

On-the-Fly Spatio-Temporal Human Segmentation of 3D Point Cloud Data By Micro-Size LiDAR

Yuma Okochi
Grad. Sch. of Info. Sci. and Tech.
Osaka University
Osaka, Japan
y-okochi@ist.osaka-u.ac.jp

Hamada Rizk
Grad. Sch. of Info. Sci. and Tech.
Osaka University, Osaka, Japan
Tanta University, Tanta, Egypt
hamada_rizk@ist.osaka-u.ac.jp

Hirozumi Yamaguchi
Grad. Sch. of Info. Sci. and Tech.
Osaka University
Osaka, Japan
h-yamagu@ist.osaka-u.ac.jp

Abstract—The technology of 3D recognition is evolving rapidly, enabling unprecedented growth of applications towards human-centric intelligent environments. On top of these applications human segmentation is a key technology towards analyzing and understanding human mobility in those environments. However, existing segmentation techniques rely on deep learning models, which are computationally intensive and data-hungry solutions. This hinders their practical deployment on edge devices in realistic environments. In this paper, we introduce a novel micro-size LiDAR device for understanding human mobility in the surrounding environment. The device is supplied with an on-device lightweight human segmentation technique for the captured 3D point cloud data using density-based clustering. The proposed technique significantly reduces the computational complexity of the clustering algorithm by leveraging the Spatio-temporal relation between consecutive frames. We implemented and evaluated the proposed technique in a real-world environment. The results show that the proposed technique obtains a human segmentation accuracy of 99% with a drastic reduction of the processing time by 66%.

Index Terms—3D point cloud, Micro-size LiDAR, segmentation, wearable device, time-series 3D processing

I. INTRODUCTION

The current advances in the sensing capabilities of IoT devices and surveillance cameras open the door for many novel applications [1]–[10]. These applications enrich our traditional environments leading to the next generation of human-centric intelligent environments that improve the life quality of individuals. This is achieved through the sort of contextual intelligence obtained by analyzing human mobility in indoor environments.

The 3D technologies are developing rapidly as 3D data can facilitate capturing rich information (in the form of a 3D point cloud) of the surrounding environment, including shapes and locations of objects without any visible camera that usually leads to privacy leakage. Thus, they have been adopted in many applications, including 3D point cloud segmentation [11], 3D shape classification, and 3D object detection and tracking [12]. However, most of these applications rely on large-size and expensive LiDARs, which provide dense point clouds that are then processed using computationally intensive deep neural networks. For instance, PointNet [13] is a pio-

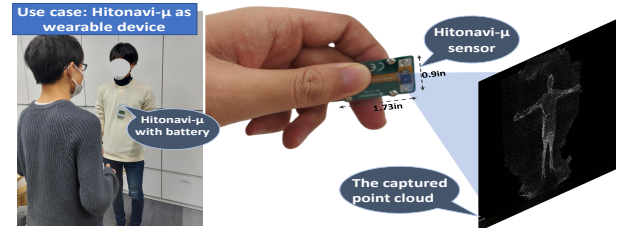


Fig. 1. A point cloud captured by Hitonavi-μ.

neering work for point cloud classification and segmentation using an order-insensitive symmetric function and Volumetric convolutional neural network (CNN) [14]. Pointpillars [15] divides a 3D point cloud into vertical pillars and convolutes them for input into a 2D CNN. However, large-size LiDARs are usually non-portable, which hinders their adoption as edge devices, specifically in mobile scenarios of the application. Additionally, deep learning models (e.g., large CNNs) require intensive processing and thus do not suit the processing power at edge devices.

In this paper, we propose a system for enabling human segmentation with a cheap possibly-wearable LiDAR, as shown in Fig. 1. We introduce a brand new design of a micro-size LiDAR that can scan and represent the surrounding environment in a privacy-preserving point cloud representation. This representation can be leveraged to recognize people in the vicinity of the sensor without revealing their identity or any sensitive information such as visible images or voices. However, the proposed system has to handle the challenge of processing time-series 3D point cloud data on-device without necessitating the availability of high-performance servers, e.g., [16], [17]. To this end, the proposed system introduces an optimized computationally efficient version of the DBSCAN density-based clustering technique for real-time grouping of 3D point clouds and identifying the relationship between clusters in sequential frames. The features of the obtained clusters are leveraged to identify the category (class) of each cluster in the scene.

