

Curved-Voxel Clustering for Accurate Segmentation of 3D LiDAR Point Clouds with Real-Time Performance

Seungcheol Park, Shuyu Wang, Hunjung Lim, and U Kang

Abstract—Given 3D LiDAR point clouds, how can we segment them fast and accurately? Fast and accurate segmentation of 3D LiDAR points is an important issue in mobile robotics with various applications in classification, tracking, SLAM, etc. Despite its importance, existing methods do not provide both speed and accuracy; in particular, methods performing segmentation in the 3D domain are too slow, disabling its use in real-time processing.

In this paper, we propose Curved-Voxel Clustering (CVC), a fast and accurate method for segmenting 3D LiDAR point clouds utilizing LiDAR-optimized curved-voxel. CVC attains fine discriminations by considering three important aspects for clustering 3D LiDAR points: distance from the sensor, directional resolutions, and rarity of points. CVC succeeds in providing real-time performance by carefully managing curved-voxels with a hash table. Especially, CVC works well on sparse 3D point clouds. Through experiments, we show that our method is up to $1.7\times$ faster and 30% more accurate than other segmentation methods. CVC enables real-time segmentation with more than 20 runs in a second.

I. INTRODUCTION

How can we segment 3D LiDAR point clouds fast and accurately? LiDAR sensors have been widely used in mobile robotics, autonomous vehicles, and other research areas because of their wide horizontal field of view and the long scan distance. Many researchers have extensively utilized LiDAR for pedestrian classification [1]–[5], multi-robot mapping [6], [7], etc.

Existing segmentation methods based on 3D LiDAR point clouds are divided into three groups: segmentation in the 3D domain [8]–[10], segmentation with occupied grid cells [11]–[13], and segmentation on a range image [14]. However, existing methods are either slow due to high computational costs, or inaccurate since they do not carefully consider the characteristics of 3D LiDAR point clouds.

We propose Curved-Voxel Clustering (CVC), a fast and accurate method for segmenting 3D LiDAR point clouds. CVC efficiently and accurately segments point clouds by 1) introducing a new spatial primitive called curved-voxel, 2) carefully considering three distinct properties of 3D LiDAR point clouds (details in Section II), and 3) an efficient hash-based data structure.

Fig. 1 shows a case study of segmenting 3D LiDAR point clouds for five people. Each color denotes a cluster. Note that CVC segments five people correctly even if they are close

Seungcheol Park, Shuyu Wang, and U Kang (corresponding author) are with Department of Computer Science and Engineering, Seoul National University, Seoul 08826, Republic of Korea. Hunjung Lim is with Samsung Electronics. ant6si@snu.ac.kr, wangshuyu79@gmail.com, ukang@snu.ac.kr, hunjung.lim@samsung.com

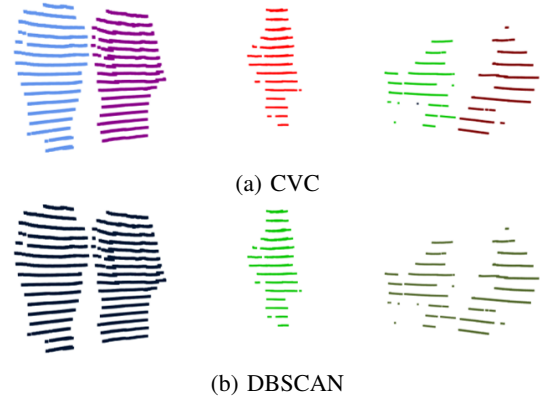


Fig. 1: Segmentation results of CVC (proposed) and DBSCAN using data recorded with a Velodyne VLP-16 scanner (best viewed in color). Only CVC segments five people correctly even if they are close to each other.

TABLE I: Comparison of our proposed method CVC and competitors. CVC is the only method that performs fast and accurate segmentation considering all the desired properties (details in Section II-A) for segmenting 3D LiDAR points.

