A. Overview

In the supplementary materials, we provide detailed information on the generation of the two datasets used for training HMFlow and HBSeg, along with corresponding benchmarks and evaluation metrics to facilitate their use and assessment in future research. Additionally, we include the implementation details of the Tokenizer in both Self-learning and Fine-tuning stages. For more comprehensive and extensive comparisons, we have expanded our comparison experiments on HuCenLife [23] to include methods tailored for modeling dynamic point cloud videos, to demonstrate the superiority of our method in capturing human motion representations. Finally, we also provide additional details regarding the size of the UniPVU-Human model and comparisons with others.

B. Human Motion Flow (HMFlow)

B.1. Implementation Details

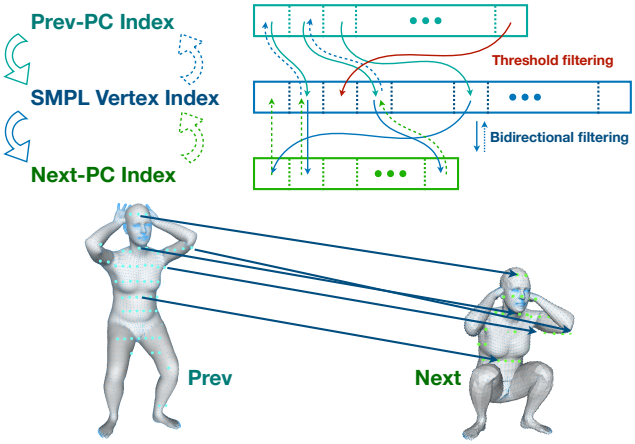

Figure 1. The pipeline of generating the flow from the previous point cloud to the next point cloud. We associate each synthetic LiDAR point to its nearest SMPL vertex, to establish the correspondence between synthetic LiDAR points across different frames by using SMPL vertices indices as medium, so that we can obtain point-wise motion flow.

Due to rotation or occlusion, point clouds may flow in and out between consecutive frames, resulting in a lack of temporal correspondence. But for the SMPL [11] mesh, each mesh vertex can be matched between consecutive frames using vertex index. Therefore, we make a large-scale synthetic dataset, LiDARFlow-Human, by scanning the SMPL mesh surfaces of consecutive frames using a simulated LiDAR to generate simulated LiDAR point clouds (As shown in Figure. 1). Each simulated LiDAR point is matched with its nearest SMPL vertex. Consequently, we use SMPL vertices as a medium to match simulated LiDAR points between frames. By subtracting coordinates, we obtain the point-wise motion flow. Moreover, we set a threshold to filter the distance and build the bidirectional connections to ensure the accuracy of the matching. Specifically, when the nearest distance from vertex to point is smaller than the defined threshold D, we think them has the unidirectional connection and we select the unidirectional connection C_{p2n} from previous point cloud to next point clouds, meanwhile we select the unidirectional connection C_{n2p} from next point clouds to previous point clouds. The bidirectional filter are used to delete the unidirectional connection without coincidence,

$$Flow_{p2n} = C_{p2n} \cap C_{n2p}.$$

Table 1. Human Motion Flow (HMFlow) Result on LiDARFlow-Human.

	EPE↓	acc_strict↑	acc_relax↑	outlier↓
FLOT [14]	0.14	83.67	95.77	0.78

B.2. Dataset and Evaluation Metrics

We will contribute LiDARFlow-Human, used for training Human Motion Flow Estimator (HMFlow), to the community with corresponding benchmarks. As shown in Table. 1, we adopt the evaluation metrics used in [6, 9, 14, 22]:

- EPE: End Point Error (meters).

$$EPE = \frac{\sum_{i=1}^{N} \left\| (\vec{f_{predict}})_i - (\vec{f_{gt}})_i) \right\|_2}{N},$$

  where $(\vec{f_{predict}})_i$ and $(\vec{f_{gt}})_i$ are point-wise predicted motion flow and ground truth motion flow, respectively.
- acc_strict: percentage of points such that $EPE_i < 0.05$ or $EPE_i / \|\vec{f_i}\|_2 < 0.05$.
- acc_relax: percentage of points such that $EPE_i < 0.1$ or $EPE_i / \|\vec{f_i}\|_2 < 0.1$.
- outlier: percentage of points such that $EPE_i > 0.3$ or $EPE_i / \|\vec{f_i}\|_2 > 0.1$.