Evaluation of the proposed system in a realistic testbed in our lab confirms its efficacy in real-time segmentation. Specifically, the proposed system achieves a segmentation F1-score of 99% with an average processing time of 33.9ms per frame. This yields 66% saving of the processing time compared to the baseline without compromising on the performance.

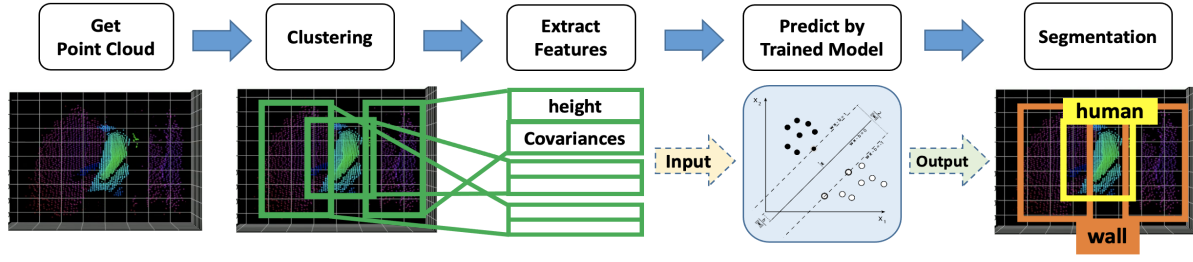


Fig. 2. Illustration of processing flow of the system

The rest of this paper is structured as follows: In Section II, we describe the sensors, and the proposed clustering method of time series data. In Section III, we describe the data collection process and provide a detailed evaluation of the system. Finally, we conclude the paper in Section IV.

II. THE SYSTEM

This section presents the processing flow of the system, including its different modules, as shown in Fig. 2.

A. The Sensing Part

LiDAR, in general, is remote sensing means used to measure the distance to the surface of surrounding objects, e.g., walls, humans, etc. Typically, it measures the time for a laser pulse to travel from the sensor to an object and reflected and thus estimates the distances in the form of 3D point cloud [2]. However, existing LiDARs in the market are **non-portable large-size LiDAR** which is quite expensive. Therefore, we introduce Hitonavi- μ : a small human sensing device with a computing unit, battery, and a micro-size portable version of the LiDAR sensor [18] as shown in Fig. 1. The LiDAR sensor has a rectangle shape of 44×16 for width, height, and depth. Thus, the sensor is relatively smaller and lighter than the consumers' LiDAR devices. The rationale behind this design is to facilitate leveraging the sensor as a wearable device, paving the road for the next generation of human-centric technology in intelligent environments. The sensor consists of an infrared laser and a CMOS image sensor and can acquire a 3D point cloud in real-time. Each 3D point in the point cloud has only x, y, and z coordinate information, and no color or other information is available. The captured point cloud is then fed to the clustering and segmentation techniques for further processing. The LiDAR sensor is connected to a computing unit which is a battery-equipped Raspberry Pi 4 Model B for processing our algorithms.

B. The Clustering Part

This section aims at efficiently clustering the surrounding objects in the scene while keeping the associated processing of clustering algorithm minimal. For ensuring the clustering performance, we leverage Density-Based Spatial Clustering of Applications with Noise algorithm (DBSCAN) [19] motivated by its advantages. It is robust to outliers and noise [20] and can work independently of the domain knowledge information e.g., the number of clusters. Additionally, unlike other clustering techniques, DBSCAN can create clusters of arbitrary shapes