Method	Speed	Accuracy	Prop. 1	Prop. 2	Prop. 3
RBNN [9]	✓				△
RBNN*	✓	△	✓		△
DBSCAN [10]		△	△		△
Cluster-all [8]	✓				△
CVC	✓	✓	✓	✓	✓

to each other, while DBSCAN incorrectly segments them into three clusters. Table I shows a comparison of CVC and other competitors in various aspects. RBNN* denotes our improved version of RBNN [9] (details in Section IV-A). CVC is the only method that is fast, accurate, and considers all the unique properties for segmenting 3D LiDAR point clouds.

The main contributions of this paper are the followings.

- **New Spatial Primitive.** We design curved-voxel, a LiDAR-optimized spatial unit reflecting distinct characteristics of 3D LiDAR point clouds.
- **Algorithm.** We propose CVC, an efficient method for segmenting 3D LiDAR point clouds by utilizing LiDAR-optimized curved-voxels and efficient hash-based data structure.
- **Experiments.** We present experimental results showing that CVC segments 3D LiDAR point clouds up to $1.7\times$ faster and 30% more accurately than other competitors


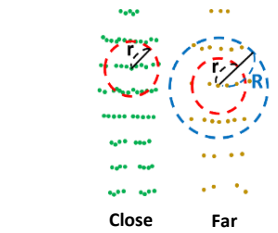



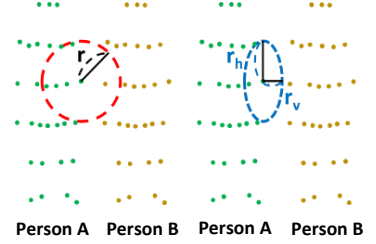
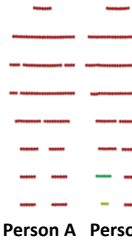
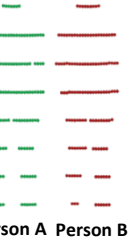

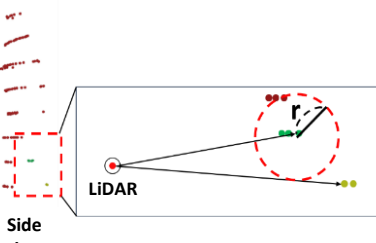
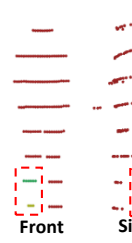

Property	Example	Illustration	RBNN [9]	CVC (proposed)
Property 1 (distance from the sensor)	 Far from the sensor	 Close Far	 Far	 Far
Property 2 (directional resolutions)	 People close to each other	 Person A Person B Person A Person B	 Person A Person B	 Person A Person B
Property 3 (rarity of Points)	 Walking person	 Side view LiDAR	 Front view Side view	 Front view Side view

Fig. 2: Three distinct properties of 3D LiDAR point clouds that need to be considered for accurate segmentation. CVC is designed to carefully consider all the properties.

do. In particular, CVC has an advantage of correctly distinguishing adjacent people.

The rest of the paper is organized as follows: desired properties and problem definition in Section II, proposed method in Section III, experiments in Section IV, related works in Section V, and conclusions in Section VI.

II. DESIRED PROPERTIES AND PROBLEM DEFINITION

In this section, we describe desired properties that segmentation methods for 3D LiDAR points should satisfy, and define the problem addressed in this paper.

A. Desired Properties for Segmenting 3D LiDAR Points

Since 3D point clouds are generated by laser scans radially emitted from a LiDAR sensor, they have the following three distinct properties. First, the distance between two nearest points grows as the points go farther from the LiDAR sensor (Property 1 in Fig. 2). Second, the vertical angular resolution is much larger than the horizontal one, since LiDAR device has few number of vertical channels (Property 2 in Fig. 2). For example, a popular LiDAR device VLP-16 has ten times lower vertical resolution than the horizontal one. Third, a LiDAR sensor gives only one point for each laser scan (Property 3 in Fig. 2). Combined with the low

vertical resolution, this gives a large depth difference in radial directions especially when we detect an inclined object.

