OrienNormNet: Orientation Normalization of 3D Body Models

Ran ZHAO, Xinxin DAI, Pengpeng HU *, Adrian MUNTEANU
Vrije Universiteit Brussel, Brussels, Belgium

https://doi.org/10.15221/22.36

Abstract

3D human body models are widely used in human-centric industrial applications, including healthcare, fashion design, body biometrics extraction, and computer animation. Prior to processing and analyzing body models, it is significantly important to rotate them to the same orientation. For instance, body measurement systems and virtual try-on systems usually assume the orientation of the body is known. These systems will output incorrect results or report errors without the correct orientation information. Unfortunately, the orientations of scanned bodies are different in practice since they are in different coordinate systems due to the setup variations of scanners. To automatically normalize the orientations of bodies is a challenging task due to the presence of pose variations, noises, and holes during 3D body scanning. In this study, we propose a novel deep learning-based method dubbed OrienNormNet for normalizing the orientation of 3D body models. As shown in Figure 1 and Figure 2, OrienNormNet directly consumes raw point clouds or mesh vertices, and it is applied in an iterative manner. First, the centroids of point clouds are translated to the origin to obtain zero-centered point clouds. Next, OrienNormNet consumes zero-centered raw point clouds and outputs coarse axis rotation angles. Finally, OrienNormNet takes the coarsely rotated point clouds from the previous processing as an updated input and outputs the refined axis rotation angles. By applying the obtained coarse and fine axis rotation angles, thousands of bodies can be adjusted to the same orientation in a few seconds. Experimental results based on synthetic datasets as well as real-world datasets validated the effectiveness of our idea.

Keywords: 3d body, orientation normalization, 3d scanning, deep learning, pose normalization

1. Introduction

3D human body models are widely used in various fields of computer vision, such as clothing design [1], [2], [3], [4], animation production [5], [6], [7], and contactless body measurement [8], [9], etc. Typically, different from other 3D objects, the orientation of a body is assumed to be not only upright but also face to the specified direction in a default coordinate system. Such an assumption is important for the success of most traditional and deep learning-based algorithms that takes human bodies as input. Arbitrary orientations will cause algorithms and systems based on the canonical orientation to output incorrect results or report errors. However, due to differences in scanning equipment, and the posture of the object, the scanned bodies are in different coordinate systems, which results in different orientations. Therefore, the scanned bodies should be adjusted to be same orientation, which is called orientation normalization.

Orientation normalization for the human body is a challenging task. It mainly contains two subgoals: (1) Upright orientation. Similar to recovering the upright orientation for general 3D objects [10], [11], human upright restoration faces the challenge of human posture diversity. Finding algorithms that converge for any pose of the human body is difficult. (2) Facing orientation. Different from orientation normalization for general 3D objects, orientation normalization for humans is not equal to uprightness. Human models, which are upright but facing in different directions, lead to methods based on the canonical orientation to output incorrect results or report errors as well. Therefore, the implementation of human body orientation normalization is to find an algorithm to unify the facing orientation of the human body while unifying the upright orientation. This algorithm needs to converge on the arbitrary human body (arbitrary body shapes and arbitrary postures) in arbitrary coordinate systems.

For very small amounts of data, we can manually adjust one by one with the help of 3D geometry processing software such as Maya, 3DS Max, and MeshLab. But such a solution fails to deal with large-scale data and is time-consuming and inaccuracy [14]. To handle large batches of data, many algorithms spring up. Existing algorithms can be mainly classified into two categories: upright orientation estimation and the support surface.

* phu@etrovub.be

Traditional upright orientation estimation usually relies on object symmetry to determine a canonical coordinate system. This kind of method aligns the models with canonical axes. Principal Component Analysis (PCA) and rank minimization are commonly employed in upright orientation estimation. PCA [12] treats the principal axes of zero-centered 3D models as canonical axes. Rank minimization is to find a coordinate system that can minimize the projection matrices [12] or third-order tensors [14] of 3D objects. The theoretical basis of this kind of method is the low-rank theory, which finds that an orientation normalized 3D object has the low-rank projection matrix and low-rank tensor. Considering the human body in most postures are not symmetric. PCA and low-rank theory both may fail. Deep learning-based upright orientation estimation methods break the bottleneck of traditional methods and can work for asymmetric objects, such as 3D Convolutional Network-based [10] and reinforcement learning-based UprightRL [15]. However, even deep learning-based methods cannot address the oriented problem after upright.

Another solution is to detect the support surface and then align the support surface to the ground to normalize the orientation. This type of method usually simplifies the surface structure of the 3D object, selects the possible contact surfaces, and selects the most suitable contact surface as the surface is in contact with the ground. For example, [16] employs UV-measurement to evaluate candidate bases, and [17] uses the combination of Random Forest and Support Vector Machine to select the best base. These methods are suitable for objects of large contact area with the ground. The deep learning-based method [11] provided a novel idea for contact surface detection, which defines the detection task as a classification task. It enables point cloud classification algorithms [18], [19] to be directly applied to the detection of contact surfaces. These deep learning-based methods are more for objects with smaller contact surfaces. Even though the requirement of the contact area is relaxed for the deep learning-based methods, it still requires the object to have a sufficient contact area to be detected. Furthermore, the detection of the contact surface is also limited by the complex posture of the human body, and these methods cannot solve the problem of facing directions even standing upright.

In this paper, we address this problem by dividing it into two sub-tasks: human upright normalization and human facing normalization, and propose a two-step iterative neural network called OrienNormNet. OrienNormNet is designed to transform the human body in any pose to the canonical facing direction in the canonical coordinate system. Compared with previous methods, our method has two main advantages: (1) our method is more robust. It can handle the human body in any posture, regardless of whether the human body has symmetry in the posture, or whether the contact area with the ground is large enough. In addition, it shows good performance on both synthetic bodies and real-world bodies with missing points and noise. (2) our method addresses upright and facing normalization.

