily on deforming a template model to produce a new 3D model. Allen at el formulated an optimization problem to find an affine transformation at each vertex of the designed template. They defined three types of error and combined them to create the objective function. They also addressed incomplete surface data and filled in missing and poorly captured areas caused by the scanner [12]. Modifying Allen's method, Hasler performed non-rigid registration with the aim of fitting poses and shapes of 3D objects from a template model [13].

Many attempts have been made to construct 3D models from 2D data such as images. This approach could reduce the cost of 3D data collection because it is merely required a set of images. However, the image data often contain noises and background, which are strenuous to eliminate. Joseph at el. proposed an algorithm to construct 3D face from an unconstrained collection of face images captured under various poses, expressions, and illuminations. Their algorithm includes four steps; they are landmark alignment, landmark driven 3D wrapping, photometric normals, and 3D surface reconstruction [14]. In the like manner, Luo applied a shape-from-shading method to construct fine geometric details of a face after using photometric consistency constraints to refine the 3D coarse shape [1]. Jason used only one depth image to generate a complete 3D model by exemplar-based approach. The random forest was built with the use of entropy dissimilarity to retrieve similar exemplary 3D shapes from their database, and the deformation was modeled using a 3D approximate thin-plate splines given symmetry constraints [15]. Chen et al. estimated the dense 3D shape by combining 2D landmarks and silhouette in the image of an object. The authors propose a dictionary-based framework, including orthogonal matching and graph embedding based on dense local correspondence [2].

The statistical approach is one of the most useful methods to reconstruct 3D objects. These methods use the training set to learn the correlation between input and output or construct an examplar space for extrapolation. Inspiring form DeCarlo's work [16], the statistics-based model has become a powerful tool for demonstrating the feature space of 3D models. In this study, measurements of human's face were used to generate 3D face shapes by variational modeling while a prototype shape was considered as the reference. Wuhrer et al. introduced a method that extrapolates the statistical inference to fit the measurements using non-linear optimization [17]. First, PCA was applied to produce a feature space of human shapes, and then shape refinement was used to correct the predicted model. Wu et al. learned a latent space of 3D objects using 3D-GAN; this approach has been studied extensively in recent years [18]. However, space spanned by the training data would restrict the diversity of generated models, which is the common drawback of the statical approaches. Furthermore, finding a large number of variables by optimizing on the small data set would lead to the under-fitting problem.

On behalf of the development of Deep Learning in recent years, Deep Neural Networks have been launching to analyze 3D models successfully. Sinha et al. modified Deep Residual Networks to generate 3D shape surfaces of rigid and non-rigid shapes directly and concluded that their Network could learn a meaningful representation of shapes through the experiments [8]. Charles's group tackled the problem of segmentation on 3D objects by proposing PoinNet, a Neural Network which employs point clouds as the input [11]. He also theoretically analyzed the robustness of the Network with the permutation invariance of input data. Generating tasks have been more blossoming based on the invention of GAN [19]. Milz et al. used conditional GAN to learn the 3D point clouds, which are the latent presentations for image translation [20]. Shu et al. integrated tree structure into graph convolutions and embedded them to the generator module of GAN to generate multi-class 3D point clouds [21]. Shi et al. proposed PointRCNN for 3D object detection [10]. Their framework combined generated 3D proposals and local spatial features to gain better box proposal refinement, which helps to give accurate predictions. Structural Point Cloud decoder proposed by Tchapmi can generate 3D shapes based on a hierarchical rooted tree principal in which each node of the tree represents a segment of the point cloud, and the root depicts the whole point cloud. The 3D encoder part was arbitrary, meaning that that the network could not see any prior knowledge about the topology structure of 3D objects [22]. Raj et al. constructed 3D textured meshes from geometry information about objects [9]. In their study, Multiview images were generated using CycleGan, which were the input to construct textured meshes by a differentiable renderer. These studies shed light on the ability of Deep Neural Networks in the literature of 3D model generator, especially the success of Convolutional Neural Networks.

III. Methodology

In this section, we demonstrate our framework, which consists of two main steps, including generating primary slices and refining 3D point cloud. To structure 3D objects, we construct a set of points placed on planes which are perpendicular to the axial height of the objects. These points formed the outline of objects. Then, we created meshes to patch the gaps on the boundary of 3D objects. Typically, if the distance between two adjacent surfaces is small, these will have nearly similar shapes. Therefore, independently constructing the marginal points on every surface is not necessary. That enables us to pick some surfaces that are considered as the main ones. These surfaces can be chosen based on available measurements or analyzing the figure of objects. The critical advantage of this approach is the reduction in the number of calculations and required measures.

Assume that the set of all surfaces being perpendicular to the axial height of a 3D object is $S = \{S_i \mid i = 1 \dots m\}$. The primary set PS is a subset of S, $PS = \left\{S_i \in S \mid i \in PI \subset \{1,2,..,m\}\right\}$ such that for all $S_i, S_j \in PS, i \neq j$, S_i and S_j do not have a common shape. We assess the degree of differences of two shapes based on observing the appearance of 3D objects. To learn the relationship between measurements and primary surfaces, we construct a mapping from an initial set to a target set

$$f_i \colon C_i = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 \leq \left(\frac{m_i}{2\pi}\right)^2\} \ \rightarrow S_i' \tag{1}$$

Such that the difference of S_i and S_i' is minizied. If we consider hollow 3D objects, these surfaces will turn into the slices defined in the following section, and C will be the circle with its radius is computed by the perimeter of the corresponding slice.

Once all principal slices had been available, the entire model would be constructed. The shape of surfaces between two principal slices gradually changes to match the shape of these two principals. Through the use of primary surfaces; Therefore, we can interpolate the whole 3D object. However, the interpolated surfaces are not as practical as the real ones. We overcome this problem by using the adjusting model based on CNN, which will be clarified in the next section.

A. Building primary slices

Our study restricts to a class of surfaces which can be written under the trigonometric formula and satisfy the following mild condition. A qualified surface is a set of points $S_{z_o} = \{(x, y, z) \in \mathbb{R}^3 | z = z_0\}$, so that for all $\theta \in [0,2\pi]$ and $r > 0$, there is no more than one point $(x, y, z) \in S_z$ satisfies Formula (2).

$$\begin{cases} x = x_0 + r\cos\theta \\ y = y_0 + r\sin\theta \end{cases} \tag{2}$$

Where (x_0, y_0) is former given and we call it as "anchor point", which is the center of a slice, Figure 2. In this study, the term "slice-structure" refers to the above definition.

The above surface description has an advantage that the redundancy of the third dimension is eliminated. A point (x, y) could be replaced by a pair (r, θ), but the θ variables are actually in common for all slices. Thereby, a slice can be written in vector from called slice-vector that its components are distances between the anchor and points on it. Moreover, this representation is invariant under translation because r does not change when translating 3D objects. Besiseds, we simply shift the components of slice-vectors when rotating 3D models.