Algorithm 1 Clustering for time series 3D point cloud data

Input: pcs : time series 3D point cloud data
Input: $minPts$: Density threshold of DBSCAN
Input: ϵ : Radius of searching range of DBSCAN
Input: d_I : Distance Threshold of *include()*
Input: d_c : Distance Threshold of *combine()*
Output: pcs_ls_list : time series pcs and labels, labels are indicating to which cluster each point belongs

```

1:  $t \leftarrow 0$ 
2:  $pc\_ls_t \leftarrow DBSCAN(pcs_t, minPts, \epsilon)$ 
3:  $bb_t \leftarrow bounding\_box(pc\_ls_t)$ 
4:  $pcs\_ls\_list[t] \leftarrow pc\_ls_t$ 
5: for  $t = 1, \dots, len(pcs) - 1$  do
6:    $pc\_ls_t \leftarrow update\_label(pcs_t, bb_{t-1})$ 
7:    $uc\_pc\_ls_t \leftarrow uncontained(pc\_ls_t)$ 
8:   if  $len(uc\_pc\_ls_t) > 0$  then
9:      $ls\_sub_t \leftarrow DBSCAN(uc\_pc_t, minPts, \epsilon)$ 
10:    if  $uc\_pc\_ls_t$  has new clusters then
11:       $pc\_ls_t \leftarrow include(pc\_ls_t, uc\_pc\_ls_t,)$ 
12:       $pc\_ls_t \leftarrow combine(pc\_ls_t)$ 
13:       $bb_t \leftarrow bounding\_box(pc\_ls_t)$ 
14:    end if
15:  end if
16:   $pcs\_ls\_list[t] \leftarrow pc\_ls_t$ 
17: end for
18: return  $pcs\_ls\_list$ 

```

and of varying densities. However, DBSCAN has a high complexity of $O(n^2)$, where n is the number of points. The problem is that clustering 3D points of each point cloud, required heavy processing which negatively affects the real-time performance when deployed on the edge (Hitonavi- μ).

To mitigate the computational overhead, we propose a modified version of DBSCAN as shown in Fig. 2. The basic idea of the modified algorithm is that consecutive frames have minor changes if a sufficient frame rate is ensured. Thus, considering the relation between those frames and only cluster the new components leads to a significant reduction of points and thus the clustering processing time. The pseudo code of the algorithm is shown in Algorithm 1. The details of each function of the algorithm is described below and the output of each function is illustrated in Fig. 3.

bounding_box(): This function finds the coordinates of the vertices of a rectangular object that covers each cluster generated by DBSCAN.

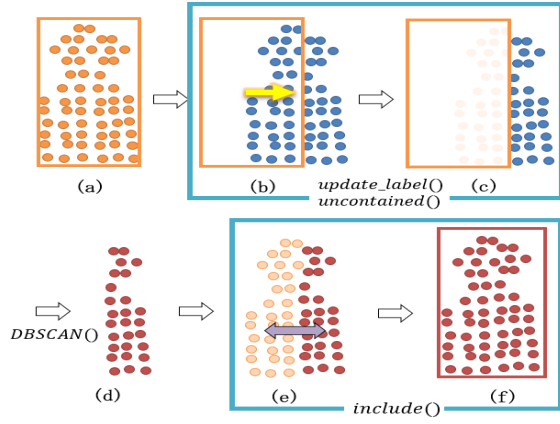


Fig. 3. Illustration of how the proposed algorithm works (this example is 2D-case, bounding boxes are rectangle). Figures (b) to (f) depicts the operation of the functions `update_label()`, `uncontained()` and `include()`, respectively. The red points are those in a cluster, and the oranges are the points labeled by `update_label()`.

update_label(): This function propagates the cluster index from one frame to its subsequent one. Specifically, it takes as input the point cloud of the frame at time t and the bounding box of the frame at time $t-1$. So, at time t , points are assigned the same cluster index of the bounding box (of the previous frame), which typically covers them. The remaining uncovered points are further processed by the `uncontained()` function.

uncontained(): This function extracts remaining points that are uncovered by any arbitrary bounding box and kept anonymous. Then feeds them to `DBSCAN()` (as in line 9 of Algorithm 1) to group them into clusters. It is worth mentioning that the clustering processing time is drastically shortened by executing DBSCAN on a significantly fewer number of points m . Thus the complexity of DBSCAN is improved from $O(n^2)$ to $O(m^2)$.

include(): This function is executed once clusters of the remaining points (sub-point cloud) are obtained. This function decides whether the obtained clusters should be merged with other clusters covered by bounding boxes or not. This is done by calculating the distance from the bounding box to the cluster center point. A cluster whose center is within a distance threshold d_I of a bounding box is then combined with the points encompassed by that bounding box using the `combine()` function.

combine(): This function calculates the distance between new clusters and clusters covered by bounding boxes and combines clusters whose inter-distance is less than a predefined threshold value d_c . It is worth mentioning that clustering noisy point clouds may lead to several sparse clusters. However, the `combine()` function can resolve this issue by combining adjacent clusters while keeping anomalous ones apart facilitating their removal.

