Rotation Invariant Automatic Rigging for 3D Human Scan Data

Yiqing Li* and Satoru Fujita†
* Hosei University, Japan
E-mail: yiqing.li.6p@stu.hosei.ac.jp Tel/Fax: +81-42-387-4545
† Hosei University, Japan
E-mail: fujita_s@hosei.ac.jp Tel/Fax: +81-42-387-4545

***Abstract*—We propose a three-step rotation-invariant automatic rigging method specifically designed for 3D human scan data. Our method first computes an interior point cloud from the input mesh model. It then estimates the point cloud's facing direction to determine the human model's orientation. Using this estimated direction, we reorient the point cloud to face forward. Finally, human body parts are segmented on the aligned point cloud. We also introduce an attention mechanism to improve the joints estimation. Based on the segmentation and the predicted attention score, we estimate the fundamental 21 joints essential for representing human motion, including key areas such as the head and neck. Since the estimated joints are semantically labeled, the joint identities and their skeletal connectivity can be readily determined. This skeleton can be directly utilized in applications for downstream tasks such as animation generation and motion retargeting. We used Vision Transformer to estimate the face direction and PointMLP-based architecture to predict the skeleton of the input 3D scan data. We trained the architecture on our original 3D scan dataset, including five common poses. Our method outperforms prior work in terms of both the number and accuracy of the estimated joints, and it is robust to variations in the input model's orientation.**

I. Introduction

3D human models are used in various fields, including animation, virtual fitting, and gaming. Embedding a skeletal structure into a mesh is necessary to enable movement in these models. Traditionally, animators conduct this process manually. However, manual rigging is time-consuming due to the large amount of redundant work involved. Automatic rigging techniques can significantly accelerate this process and reduce manual labor. With the recent advancements in neural networks, several learning-based methods for automatic rigging of animated characters have emerged. These approaches typically rely on clean, well-posed mesh models created or refined by professional animators. As a result, they often struggle to handle raw 3D scan data, which tends to have complex surface geometries and lacks post-processing or cleanup. Moreover, when using raw scans, variability in body orientation, pose, and clothing further increases the difficulty of automatic rigging. Such real-world variability makes it challenging to design systems that can robustly infer anatomically plausible skeletons and skin weights from the raw input.

To address these challenges, we propose a deep-learning based method that allows automatic rigging regardless of the model's rotation or posture. Our process consists of three steps. First, inspired by 'Pinocchio [1]', we extract the interior point cloud from the mesh. We found that the interior point cloud better understands the rotation and segmentation of 3D models. Next, we split the 3D models horizontally, and each resulting point cloud is projected onto a 2D binary image. These images are then analyzed using Vision Transformer(ViT)[2] to estimate the orientation vectors. Finally, we reorient the interior point cloud using the estimated orientation vectors. We then apply PointMLP [3] to perform semantic segmentation into 21 predefined body part regions and to predict an attention score for each point, indicating the likelihood that it is spatially close to a specific joint location. The overview of our method is illustrated in Fig. 1. During the evaluation, we showed that our method outperforms prior work regarding the number and accuracy of the estimated joints, and it is robust to variations in the input model's orientation and pose.

In summary, we contribute a rotation-invariant automatic rigging method designed explicitly for raw 3D scan data. Our method is robust to the input model's orientation. By leveraging semantic segmentation on an interior point cloud representation, our approach enables accurate estimation of 21 anatomically meaningful joints. Experimental results demonstrate that our method outperforms existing approaches such as RigNet [4] and TARig [5] in both the number and positional accuracy of the predicted joints.

II. Related Work

A. Orientation estimation

Existing studies on 3D human model analysis often assume that the input model is already aligned in a fixed orientation. However, the facing direction is unknown in practical scenarios such as raw 3D scans. Li and Fujita [6] introduced a direction estimation approach that handles arbitrary input orientations without manual alignment. This work is inspired by the approximate medial surface proposed by [1] and begins by computing an interior point cloud from the input mesh model. Then, Li slices the point cloud horizontally and projects each slice onto a 2D binary image of 64×64 pixels. These binary images serve as input to a Vision Transformer. Each image is divided into four patches, which are flattened and linearly projected into D-dimensional token vectors. As ViT [2], this method uses a standard Transformer encoder composed of multi-head self-attention (MSA) layers and feedforward