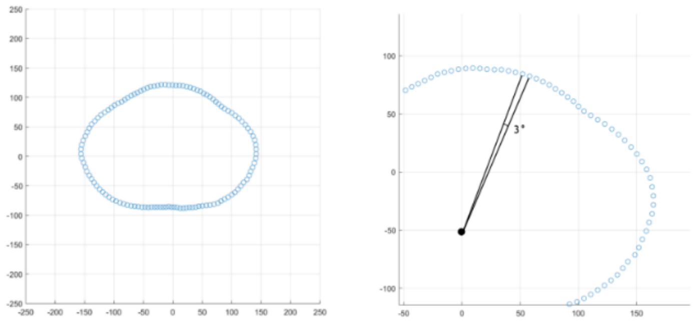


Figure 2. An example satisfied the proposed slice structure. Each point is 3^o apart, the anchor point is at the origin.

The principal slices are generated by deforming initial circles with the drive of samples in a training set. Let $PI \subset \{1,2,\dots,m\}$ is the index set of main slices, we approximate the target slices $S'_i, i \in PI$ by Formula (3). Let S_i, $i = 1,\dots,m$ is the $n-$ dimensional vector representing the i^{th} slice, the k^{th} component of it is the distance between the center and the point at $\theta = 2\pi\frac{k-1}{n},\ k = 1\dots n$.

We define the deformation function $f_i: \mathbb{Z}^{+^{n\times 1}} \to \mathbb{Z}^{+^{n\times 1}}$ as

$$S'_i = f_i(X_i) = (W^2)^T \alpha((W^1)^T X_i) \tag{3}$$

Where $W^1 \in \mathbb{R}^{n\times L_1}, W^2 \in \mathbb{R}^{L_1\times n}$, α is a non-linear function, X_i is an initial slice, f_i is known as Multilayer Neural Network (MNN). Algorithm 1 summarizes the learning procedure of generating principal slices.

The key idea of the first model is to deform an initial shape into the desired shape controlled by perimeters and the training data. Using circumferences alone to construct 3D objects is insufficient since the shape of objects varies even though its measurements are common. Therefore, our approach also based on the shape of objects that can be investigated by MNN model from the training set. In this case, the learning model seeks for positions on initial slices that need to be shrunk or dilated, Figure 3. We choose this learning model because it automatically seeks an appropriate transformation. Moreover, all primary slices are generated by the typical architecture; thus, it is adaptable in terms of implementation.

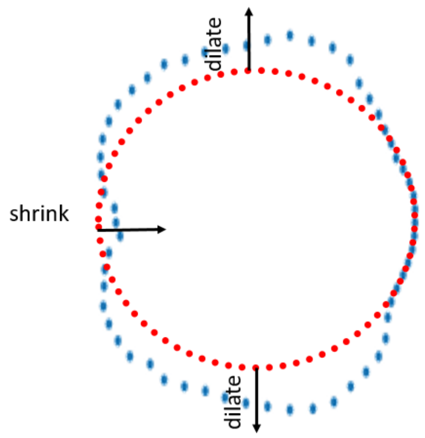


Figure 3. Deforming an initial slice (red circle) into the target slice (blue curve). The learning model explores the way to translate the red points to match the nearest blue points.

Algorithm 1: Building primary slices form measurements

```
Input: Set of measurements $P$, Set of 3D shapes in which $S^{(i)} \in D$ is a sample following the slice-structure.
Output: Set of learned parameters $W$
for $h \in SP$ do
    $loss_h = 0$
    for each sample $S^{(i)} \in D$ do
        $w =$ length of vector $S_h^{(i)}$
        $r = \left(P_h^{(i)}\right)/2\pi$
        init $X$ as $w$-dimensions array with $X_k = r$
        $Y = f(X)$
        $\max = \max Y$
        $\min = \min Y$
        $loss_h = loss_h + \frac{1}{w}\left\|Y - S_h^{(i)}\right\| + \left(2\pi\sqrt{\frac{(max)^2+(min)^2}{2}} - P_h^{(i)}\right)$
    $W^{(h)} = argmin(loss_h)$
```

Algorithm 2: Constructing 3D point cloud

```
Input: Set of generated primary slices $D'$ in which $S'^{(i)} \in D'$ comes from Algorithm 1, Set of 3D shapes in which $S^{(i)} \in D$ is a sample following the slice-structure
Output: Set of learned parameters $W$
$loss = 0$
for each sample $S^{(i)} \in D$ do
    $n =$ the number of rows of $S^{(i)}$
    $m =$ the number of column of $S^{(i)}$
    init $X$ as $n \times m$ matrix
    for $k$ from 1 to $m$ do
        for $h$ from 1 to $n$ do
            start = $\max\{x \in PI | x \leq h\}$
            end = $\min\{x \in PI | x \geq h\}$
            $X_{hk} = S'^{(i)}_{start,k} + (S'^{(i)}_{start,k} - S'^{(i)}_{end,k}) * (h - start)/(end - start)$
    $Y = g(X)$
    $loss = loss + \frac{1}{mn}\|Y - S^{(i)}\|$
$W = argmin(loss)$
```

B. Generating entire point cloud

Based on the output of the above step, we carry out linear interpolation all the remaining slices. Considering $\theta = 2\pi\frac{(k-1)}{n}$, we calculate $s'_{ik}, i \notin SP$ based on $s'_{ik}, i \in SP$, Figure 4. The premiment reason of using interpolate here is that we want to perform upsampling from primary slice to produce the others. There are two main ways to do upsampling in CNNs, interpolation and transposed convolution [23], but linear interpolation is simpler and we find it give interesting results. Otherwise, transposed convolution will add more extra parameters to the learning model and it is only useful in case training an end-to-end model. Interpolation is mandatory to our approach because it helps to increase the size of receptive fields [24] that needs for reconstruction.

The interpolated slices are subsequently the input for the second model. We construct the second synthesizer based on Convolutional Neural Network (CNN) [25] because its kernels can capture local characteristics, and that is useful when inspecting relations between adjacent slices. This model corrects wrong interpolated points by using the information on the training set via CNN architecture. Local structures of 3D shapes are retained by convolutional layers in CNN hence resulting in fine appearance. We define the CNN model as a function $g\colon {\mathbb{R}^{+}}^{m\times n} \to {\mathbb{R}^{+}}^{m\times n}$, Formula (4).

$$Y = g(X) = W^{L} * \alpha_{L-1}\left(W^{L-1} * \alpha_{L-2}\left(\dots \alpha_{3}(W^{3} * X)\right)\right) \tag{4}$$

Where $a_l, l = 3, \dots, L-1$ stands for a non-linear activation function, X is formed by stacking both principal and interpolated slices in a row.

A rational choice of loss function for this problem is Mean Square Error (MSE). In our study, MSE calculates the difference between the generated and the actual distance. This measure provides the similarity between a created object and the real one. We use this metric to evaluate the error on both learning models, Algorithm 1&2. We also add the measurement error into the loss function of the first model. This error is a constraint to match the sizes of generated shapes to the sizes of original objects.