C. Human Body Segmentation (HBSeg)

C.1. Implementation Details

To address the absence of 3D human body part segmentation datasets based on LiDAR point clouds, we create a synthetic dataset of 1 million LiDAR human point cloud instances, named LiDARPart-Human, which uses the AMASS dataset for 3D human meshes and simulates LiDAR scans from various perspectives and distances (Figure. 2). These scans incorporate random occlusions and noise to reduce the gap between synthetic and real data. The SMPL mesh vertices, known for their ordered and regular structure, provide 24 human body part labels, but due to the sparsity of LiDAR point clouds, we simplify these to

Table 2. Human Body Segmentation (HBSeg) Results on LiDARPart-Human.

	head	left-arm	right-arm	up-body	low-body	upleft-leg	upright-leg	lowleft-leg	lowright-leg	mIoU↑
PointNet [15]	88.2	51.2	46.6	52.1	62.6	45.8	36.2	67.4	60.2	56.7
PointNet++ [16]	88.6	69.5	69.9	65.4	82.2	82.7	82.5	89.1	89.4	79.9
PointMLP [12]	92.0	76.1	75.2	76.7	88.0	86.3	85.8	92.8	92.3	85.0
PointNeXt [17]	95.1	82.7	81.9	83.1	91.9	91.2	90.8	96.1	96.0	**89.9**

Table 3. Supplementary Comparison Experiments on HuCenLife [23]. "DM" stands for "Dynamic Method," indicating whether it is a method used for modeling dynamic point cloud videos. For the static methods, which are designed for processing static point clouds, we apply them on each frame of the point cloud sequence and then fuse these frame features after the encoder network by element-wise adding. "SL" stands for "Self-learning", signifying whether the method employs a self-learning mechanism.

	DM	SL	lift	carry	move	pull_push	sco-bal	hum-inter	fitness	entertain	sports	bend-over	sit	walk-stand	mAcc
PointNet [15]	✗	✗	45.5	48.8	33.3	84	59.4	2.6	65.3	49.3	34.8	29.2	54.3	61	47.3
PointNet++ [16]	✗	✗	49.5	45.7	35.6	52.7	59	6	28.6	43.8	41.2	31.9	38.8	55	40.7
PointMLP [12]	✗	✗	48.5	47.7	57.7	80.1	80.3	36.1	75.7	60.8	39.5	54.9	55.8	59.7	58.1
PointNeXt [17]	✗	✗	48.1	56.6	34.1	80	85.6	22.6	50	38	25.7	25.5	63.1	70.9	50
PCT [7]	✗	✗	39.7	54.9	52.3	80.2	89.8	9.8	63.3	73.6	37.7	62.5	51	75.8	57.6
HuCenLife [23]	✗	✗	45	44.4	52.7	81.2	86.7	23.1	81.2	54.8	41.7	54.8	53.2	70	57.4
PSTNet [5]	✓	✗	30.2	22.6	61.4	64.7	74.6	21.6	20.8	82.4	39.7	51.1	36	15.4	43.4
PSTNet++ [3]	✓	✗	31.8	35.4	19.4	77.4	52.1	44.8	65.3	52.8	51.6	43.8	63	65.3	50.2
P4Transformer [2]	✓	✗	52.6	44.1	20.6	83.8	67.5	28.1	35.4	68.7	50.6	38.8	62.6	63.8	51.4
PST-Transformer [4]	✓	✗	54.2	40.3	23.4	82.6	78.5	21.8	25	51.9	37.7	68.1	79	74.5	53.1
PPTr [21]	✓	✗	48.2	46	18	79.1	71.5	20	44.7	63.7	52.4	35.6	65.4	70	51.2
PointMAE [13]	✗	✓	53.4	53.1	47.2	84.9	88.8	7.8	71.4	76.8	39.2	57.9	41.8	74.2	58
MaST-Pre [18]	✓	✓	32.8	39.9	48.4	84.5	87.4	31.4	70.7	59.1	43.3	51.7	66.9	32.5	54.1
PointCMP [19]	✓	✓	25.6	8.3	56.2	78.8	71.9	7.8	65.3	58.6	52.9	55.1	72.9	19.5	47.7
UniPVU-Human	✓	✓	27.1	37.3	57.1	82.6	84	24.7	**85.4**	52.1	**53.9**	**93.8**	67.3	**76.1**	**61.8**

