CH-010

Orientation Estimation of 3D Human Scans Using Interior Point Clouds and Vision Transformer

李　伊晴†　　藤田　悟†
Yiqing Li　　Satoru Fujita

Abstract: We propose a novel method for estimating the facing direction of human 3D scan data using Vision Transformers (ViT). Unlike existing approaches that rely on clean, standardized poses, our method handles noisy scans of various poses by extracting interior point clouds, which are more stable and informative. Interior point clouds are then sliced horizontally and projected into 2D binary images for ViT-based orientation prediction. Our results improved robustness and accuracy, highlighting the method's potential for applications such as automatic rigging, 3D scan alignment, and avatar animation.

1. Introduction

Estimating the facing direction of human 3D models is important in many applications, including automatic rigging, 3D scan alignment, and AR/VR avatar control. While existing methods often assume standard initial poses such as T- or A-poses, real-world scan data includes models in varied natural standing poses with unknown orientations.

To overcome this limitation, we propose a method based on Vision Transformers (ViT) [1] to estimate the orientation of human 3D scan data. Since human 3D scan data has a complicated surface, we first extracted the interior point cloud from the input mesh. The interior point cloud consists of points slightly inside the surface and points lying on the internal center line of the 3D model, which can speed up the training phase and help the neural network to understand the orientation accurately. Then, the 3D model is horizontally segmented, and each segment is projected onto a 2D binary image. These images are fed into ViT to estimate the model's orientation vector.

This paper contributes an interior point cloud representation that consists of points located slightly inside the mesh surface. The interior point cloud showed better results during our experiments than surface points. Moreover, since it contains volumetric cues indicative of potential joint locations, it also has the potential to support downstream tasks such as pose estimation with improved accuracy. Building on this representation, we introduce a Vision Transformer-based method to estimate the facing direction of 3D human models accurately. This approach overcomes the limitations of traditional surface-based methods and contributes a robust orientation estimation technique applicable to unaligned human scan data.

2. Related Work

2.1 Vision Transformer

Vision Transformer (ViT) [1] divides an image into smaller patches, which are then treated as tokens and input into the Transformer model. This approach makes feature extraction from images a practical task. In this study, we leverage ViT's advanced feature extraction capabilities to estimate the orientation of a human body from 2D binary images obtained by projecting vertically segmented point clouds.

2.2 3D Point cloud learning

In recent years, many neural networks have focused on learning the geometric structure of objects directly from 3D point clouds. PointNet[2], introduced in 2017, was the first to perform point cloud processing by extracting global features through max-pooling of per-point features generated by MLP. In 2019, EdgeConv[3] was developed, which dynamically transferred features from point to point based on the edges of the mesh. This method could extract global features by incorporating local features. As a result, EdgeConv is capable of capturing the fine-grained geometric properties of point clouds effectively. In 2022, PointMLP[4] reevaluated the network architecture for point cloud learning and found that progressively extracting local features using residual MLP blocks was sufficient to capture both local and global features.

These point-based approaches were more efficient in terms of memory and computation than voxel-based methods. However, due to point clouds' inherently sparse and disordered nature, extracting stable and consistent local features remains challenging. This irregularity often requires denoising or sampling strategies to achieve robust feature representation. On the other hand, our method reduces the influence of surface noise by projecting points located slightly beneath the mesh surface and around potential joint regions onto 2D planes. This strategy reduces sensitivity to mesh irregularities and enables efficient training with minimal preprocessing. Unlike conventional approaches that often require careful sampling or denoising, our method directly handles raw scan data without additional refinement.

2.3 Automatic rigging and orientation estimation

Since 2007, researchers have explored an automatic rigging method. In 2007, Ilya Baran and Jovan Popovic developed an automated rigging technique, introducing 'Pinocchio' [5]. They inserted multiple spheres into the characters' models and constructed a graph passing through the centers of these spheres. Subsequently, they removed unnecessary lines and generated approximate alignment of skeletons. In 2020, Xu et al introduced a neural network-based rigging method called RigNet [6] for animating character models. This method utilizes the mesh model's shape, joint locations, skeletal structure, and skinning information. RigNet demonstrated the ability to predict joint positions and skeletal structures accurately. In 2023, Ma et al [7] observed that the predicted joints of RigNet lack anatomical awareness, which necessitates additional manual annotation for motion retargeting. This issue also results in invalid skeletons