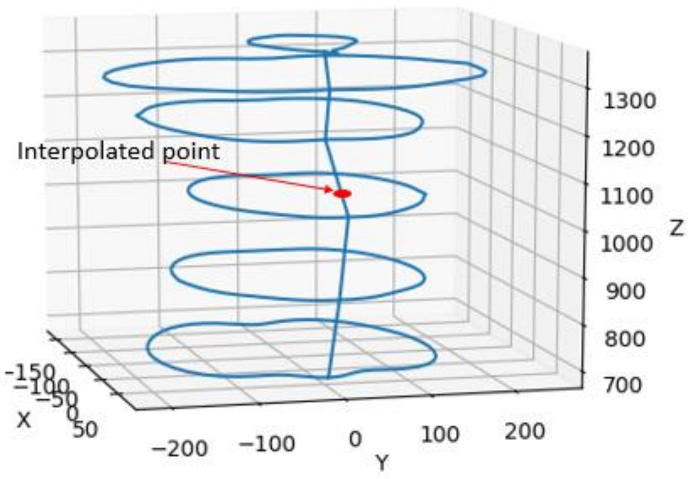


Figure 4. Interpolation of intermediate slices. Having the principal slices (blue shapes) enables us to calculate the middle slices (red) through the use of linear interpolation.

IV. An application for building a 3D human model

A. Dataset

Two universities in Vietnam independently developed the datasets used in this study. The details of the dataset are summarized in Table 1.

Table 1. Summary of the dataset.

Authority	Gender	Age	The number of avatars	The number of damaged avatars
Industrial University of Ho Chi Minh City (IUH)	Male	20-60	1106	65
Hanoi University of Science and Technology (HUST)	Female	20-60	600	63

A total of 1706 persons were recruited for this study. All samples took part in the experiment were Vietnamese without deformities on their body. Also, there were variations in age and occupation among the participants. 3D scanning devices generated each example on both datasets, and they were saved under '.obj' format. Each person only provided one 3D scan of the body, hence the number of participants and samples is equal. Participants were suggested wearing a tight suit and complied with the standard pose when scanning their body. Samples violating recommended rules and not scanned carefully were categorized into the damaged category. This group accounted for 128 samples.

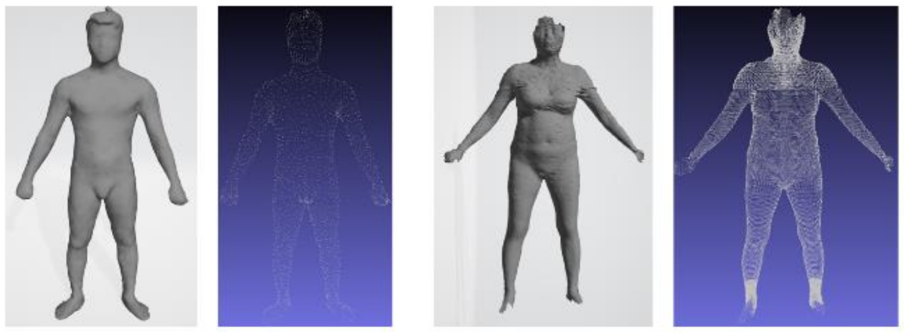


Figure 5. The avatars in the male and female dataset along with their point cloud.

Two datasets were built from different devices, thus having some distinct features, Figure 5. The most remarkable thing is that the point density of the male avatars (about 5000 points) is not as dense as that of females (about 44000 points). 3D female avatars satisfy the proposed slice-structure, and each vertex has already been distributed into one of five parts, including the torso, two legs, and two arms. Each point on torso slices, leg slices, arm slices is 3, 5, 10 degrees apart, respectively. Besides, all slices are equally spaced in height by the same distance.

Meanwhile, the male dataset does not meet ideal conditions as same as those of the female. Not only it has no predefined boundary between two body parts but also the point clouds do not follow the slice-structure. However, the authors of the man dataset provided a set of landmarks for each avatar. By exploiting these reference points, we can separate body parts on the men models, Figure 6. The markers are not necessary to belong to the point cloud of objects. Moreover, our slice-structure is easily achievable with appropriate preprocessing steps.

While analyzing the database, we notice that there are several damaged examples in both datasets. The disfigurement in the female dataset almost come from scanning devices. Otherwise, the damaged examples in the male dataset are due to the non-cooperation of the participants, Figure 7. Overall, there are 65 unqualified samples in the male dataset and 63 unqualified samples in the female dataset.

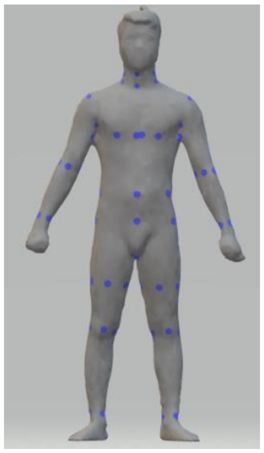


Figure 6. Anatomical landmarks of a male model. The anatomical points were spotted in three-dimensional space; each point relates to a prominent position on the man's body.

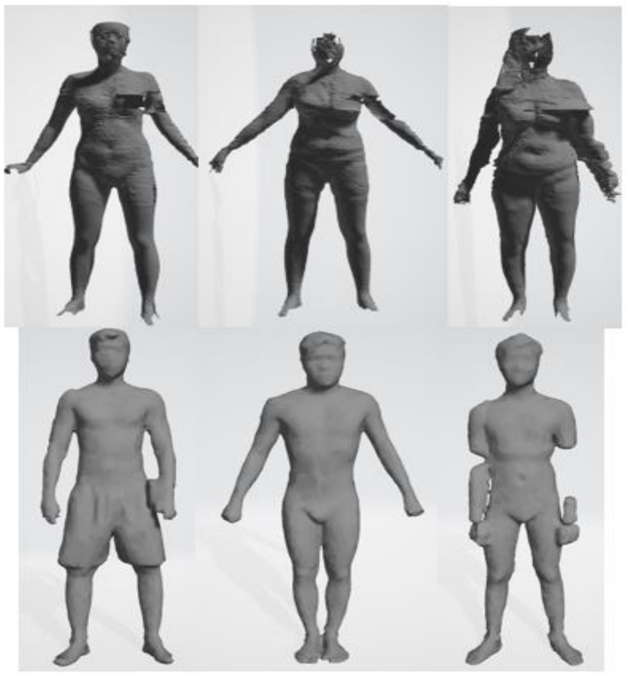


Figure 7. Damaged 3D human models.

B. Preprocessing

We split a 3D human model into five parts based on the landmarks as the following manner, Figure 8.a.

Torso

- Upper torso: From neck to armpit, limited by the left elbow and right elbow.
- Lower torso: From armpit to hip, limited by the left and right hip or limited by the left and right stomach.

Arm (Left/Right)

- Upper arm: From armpit to the elbow, limited by armpit and elbow.
- Lower arm: From elbow to wrist, limited by elbow and wrist.

Leg (Left/Right)

- Upper leg: From hip to knee, limited by crotch point and knee.
- Upper leg: From knee to ankle, limited by knee and ankle.

After all body parts were divided, we construct surfaces that perpendicular to the high axis. Assuming that the set of all points belonging to one human section is S, we assigned

$$(x_0, y_0, z_0) := \left(x_0, y_0, \min H_z + i\frac{d_z}{m-1}\right) \tag{5}$$

If

$$\min H_z + i\frac{d_z}{m-1} \le z_o < \min H_z + (i+1)\frac{d_z}{m-1} \tag{6}$$

Where $H_z = \{z \in \mathbb{R}, (x, y, z) \in S\}, d_z = \max H_z - \min H_z$ is the height of a body part, $i = 0, \ldots, m-1$ with m denotes the number of slices (50 in our experiment).