979-8-3315-7206-8/25/$31.00

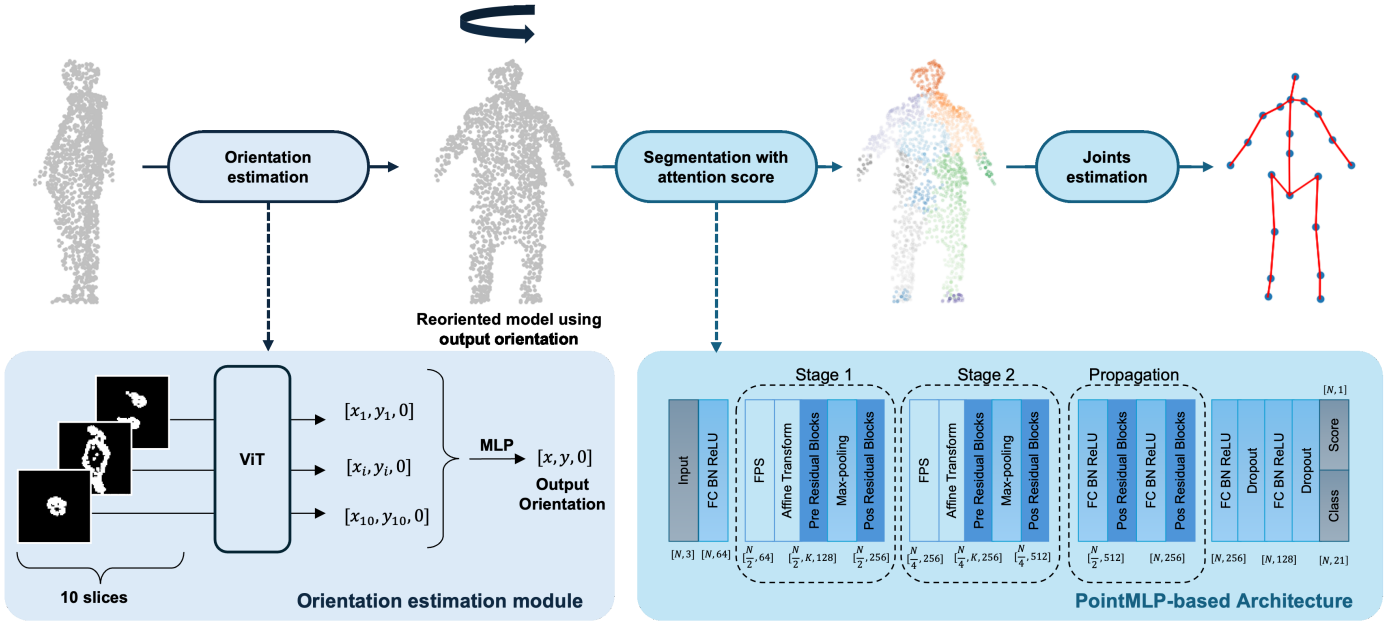

Fig. 1. Overview of our approach. We begin with an interior point cloud extracted from a 3D human model that may be arbitrarily rotated. The first step is to estimate the face orientation of the point cloud using a Vision Transformer. The input point cloud is reoriented using this face orientation. Next, we predict the segmentation and attention score for each point and calculate the cluster centers of each segment part, which are considered the joints of the input model.

Multi-Layer Perceptron (MLP) blocks to process these token embeddings. Finally, the per-slice orientation predictions are aggregated using an additional MLP, which produces a single global orientation vector representing the estimated facing direction of the human model.

B. 3D point cloud segmentation

Learning directly from 3D point clouds has recently become a prominent direction in geometric deep learning. Unlike voxel-based methods, which suffer from high memory consumption and resolution loss, point-based methods maintain high geometric accuracy without discretizing the input shape and are computationally efficient.

Early methods like PointNet [7] pioneered a direct point cloud processing method using symmetric functions (e.g., max pooling) to extract global features from unordered point sets. PointNet++ [8] extended this idea by introducing hierarchical feature learning with local neighborhoods, significantly improving performance in segmentation tasks. Subsequent works aimed to enhance the modeling of local structures. For example, DGCNN [9] introduced EdgeConv, which dynamically builds a local graph around each point and captures relationships through edge-based feature aggregation. EdgeConv proved highly effective for tasks like part segmentation and scene parsing. More recently, PointMLP [3] proposed a lightweight yet practical architecture that leverages spatially structured local learning through residual MLP blocks. This architecture resolves the challenges posed by convolutional and attention-based models, providing a straightforward and effective solution for 3D point cloud analysis. In particular, their method achieved state-of-the-art accuracy on the ModelNet40 benchmark and the real-world ScanObjectNN dataset. Inspired by the simplicity and powerful performance of PointMLP, we employ it in our method to segment the interior point cloud into body parts.