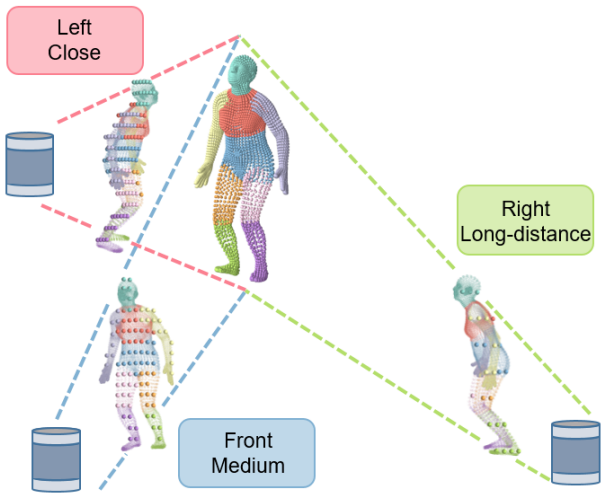

Figure 2. We create a synthetic dataset of 1 million LiDAR human point cloud instances, using the AMASS dataset for 3D human meshes and simulating LiDAR scans from various perspectives and distances for enhancing the diversity of the samples, so as to better simulate the distribution of real-world data.

9 main categories: head, left-arm, right-arm, up-body, low-body, upleft-leg, upright-leg, lowleft-leg, and lowright-leg. Each LiDAR point is automatically labeled with the nearest vertex's body part label.

C.2. Dataset and Evaluation Metrics

Similar to LiDARFlow-Human (Section. B), we also establish a benchmark on LiDARPart-Human and will make it public. As shown in Table. 2, the evaluation metric for LiDARPart-Human is the mean Intersection over Union (mIoU), which is the average of the IoUs calculated for each of the 9 human body parts.

D. The Network Design Details for the Tokenizer

As previously mentioned, in the self-learning module, the motion flow features F are not fused with the part patches features P. This design prevents premature leakage of location information of masked tokens to the STEncoder. The network design details for the Tokenizer are illustrated in Figure. 3. During self-learning, each point of P is mapped to a feature vector using several shared MLPs. Subsequently, max-pooled features are concatenated to each feature vector. These are then processed through several MLPs to expand their dimension to $C = 384$. During fine-tuning, the same operation is applied to the motion flow F. The features of P and F are then fused through element-wise addition. Finally, a max-pooling layer is applied to derive the part token T.

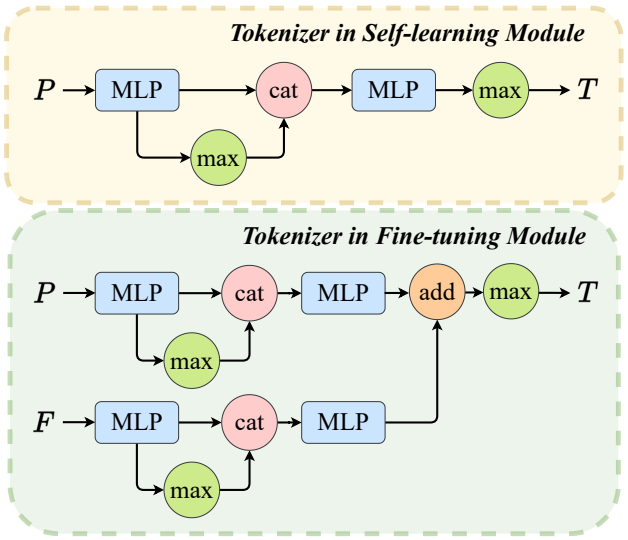

Figure 3. The network design details for the tokenizer. The primary distinction between the tokenizer in the fine-tuning module and that in the self-learning module lies in the integration of the motion flow features, denoted as F.

Table 4. Comparative Analysis of Model Parameter Numbers in Transformer-Based Dynamic Point Cloud Methods.