By denoting $S_i = \left\{(x_0, y_0, z_0) \middle| z_0 = \min H_z + i\frac{d_z}{m-1}\right\}$, we are ready to create the slice vectors. We first calculate the position of an anchor point on each slice. The mean formula is suitable to find anchor points, Formula 7.

$$\left(x^{(i)}, y^{(i)}, z^{(i)}\right) = \frac{1}{|S_i|}\sum_{(x,y,z)\in S_i}(x, y, z) \tag{7}$$

However, the formula above expects the distribution of points on a slice must provide enough information to approximate the center point accurately. Although the density on male avatars is not ideal to use Formula (7), the landmark set enables us to approximate the skeleton of male avatars and deduce the anchor points. Take the torso for an example, we constitute its skeleton by the line connecting the center of four landmarks at the neck and the crotch point, Figure 8.b. Once the anchor lines (skeleton) are found, it is straightforward to compute anchor points at any height. On the other hand, the number of points per slice of female avatars is sufficient to apply Formula

(7). In this work, we carefully choose a person as the template whose anchor lines were utilized to construct new 3D models.

We convert all slices into the form defined in section III.A, then we formulated the slices vectors, which were the primary data structure used in our algorithms. Given a point $(x_0, y_0, z_0) \in S_i$, the angle established by the anchor point $\left(x^{(i)}, y^{(i)}, z^{(i)}\right)$ and this point was computed by Formula (8).

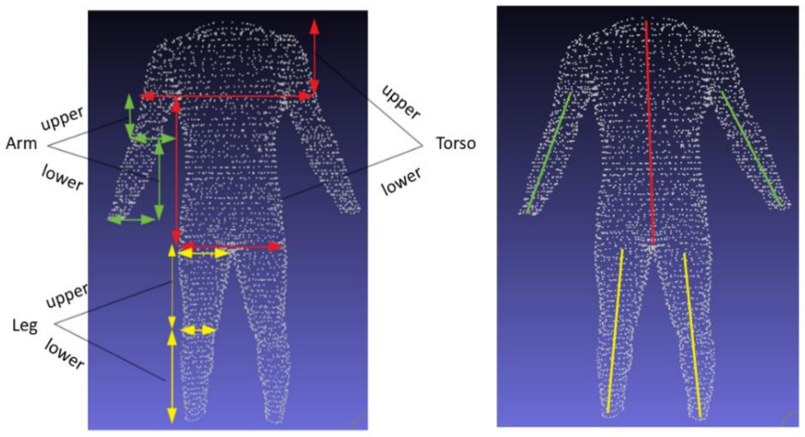
$$a_0 = \arctan\left(\frac{y_0 - y^{(i)}}{x_0 - x^{(i)}}\right) \tag{8}$$

The j^{th} component of a slice vector represents the distance between the anchor point and the point at $\theta = 2\pi\frac{j-1}{n}$, n is the dimension of the slice-vectors. One point was assigned to the j^{th} position if satisfying the following condition.

$$2\pi\frac{j-1}{n} \le a_0 < 2\pi\frac{j}{n} \tag{9}$$

Where $j = 1, \dots, n$. The value of the j^{th} component of slice vectors was directly calculated by Euclidian metric, Formula (10).

$$d_j^{(i)} = \sqrt{(x_0 - x^{(i)})^2 + (y_0 - y^{(i)})^2} \tag{10}$$



a b

Figure 8. Preprocessing a 3D human model. a) Dividing the body parts, b) Determining the anchor lines.

Both man and female avatars suffered from missing data problem. In the female dataset, the reasons are the carelessness during the scanning process and the outdated equipment. While data missing on preprocessed male avatars is inevitable because their original point cloud is not ideal. Furthermore, the point density is not dense enough to divide male bodies into many slices. We solve this problem by conducting linear interpolation on grid data of slice vectors, Figure 9.c.

C. Measurements

The male dataset supplies us a set of anthropometric measurements with 178 categories comprising slice perimeters, widths, and heights of body parts. Nevertheless, the measurements are not in the same unit with distances computed on the point clouds. On the other hand, the female dataset provides no measurements. Due to these reasons, we recalculate the measures to be consistent in both datasets. A straightforward way to compute slice circumferences is summing all distances of two adjacent points, but it is not realistic when measuring non-convex shapes. We propose the use of the circumference of the convex hull of a slice as its measurements. To fit the convex hull of slice shapes, we use the Gift wrapping algorithm [26].

Measurements are calculated on the convex hull of primary slices. Summary, there are 28 slice measurements, but we could reduce the number of measures to 17 because of the similarity of the right and left side. Besides, it is necessary to record the height (length) of each body part to build up 3D human models entirely. Therefore, there are 20 measurements in total. The primary positions are statistically chosen based on the dataset and the standard ratio of the human body [27], Figure 10.

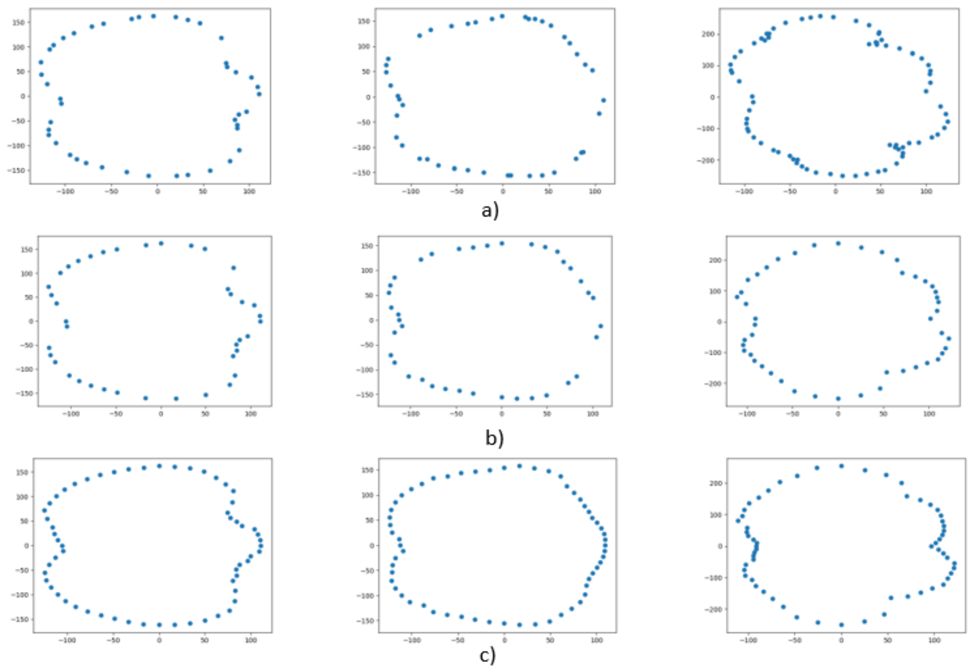


Figure 9. Constructing the slice structure. a) original slices, b) selecting slice components, c) filling missing data.