C. Automatic rigging

Since the introduction of Pinocchio [1] by Baran and Popović in 2007, automatic rigging has been an active area of research. Pinocchio constructs an approximate skeleton by fitting spheres inside the character mesh and connecting their centers to form a medial axis, which is then refined into a plausible skeleton structure. More recently, neural network-based methods such as RigNet [4] have emerged, which predict joint locations and skeletal structures directly from 3D mesh models using supervised learning. Ma et al. [5] pointed out that RigNet often produces anatomically incorrect joints, leading to invalid skeletons that require manual correction. To overcome this, they proposed a template-aware, bone-flow-guided method for more accurate and stable rigging. These methods leverage information about the mesh surface, bone topology, and skinning weights to automate the rigging process. However, these approaches assume that the input character models are clean, manually refined, and oriented in a canonical pose (e.g., T-pose or A-pose). As a result, they struggle with raw 3D scans due to their complex geometry and require a fixed input orientation, which limits their applicability to varied human poses and directions. We propose a method invariant to the input model's orientation and pose to address these limitations. By estimating the facing direction of the scanned human mesh and aligning it before segmentation and rigging, our method enables reliable skeleton prediction, even

under challenging real-world scan conditions.

III. Rotation Invariant Automatic Rigging

A. Orientation estimation

Since prior methods assume a fixed orientation and cannot be directly applied to arbitrarily rotated 3D models, we first estimate the facing direction of the input model. We reorient the model according to the estimated orientation before further processing. To achieve the best performance of orientation estimation, we follow the work [6] and use the interior point clouds as the input to our pipeline. Interior point clouds contain points situated slightly beneath the mesh surface and regions likely to correspond to underlying skeletal structures. Compared to mesh-based methods, this representation is more robust to noise, surface irregularities, and missing data. We generate the interior point cloud by constructing an octree-based spatial partitioning of the 3D model and selecting cells that lie strictly inside the mesh using a Signed Distance Field (SDF). The centers of these interior cells are then treated as candidate points. A sphere packing algorithm is applied to refine the representation further to reduce point density while preserving geometric consistency. Spheres are placed starting from regions farthest from the mesh surface, prioritizing larger spheres to cover core skeletal areas and filling in remaining gaps with smaller spheres. The centers of the selected spheres are the input to our orientation estimation pipeline.

We estimate the orientation of a 3D human model through a Vision Transformer (ViT)-based framework that operates on interior point clouds as shown in Fig. 1. We partition the interior point cloud into horizontal slices, and project each slice onto a 2D binary image. These images are divided into non-overlapping patches, embedded into token vectors, and processed through a standard Transformer encoder. Our model predicts one local orientation vector for each slice, and all the local orientation vectors are aggregated via an MLP to produce a global orientation prediction. We optimize the model by minimizing the L2 loss between the final predicted orientation vector and the ground-truth direction during the training phase.

B. PointMLP-based architecture

Previous methods, such as the approach proposed by [4], focused on extracting skeletal structures from 3D mesh models but did not explicitly estimate semantically meaningful joints (e.g., "left elbow" or "right knee"), which are crucial for animation and high-level motion understanding. In contrast, our method performs semantic segmentation on the input point cloud by assigning each point to a specific body part. We then infer joint positions as the representative centers of these segmented regions. Simply computing the centroid of all segmented points can lead to inaccurate joint localization due to spatial bias. Therefore, we also introduce an attention mechanism to improve robustness.

We adopt a PointMLP-based architecture to extract per-point features and to predict semantic part labels and joint attention scores jointly. Given a set of points $P = \{p_i \mid i = 1, \ldots, n\} \in \mathbb{R}^{n\times 3}$, where n indicates the number of points in a Cartesian space. As illustrated in Fig. 1, the architecture is composed of two main components: (1) a hierarchical feature extraction module that progressively resamples the point cloud P while learning high-level representations for each point p_i and (2) a feature propagation module that interpolates and refines features back to the original point resolution.