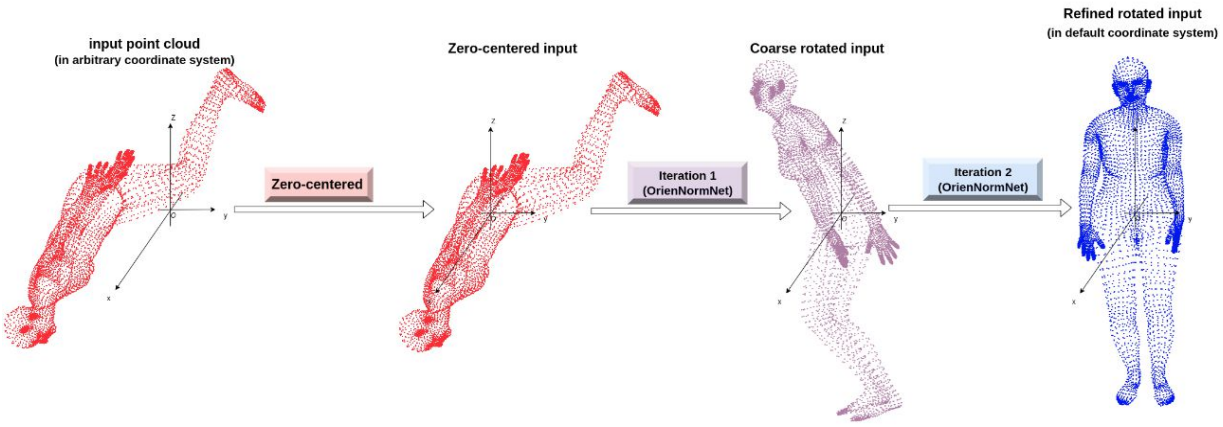

Fig. 1. Overview of the proposed method: Given a point cloud in an arbitrary coordinate system, we first translate its centroid to the origin to obtain zero-centered input. The preprocessed point cloud will be fed into OrienNormNet to normalize its orientation in an iterative manner. The first iteration takes a zero-centered point cloud as input. After the 1st processing of OrienNormNet, we can obtain the coarsely rotated angles to rotate the input point cloud with its head up but facing in various directions. Then the OrienNorNet with the same architecture but different learned weights takes the coarsely rotated point cloud as input and output refined rotation angles, which can make the head-up point cloud face the expected direction.

2. Method

2.1. Problem statement

Given a human point cloud (or mesh vertices) $S = \{x_i = (x_{i1}, x_{i2}, x_{i3}) \in \boldsymbol{R}^3 | i = 1,2, \dots, n\}$ with n points in an arbitrary coordinate system, there exists a transformation matrix T to transform S from a random orientation to a canonical one.

$$\hat{S} = T \cdot S \qquad (1)$$

Here, T can be split into translation and rotation.

2.1.1. translation

The translation aims to shift the point cloud so that its centroid is the origin of the canonical coordinate system. For each S, denote the centroid of S as (c_1, c_2, c_3), which is

$$(c_1, c_2, c_3) = \frac{1}{n}\sum_{i=1}^{n}(x_{i1}, x_{i2}, x_{i3}) \qquad (2)$$

The zero-centered S is

$$\bar{S} = (\overline{x_{i1}}, \overline{x_{i2}}, \overline{x_{i3}})_{i=1}^{n} = (\mathrm{x_{i1}} - \mathrm{c_1}, \mathrm{x_{i2}} - \mathrm{c_2}, \mathrm{x_{i3}} - \mathrm{c_3})_{i=1}^{n} \qquad (3)$$

After translation, $\bar{S}$ has been in the canonical coordinate system. The centroid of $\bar{S}$ is the origin of the canonical coordinate system.

2.1.2. Rotation

The rotation matrix R, $R = (r_{ij})_{3\times3} \in SO(3)$, is to adjust the orientation of $\bar{S}$. R can be represented by the multiplication of three basic rotation matrices $R_x(\alpha)$, $R_y(\beta)$ and $R_z(\gamma)$, which can rotate an object by roll angle α, pitch angle β, and yaw angle γ along the x-axis, y-axis, and z-axis, respectively.

$$\begin{aligned} R &:= R_z(\gamma)R_y(\beta)R_x(\alpha) \\ &= \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \qquad (4) \\ &= \begin{bmatrix} \cos\beta\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \cos\beta\sin\gamma & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ -\sin\beta & \sin a\cos\beta & \cos a\cos\beta \end{bmatrix} \end{aligned}$$

Therefore, the orientation normalized S is

$$\hat{S} = \bar{S} \cdot R = \left(\sum_{j=1}^{3}\overline{x_{ij}}\, r_{j1}, \sum_{j=1}^{3}\overline{x_{ij}}\, r_{j2}, \sum_{j=1}^{3}\overline{x_{ij}}\, r_{j3}\right)_{i=1}^{n} \qquad (5)$$

Through experiments, we found that due to the diversity of human poses and body shapes, directly regressing the rotation matrix is not very good. To improve the performance of the proposed method, we propose a coarse-to-refine strategy by a two-step iteration, which normalizes the orientation of the human body through two rotations, whose rotation matrices are denoted as R_1 and R_2. Each rotation matrix $R^{(t)}$ $(t = 1,2)$ can be further split into rotation matrices along the x-axis $(R_x(\alpha^{(t)}))$, y-axis $(R_y(\beta^{(t)}))$ and z-axis $(R_z(\gamma^{(t)}))$, which are determined by the rotation angles $\alpha^{(t)}$, $\beta^{(t)}$, and $\gamma^{(t)}$ along the x-axis, y-axis, and z-axis, respectively.

$$\begin{aligned} R &= (r_{ij})_{3\times3} = \prod_{t=1}^{2} R^{(t)} \qquad (6) \\ &= \prod_{t=1}^{2} R_z(\gamma^{(t)})R_y(\beta^{(t)})R_x(\alpha^{(t)}) \end{aligned}$$

Therefore, the task of finding R is equal to finding $\{R^{(t)}\}_{t=1}^{2}$, which is equal to finding yaw, pitch, and roll angles set $\{(\alpha^{(t)}, \beta^{(t)}, \gamma^{(t)})\}_{t=1}^{2}$.