In order to correctly segment 3D point clouds from the LiDAR sensor with the aforementioned properties, a good segmentation method needs to satisfy the following desired properties.

- **Property 1 (consider distance from the sensor):** group points correctly, regardless of their distances from the LiDAR sensor.
- **Property 2 (consider directional resolutions):** consider the difference of horizontal and vertical angular resolutions.
- **Property 3 (consider rarity of points):** correctly group points detected from successive vertical laser scans even if they have a large depth difference in radial directions.

Fig. 2 illustrates examples of segmenting 3D LiDAR point clouds using RBNN [9] which groups points in a circle of fixed radius r for every point. The third column shows failed cases of RBNN (in red) and desired ones (in blue). For Property 1, which states that the distance between two nearest points grows as they go farther from the sensor, RBNN cannot group points correctly since it uses a fixed-radius circle. We need a method to change radius dynamically in proportion to the distance from the

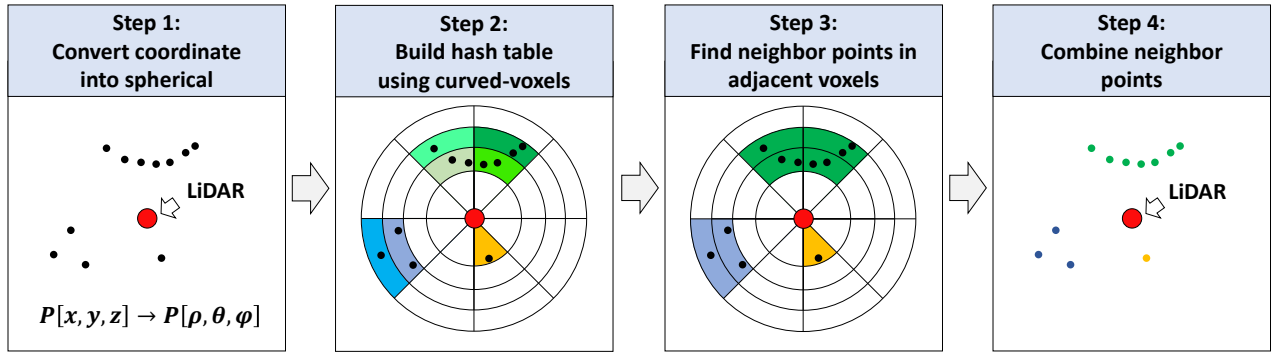


Fig. 3: Steps of CVC, using a bird-eye view. In Step 1, we convert cartesian coordinate into spherical one. In Step 2, we build a hash table that maps curved-voxel indices to indices of points included in each voxel. In Step 3, we find the neighbor points inside 9 voxels surrounding each target voxel including the voxel containing the target point. In the final Step 4, we combine neighbor points into a cluster. Note that nearby points are grouped to clusters of distinct colors.

sensor. For Property 2, which states that the distance between two nearest neighbor points in the vertical direction is substantially larger than that in the horizontal one, RBNN cannot correctly segment adjacent people by a circle since the minimum radius should be larger than the minimum distance between nearest neighbors in vertical direction. We need a method that has independent radius for each direction. For Property 3, which states that two vertically neighboring points out of an inclined object have large depth difference due to the low vertical resolution, RBNN cannot group them correctly since its fixed radius is not enough. We need a method that is able to group nearby points with a depth difference. In Section III, we propose CVC which satisfies all of these desired properties, as shown in the fifth column of Fig. 2, unlike RBNN in the fourth column of the figure.

B. Problem Definition

Based on the desired properties, we formally define the problem of segmenting 3D LiDAR point clouds as follows.