†法政大学　Hosei University

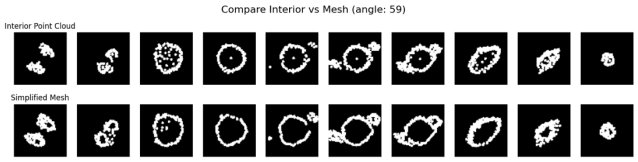

Figure 1: Interior point clouds (top) and horizontal slices of the mesh model (bottom) orthogonal to the vertical axes.

that require manual correction. To address these challenges, Ma introduced a template-aware bone-flow-guided method to improve the accuracy and stability of automatic rigging. However, since all these automatic rigging methods rely on character models in a fixed pose and orientation, they cannot rig rotated models at arbitrary angles. To overcome this limitation and enable robust rigging of arbitrary 3D scans, we aim to estimate the facing direction of human models through orientation estimation.

3. Orientation Estimation

3.1 Interior point cloud

We have found that using the interior point cloud enhances the understanding of rotation compared to mesh surfaces. 3D scan data often contains noise, missing parts, and complex surface details. In contrast, interior point clouds are less affected by surface resolution or quality, and they can represent consistent geometric features. For example, as shown in Figure 1, the second column illustrates a 2D projection of the legs where mesh noise appears between the two limbs. From the cross-sectional view, these two legs are perceived as a single connected structure. In contrast, this noise is absent in the corresponding interior point cloud representation. In the fifth column, the mesh exhibits an incomplete surface, whereas the interior point cloud maintains a more continuous structure.

To compute the interior point cloud, we followed the work of Baran et al [2]. First, we hierarchically partition the point cloud using an octree and divide the cells larger than a specified tolerance. We set the maximum octree depth to 4, resulting in the smallest cube size of $L/2^4$, where L is the edge length of the cube enclosing the entire mesh model. In our work, we set the tolerance to $1/125$ of the edge length of this smallest cube. Next, we utilize the signed distance field (SDF) to extract only the cells located inside the model. We consider the center of these cells to be the interior point cloud.

Furthermore, we apply a sphere packing method to reduce the density of the interior point cloud $P_i \in \mathbb{R}^3$, which is defined as $P = \{p_1, \dots, p_n\}$. The point cloud consists of mesh vertices is denoted as M. The sphere packing process starts by selecting the interior point that is farthest from the mesh surface. Specifically, for each candidate center p_i, we compute its nearest neighbor m_i in the mesh surface M. The initial sphere radius is then determined as $r_i = d(p_i, m_i)$. Next, we sort all candidate spheres (p_i, r_i) in descending order of r_i and construct a sorted list $L = \{(p_1, r_1), (p_2, r_2), \dots, (p_m, r_m)\}$ such that $r_1 > r_2 > \cdots > r_m$. The sphere packing process starts from the largest sphere and iteratively selects subsequent spheres from L. A sphere (p_i, r_i) is added to the final sphere set S only if the center point p_i does not lie inside any of the spheres in S:

$$(p_i, r_i) \in L, \quad \forall (p_j, r_j) \in S, \quad d(p_i, p_j) \geq r_j \quad (1)$$

This ensures that larger spheres are placed first, mainly around the skeletal regions, while smaller ones fill the remaining spaces. We use the sphere centers from the final sphere set S as the interior point cloud for the subsequent orientation estimation.

Table 1: Comparison of orientation estimation using sampled mesh models and interior point clouds.

	Acc ↑	r ↑	MSE ↓
Mesh	0.6756	0.5698	63.0720
Inpcd	**0.7244**	**0.7362**	**43.8236**

3.2 Orientation vector estimation

To estimate the facing direction of 3D human scan models, our method leverages interior point clouds and their 2D projections. Specifically, we divide the point cloud into ten horizontal slices orthogonal to the vertical axis and project each slice onto 2D binary images. Horizontal cross-sections viewed from above can retain useful information about the body's facing direction. For example, the shape and orientation of the feet and the head are visible in these slices. In contrast, silhouettes obtained from front or side views tend to lack directional cues. Furthermore, using multiple slices instead of one slice allows the model to capture geometric features from different heights of the body. The details of our method are described below.