We train an iteration-based neural network to learn the final mapping from $\bar{S}$ to (α, β, γ).

$$\begin{aligned} \Phi: \ \boldsymbol{R}^{n\times 3} &\rightarrow \boldsymbol{R}^{3} \\ \bar{S} &\mapsto (\alpha, \beta, \gamma) \end{aligned} \qquad (7)$$

Here, Φ consists of two mapping $\phi_t : \widehat{S^{(t)}} \mapsto (\alpha^{(t)}, \beta^{(t)}, \gamma^{(t)})\ (t = 1,2)$. $\widehat{S^{(1)}}$ represents $\bar{S}$. The t-th iteration can learn the mapping ϕ_t. Let $\widehat{S^{(3)}}$ represent $\hat{S}$. We have

$$\widehat{S^{(t+1)}} = \widehat{S^{(t)}} \cdot R^{(t)} = \widehat{S^{(t)}} \cdot R_z(\gamma^{(t)}) R_y(\beta^{(t)}) R_x(\alpha^{(t)}) \qquad (8)$$

2.2. Dataset

A large number of human bodies are required to train our proposed neural network, which must have different postures, and body shapes, and be located in different coordinate systems. Therefore, we train our neural network using synthetic data instead of real-world data.

Similar to [20], we synthesize 150k human bodies by leveraging the SMPL [21] model as ground-truth bodies. Then we add random translation to each point cloud and save the rotation angles tuple $(\alpha_{GT}, \beta_{GT}, \gamma_{GT})$ as ground-truth angles.

We randomly split the human bodies into 99% training set, 0.7% validation set, and 0.3% testing set.

2.3. OrienNormNet

Since we divide the orientation normalization into two steps: upright normalization and facing normalization, the **Orien**tal **Norm**alization **Net**work (OrienNormNet) we proposed implements a coarse-to-refine normalization process in two iterations to handle this complex task.

OrienNormNet is a PointNet-based [18] encoder-decoder framework, as Fig.2 shows. The encoder consists of two stacked simplified PointNet. For two iterations, the input of the first shared MLP with hidden layers of 128 and 256 nodes is $\widehat{S^{(t)}} = \left(\widehat{x_{ij}^{(t)}}\right)_{n\times 3}$ $(t = 1,2)$, where $\widehat{S^{(1)}}$ represents $\bar{S}$ and $\widehat{S^{(2)}}$ is the coarse rotated point cloud obtained from the first iteration. The output of the first shared MLP is the local feature matrix $F_{local}^{(t)} = \left(f_{ij}^{(t)}\right)_{n\times 256}$ $(t = 1,2)$, which will extract a global feature $F_{global}^{(t)} = \left(f_i^{(t)}\right)_{i=1}^{256}$ $(t = 1,2)$ by max-pooling for each column.

$$F_{local}^{(t)} = \left(f_{ij}^{(t)}\right)_{n\times 256} = MLP^{1}\left(\widehat{S^{(t)}}\right) \qquad (9)$$

$$\begin{aligned} F_{global}^{(t)} &= \left(f_1^{(t)}, f_2^{(t)}, \dots, f_{256}^{(t)}\right) = maxpool\left(F_{local}^{(t)}\right) \\ &= \left(max\left(f_{i1}^{(t)}\right), max\left(f_{i2}^{(t)}\right), \dots, max\left(f_{i256}^{(t)}\right)\right) \end{aligned} \qquad (10)$$

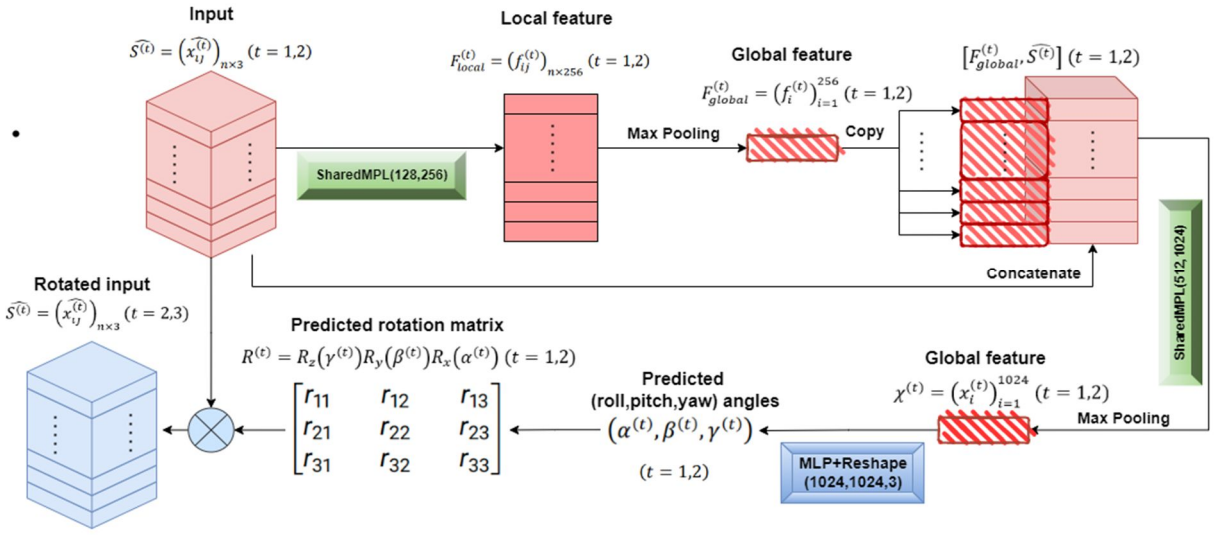

Fig. 2. The architecture of OrienNormNet. Point clouds are fed into a shared MPL layer to learn point-wise features. Then, point-wise features are fused into a global feature by the max-pooling operation. The global feature is stitched to each point of input to obtain a new point matrix. By repeating the learning strategy and there shaping operation, the new point matrix is converted to a set of axis rotation angles, which adjusts the orientation of the input point cloud.