Problem 1 (SEGMENTING 3D LiDAR POINT CLOUDS):

- **Given :** 3D point clouds from LiDAR sensor,
- **Segment :** the points accurately and efficiently, satisfying the following properties:
 - 1) **Consider distance from the sensor:** group points correctly, regardless of their distances from the LiDAR sensor.
 - 2) **Consider directional resolutions:** consider the difference of horizontal and vertical angular resolutions.
 - 3) **Consider rarity of points:** correctly group points detected from successive vertical laser scans even if they have a large depth difference in radial directions.

III. PROPOSED METHOD

We propose CVC, a fast and accurate segmentation method for 3D LiDAR data. We first give an overview of CVC in Section III-A. We then describe details of our method in Sections III-B and III-C.

A. Overview

CVC efficiently segments objects from 3D LiDAR point clouds. The main challenges of segmenting 3D LiDAR points are as follows:

- 1) **Maximize segmentation efficiency.** How can we efficiently segment thousands of points in real-time?
- 2) **Maximize segmentation accuracy.** How can we correctly segment each object even when they are closely placed?

We address the above challenges with the following ideas:

- 1) **Curved-Voxel: a new type of spatial primitive in a spherical coordinate (Section III-B).** Curved-voxel satisfies the properties in Section II-A.
- 2) **Curved-Voxel Clustering (CVC) : a new segmentation algorithm using curved-voxel (Section III-C).** We propose CVC, a fast and accurate segmentation method for 3D LiDAR data based on curved-voxel.

CVC (Algorithm 1) comprises four steps. First, we convert cartesian coordinates into spherical ones, $P = [\rho, \theta, \phi]$, where ρ is the radial distance from the sensor, θ is the azimuth angle, and ϕ is the polar angle. Second, we build a hash table to map a curved-voxel index to indices of points inside the voxel. Third, we find the neighboring points of each point in adjacent curved-voxels using a hash table. Finally, we combine each point and its neighbors into one cluster. Figure 3 shows an example of running CVC through a two-dimensional illustration.

B. Curved-Voxel: New Type of Spatial Primitive in Spherical Coordinate

We propose curved-voxel, a new type of voxel to satisfy the three properties discussed in Section II-A.

Definition 1 (CURVED-VOXEL): a spatial unit consisting of three-dimensional spherical coordinates. The i, j , and k -th curved-voxel $CV_{i,j,k}$ contains points in a spherically shaped voxel as follows:

$$CV_{i,j,k} = \{ P(\rho, \theta, \phi) \mid \Delta\rho * i \leq \rho < \Delta\rho * (i+1), \Delta\theta * j \leq \theta < \Delta\theta * (j+1), \Delta\phi * k \leq \phi < \Delta\phi * (k+1) \} \quad (1)$$

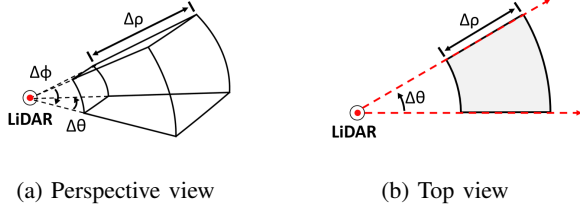


Fig. 4: (a) Perspective view of a curved-voxel where $\Delta\rho$, $\Delta\theta$, and $\Delta\phi$ are unit sizes for each spherical direction. (b) Top view of a curved-voxel (in gray) and two laser beams emitted from the sensor (in red). A curved-voxel has edges that are parallel to the laser beam.

where each $P(\rho, \theta, \phi)$ is in spherical coordinate with the radial distance ρ , azimuth angle θ , and polar angle ϕ . $\Delta\rho$, $\Delta\theta$, and $\Delta\phi$ are unit size parameters for each spherical direction.

Fig 4 illustrates a curved-voxel which satisfies the desired properties for segmenting 3D LiDAR point clouds as follows.