The hierarchical feature extraction module consists of two successive stages, each of which downsamples N_s points using the Farthest Point Sampling (FPS) algorithm. Here, s denotes the stage index. For each sampled point, we select $k = 12$ neighbors from those whose distance is smaller than the sum of their absolute SDF values. By incorporating SDF values, we prevent the connection of points that are spatially close yet separated by the geometry, such as points between the arm and torso. Then, we use an affine transformation module to process the sampled point and its neighbors. This affine transformation module can be described as follows:

$$\begin{aligned} \{f_{i,j}\} &= \alpha \otimes \frac{\{f_{i,j}\} - f_i}{\sigma + \epsilon} + \beta, \\ where\ \sigma &= \sqrt{\frac{1}{k \times n \times d} \sum_{i=1}^{n} \sum_{j=1}^{k} (f_{i,j} - f_i)^2}. \end{aligned} \tag{1}$$

Here, $\{f_{i,j}\}_{j=1,\ldots,k} \in \mathbb{R}^{k\times d}$ is the grouped features of k neighbors around f_i. The dimension of each neighbor feature $f_{i,j}$ is d. α and β are learnable parameters. $\otimes$ denotes Hadamard product. To ensure numerical stability and avoid division by zero, we add a small constant $\epsilon = 1e^{-5}$ to the denominator.

We use residual blocks to extract features from these transformed points before and after a max-pooling operation, respectively. We denote the pre-aggregation residual blocks as $\mathcal{F}_{pre}(\cdot)$ and the pos-aggregation residual blocks as $\mathcal{F}_{pos}(\cdot)$. Note that both $\mathcal{F}_{pre}(\cdot)$ and $\mathcal{F}_{pos}(\cdot)$ consist of three residual blocks. Each residual block consists of two fully connected layers, each followed by a batch normalization, with a ReLU activation applied between them. The feature extraction process in one stage can be represented as below:

$$g_i = \mathcal{F}_{pos}(\mathcal{A}(\mathcal{F}_{pre}(f_{i,j}), |j = 1, \ldots, k))), \tag{2}$$

where the aggregation function $\mathcal{A}(\cdot)$ is max-pooling.

The feature propagation module is designed to restore higher resolution point-wise features from the downsampled point cloud representations. Given a set of original points N and their corresponding down-sampled point clouds N_s at each stage, the module first applies inverse distance weighted interpolation. For each original point in N, we aggregate the features of the three nearest sampled points from N_s using weights inversely proportional to their distances. Then, we concatenate the original point features with the interpolated features. This concatenated representation is then passed through a fully connected layer, followed by batch normalization and a ReLU activation, to fuse the features. Finally, the fused features are refined using the pos-aggregation residual blocks, enhancing the quality of the restored features. The feature propagation process is formulated as:

$$y_i = \text{Propagation}(g_{i1}, g_{i2}, g_{i3}), \tag{3}$$

where y_i denotes the interpolated feature at the i-th point after propagation. The inputs g_{i1}, g_{i2}, g_{i3} represent the features of the three nearest neighbors of point i, selected from the sampled point set N_s. We apply this feature propagation module twice to progressively recover the original point-wise resolution, corresponding to the two stages of downsampling applied during hierarchical feature extraction. Before the final prediction, we apply two fully connected layers with batch normalization, ReLU activation, and dropout for regularization. We utilize two separate linear layers to predict the segmentation label and attention score for each point. Full architecture details are provided in Fig. 1.

We adopt multi-margin loss for the segmentation label in our framework, which is defined as:

$$\mathcal{L}_{seg}(x, y) = \frac{1}{l} \sum_{i}^{l} \max(0, 1 - x_y + x_i), \tag{4}$$

where $x \in \mathbb{R}^l$ denotes the predicted scores for all candidate joints, and $y \in \{1, \ldots, l\}$ represents the ground-truth joint index. In this study, $l = 21$, corresponding to the number of joints. The term x_y refers to the predicted score for the ground-truth joint, while x_i corresponds to the predicted score for the i-th joint. To supervise the attention score outputs, we adopt the Mean Squared Error (MSE) loss. Given a ground-truth score $y \in \mathbb{R}$ and the predicted score $x \in \mathbb{R}$ the MSE loss is defined as:

$$\mathcal{L}_{MSE}(x, y) = \gamma(y - x)^2, \tag{5}$$

where γ denote the weight of MSE loss. When the MSE loss is overly dominant, it can lead to segmentation failures. We observe that reducing the weight γ of MSE loss to 0.015 improves its overall performance.