Fig. 3 shows the operation of the different functions of the proposed algorithm. The output of the proposed algorithm is a set of clustered points.

C. The Segmentation Part

This section describes how clusters are assigned labels and thus point clouds are segmented. Towards achieving that, seven

features are extracted from each cluster. One is the height of the cluster from the ground (i.e., the y-coordinate), and the other six are the elements of the variance-covariance matrix in 3D space. The intuition behind using the height of clusters is that it can facilitate the discrimination between walls and humans. Mainly in indoor environments, the height of walls is usually higher than the height of humans. Variance and covariance are used to further enhance the discrimination of humans. This has been shown to be useful for recognizing differences in cluster shapes (i.e., the vertical and horizontal spread of a point cloud). Based on these features, we train a classification model to identify the class of each group of points. Typically, Support Vector Machine Classification (SVC) model is adopted due to its favorable performance for real-time applications.

III. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of the proposed technique in a real-world testbed. Hitonavi- μ is fixed at height of 1m and used to capture the point cloud, in a large enough room has area of $15m \times 25m$, with an average frame rate of 10 fps. The captured point clouds constitute the dataset of 1776 frames which is pre-processed by removing outliers and annotating points belong to each object (e.g., human and walls) in the frame. This annotation process is important for training and testing the proposed techniques. For evaluation purposes, we consider a hold-out test set having 240 frame of six patterns of human mobility including: passage of people in the left and right directions, approach of people, and departure of people, with and without walls in the picture.

The proposed modified clustering technique is compared to the baseline DBSCAN algorithm, which iterates on every 3D point in the point cloud of every frame. The evaluation metrics are processing time, precision, recall, and F1-score. The processing time is the average time taken to segment all frames 20 times. The precision, recall, and F1-score are measured at the point level. We implemented the proposed techniques with Python programming language using Scikit-learn library (which provides the basic DBSCAN implementation) on Raspberry Pi 4 Model B.

The experimental results are summarized in Table I and Table II. The processing time of the proposed algorithm and the baseline is reported in Table I. The table confirms that the proposed clustering algorithm surpasses the baseline in all test cases by an average of 66% less processing time. In particular, the processing time is significantly reduced by 71.1% in empty environments (where only walls are captured). This is due to the fact that the data does not change much in consequent frames. However, in environments of high human mobility, the proposed algorithm obtains a remarkable reduction in the processing time by at least 60.6%. This result validates the real-time performance of the proposed algorithm without compromising on the segmentation performance as reported in Table II. Fig. 4 shows the segmentation results of a frame segmented using (a) the basic DBSCANs and (b) using the proposed technique. The figure confirms the favorable

TABLE I
EXPERIMENTAL RESULT: ESTIMATION TIME PER FRAME

Dataset for test		Basic DBSCANs	Modified DBSCANs	Reduction (%)
1	One Wall	84.2	23.3	71.1
2	Human-passing	85.7	33.7	60.6
3	Human-approaching	115.1	44.0	61.8
4	Human-departing	118.7	34.7	70.8
			Average Reduction	66

TABLE II
EXPERIMENTAL RESULT: EVALUATION METRICS

Dataset for test			1	2	3	4
			One Wall	Human passing	Human approaching	Human departing
Basic DBSCANs	Human	Precision	-	99.6	98.9	99.7
		Recall	-	99.9	99.2	100.0
		F1-score	-	99.8	99.0	99.9
	Wall	Precision	100.0	100.0	99.3	100.0
		Recall	100.0	99.9	99.1	99.8
		F1-score	100.0	100.0	99.2	99.9
Modified DBSCANs	Human	Precision	-	71.9	99.6	92.4
		Recall	-	89.7	99.4	87.1
		F1-score	-	79.8	99.5	89.6
	Wall	Precision	100.0	98.5	99.5	90.5
		Recall	99.3	95.1	99.6	94.5
		F1-score	99.6	96.8	99.6	92.4

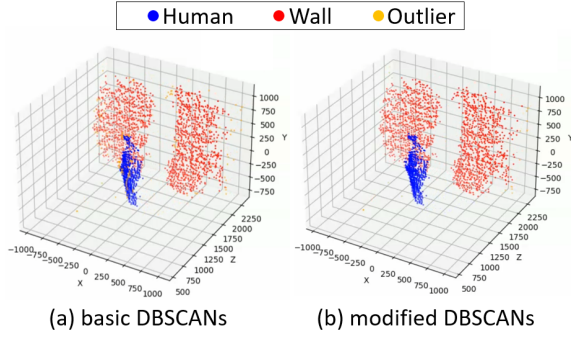


Fig. 4. Segmentation Result

performance of the proposed technique as a real-time human segmentation algorithm for our brand new micro-size sensor.