- 1) **Consider distance from the sensor.** A curved-voxel has four edges (in ρ direction) parallel to the laser beams emitted from the sensor. The distance between these edges grows when the voxel goes farther from the sensor. Since we assume all points inside a curved-voxel and its neighboring voxels belong to a same cluster (Section III-C), points are grouped correctly regardless of their distances from the sensor.
- 2) **Consider directional resolutions.** Two independent parameters $\Delta\theta$ and $\Delta\phi$ allow us to adjust horizontal and vertical unit sizes of curved-voxel, respectively. Thus, we consider directional resolutions by adjusting these parameters in proportion to angular resolutions in both directions.
- 3) **Consider rarity of points.** Independent size parameter $\Delta\rho$ enables a curved-voxel to group points even if they have depth difference in radial directions.

We discuss how to perform accurate and efficient clustering using curved-voxels in the next section.

C. Curved-Voxel Clustering (CVC) : New Segmentation Algorithm Using Curved-Voxel

We describe our proposed segmentation method CVC in Algorithm 1. We first convert the coordinate of points into spherical one which consists of ρ , θ and ϕ (line 2). Then, we build a hash table that maps each curved-voxel index to indices of points inside the voxel (line 3). Note that we maintain sparse representations of the hash table, storing only voxels that contain at least a point. To do that, we 1) compute voxel indices by dividing raw coordinate of each point by the three size parameters $\Delta\rho, \Delta\theta$, and $\Delta\phi$, 2) build a preliminary hash table that maps each point index to the index of the curved-voxel containing the point, and 3) invert the preliminary hash table to get the final hash table. As a result, we generate a space-efficient hash table that contains

Algorithm 1 CVC: Curved-Voxel Clustering

Input: 3D point clouds p^r , and curved-voxel size parameters $\Delta\rho, \Delta\theta$, and $\Delta\phi$

Output: list *labels* of clusters of points

- 1: **Initialize:** number n of points $\leftarrow |p^r|$,
labels of clusters of points \leftarrow length n list of zeros,
cluster index $id \leftarrow 0$
- 2: $p \leftarrow \text{convert-to-spherical}(p^r)$
- 3: hash-table, voxel-index
 $\leftarrow \text{build-hash-table}(p, \Delta\rho, \Delta\theta, \Delta\phi)$
- 4: **for** every point p_i **do**
- 5: **if** p_i is already included in a cluster **then**
- 6: **continue**
- 7: **end if**
- 8: neighbors
 $\leftarrow \text{find-neighbors}(\text{voxel-index}[i], \text{hash-table})$
- 9: combine-clusters(p_i , neighbors)
- 10: **if** p_i is still not included in any cluster **then**
- 11: increase id
- 12: assign-cluster(p_i , neighbors, id)
- 13: **end if**
- 14: **end for**
- 15: **return** *labels*

information of only non-empty curved-voxels. After building the hash table, we visit each point p_i to find neighbor points in 27 ($= 3^3$) voxels surrounding the target voxel containing p_i using a hash table, and combine them as a cluster (lines 4~14). Finally, we return the cluster labels of each point.

There are two main ideas that accelerate our algorithm. First, as introduced in a recent study [9], we skip points that are already included in a cluster (lines 5~7), avoiding redundant computations. Second, we utilize a hash table which finds neighbor points of a point in $O(1)$ time. Thanks to these ideas, CVC takes $O(n)$ time to search for neighbors, where n is the number of points, which is faster than $O(n \log n)$ of existing methods, as shown in the following lemma.

Lemma 1 (COST OF FINDING NEIGHBORS):

The expected time complexity of finding neighbors in CVC is $O(n)$, where n is the number of points.

Proof: Finding neighbors is related to lines 5~8 of Algorithm 1. Lines 5~7 is called $O(n)$ times, while line 8 is called $O\left(\frac{n}{k_{\text{average}}}\right)$ times where k_{average} is the average number of neighbors in adjacent curved-voxels over all queries. Thus, the total complexity of finding neighbors is $O(n)$. ■

IV. EXPERIMENTS

We aim to answer the following questions to evaluate the performance of our method CVC.