The combination of $F_{global}^{(t)}$ and $\widehat{S^{(t)}}$ is the input for the second PointNet with hidden layers of 512 and 1024 nodes and max-pooling, which will output the final feature vector $\chi^{(t)} = \left(x_i^{(t)}\right)_{i=1}^{1024} (t = 1,2)$.

$$\left[F_{global}^{(t)}, \widehat{S^{(t)}}\right] = \left(f_1^{(t)}, f_2^{(t)}, \dots, f_{256}^{(t)}, \widehat{x_{i1}^{(t)}}, \widehat{x_{i2}^{(t)}}, \widehat{x_{i3}^{(t)}}\right)_{i=1}^{n} \quad (11)$$

$$\chi^{(t)} = \left(x_1^{(t)}, x_2^{(t)}, \dots, x_{1024}^{(t)}\right) = maxpool\left(MLP^2\left(\left[F_{global}^{(t)}, \widehat{S^{(t)}}\right]\right)\right) \quad (12)$$

The decoder includes three fully connected (FC) layers with 1024,1024, and 3 nodes to decode $\chi^{(t)}$ to $\left(\alpha^{(t)}, \beta^{(t)}, \gamma^{(t)}\right)$ $(t = 1,2)$.

2.4. Loss function

Our OrienNormNet has two steps. The first step is to learn three axis-rotation angles, which can rotate $\overline{S}$ to be almost upright. We divide the loss function L into body loss L_{body} and angle loss L_{angle} to supervise the first step.

$$L = L_{body} + L_{angle} \quad (13)$$

We refine the upright posture and rotate the facing direction for the second step. The refinement step is a small adjustment based on the first step. Therefore, we only need to adapt L_{body} to supervise the learning process of optimization angles.

2.4.1. Body loss

We set the SMPL model in the canonical coordinate system as ground truth, denoted as S_{GT}. The body loss is defined as the mean square error (MSE) between $\widehat{S^{(t)}}(t = 2,3)$ and S_{GT}.

$$L_{body}\left(\widehat{S^{(t)}}, S_{GT}\right) = \frac{1}{\left|\widehat{S^{(t)}}\right|} \sum_{i=1}^{\left|\widehat{S^{(t)}}\right|} \sum_{j=1}^{3} \left(\widehat{x_{ij}^{(t)}} - s_{ij}\right)^2 \quad (14)$$

2.4.2. Angle loss

When transforming the point cloud of the canonical orientation into the data required for the experiment, we record the rotation angles $(\alpha_{GT}, \beta_{GT}, \gamma_{GT})$ as the ground truth. The MSE between the $(\alpha^{(1)}, \beta^{(1)}, \gamma^{(1)})$ and $(\alpha_{GT}, \beta_{GT}, \gamma_{GT})$ is the angle loss we define.

$$L_{angle}\left(\left(\alpha^{(1)}, \beta^{(1)}, \gamma^{(1)}\right), (\alpha_{GT}, \beta_{GT}, \gamma_{GT})\right) = \frac{1}{3}\left[\left(\alpha^{(1)} - \alpha_{GT}\right)^2 + \left(\beta^{(1)} - \beta_{GT}\right)^2 + \left(\gamma^{(1)} - \gamma_{GT}\right)^2\right] \quad (15)$$

3. Experiments and results

3.1. Environment

The training operation is set on a desktop PC equipped with processor (Intel(R) Core i9-9900 CPU @ 3.10GHz × 16), graphics (GeForceRTX 2080 Ti/PCIe/SSE2), 62,7 GiB memory, and Ubuntu 16.04 LTS. All the training and test are run under CUDA 9.0, cudnn 7.0.5, python 3.6, and Tensorflow-GPU 1.0.5.

3.2. Evaluation criteria

Since all the datasets we employ have ground-truth rotation angles, and the values of (α, β, γ) can be recovered from the rotation matrix $R = R^{(2)} \cdot R^{(1)}$. we adapt the absolute error, mean absolute error, and accuracy to quantitatively evaluate the performance of our method.

3.2.1. Absolute error

The absolute error is the difference between the predicted rotation angle along one axis and the corresponding ground-truth angle. This error shows the performance on each axis, which is defined as

$$AE\left(A_{predict}, A_{true}\right) = \left|A_{predict} - A_{true}\right| \quad (16)$$

where $A_{predict} = \{\alpha, \beta, \gamma\}$, and $A_{true} = \{\alpha_{GT}, \beta_{GT}, \gamma_{GT}\}$ are the corresponding ground-truth angles set.

3.2.2. Mean error

The mean error shows the overall performance of our method by computing the mean angular deviation between three rotation angles and their ground-truth angles.

$$ME = \frac{1}{3}[AE(\alpha, \alpha_{GT}) + AE(\beta, \beta_{GT}) + AE(\gamma, \gamma_{GT})] \quad (17)$$

3.2.3. Accuracy

We set a tolerance threshold $\tau = 5°$. The predicted angle with absolute error within the threshold is considered the correct angle for prediction. We take the ratio of the correct number of predictions to the sample number (denoted as N) as the accuracy rate.

$$Accuracy = \frac{\#\{AE\left(A_{predict}, A_{true}\right) < \tau\}}{N} \quad (18)$$

And the mean accuracy is the proportion of predictions with ME (Eq.16) small than τ.

$$MeanAccuracy = \frac{\#\{ME < \tau\}}{N} \quad (19)$$

3.3. Results

We tested our algorithm on both synthetic and real-world bodies. Our experimental results consist of three parts: (1) Experimental results on unseen synthetic data. We used 450 unseen synthetic human data for testing. These bodies do not have any noise nor any holes. (2) Test results of FAUST [22] data. We use 200 human bodies from the FAUST dataset, which are noise-free with some missing points in the hands and feet. (3) Experimental results of real-world datasets in [23]. We use 48 human bodies from the dataset provided by [23], which have more noise in the extremities with some holes. We conducted qualitative and quantitative assessments of these experimental results, respectively.