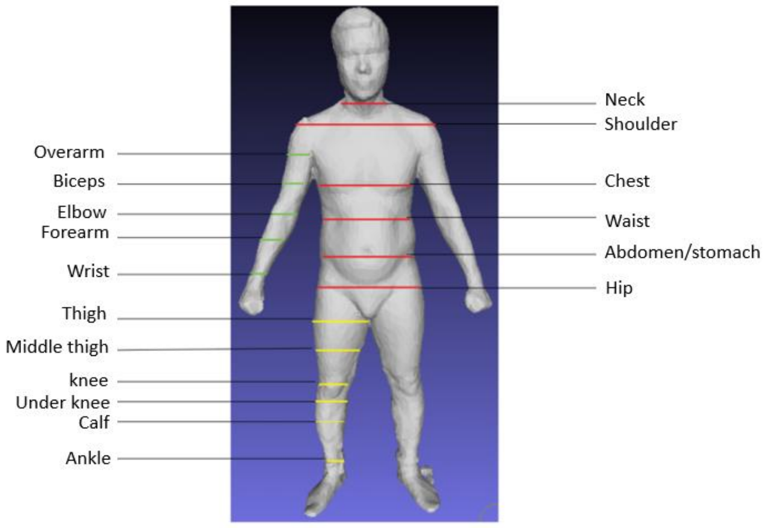


Figure 10. The position of primary slices.

D. Learning model

To construct primary slides, we build the Neural Network (NN) models with one hidden layer, as described in Section III.A. These model deform input slices into target slices, Figure 11.

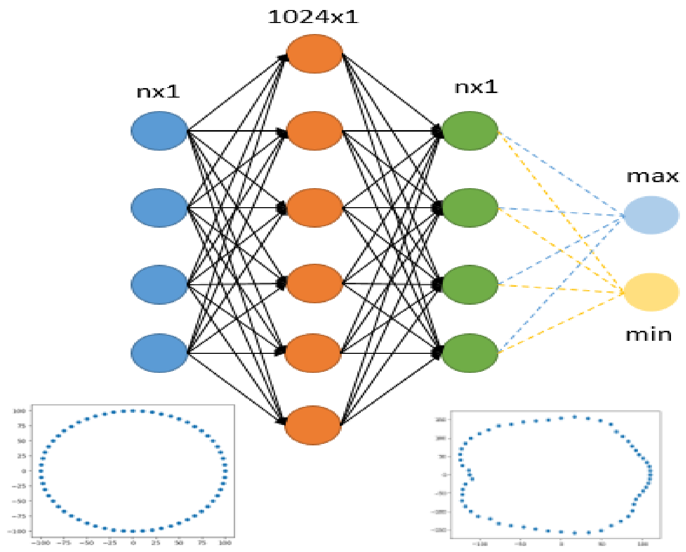


Figure 11. The Neural Network model for generating the primary slices.

These models take an initial circle as input and learn the deformation from the input shape into the target shape. The radius of the initial ring is $r = \frac{p}{2\pi}$, where p is the perimeter of the slice being considered. The input and output size depends on the body part, in our experiment, n equal 20, 30, 60 for the arm, leg, torso, respectively. The hidden layer of this network had 1024 units containing shape features. The error between a predicted (y') and the actual slice (y) is calculated by Formula (11).

$$L(y,y') = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i')^2 + \gamma\left(2\pi\sqrt{\frac{(\max y')^2 + (\min y')^2}{2}} - p\right) \tag{11}$$

Where the second error term comes from the difference between the approximation of the circumference of the generated slice and the actual one. The objective function reflects the error not only at each component (local information) bust also the perimeter of the slice (global information). λ is the hyper parameter controlling the ratio of two losses.

Once all the primary slices were found, linear interpolation is used to infer the remaining slices. These interpolated slices are the input to the second NN model, as stated in section III.B, Figure 12. We use ReLU [28] as the activation function in both architectures.

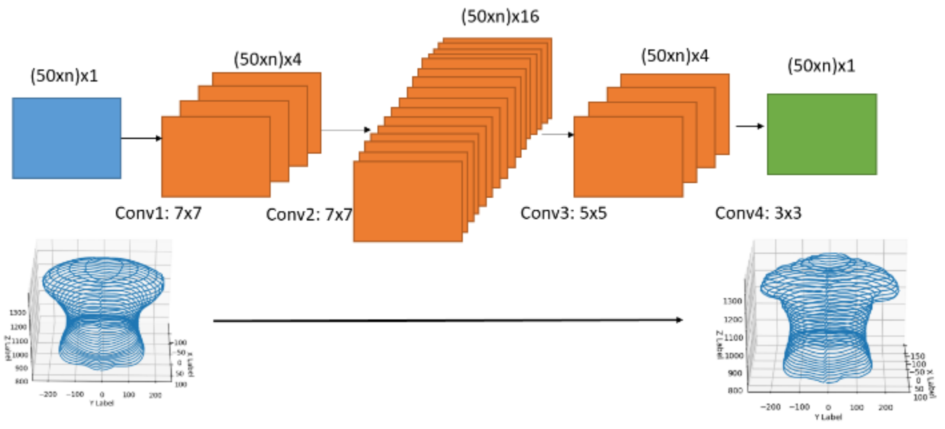
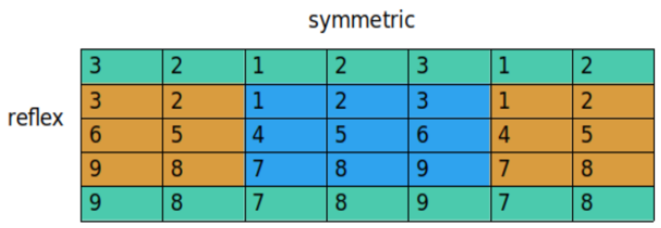


Figure 12. The Neural Network model for adjusting the entire slices.

The convolutional layers help to extract local correlations of 3D shapes since they act as multiple filters concerning adjacent slices. As a result, the remaining slices would be corrected based on the primary slices and the learning data. The most cautious thing when building CNN in this circumstance is padding. We perform reflexive padding in vertical and symmetric padding in horizontal, Figure 13. Padding this way retains circularly linked characteristic of each slice.


symmetric

reflex

3	2	1	2	3	1	2
3	2	1	2	3	1	2
6	5	4	5	6	4	5
9	8	7	8	9	7	8
9	8	7	8	9	7	8

Figure 13. Padding strategy.

We define the loss function on the second model by using MSE, Formula 12.

$$L(y',y) = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(y_{ij} - y_{ij}'\right)^2 \tag{12}$$

Where y, y' are respectively a matrix of actual and predicted distances to the anchor points of a body part.