- **Q1. Efficiency on synthetic data (Section IV-B).** How efficiently does CVC segment synthetic objects with various distances, compared to other methods?
- **Q2. Efficiency on real-world data (Section IV-C).** How efficiently does CVC segment real-world objects, compared to other methods?

TABLE II: Description of real-world 3D LiDAR dataset. Every data set is attained by Velodyne VLP-16 with rotation speed 10 Hz.

Dataset	Time (sec)	Points	Environment
L-CAS 1 [1]	1140	15423	hall
L-CAS 2 [1]	720	15064	hall
L-CAS 3 [1]	1079	15823	hall
HALL	62	9468	hall
OFFICE	86	12527	office

- **Q3. Accuracy (Section IV-D).** What is the accuracy of CVC in segmenting real-world data compared to other methods?

A. Experiment Setup

Systems. All experiments are carried out with Robot Operating System (ROS) Kinetic and Gazebo 7 on Ubuntu 16.04 LTS, in a single machine equipped with an Intel i7-8700 processor and 32GB memory.

Methods. We compare CVC with RBNN*, DBSCAN [10] and Cluster-all method [8]. RBNN* is a modified version of RBNN [9] to make the radius r proportional to the distance from the sensor. By this modification, RBNN* is able to satisfy the desired property 1 to improve the accuracy. We assign parameters of RBNN based on the recent study [1], and implement Cluster-all based on CVC. All of the methods are implemented in Python with NumPy and SciPy library to handle array operation and kd-tree structure.

Dataset. We use both real-world and synthetic datasets. TABLE II shows real-world data we use. L-CAS 1, 2, and 3 are open-source datasets from Lincoln Centre for Autonomous Systems Research (L-CAS) ¹. HALL and OFFICE are datasets collected from our workplace, in a main hall and an office room, respectively. We remove ground plane in the raw dataset by keeping only the points p with heights $z \geq z_{min}$ where z_{min} is the height threshold considering the sensor height. We also generate synthetic data using Gazebo 7. All data are based on Velodyne VLP-16 3D LiDAR equipped with a rotation scanner with speed 10Hz.

B. Efficiency on Synthetic Data

We compare segmentation performance over a range of distances on synthetic data generated by Gazebo 7. We measure the segmentation frequency for five thousand points with varying distances from 2m to 20m. Figure 5 illustrates the experiment results. The figure shows that CVC performs the fastest for almost all distances compared to other algorithms. Cluster-all method performs faster than CVC when the object is close to the LiDAR sensor ($< 5m$), but CVC also shows a high frequency ($> 30Hz$), satisfying real-time performance. Moreover, the frequency of CVC is not significantly affected by the distance from the sensor since the size of a curved-voxel grows when it goes far from the sensor. On the contrary, Cluster-all method performs much slower as the

¹<https://lcas.lincoln.ac.uk/wp/research/data-sets-software/l-cas-3d-point-cloud-people-dataset/>

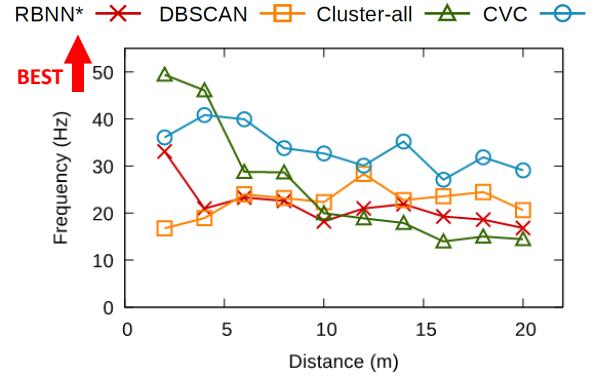


Fig. 5: Segmentation performance over different distances for synthetic data. CVC gives the largest segmentation frequency in most cases.