We estimate the 3D human model's orientation using ViT. First, we divide the interior point cloud $P \in \mathbb{R}^{N\times 3}$ into ten horizontal slices. Each slice is denoted as i. Each slice is then projected onto a 2D plane, represented as a binary image $I^i \in \{0,1\}^{H\times W}$, where $H = W = 64$. Next, we divided the image into $N = H \times W/S^2$ non-overlapping 2D patches, where S is the patch size. Based on empirical evaluations, we found that setting $S = 32$ $(i.e., N = 4)$ yielded the best performance in our orientation estimation task. Each patch is flattened and projected into a D-dimensional token vector using a linear embedding:

$$z_0^i = [x_{class}; x_s^1 E; x_s^2 E; \dots; x_s^N E] + E_{pos} \quad (2)$$

where $x_s^j \in \mathbb{R}^{S^2}$ is the j-th flattened patch, $E \in \mathbb{R}^{S^2\times D}$ is the embedding matrix, and $E_{pos} \in \mathbb{R}^{N\times D}$ is the positional encoding. These tokens are then processed through a standard Transformer encoder composed of multiheaded self-attention (MSA) and MLP blocks:

$$z_l' = MSA\big(LN(z_{l-1})\big) + z_{l-1}, \qquad z_l = MLP\big(LN(z_l')\big) + z_l' \quad (3)$$

where $l = 1 \dots L$ denotes the Transformer layer and LN is Layer-norm. Finally, we predict an orientation vector $d_i = LN(z_L^0) = [x_i, y_i, 0] \in \mathbb{R}^3$ for each binary slice i. These per-slice orientation predictions are subsequently aggregated through a Multi-Layer Perceptron (MLP) to produce a single global orientation vector:

$$D = MLP([d_1; d_2; \dots; d_{10}]) \in \mathbb{R}^3 \quad (4)$$

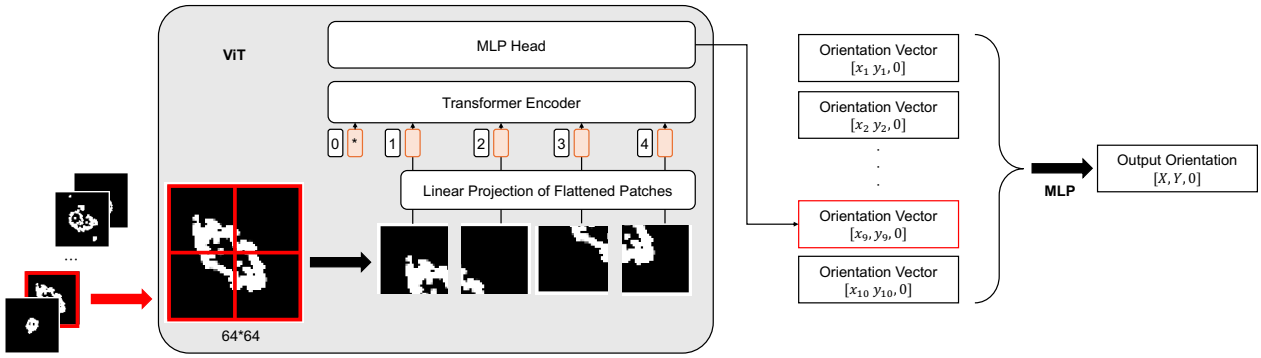

Figure 2: Orientation Vector Estimation

where $[;]$ denotes vector concatenation and d_i represents the predicted orientation vector from the i-th slice. This MLP consists of two fully connected layers, a ReLU activation and a dropout layer with a probability of 0.2. The final vector $D \in \mathbb{R}^3$ represents the estimated global orientation of the input model. The overview of our method is shown in Figure 2.

Only the final orientation vector D is utilized during training phase for loss computation. We adopt $\mathcal{L}^2$ loss function:

$$L^2 = \frac{1}{M}\sum_{i=1}^{M} \|D_i - G_i\|_2 \quad (5)$$

to measure the discrepancy between the predicted orientation D_i vector and the ground-truth G_i. Here, M is the number of input models and $\|\cdot\|_2$ denotes the L2 norm.