V. Experiments and results

We train our NN models on the Linux server with 24 GB RAM, GPU with 12 GB RAM, and Xeon CPU with 2.2Ghz. The programming language is Python, and the main libraries are PyTorch and numpy. Adam algorithm [29] is adopted to minimize the objective function; the meta parameters are set according to the recommendation of the authors (learning rate $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$). We evaluate the error by the relative error, Formula 13.

$$L(y, y') = \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{m}\frac{|y_{ij} - y'_{ij}|}{y_{ij}} \tag{13}$$

Where y, y' are respectively a matrix of the actual and predicted distances to the anchor point of a body part. The error formula is not affected by size heterogeneity on different body parts as well as on different datasets. In the male dataset, we use 1066 samples as training data and 100 samples as test data, while 500 and 100 as training and test data in the female dataset, the samples are selected randomly. Table 2 shows the average error on each primary slice after training 1000 epochs on the male and female dataset.

Table 2. Average error on each primary slice on the training data of male and female dataset (full dataset).

Slice	Train/Male (%)	Train/Female (%)
Hip	8.39	4.68
Abdomen	4.41	3.46
Waist	5.34	4.90
Chest	5.80	11.25
Shoulder	6.18	9.37
Neck	13.55	21.61
Left Wrist	7.77	10.99
Left Forearm	5.53	7.48
Left elbow	4.21	7.72
Left Biceps	6.22	8.80
Left Overarm	10.54	15.66
Right Wrist	13.43	8.12
Right Forearm	12.69	7.63
Right Elbow	8.36	7.01
Right Biceps	8.84	6.69
Right Overarm	12.20	12.64
Left Ankle	7.93	7.38
Left Calf	7.73	4.62
Left Under knee	11.80	3.30
Left Knee	6.31	4.40
Left Middle Thigh	3.63	4.72
Left Thigh	6.46	9.67
Right Ankle	9.54	6.64
Right Calf	7.52	4.66
Right Under Knee	10.67	3.09
Right Knee	5.80	4.45
Right Middle Thigh	3.41	5.05
Right Thigh	6.03	10.21

Learning the relationship between the size and corresponding slice shape is a severe problem because of the curse of dimension. Even though the input is just a scalar, we have to predict the slice vector with at least 20 components. We propose using initial shapes to settle out this problem. The initial shape not only is a rough approximation of the target slice but also helps the NN model increase the number of parameters and avoid under-fitting. In our work, we limit the class of initial shapes to circles that their radius was calculated by the slice perimeters. Geometrically, the first NN models act as a figure deformation controlled by the slice sizes. The NN models are the non-linear transformations from straight lines to the particular slice-vector "curves," which are the curves describing the magnitude of slice-vectors. These curves are similar in shape if they are in the same positions, Figure 14.

In the Torso part, the neck slices have the highest average error due to these slices are not completely separated from the head. Moreover, the majority of the anatomical landmarks at the neck are placed at the wrong locations (collar or chin). Accordingly, the shape of the neck slices varies considerably, as seen in Figure 15. The same circumstance happens with overarm slices. The boundaries between arms and shoulder are not accurately determined based on the landmarks. Another problem is the lack of a large number of components on overarm slice-vectors because of the obstructed locations such as armpits, where the 3D scanner usually ignores.

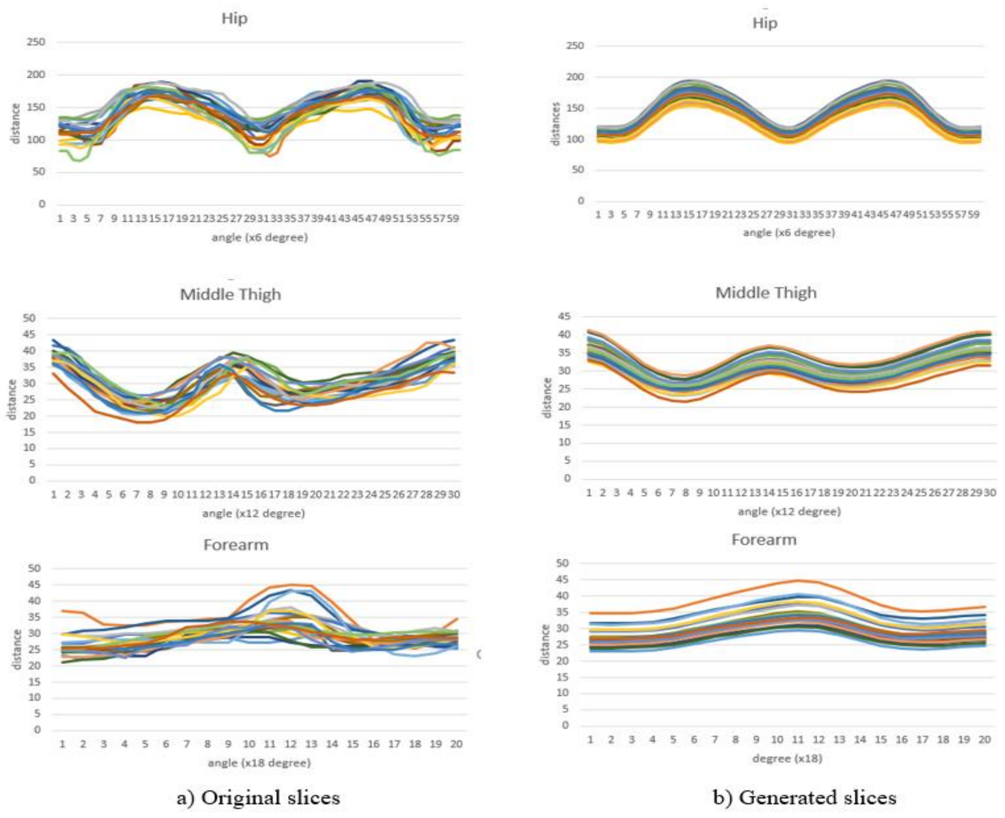


Figure 14. The slice-vector "curves" of the wrist, hip and thigh of 20 examples in the male dataset.

.

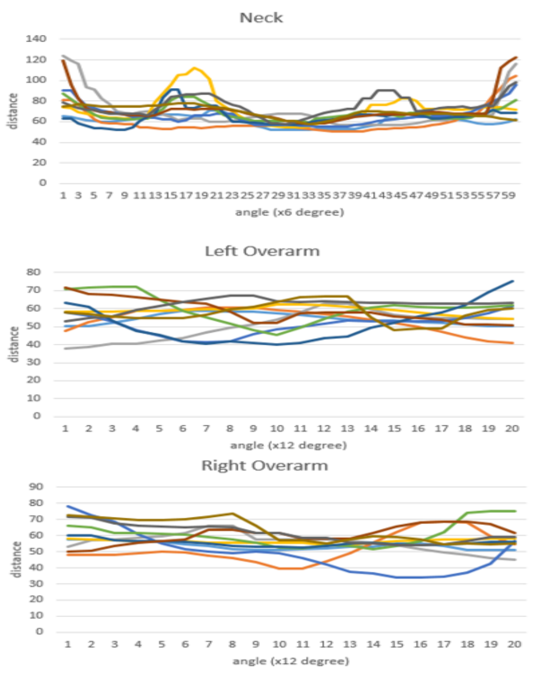


Figure 15. The slice vector “curves” of the neck, left and right overarm of 10 examples in the male dataset.

Table 3. Average error on each part of a 3D male and female model after activating the CNN model. The test set is comprised of damaged and undamaged samples. The two last columns are the average value of the damaged and undamaged test set.