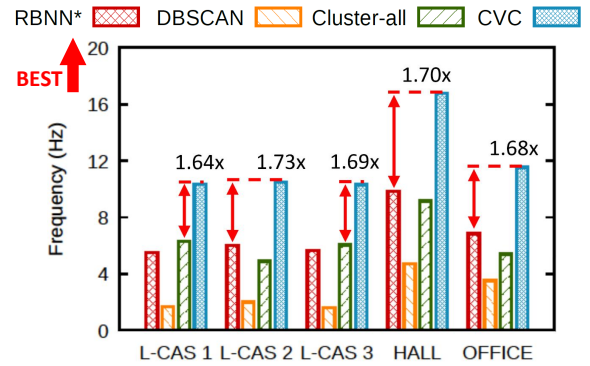


Fig. 6: Segmentation performance on real-world data. CVC efficiently segments real-world 3D point clouds, providing up to $1.7\times$ higher frequency of segmentation.

distance from the sensor grows since it has a fixed size of cubic-shaped voxels.

C. Efficiency on Real-World Data

Figure 6 shows the segmentation performance on five real-world datasets which contain various objects. We select 100 frames from each dataset and compute average segmentation frequency. CVC performs faster than other algorithms in all datasets, thanks to skipping already labeled points and utilizing a hash table. CVC is up to $1.7\times$ faster than other methods, and provides real-time segmentation results with segmentation frequency greater than 20 Hz.

D. Accuracy

Figure 7 shows real-world segmentation accuracy using labeled data in L-CAS dataset. We randomly select 100 pedestrian-labeled data and 50 group-labeled data except for partially visible one. After that, we manually check the results of each method whether it segmented each object correctly or not. CVC accurately segments almost all pedestrian cases and over the half of group cases. It shows the highest accuracy among the four methods, especially in group cases,

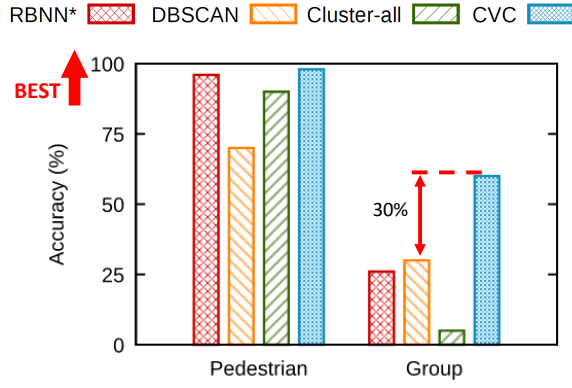


Fig. 7: Segmentation accuracy for real-world data. CVC provides the highest accuracy.

achieving up to 30% higher score than others. This improved accuracy comes from the LiDAR-optimized curved-voxel which satisfies all the desired properties.

Although providing the best accuracy, there are two cases that CVC fails to distinguish each person. In the first case, two people are so close that they look like a connected object, i.e. their distance is smaller than the resolution of LiDAR. In this situation, we might consider more information like surface normal to improve the segmentation quality. In the second case, there are noise points between two adjacent people. These noises prevent CVC from correctly segmenting each person into its own cluster. To address this problem we might further consider the density of points to remove these noises.

V. RELATED WORKS

In this section, we discuss related works on segmenting 3D LiDAR data.

Existing segmentation methods are typically divided into three groups: 1) segmentation in the 3D domain [8]–[10], 2) segmentation with occupied grid cells [11]–[13], and 3) segmentation on a range image [14]. The methods performing segmentation in 3D domain usually compute sophisticated features to find neighboring points and combine them: Cluster-all method [8] utilizes cubic-shaped voxels, while RBNN [9] and DBSCAN [10] utilize a sphere to find neighbors. The problem of these methods is their long running time due to their huge required computations. The methods [11]–[13] that perform segmentation with occupied grid cells run quickly as a result of dimensionality reduction. However, such reduction also limits the accuracy, and they often provide under-segmented results. The method [14] based on a range image performs fast segmentation because it can directly find adjacent points using row and column indices. However, it tends to show over-segmented results for flat objects (e.g. walls).