C. Joints estimation

After segmentation, we apply DBSCAN[10] to each set of points belonging to the same predicted label l. This removes outliers, as isolated noisy points fail to form dense clusters. Then, we compute a weighted centroid $c_l \in \mathbb{R}^3$ of the remaining points for each label. Let N_l be the set of points assigned to label l. The centroid is defined as:

$$w_i = |SDF_i| \cdot s_i, c_l = \frac{\sum_{i \in N_l} w_i \cdot x_i}{\sum_{i \in N_l} w_i + \epsilon}, \tag{6}$$

where w_i is the product of the absolute signed distance field (SDF) and a learnable attention score s_i for each point p_i. The learnable attention score s_i is based on the inverse distance to the nearest ground truth joint location which is defined as:

$$s_i = \frac{1}{\|p_i - j_{nearest}\| + \epsilon}. \tag{7}$$

To avoid division by zero, we add a small constant $\epsilon = 1e-8$ to the denominator. These scores reflect the likelihood of each point being close to the true joint. Since our interior point cloud contains both points near the skeleton and those close to the surface, assigning larger weights to points that are both closer to the joints (via the attention score) and farther from the surface (via the absolute SDF) significantly enhances the accuracy of joint estimation.

IV. Experiments

A. Datasets and experimental setup

We collected a dataset of 3D human models using a full-body scanner equipped with cameras arranged on 15 poles. Each scan produces 68–78 images, which are reconstructed into 3D models using RealityCapture 1.4. We manually rigged all models with 21 joints. The dataset comprises 91 training and 10 testing models in five poses, including A-pose and various arm and leg configurations. For orientation estimation, each model was rotated five times from 0° to 359°, resulting in 455 training and 45 testing samples.

Our method was implemented and evaluated on an NVIDIA A100. We implemented the model using PyTorch. For the orientation estimation task, we used the Adam optimizer with a learning rate of $1e-3$ and applied a learning rate scheduler with a step size of 10 and a decay factor of $\gamma = 0.9$. We trained the orientation estimation model for 30 epochs with a batch size of 4. For the segmentation and attention score estimation, we used the Adam optimizer with a learning rate of $1e-4$. We set the batch size to 1 and trained the model for 30 epochs. The weight for the MSE loss was set to 0.015. We applied DBSCAN to extract the primary cluster from each segmented region. We set the threshold for attention scores to 0.6 and maximum neighborhood distance $\epsilon = 0.25$ for the cluster.

To evaluate the accuracy of joint estimation, we employ four metrics: CD-J2J, IoU, precision, and recall. CD-J2J (Chamfer Distance between Joints) measures the average Euclidean distance from each predicted joint to its nearest ground-truth joint. IoU is computed by first solving the optimal assignment between predicted and ground-truth joints using the Hungarian algorithm and then calculating the proportion of matched predicted joints relative to the total number of ground-truth joints. Precision indicates the ratio of predicted joints whose distances to their nearest ground-truth joints are within a threshold of 5 cm. Recall, on the other hand, represents the proportion of ground-truth joints that are matched by predicted joints within the same threshold.

B. Rotation Robustness Evaluation

We first compare our method to two existing rigging methods. As shown in Table I, our method achieves the lowest CD-J2J value and the highest precision among all compared approaches, indicating superior accuracy in both the spatial proximity of predicted joints and the reliability of joint localization. As illustrated in Fig. 2, our method accurately estimates joint positions of limb joints, particularly in the hands and toes. However, the positional variability of the arms leads to segmentation failures, resulting in mislabeled points for the elbows. These outliers significantly bias the centroid computation, leading to inaccurate joint localization as shown in Fig. 2(c). RigNet demonstrates relatively low CD-J2J and

TABLE I
COMPARISON OF RIGGING PERFORMANCE ON RANDOMLY ROTATED 3D HUMAN MODELS.