Part	Train Male (%)	Train Female (%)	Test Male Undamaged (%)	Test Female Undamaged (%)	Test Male Damaged (%)	Test Female Damaged (%)	Test Male (%)	Test Female (%)
Torso	7.42	11.26	6.46	9.74	10.73	15.79	8.59	12.765
Left arm	8.24	15.20	6.68	15.83	10.87	26.11	8.77	20.97
Right arm	12.99	12.23	10.83	12.33	14.72	13.82	12.77	13.075
Left leg	8.78	7.26	7.59	7.40	11.22	9.92	9.40	8.66
Right leg	8.39	8.01	7.81	7.45	11.81	13.26	9.81	10.35
Average	**9.16**	**10.59**	**7.87**	**10.55**	**11.87**	**15.78**	**9.86**	**13.16**

Table 3 illustrates the results after training the CNN models to entirely construct a human body. To conduct this section, we also use Adam algorithm with 1000 epochs. We choose 50 good samples and 50 damaged samples to form the test set. Thus, we could analyze the influence of destructive patterns on the overall test accuracy. The results show that the errors in the undamaged test set approximate to the training errors, while the errors in the damaged test set are not as low as the good ones. Taken as a whole, we conclude that our framework is non-sensitive to the small number of damaged samples. Moreover, the number of samples in the training set is sufficient to make inference on the shape of test samples.

After removing substandard patterns, we conduct a new training process on the latest training and test sets, and the results are given in Table 4. In the male dataset, there are 1001 training samples and 100 test samples, while there are 437 and 100 samples as training and test data in the female dataset. The average errors after feeding the interpolated primary slices into the CNN models are lower than the errors of themselves when compared to the ground truth.

Once all necessary slices are ready to build a 3D model, we carry out remeshing by using the triangular meshes. This simple rule constitutes a mesh by using three points. The points at $(i,j),(i,j+1),(i+1,j)$ on two adjacent slices will form a mesh. Likewise, the points at $(i+1,j),(i,j+1),(i+1,j+1)$ will also produce a mesh, Figure 16. As mentioned in the Introduction, this phase manipulates the meshes to fabricate the skins of 3D objects.

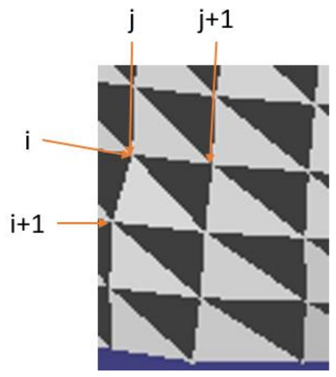


Figure 16. Remeshing based on the triangular meshes.

Table 4. Average error on each part of 3D a male and female model on the undamaged datasets before and after activating the CNN model.

Part	Train Male (Before)	Train/ Female (Before)	Test/ Male (Before)	Test/ Female (Before)	Train/ Male (After)	Train/ Female (After)	Test/ Male (After)	Test/ Female (After)
Torso	6.90	8.04	7.21	8.27	6.68	7.13	7.94	7.94
Left arm	7.62	12.77	8.31	13.81	7.15	12.60	7.67	13.83
Right arm	8.14	10.81	8.27	10.94	7.15	10.70	7.35	10.72
Left leg	7.02	6.32	7.74	6.51	7.31	5.77	7.90	5.96
Right leg	7.53	6.63	8.12	7.08	6.90	5.89	7.39	6.13
Average	**7.42**	**8.91**	**7.93**	**9.32**	**7.03**	**8.41**	**7.65**	**8.91**

The average training and test time per body part are shown in Table 5. In overall, the proposed framework took less than half a second to generate a full human body with the parameters of learning models had been trained and stored.

Table 5. Training and test time calculated on average values on both datasets.

Part	Train (s)	Test a sample (s)
Torso (6 primary slices – first model)	474	0.013
Torso (all slices – second model)	1484	0.121
Arm (10 primary slices – first model)	540	0.032
Arm (all slices – second model)	133	0.098
Leg (12 primary slices – first model)	780	0.035
Leg (all slices – second model)	764	0.108
Total	**4175**	**0.407**

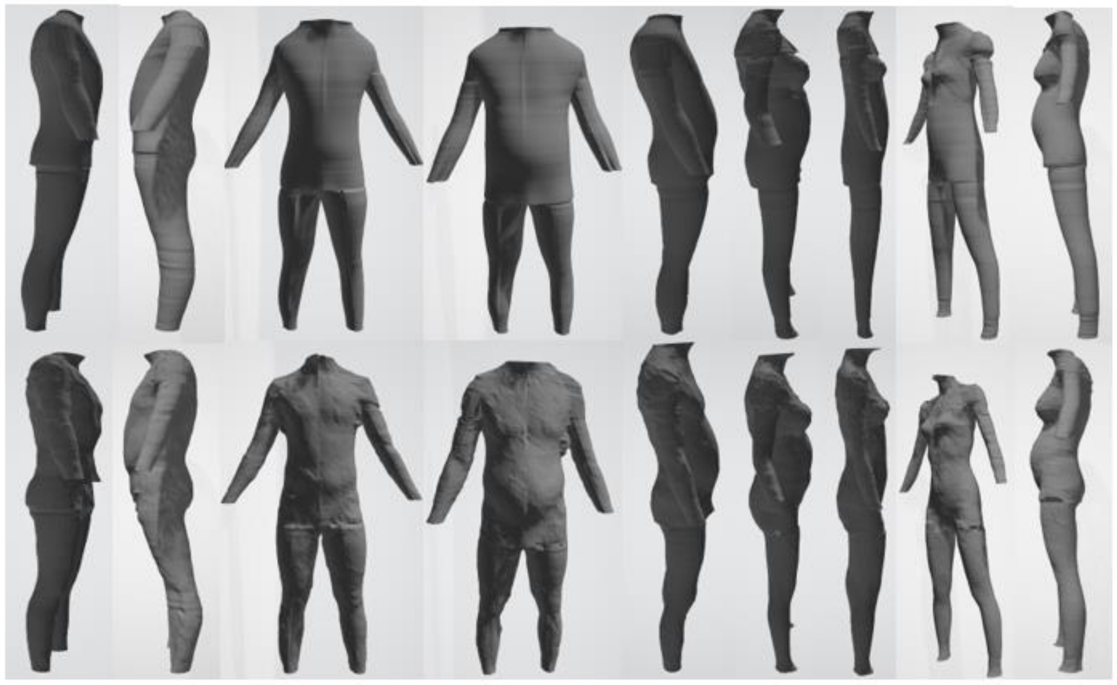


Figure 17. The 3D avatars of male and female: First row: generated shapes, Second row: original shapes.

VI. Discussions and Conclusions

Generating 3D models has been becoming an attractive field in recent years. Constructing 3D shapes is not a trivial task since the complexity of the models usually demands careful design, powerful computer hardware, and modern scanning devices. To tackle this problem, we introduced a novel method to create a new 3D model by merely taking the measurements as input. Our main contributions including (1) describing a formula to represent 3D data under slices of the point cloud, (2) introducing a two-stage framework based on Neural Networks for generating the primary slices and impaling entire slices, (3) conducting the experiment and unveiling a benchmark on the IUH and HUST 3D human dataset.