3.3.1. Qualitative evaluation

Fig.3, Fig.4, and Fig.5 visualize the zero-centered inputs and the corresponding results of unseen synthetic data, FAUST data, and a real-world dataset from [23], respectively. These figures illustrate that our method works well on both synthetic bodies and real-world bodies although holes, noises, and complicated postures exist. The first two columns in these three figures show the input bodies. They are in different orientations with different postures. The last two columns present the results for two steps. The third column proves that the first step can coarsely rotate the input to almost upright, while the last column shows the refinement step is capable to realize full orientation normalization (standing upright and facing to the canonical direction) based on the coarsely rotating bodies.

3.3.2. Quantitative evaluation

We perform a local and global quantitative evaluation of our results in terms of angular errors.

We use Eq.16 and Eq.18 to analyze the errors of the rotation angles along the three coordinate axes one by one to evaluate the performance of our method along each coordinate axis. Fig.6 illustrates the error distributions of α, β, and γ for three different datasets. Our method performs similarly on synthetic and real-world data, where α and γ are mostly between 0 ° to 3 °, while β is mostly between 0 ° and 4 °.

In addition, the first three rows of data in Table 1, 2, and 3 respectively give the specific values of the rotation angle errors of the three datasets. The average errors of α, β, and γ are all less than 3 °, the minimum errors are less than 0.2 °, and the maximum errors do not exceed 10 °. When the threshold (Eq. 18) is set to 5 °, the accuracy of α, β, and γ predicted in the three datasets are all above 85%. On the predicted results of the real-world dataset (FAUST and data from [23]), the accuracy of α and γ reach 100%.

On the other hand, we employ Eq. 17 and Eq.19 to evaluate the overall performance of our method. The last columns of the three boxplots in Fig.6 visualize that for three different datasets, the mean errors of the rotation angles are concentrated between 1 ° and 2.5 °. The last rows of Table.1, Table.2, and Table.3 show that for the three different datasets, the average of the mean errors is approximate 2 °, the maximum is close to 4.5 °, and the minimum is below 1 °. Setting the threshold (Eq.19) to 5 °, the mean accuracy of the predicted angles achieves 100%.

4. Conclusions

In order to solve the human orientation normalization task, we propose a novel neural network with two iterations, called OrienNormNet, which is a coarse-to-refine PointNet-based neural network constrained by special loss functions. Experiments show that OrienNormNet can be trained on synthetic data and practically generalize well to real-world data. OrienNormNet works as well for real-world data with noise and holes and can constrain the mean rotation error to 5 °. Therefore, our method can adjust a large batch of scanned human bodies to the same orientation within seconds by applying the obtained coarse and fine axis rotation angles. Compared with other orientation normalization methods, our method is fully automatic and fast and can work well for bodies under different postures. Moreover, our method is effective not only for the human body but also for the orientation normalization of other objects that need to constrain the facing direction.

However, our method also has some limitations. First, our method is mainly aimed at the complete point clouds. Even if there are holes, the holes do not affect the integrity of the point clouds. Research on orientation normalization of partial point clouds would be interesting work. Furthermore, although our method can achieve orientation normalization for other objects, it requires retraining of the network. Extending this method to arbitrary objects is also meaningful.