4. Experiments

We conducted all experiments independently on two different GPUs: an NVIDIA A100 and an NVIDIA GeForce RTX 4070. We implemented the model using PyTorch and trained it using the Adam optimizer with an initial learning rate of 1e-3. We applied a learning rate scheduler with a step size of 10 and a decay factor of $\gamma = 0.9$. We trained the model for 30 epochs with a batch size of 4.

4.1 Dataset

The human 3D model dataset is collected by a 3D human body scanner. The cameras of this scanner are set up in fifteen poles, each of which holds five cameras at different heights, arranged around the human. The scanner simultaneously captures 68 to 78 photos of the human from different directions. We utilize Reality Capture 1.4 to generate a 3D model from the captured pictures and manually rig the skeleton for the training data. We collected 91 models for training and 10 models for testing. These models include five different poses. The poses include A-pose, pose with. both hands raised, pose with one hand in front of the body in alternation, and A-pose with knees bent. We randomly rotated all models from 0 to 359 degrees 5 times for the orientation estimate dataset. Consequently, we used 455 models for training and 45 for testing during the orientation estimation phase.

We calculate the interior point cloud for all models following Section 3.1. The average number of points in an interior point cloud is 2225, while the simplified mesh model has 1384 points on average.

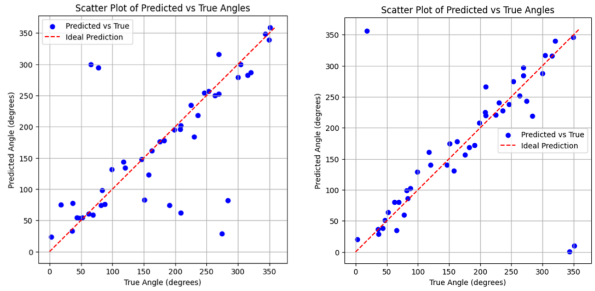

Figure 3: Scatter Plots of Predicted Versus Ground-truth Orientation Angles: (Left) Results using mesh surface point clouds. (Right) Results using interior point clouds.

4.2 Orientation vector estimation

As described in Section 3.2, we divided each interior point cloud into 10 horizontal segments orthogonal to the vertical axis. We projected each segment onto a 2D binary image of size 64×64 and further split the image into non-overlapping 2D patches of size 32×32 before feeding them into the Transformer.

To evaluate whether interior point clouds provide more informative features for orientation estimation than mesh surfaces, we conducted a comparative experiment using both mesh models and their corresponding interior point clouds. As summarized in Table 1, the results indicate that interior point clouds consistently yield superior orientation estimation performance.

We calculated accuracy by measuring whether the absolute error between the predicted and ground-truth orientation vectors was less than 45°, which was set as the tolerance threshold. In addition to accuracy (Acc), we report the Pearson correlation coefficient (r) and mean squared error (MSE) between the predicted and ground-truth values. The results demonstrate that the interior point cloud significantly improves both the accuracy and stability of orientation estimation.

Figure 3 shows the evaluation of orientation estimation on the test dataset using mesh surface point clouds (left) and interior point clouds (right). In the best case in the experiment using interior point clouds, accuracy exceeded 95.5%, and the orientation estimation error follows a distribution centered around 5.27 ± 21.72. On the other hand, while using simplified mesh surface point clouds, the best accuracy was 77.8%, and the orientation estimation error follows a distribution centered around -6.82 ± 53.72.

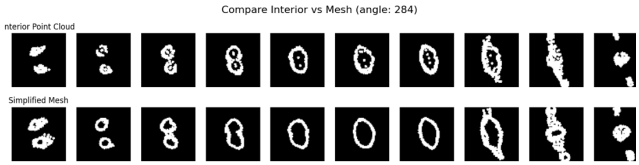

Figure 4: Comparison of 2D projections: (Top) Horizontal slices of the interior point cloud, highlighting plausible joint positions and subtle limb curvatures. (Bottom) Horizontal slices of the mesh surface model, resulting in an ambiguous silhouette with an elliptical torso.