Our proposed CVC provides faster and more accurate performance compared to existing methods thanks to the curved-voxel primitive and an efficient algorithm.

VI. CONCLUSIONS

In this paper, we propose CVC, a fast and accurate method for segmenting 3D LiDAR points. We design curved-voxel, a new spatial primitive to consider distinct characteristics of 3D LiDAR points. We also propose an efficient hash-based data structure to speed up segmentation. CVC is up to $1.7\times$ faster and 30% more accurate than other segmentation methods. Furthermore, CVC provides segmentation results more than 20 times in a second, enabling real-time segmentation on real-world data.

ACKNOWLEDGMENT

This work is supported by Samsung Electronics Co., Ltd.

REFERENCES

- [1] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for human classification in 3d lidar-based tracking," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, 2017, pp. 864–871.
- [2] Z. Yan, L. Sun, T. Duckett, and N. Bellotto, "Multisensor online transfer learning for 3d lidar-based human detection with a mobile robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018*.
- [3] J. Shackleton, B. V. Voorst, and J. A. Hesch, "Tracking people with a 360-degree lidar," in *Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2010, Boston, MA, USA, August 29 - September 1, 2010*, 2010, pp. 420–426.
- [4] K. Kidono, T. Miyasaka, A. Watanabe, T. Naito, and J. Miura, "Pedestrian recognition using high-definition LIDAR," in *IEEE Intelligent Vehicles Symposium (IV), 2011, Baden-Baden, Germany, June 5-9, 2011*, 2011, pp. 405–410.
- [5] C. Premebida, G. Monteiro, U. Nunes, and P. Peixoto, "A lidar and vision-based approach for pedestrian and vehicle detection and tracking," in *IEEE Intelligent Transportation Systems Conference, ITSC 2007, Seattle, WA, USA, 30 September-3 October 2007*.
- [6] K. M. Wurm, C. Stachniss, and W. Burgard, "Coordinated multi-robot exploration using a segmentation of the environment," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, September 22-26, 2008, Acropolis Convention Center, Nice, France, 2008*, pp. 1160–1165.
- [7] R. Dubé, A. Gavel, H. Sommer, J. I. Nieto, R. Siegwart, and C. Cadena, "An online multi-robot SLAM system for 3d lidars," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*.
- [8] B. Douillard, J. P. Underwood, N. Kuntz, V. Vlaskine, A. J. Quadros, P. Morton, and A. Frenkel, "On the segmentation of 3d LIDAR point clouds," in *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*.
- [9] K. Klasing, D. Wollherr, and M. Buss, "A clustering method for efficient segmentation of 3d laser data," in *2008 IEEE International Conference on Robotics and Automation, ICRA 2008, May 19-23, 2008, Pasadena, California, USA, 2008*, pp. 4043–4048.
- [10] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA, 1996*.
- [11] J. Behley, V. Steinhage, and A. B. Cremers, "Laser-based segment classification using a mixture of bag-of-words," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, November 3-7, 2013*, 2013, pp. 4195–4200.
- [12] M. Himmelsbach, F. von Hundelshausen, and H. Wünsche, "Fast segmentation of 3d point clouds for ground vehicles," in *IEEE Intelligent Vehicles Symposium (IV), 2010, La Jolla, CA, USA, June 21-24, 2010*.
- [13] D. Korchev, S. Cheng, Y. Owechko, and K. Kim, "On real-time lidar data segmentation and classification," 07 2013.
- [14] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3d laser scans for online operation," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2016, Daejeon, South Korea, October 9-14, 2016*, 2016, pp. 163–169.