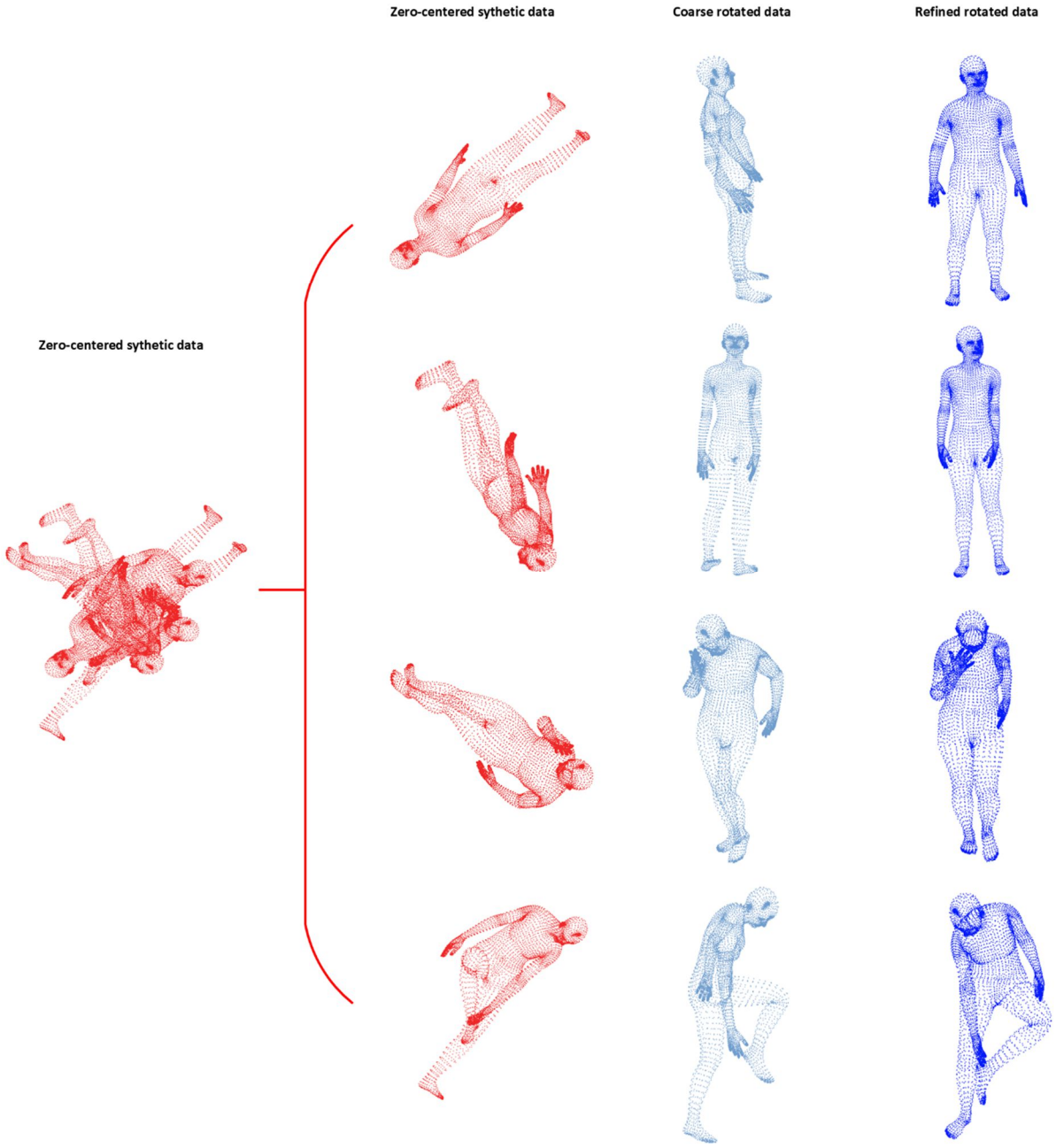

Fig. 3. Our results are based on the unseen synthetic data. All the synthetic data are pure without holes. We show the zero-centered input data in the first column, the coarsely rotated results from the first iteration in the second column, and the refined results from the second iteration in the last column. For synthetic data, the first iteration can rotate the input body in arbitrary postures to almost upright. The second iteration can fix this problem to rotate the almost upright body to canonical orientation.

Table 1. Average, maximum, and minimum AE & ME and accuracy on 450 unseen synthetic bodies.

Synthetic	Accuracy (%)	Average (°)	Maximum (°)	Minimum (°)
$AE(\alpha, \alpha_{GT})$	98	1.912	6.353	0.001
$AE(\beta, \beta_{GT})$	99	1.791	5.747	0.007
$AE(\gamma, \gamma_{GT})$	92	2.237	7.681	0.004
ME	100	1.980	4.476	0.135

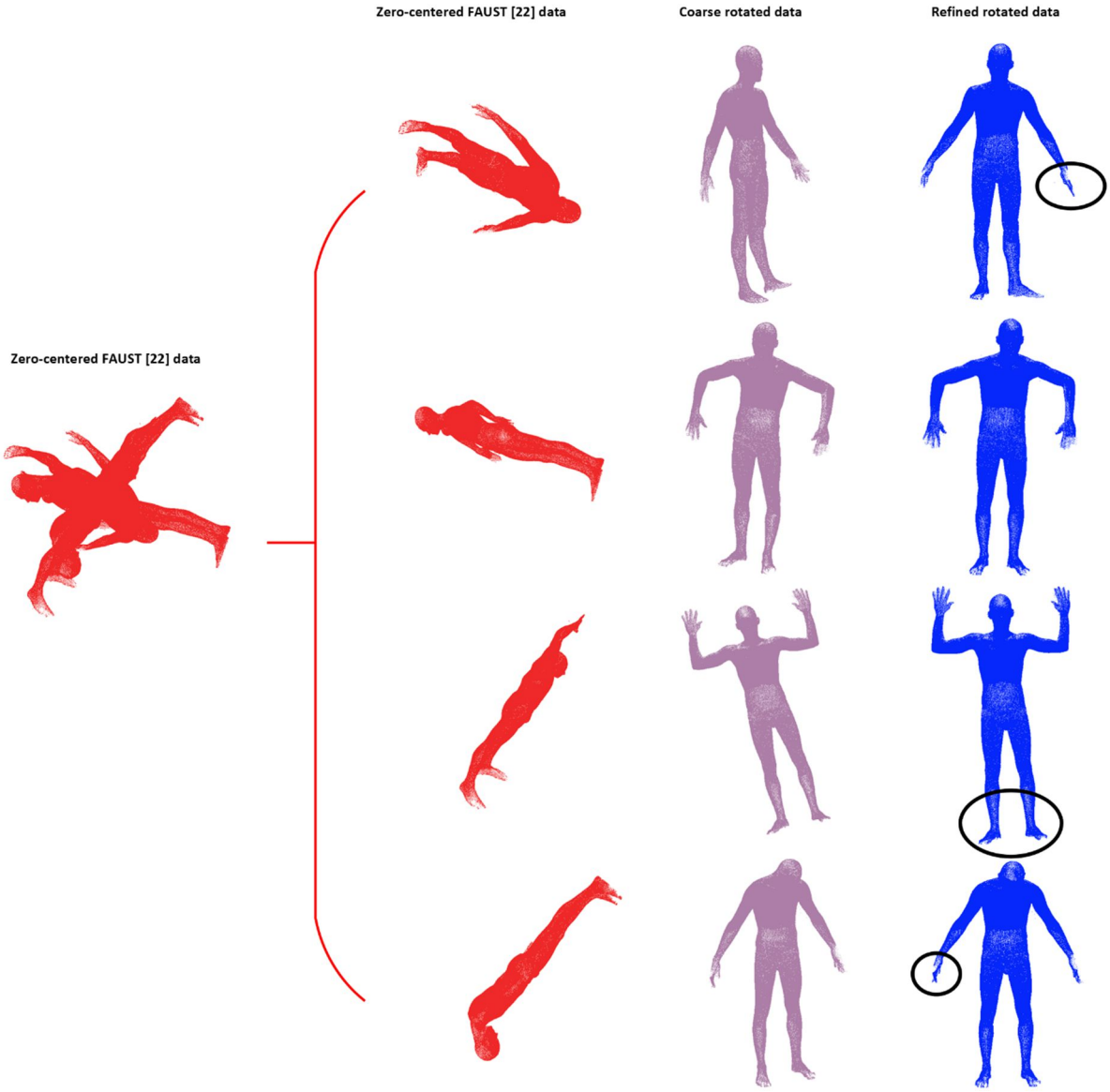

Fig. 4. Our results are based on the real-world scanned bodies from FAUST data [22]. These point clouds have obvious missing points (black ellipses) on the hands and feet. Similar to the synthetic data (Fig. 3.), the first column shows the zero-centered point clouds, and the results of coarse and refined rotations are in the middle and last columns, respectively.

Table 2. Average, maximum, and minimum AE & ME and accuracy on 200 FAUST [22] bodies.