Through experiments, we observed that when using mesh models, several samples failed to achieve accurate orientation estimation, particularly in the rotation ranges of 50–100° and 150–300°, as shown in Figure 3 (left). Notably, when experimenting on the mesh surface dataset, models rotated between 50–100° were never estimated correctly. Further investigation showed that these samples corresponded to human models with both arms raised, as illustrated in Figure 5. When projected onto a 2D plane, the binary images derived from mesh models—such as the example in Figure 4 (bottom)—often result in ambiguous silhouettes, with the torso region appearing as a simple ellipse which is difficult to identify the front and the back side of human body. In contrast, as shown in Figure 4 (top), the projections from interior point clouds represent plausible joint positions that introduce slight curvature around limb regions. These subtle geometric cues lead to more accurate orientation predictions when using interior point clouds.

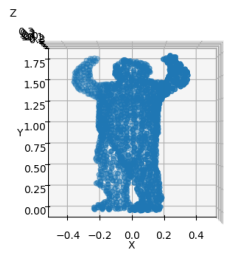

Figure 5: Example of a 3D mesh model with raised arms, resulted in incorrect orientation estimation.

5. Discussion

The experimental results in Section 4 confirm that transforming interior point clouds into binary images and employing a Vision Transformer (ViT) allows for reliable estimation of the orientation of 3D human models. By comparing the evaluation results of the interior point cloud and mesh model, we found that using the interior point cloud improved the accuracy and stability of orientation estimation. We attribute this improvement in accuracy to two main factors. First, interior point clouds are less affected by surface noise and missing parts in 3D scan data. Second, projections of mesh models often result in elliptical silhouettes where the distinction between the front and back sides becomes ambiguous.

On the other hand, our dataset has limitations. The current dataset includes only five human poses, which do not fully represent the wide variety of real-world human postures. Consequently, the model may produce unstable predictions when applied to uncommon or complex poses. Furthermore, the geometry of the mesh surface can vary significantly depending on the clothing worn by the scanned models. For instance, when the input model wears a coat or a long dress, the lower body region may appear as one column, obscuring the separation between the legs instead of two distinct legs. This ambiguity in shape can negatively impact the accuracy of orientation estimation. Expanding the training dataset to include more diverse postures and clothing is crucial.

In future work, we aim to apply this orientation estimation method to the automatic rigging of 3D human models without relying on a fixed forward direction. Because the orientation vector is derived directly from internal geometry, this approach can support rigging even for scanned models with arbitrary orientations.

6. Conclusion

Existing applications that utilize 3D human models, such as automatic rigging, animation, and pose analysis, often assume that the models are presented in a standardized pose and orientation. However, scanned human models can appear in arbitrary poses and facing directions in real-world settings. To address this gap, we proposed a method for estimating the facing direction of such models as a preprocessing step for downstream tasks such as automatic rigging.

In this study, we presented a novel method for estimating the orientation of 3D human scan data by utilizing interior point clouds and a Vision Transformer-based architecture. Through experiments, we demonstrated that our approach enables accurate and robust orientation prediction, outperforming methods based on mesh surface features. In future work, we aim to expand our dataset to include a broader range of poses and clothing and to apply our approach toward fully automated rigging of human models with arbitrary orientations and postures.

Acknowledgment

This work was conducted as a “collaborative study under the NIJL Project”.

References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, CoRR, abs/2010.11929, (2020).
[2] Charles Ruizhongtai Qi, Hao Su, Mo Kaichun, Leonidas J. Guibas, PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp.652-660, 2017.
[3] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, Justin M. Solomon, Dynamic Graph CNN for Learning on Point Clouds, ACM Transactions on Graphics, Volume: 38, Issue: 5, Article No.: 146, pp.1–12, 2019.
[4] Ma Xu, Qin Can, You Haoxuan, Ran Haoxi, Fu Yun, Rethinking network design and local geometry in point cloud: A simple residual MLP framework, arXiv preprint arXiv:2202.07123, 2022.
[5] Ilya Baran, Jovan Popovic, Automatic rigging and animation of 3D characters, ACM Transactions on Graphics, Volume: 26, Issue: 3, pp.72–es, 2007.
[6] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chiris Landreth, Karan Singhacm, RigNet: Neural Rigging for Articulated Characters, Transactions on Graphics, Volume: 39, Issue: 4, Article No.: 58, pp.58:1-58:14, 2020.
[7] Ma, Jing, Zhang, Dongliang, Tarig: TARig: Adaptive template-aware neural rigging for humanoid characters, Computers & graphics, Volume: 114, Number: Aug., pp.158-167, 2023