The results confirm the effectiveness of our approach due to the generated 3D point cloud models are fine enough for visualization with small errors during the reasonable running time, Figure 16. Our proposed framework not only explores the correlation between the shapes and sizes of a human body but also captures the local information among adjacent slices. Instead of directly inferring a whole 3D model, we divided the objective model into specific parts and defined suitable NN architecture for each one. In the spirit of learning detail slice shapes rather than learning overall structures, the hierarchical learning strategy is introduced in which the shape of slices corresponding to user-defined measurements are the foundation of the shape of other slices. The key idea to generate a new slice shape is to deform an initial shape depending on the training dataset. Because every single step of our method is no need to change the coordinate or reduce the dimension, we ensure that generated point clouds still look like the samples in the training data. The slice structure used in this study is not restricted in static cases. It is useful when applying to 3D dynamic models via a morphable skeleton.

The vital weakness of our approach is data deficiency. The presented learning models suffer from the under-fitting problem; hence, they cannot achieve the ideal generalization. Although having the same measurements, two slices have differences in shape, and this causes high errors in the training set, as shown in Table 2. However, the information contained in the training data is well captured by the learning models. We can conclude that based on the approximation in the errors of training and test data. The current study has only investigated the construction of point clouds. Therefore, any application requiring 3D models with sufficient mesh reconstruction might need more processing steps. Although the slice-structure is rather simple, it is challenging to achieve its status, especially when disjointing 3D shapes with intricate designs. Another shortcoming is that we are unable to compare the present study's finding with other previous studies because of the difference in the dataset and evaluating metrics.

In conclusion, this study outlined the problem of 3D modeling automatically. We have devised a strategy to encode 3D point clouds so that CNN can be implemented to learn the local information about its shapes. The present framework guarantees that a 3D human body will be constructed wholly given a set of essential anthropometric data. Moreover, the proposed method can be generalized to generate other types of 3D shapes. Our study encourages a new way to apply Deep Learning models to analyze 3D shapes.

Acknowledgement

The 3D human datasets are sponsored by Industrial University of Ho Chi Minh City (IUH) and Hanoi University of Science and Technology (HUST), Vietnam. We gratefully acknowledge their support with the detailed information of these datasets. Primarily, we would like to thank Faculty of Garment Technology and Fashion Design, Industrial University of Ho Chi Minh City for the powerful computing device assisting us to complete this study.

References

[1] L. Jiang, J. Zhang, B. Deng, H. Li and L. Liu, "3D Face Reconstruction With Geometry Details From a Single Image," *IEEE Transactions on Image Processing* , vol. 27, no. 10, pp. 4756-4770, 2018.

[2] C. Kong, H. C. Lin and S. Lucey, "Using Locally Corresponding CAD Models for Dense 3D Reconstructions From a Single Image," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[3] X. Yan, J. Yang, E. Yumer, Y. Guo and H. Lee, "Perspective Transformer Nets: Learning Single-View 3D Object Reconstruction without 3D Supervision," in *Advances in Neural Information Processing Systems*, 2016.

[4] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang and J. Xiao, "3D ShapeNets: A Deep Representation for Volumetric Shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[5] S. Y. Baek and k. Lee, "Parametric human body shape modeling framework for human-centered product," *Computer-Aided Design,* vol. 44, no. 1, pp. 56-67, 2012.

[6] B. Allen and B. Curless, "Exploring the Space of Human Body Shapes: Data-driven Synthesis under Anthropometric Control," in *SAE Technical Paper*, 2004.

[7] C. Wang, "Parameterization and parametric design of mannequins," *Computer-Aided Design,* vol. 37, no. 1, pp. 83-98, 2005.

[8] A. Sinha, A. Unmesh and Q. Huang, "SurfNet: Generating 3D shape surfaces using deep residual networks," 2017.

[9] A. Raj, C. Ham, C. Barnes, V. Kim, J. Lu and J. Hays, "Learning to Generate Textures on 3D Meshes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[10] S. Shi, X. Wang and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[11] R. Q. Charles, H. Su, M. Kaichun and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[12] B. Allen, B. Curless and Z. Popovic, "The space of human body shapes: reconstruction and parameterization from range scans," *ACM Transactions on Graphics,* vol. 22, no. 3, pp. 587-594, 2003.

[13] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn and H.-P. Seidel, "A Statistical Model of Human Pose and Body Shape," *Computer Graphics Forum,* pp. 337-346, 2009.

[14] J. Roth, Y. Tong and X. Liu, "Unconstrained 3D Face Reconstruction," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[15] J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin and D. Hoiem, "Completing 3d object shape from one depth image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[16] D. DeCarlo, D. Metaxas and M. Stone, "An anthropometric face model using variational techniques," *SIGGRAPH,* vol. 98, pp. 67-74, 1998.

[17] S. Wuhrer and C. Shu, "Estimating 3D human shapes from measurements," *Machine Vision and Applications,* vol. 24, no. 6, pp. 1133-1147, 2013.

[18] J. Wu, C. Zhang, T. Xue, B. Freeman and J. Tenenbaum, "Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling," in *Advances in neural information processing systems*, 2016.

[19] I. J. Goodfellow, J. P. Abadie, M. Mirza, B. Xu, D. W. Farley, S. Ozair, A. Courville and Y. Bengio, "Generative Adversarial Networks," Advances in neural information processing systems, 2014.

[20] S. Milz, M. Simon, K. Fischer and M. Poepper, "Points2Pix: 3D Point-Cloud to Image Translation using conditional Generative Adversarial Networks," *arXiv preprint arXiv:1901.09280,* 2019.

[21] D. W. Shu, S. W. Park and J. Kwon, "3D Point Cloud Generative Adversarial Network Based on Tree Structured Graph Convolutions," *arXiv preprint arXiv:1905.06292 (2019),* 2019.

[22] L. P. Tchapmi, V. Kosaraju, H. Rezatofighi, I. Reid and S. Savarese, "TopNet: Structural Point Cloud Decoder," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[23] J. Long, E. Shelhamer and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," in *Proceedings of the IEEE conference on computer vision and pattern* , 2015.

[24] W. Luo, Y. Li, R. Urtansun and R. Zemel, "Understanding the Effective Receptive Field in Deep Convolutional Neural Networks," in *Advances in neural information processing systems*, 2017.

[25] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE,* vol. 86, no. 11, pp. 2278-2324, 1998.

[26] R. Jarvis, "On The Identification Of The Convex Hull Of A Finite Set Of Points In The Plane," *Processing,* vol. 2, pp. 18-21, 1973.

[27] R. Davis and Altervogt, "Golden mean of the human body," *Fibonacci Quarterly,* vol. 17, pp. 340-344, 1979.

[28] V. Nair and G. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in *Proceedings of the 27th international conference on machine learning (ICML-10),* 2010.

[29] D. Kingma and B. Jimmy, "Adam: A Method for Stochastic Optimization," in *3rd International Conference for Learning Representations*, San Diego, 2015.