FAUST [22]	Accuracy (%)	Average (°)	Maximum (°)	Minimum (°)
$AE(\alpha, \alpha_{GT})$	100	1.564	4.280	0.002
$AE(\beta, \beta_{GT})$	86	2.680	9.908	0.003
$AE(\gamma, \gamma_{GT})$	100	1.650	4.588	0.012
ME	100	1.965	4.731	0.115

Table 3. Average, maximum, and minimum AE & ME and accuracy on 48 real-world bodies from [23].

Data from [23]	Accuracy (%)	Average (°)	Maximum (°)	Minimum (°)
$AE(\alpha, \alpha_{GT})$	100	1.705	3.189	0.192
$AE(\beta, \beta_{GT})$	92	2.294	8.393	0.157
$AE(\gamma, \gamma_{GT})$	100	1.917	4.357	0.130
ME	100	1.972	4.434	0.629

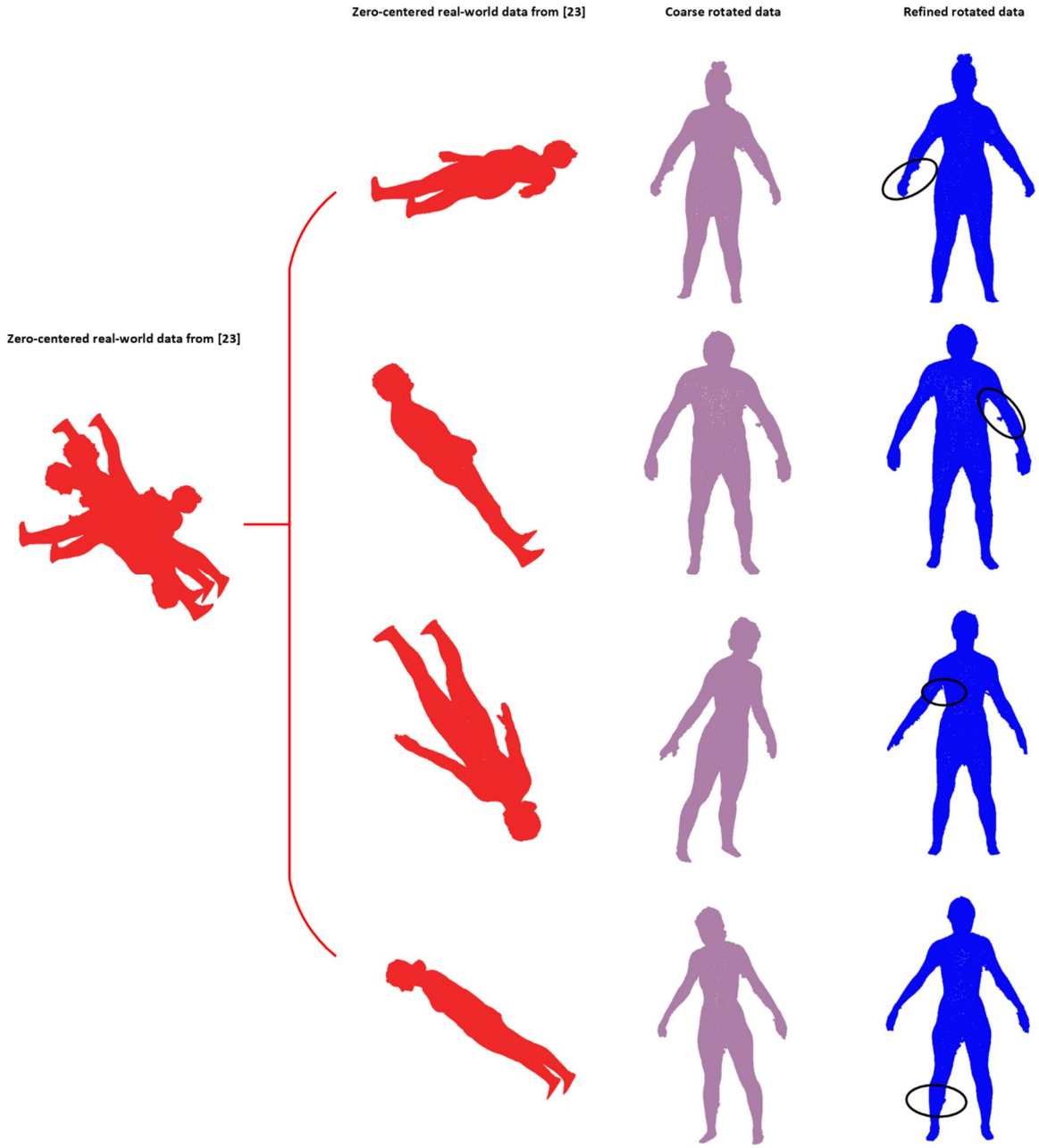

Fig. 5. Our results are based on real-world data from [23]. These point clouds have noise (black ovals) on the arms, underarms, and lower extremities. Our method can still achieve orientation normalization in two steps even with noise.

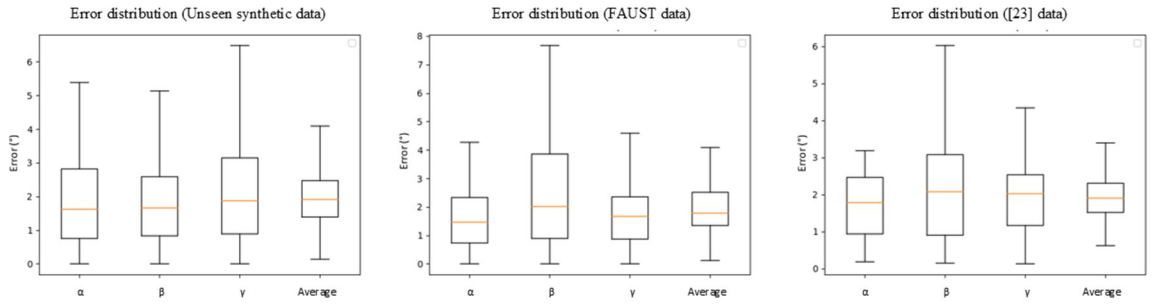

Fig. 6. Three boxplots illustrate the error distributions for three different datasets. For each boxplot, the first three boxes show the angular deviation along three axes, while the last box presents the mean angular deviation.