	Num of Params(M)↓	
	Self-learning	Fine-tuning
P4Transformer [2]	/	40.37
PST-Transformer [4]	/	60.36
PPTr [21]	/	120.7
MaST-Pre [18]	140.76	120.66
UniPVU-Human	**34.92**	**22.48**

E. Supplementary Comparison Experiments on HuCenLife [23]

For more comprehensive and extensive comparisons, we supplement our comparison experiments on HuCenLife [23] with methods specifically designed for modeling dynamic point cloud videos [2–5, 21]. As can be seen from Table. 3, although these methods perform better in categories that require modeling motion features for accurate recognition (Fitness, Sports, Bend-Over, Walk-Stand) than static point cloud methods, there is still a significant performance gap compared to our UniPVU-Human. This confirms the superiority of our method in capturing human motion representations. We also compared our method with self-learning approaches based on contrastive learning [19]. The experimental results demonstrate the superiority of our self-learning mechanism.

F. Comparative Analysis of Model Parameter Numbers

Transformer[20]-based methods [1, 7, 10, 24] have achieved considerable performance in point cloud feature extraction. However, their large model size typically results in significant computational demands. As we can see from Table. 4, the parameter number of other transformer-based dynamic point cloud methods [2, 4, 18, 21], are several times greater than that of our UniPVU-Human. Our model maintains a parameter number of twenty to thirty million in both the self-learning and fine-tuning stages, which is comparable to ResNet-50 [8]. Therefore, our UniPVU-Human achieves better performance with fewer parameters, making it a lightweight and effective model well-suited for real-world applications.

References

[1] Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. *arXiv preprint arXiv:2305.11487*, 2023. 3

[2] Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point 4d transformer networks for spatio-temporal modeling in point cloud videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14204–14213, 2021. 2, 3

[3] Hehe Fan, Xin Yu, Yi Yang, and Mohan Kankanhalli. Deep hierarchical representation of point cloud videos via spatio-temporal decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9918–9930, 2021. 2

[4] Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point spatio-temporal transformer networks for point cloud video modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):2181–2192, 2022. 2, 3

[5] Hehe Fan, Xin Yu, Yuhang Ding, Yi Yang, and Mohan Kankanhalli. Pstnet: Point spatio-temporal convolution on point cloud sequences. *arXiv preprint arXiv:2205.13713*, 2022. 2, 3

[6] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang. Hplflownet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3254–3263, 2019. 1

[7] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7:187–199, 2021. 2, 3

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[9] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Pro-*

ceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 529–537, 2019. 1

[10] Yahui Liu, Bin Tian, Yisheng Lv, Lingxi Li, and Fei-Yue Wang. Point cloud classification using content-based transformer via clustering in feature space. *IEEE/CAA Journal of Automatica Sinica*, 2023. 3

[11] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 851–866. 2023. 1

[12] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123*, 2022. 2

[13] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, pages 604–621. Springer, 2022. 2

[14] Gilles Puy, Alexandre Boulch, and Renaud Marlet. Flot: Scene flow on point clouds guided by optimal transport. In *European conference on computer vision*, pages 527–544. Springer, 2020. 1

[15] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas 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*, pages 652–660, 2017. 2

[16] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2

[17] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 35:23192–23204, 2022. 2

[18] Zhiqiang Shen, Xiaoxiao Sheng, Hehe Fan, Longguang Wang, Yulan Guo, Qiong Liu, Hao Wen, and Xi Zhou. Masked spatio-temporal structure prediction for self-supervised learning on point cloud videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16580–16589, 2023. 2, 3

[19] Zhiqiang Shen, Xiaoxiao Sheng, Longguang Wang, Yulan Guo, Qiong Liu, and Xi Zhou. Pointcmp: Contrastive mask prediction for self-supervised learning on point cloud videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1212–1222, 2023. 2, 3

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[21] Hao Wen, Yunze Liu, Jingwei Huang, Bo Duan, and Li Yi. Point primitive transformer for long-term 4d point cloud video understanding. In *European Conference on Computer Vision*, pages 19–35. Springer, 2022. 2, 3

[22] Wenxuan Wu, Zhiyuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin. Pointpwc-net: A coarse-to-fine network for supervised and self-supervised scene flow estimation on 3d point clouds. *arXiv preprint arXiv:1911.12408*, 2019. 1

[23] Yiteng Xu, Peishan Cong, Yichen Yao, Runnan Chen, Yuenan Hou, Xinge Zhu, Xuming He, Jingyi Yu, and Yuexin Ma. Human-centric scene understanding for 3d large-scale scenarios. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20349–20359, 2023. 1, 2, 3

[24] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021. 3