IV. CONCLUSIONS AND FUTURE WORK

We proposed a computational efficient clustering-based technique for processing time-series 3D point cloud data as an important step to segment humans in the surrounding environment. We also proposed a brand new design of mobile/wearable LiDAR opening the door for a wide range of applications. We showed how the proposed technique leveraged the spatial relation between clusters among consecutive frames to significantly reduce the processing time. Then, statistical features are extracted from the obtained clusters enabling human recognition and segmentation. Implementation of the proposed technique in our lab yielded a fine-grained segmentation performance and real-time processing in six test cases. This was done with 66% saving in a reduction in the computational complexity and processing time.

In the future, we plan to investigate and develop the proposed system in crowded environments. We will also study the effect of changing the parameters on the system performance.

ACKNOWLEDGMENT

This work was supported in part by A-STEP grant (no. JPMJTR20RV) from Japan Science and Technology Agency).

REFERENCES

- [1] S. Yamada, H. Rizk, and H. Yamaguchi, "An accurate point cloud-based human identification using micro-size lidar," in *International Conference on Pervasive Computing and Communications Workshops and Affiliated Events (PerCom Workshops)*. IEEE, 2022.
- [2] H. Rizk, H. Yamaguchi, M. Youssef, and T. Higashino, "Gain without pain: Enabling fingerprinting-based indoor localization using tracking scanners," in *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, 2020, pp. 550–559.
- [3] H. Katayama, T. Mizomoto, H. Rizk, and H. Yamaguchi, "You work we care: Sitting posture assessment based on point cloud data," in *International Conference on Pervasive Computing and Communications Workshops and Affiliated Events (PerCom Demos)*. IEEE, 2022.
- [4] H. Rizk, D. Ma, M. Hassan, and M. Youssef, "Indoor localization using solar cells," in *the Conference on Pervasive Computing and Communications and Affiliated Events (PerCom WiP)*. IEEE, 2022.
- [5] I. Fahmy, S. Ayman, H. Rizk, and M. Youssef, "Monofi: Efficient indoor localization based on single radio source and minimal fingerprinting," in *the 29th International Conference on Advances in Geographic Information Systems*, 2021, pp. 674–675.
- [6] V. Erdélyi, H. Rizk, H. Yamaguchi, and T. Higashino, "Learn to see: A microwave-based object recognition system using learning techniques," in *the International Conference on Distributed Computing and Networking*, 2021, pp. 145–150.
- [7] K. Alkielek, A. Othman, H. Rizk, and M. Youssef, "Deep learning-based floor prediction using cell network information," in *the 28th International Conference on Advances in Geographic Information Systems*, 2020, pp. 663–664.
- [8] H. Rizk, H. Yamaguchi, T. Higashino, and M. Youssef, "A ubiquitous and accurate floor estimation system using deep representational learning," in *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, 2020, pp. 540–549.
- [9] H. Rizk, M. Abbas, and M. Youssef, "Device-independent cellular-based indoor location tracking using deep learning," *Pervasive and Mobile Computing*, vol. 75, p. 101420, 2021.
- [10] H. Rizk, T. Amano, H. Yamaguchi, and M. Youssef, "Smartwatch-based face-touch prediction using deep representational learning," in *mobile ubiquitous*, EAI. Springer, 2021.
- [11] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1887–1893.
- [12] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 1907–1915.
- [13] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.
- [14] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5648–5656.
- [15] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705.
- [16] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *arXiv preprint arXiv:1706.02413*, 2017.
- [17] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, "2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 547–12 556.
- [18] MagikEye Inc., "MagikEye ILT001," <https://support.magik-eye.com/ILT001/>, visited in 2021/11/10.
- [19] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.
- [20] A. Elmogy, H. Rizk, and A. M. Sarhan, "Ofcod: On the fly clustering based outlier detection framework," *Data*, vol. 6, no. 1, p. 1, 2021.