5. Acknowledgements

This work was supported in part by the Innoviris under Project AI43D in close collaboration with Spentys and in part by FWO under Project G084117.

References

[1] R. White, K. Crane, and D. A. Forsyth, “Capturing and animating occluded cloth,” ACM Transactions on Graphics (TOG), vol. 26, no. 3, pp. 34–es, 2007, https://doi.org/10.1145/1276377.1276420

[2] L. Duan, Z. Yueqi, W. Ge, and H. Pengpeng, “Automatic three-dimensional-scanned garment fitting based on virtual tailoring and geometric sewing,” Journal of Engineered Fibers and Fabrics, vol. 14, p. 1558925018825319, 2019, https://doi.org/10.1177/1558925018825319

[3] P. Hu, E. S. Ho, N. Aslam, T. Komura, and H. P. Shum, “A new method to evaluate the dynamic air gap thickness and garment sliding of virtual clothes during walking,” Textile research journal, vol. 89, no. 19-20, pp. 4148–4161, 2019, https://doi.org/10.1177/0040517519826930

[4] P. Hu, N. Nourbakhsh, J. Tian, S. Sturges, V. Dadarlat, and A. Munteanu, “A generic method of wearable items virtual try-on,” Textile Research Journal, vol. 90, no. 19-20, pp. 2161–2174, 2020, https://doi.org/10.1177/0040517520909995

[5] S. I. Park and S.-J. Lim, “Template-based reconstruction of surface mesh animation from point cloud animation,” ETRI journal, vol. 36, no. 6, pp. 1008–1015, 2014, https://doi.org/10.4218/etrij.14.0113.1181

[6] A. E. Ichim, S. Bouaziz, and M. Pauly, “Dynamic 3d avatar creation from hand-held video input,” ACM Transactions on Graphics (ToG), vol. 34, no. 4, pp. 1–14, 2015, https://doi.org/10.1145/2766974

[7] S. Zeitvogel and A. Laubenheimer, “Towards end-to-end 3d human avatar shape reconstruction from 4d data,” in 2018 International Symposium on Electronics and Telecommunications (ISETC). IEEE, 2018, pp. 1–4, https://doi.org/10.1109/ISETC.2018.8583952

[8] Y. Zhong, D. Li, G. Wu, and P. Hu, “Automatic body measurement based on slicing loops,” International Journal of Clothing Science and Technology, 2018, https://doi.org/10.1108/IJCST-06-2017-0086

[9] P. Hu, N. N. Kaashki, V. Dadarlat, and A. Munteanu, “Learning to estimate the body shape under clothing from a single 3-d scan,” IEEE Transactions on Industrial Informatics, vol. 17, no. 6, pp. 3793–3802, 2020, https://doi.org/10.1109/TII.2020.3016591

[10] Z. Liu, J. Zhang, and L. Liu, “Upright orientation of 3d shapes with convolutional networks,” Graphical Models, vol. 85, pp. 22–29, 2016, https://doi.org/10.1016/j.gmod.2016.03.001

[11] X. Pang, F. Li, N. Ding, and X. Zhong, “Upright-net: Learning upright orientation for 3d point cloud,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 14 911–14 919.

[12] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, “Rotation invariant spherical harmonic representation of 3 d shape descriptors,” in Symposium on geometry processing, vol. 6, 2003, pp. 156–164.

[13] Y. Jin, Q. Wu, and L. Liu, “Unsupervised upright orientation of man-made models,” Graphical Models, vol. 74, no. 4, pp. 99–108, 2012, https://doi.org/10.1016/j.gmod.2012.03.007

[14] W. Wang, X. Liu, and L. Liu, “Upright orientation of 3d shapes via tensor rank minimization,” Journal of Mechanical Science and Technology, vol. 28, no. 7, pp. 2469–2477, 2014, http://doi.org/10.1007/s12206-014-0604-6

[15] L. Chen, J. Xu, C. Wang, H. Huang, H. Huang, and R. Hu, “Uprightrl: Upright orientation estimation of 3d shapes via reinforcement learning,” in Computer Graphics Forum, vol. 40, no. 7. Wiley Online Library, 2021, pp. 265–275, https://doi.org/10.1111/cgf.14419

[16] C.-K. Lin and W.-K. Tai, “Automatic upright orientation and good view recognition for 3d man-made models,” Pattern recognition, vol. 45, no. 4, pp. 1524–1530, 2012, https://doi.org/10.1016/j.patcog.2011.10.022

[17] Z. Liu, J. Zhang, and L. Liu, “Upright orientation of 3d shapes with convolutional networks,” Graphical Models, vol. 85, pp. 22–29, 2016 , https://doi.org/10.1016/j.gmod.2016.03.001

[18] C. R. Qi, H. Su, K. Mo, 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, pp. 652–660 , https://doi.org/10.1109/CVPR.2017.16

[19] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, “Dynamic graph cnn for learning on point clouds,” Acm Transactions On Graphics (tog), vol. 38, no. 5, pp. 1–12, 2019, https://doi.org/10.1145/3326362

[20] P. Hu, R. Zhao, X. Dai, and A. Munteanu, “Predicting high-fidelity human body models from impaired point clouds,” Signal Processing, vol. 192, p. 108375, 2022, https://doi.org/10.1016/j.sigpro.2021.108375

[21] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, “Smpl: A skinned multi-person linear model,” ACM transactions on graphics (TOG), vol. 34, no. 6, pp. 1–16, 2015, https://doi.org/10.1145/2816795.2818013

[22] F. Bogo, J. Romero, M. Loper, and M. J. Black, “Faust: Dataset and evaluation for 3d mesh registration,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3794–3801, https://doi.org/10.1145/2816795.2818013

[23] N. N. Kaashki, P. Hu, and A. Munteanu, “Anet: A deep neural network for automatic 3d anthropometric measurement extraction,” IEEE Transactions on Multimedia, 2021, https://doi.org/10.1109/TMM.2021.3132487