Method	CD-J2J(%) ↓	Precision(%) ↑	Recall(%) ↑
Ours	**5.0**	**57.7**	57.5
RigNet	6.7	16.2	**75.1**
TARig	7.1	32.9	32.9

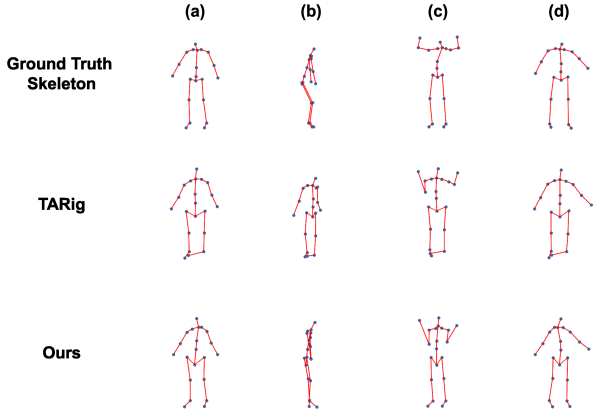

Fig. 2. Skeleton comparison between ground-truth skeletons, TARig results, and ours for four poses (a–d): A-pose, bent knees, raised arms, and left arm forward.

an exceptionally high recall but suffers from low precision. This suggests that while RigNet predicts a large number of joint candidates, many of these predictions are inaccurate. TARig consistently outputs 21 joints and exhibits a balanced but low performance across all metrics due to its lack of orientation correction. As the pose (b) shown in Fig. 2, because the method assumes a fixed frontal direction, it often fails to adapt to models that are facing sideways, leading to significant misalignment in joint placement. Fig. 2 also indicates that TARig consistently struggles to accurately estimate toe joint positions across all poses.

C. Ablation study

To evaluate the contribution of each component within our framework, we conduct a comprehensive ablation study, systematically isolating and analyzing the impact of individual modules. We compare the complete model (Ours) with four variants: without score (w/o Score), without SDF (w/o SDF), without both (w/o Score and SDF), and without DBSCAN (w/o DBSCAN). Additionally, we investigate whether incorporating our orientation estimation module into an existing method (TARig) enhances its performance. We also evaluate the impact of removing the orientation estimation from our pipeline to assess its overall contribution.

As shown in Table II, removing either the attention score or the SDF individually does not cause a substantial degradation in CD-J2J, precision, or recall. In contrast, removing both simultaneously results in a notable decline, with CD-J2J increasing from 5.0% to 5.9% and precision/recall decreasing from 57.7% to 42.2% and from 57.5% to 42.1%, respectively.

TABLE II
ABLATION STUDY EVALUATING THE EFFECTIVENESS OF ATTENTION SCORE AND SIGNED DISTANCE FIELD.

Method	CD-J2J(%) ↓	Precision(%) ↑	Recall(%) ↑
Ours	**5.0**	**57.7**	**57.5**
w/o Score	5.0	57.4	57.2
w/o SDF	5.0	57.5	57.4
w/o Score and SDF	5.9	42.2	42.1
w/o DBSCAN	5.0	57.4	57.3

TABLE III
ABLATION STUDY EVALUATING THE EFFECTIVENESS OF ORIENTATION ESTIMATION.

Method	CD-J2J(%) ↓	Precision(%) ↑	Recall(%) ↑
Ours	**5.0**	**57.7**	**57.5**
Ours w/o Ori	7.6	33.2	33.2
TARig	7.1	32.9	32.9
TARig w/ Ori	**5.2**	**54.4**	**54.4**

This result indicates that each component provides only limited improvement when used individually, but their combined use is necessary to maintain stable performance. Additionally, using DBSCAN to remove outliers also slightly improves the precision and recall of joint localization. Overall, although the proposed framework does not drastically alter the estimation accuracy, these design choices collectively contribute to improving robustness and accuracy.

In the second experiment, we found that removing orientation estimation (Ours w/o Ori) increased the CD-J2J from 5.0% to 7.6% and reduced precision from 57.7% to 33.2%, as shown in Table III. This result highlights the importance of aligning the input point cloud to a canonical orientation before rigging. Furthermore, when we apply the orientation estimation to TARig (TARig w/ Ori), its performance improves significantly. CD-J2J is reduced from 7.1% to 5.2%, and precision and recall both increase from 32.9% to 54.4%. These results not only validate the effectiveness of our orientation estimation but also demonstrate its transferability to improve existing rigging frameworks that do not account for arbitrary model rotations.

To further analyze the effectiveness of our method across individual joints, we conducted a joint-wise evaluation using our complete model. We categorized joints based on their precision scores in Table IV: joints with precision below 50% are shown on the left, while those with precision above 50% are listed on the right for comparative analysis. Notably, the ankles, knees, toes, and clavicles achieve particularly high precision and recall values, suggesting that these joints are consistently segmented and accurately localized. Their high performance can be attributed to the fact that they often appear in stable and distinct positions.

In contrast, the head and hip center show the lowest precision. Central joints, such as the chest, spine, and hips, fall within the range of 39.1% to 57.3%. These joints deviate from manual annotations but are near the centers of the segmentation clusters, reflecting a tendency of the method to localize joints at cluster centroids. Hands, elbows, and shoulders exhibit moderate precision, ranging from 43.3% to 60.7%, due to the

TABLE IV
JOINT-WISE PRECISION ANALYSIS OF OUR FULL METHOD. WE HIGHLIGHT JOINTS WITH PARTICULARLY LOW PRECISION (<50%), AND THOSE WITH HIGH PRECISION (>50%).

	Hips Center	Head	Neck	Spine	Elbows	Chest	Hands	Toes	Clavicles	Shoulders	Hips	Knees	Ankles
CD-J2J(%)	8.9	6.6	5.9	5.8	11.1	5.1	5.9	4.5	4.2	5.3	5.6	3.8	2.7
Precision(%)	9.8	27.1	38.2	39.1	43.3	57.3	57.4	65.5	73.7	60.7	50.7	74.2	92.9

TABLE V
STEP-WISE RUNTIME COMPARISON BETWEEN TARIG AND OUR METHOD. TIMES ARE REPORTED IN SECONDS.

Method	Data preparation	Rotation	Joints estimation	Total
Ours	1022.37	0.44	0.33	0.77
TARig	10.94	-	< 0.01	< 0.01

variety of upper body poses.

D. Performance Analysis

As shown in Table V, the mesh models used for TARig experiments have an average of 1,598 points per model, whereas the interior point clouds used in our experiments contain approximately 2,200 points. We conducted all experiments on the same A100 GPU. Although the total inference time of our method is roughly 100 times higher than TARig, it is around 0.7 seconds per model, which remains reasonable. The primary computational bottleneck arises from calculating the interior point cloud. Reducing this step will be important for practical deployment in real-world applications.

V. DISCUSSION

Our method achieved the highest overall accuracy among the compared approaches, demonstrating both geometric precision and robust joint estimation. The ablation study revealed that combining orientation estimation and attention score significantly contributes to performance. Furthermore, our joint-wise analysis reveals that the method performs exceptionally well on legs. At the same time, accuracy decreases for joints located in the elbows due to the pose variability in our dataset. In future work, we aim to develop a more generalizable approach that can handle a broader range of poses and reduce the time required for interior point cloud computation, thereby increasing the robustness and applicability of our method in real-world scenarios.

VI. CONCLUSION

In this paper, we propose a rotation-invariant automatic rigging method that accurately estimates joint positions regardless of model orientation. Given a 3D human scan in an arbitrary pose and direction, our approach computes the interior point cloud and predicts the facing direction using a Vision Transformer. The model is then reoriented to a canonical direction. We use a PointMLP-based architecture to segment the interior points and predict attention scores. Finally, we compute the cluster centers as the final joint positions.

Experiments demonstrate the effectiveness of our method. Ablation studies confirm that both orientation estimation and attention scoring are essential. Future work will aim to extend our method to support a wider variety of poses, reduce the time required for interior point cloud computation, and validate the approach on additional existing datasets, thereby enhancing its robustness and practical utility.

ACKNOWLEDGMENT

This work was conducted as a "collaborative study under the NIJL Project <Research No.K452052428>."

REFERENCES

[1] I. Baran and J. Popović, "Automatic rigging and animation of 3d characters," *ACM Trans. Graph.*, vol. 26, no. 3, 72–es, Jul. 2007.

[2] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[3] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu, "Rethinking network design and local geometry in point cloud: A simple residual MLP framework," in *International Conference on Learning Representations*, 2022.

[4] Z. Xu, Y. Zhou, E. Kalogerakis, C. Landreth, and K. Singh, "Rignet: Neural rigging for articulated characters," *ACM Trans. on Graphics*, vol. 39, 2020.

[5] J. Ma and D. Zhang, "Tarig: Adaptive template-aware neural rigging for humanoid characters," *Comput. Graph.*, vol. 114, no. C, pp. 158–167, Aug. 2023.

[6] Y. Li and S. Fujita, "Orientation estimation of 3d human scans using interior point clouds and vision transformer," unpublished.

[7] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 77–85.

[8] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Curran Associates Inc., 2017, pp. 5105–5114.

[9] 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 Trans. Graph.*, vol. 38, no. 5, 2019.

[10] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996.