A Novel Coding Scheme for Large-Scale Point Cloud Sequences Based on Clustering and Registration

Xuebin Sun, Yuxiang Sun, *Member, IEEE*, Weixun Zuo, Shing Shin Cheng, and Ming Liu, *Senior Member, IEEE*

***Abstract*— Due to the huge volume of point cloud data, storing and transmitting it is currently difficult and expensive in autonomous driving. Learning from the high-efficiency video coding (HEVC) framework, we propose a novel compression scheme for large-scale point cloud sequences, in which several techniques have been developed to remove the spatial and temporal redundancy. The proposed strategy consists mainly of three parts: intracoding, intercoding, and residual data coding. For intracoding, inspired by the depth modeling modes (DMMs), in 3-D HEVC (3-D-HEVC), a cluster-based prediction method is proposed to remove the spatial redundancy. For intercoding, a point cloud registration algorithm is utilized to transform two adjacent point clouds into the same coordinate system. By calculating the residual map of their corresponding depth image, the temporal redundancy can be removed. Finally, the residual data are compressed either by lossless or lossy methods. Our approach can deal with multiple types of point cloud data, from simple to more complex. The lossless method can compress the point cloud data to 3.63% of its original size by intracoding and 2.99% by intercoding without distance distortion. Experiments on the KITTI dataset also demonstrate that our method yields better performance compared with recent well-known methods.**

***Note to Practitioners*—This article deals with the problem of efficient compression of point cloud sequences that come from light detection and ranging (LiDARs) mounted on autonomous mobile robots. The vast amount of point cloud data could be an important bottleneck for transmission and storage. Inspired by the HEVC algorithm, we develop a novel coding architecture for the point cloud sequence. The scans are divided into intraframe and interframe, which are encoded separately using different techniques. Our method can be used for the compression of LiDAR point cloud sequences or dense LiDAR point cloud map and will significantly reduce the transmission bandwidth and storage spaces. We have to admit that although our method is less effective for real-time solutions, it can be highly efficient for off-line applications. Future studies will concentrate on further optimizing the coding algorithm to reduce the computational complexity and trying to find a balance between them.**

***Index Terms*— Cluster-based prediction, compression, depth modeling mode (DMM), point cloud sequence, registration.**

Manuscript received October 23, 2020; revised April 10, 2021; accepted May 7, 2021. This article was recommended for publication by Associate Editor P. Tokekar and Editor D. O. Papa upon evaluation of the reviewers' comments. *(Corresponding authors: Shing Shin Cheng; Ming Liu.)*

Xuebin Sun is with the College of Electrical and Information Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: sunxuebin@szu.edu.cn; sunxuebin@tju.edu.cn).

Yuxiang Sun is with the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Hong Kong (e-mail: yx.sun@polyu.edu.hk; sun.yuxiang@outlook.com).

Weixun Zuo is with Shenzhen Unity Drive Innovation Technology Company Ltd., Shenzhen 518000, China (e-mail: zuoweixun@gmail.com).

Shing Shin Cheng is with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: sscheng@cuhk.edu.hk).

Ming Liu is with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong (e-mail: eelium@ust.hk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TASE.2021.3082196.

Digital Object Identifier 10.1109/TASE.2021.3082196

I. Introduction

A. Motivation

ADVANCES in autonomous driving technology have widened the use of 3-D data acquisition techniques. Light detection and ranging (LiDAR) or a 3-D camera is almost indispensable for mobile robots. A number of critical techniques are performed based on point cloud data, such as simultaneous localization and mapping (SLAM) [1], path planning [2], [3], obstacle avoidance [4], and navigation [5], [6]. Point cloud data usually consist of a large number of points, including the location information of spatial objects and one or more attributes such as color and normal. For example, the Velodyne HDL64 LiDAR can measure more than 120 000 points per frame of the point cloud data. Such data require a large amount of space to store it and is difficult to share in real time with current technology. Therefore, the compression of point cloud data has become an urgent problem for autonomous driving.

LiDAR data have the characteristics of being large scale, and having an uneven distribution, and huge volume. It is difficult to remove the temporal and spatial redundancies. As the point cloud data of a vehicle-mounted LiDAR are orderly, it can be converted into a range image. A range image is a grayscale image recording of the distance information of objects from the LiDAR. However, it is hardly optimal to compress the point cloud data directly using image coding technology. Because traditional image or video coding algorithms, such as JPEG2000 [7], JPEG-LS, and high-efficiency video coding (HEVC) [8], can only encode 8-, 10-, or 12-bit integer pixel values, this is unsuitable for floating-point LiDAR data. The measurement range of a Velodyne HDL64 LiDAR can reach 120 m with an accuracy of 2 cm. If the range

value is represented by 8 bits, the measurement accuracy is merely 0.5 m, which will bring safety risks to the pilotless vehicle. On the other hand, different from a texture image, a range image rarely contains any texture and is characterized by sharp object edges and large homogeneous regions with nearly constant values. Using the image-based coding method is inefficient to remove the redundancy for LiDAR depth map.

The octree method has been widely researched for point cloud compression [9]. As the method is lossy, it is unsuitable for vehicle-borne LiDAR point cloud data. Because path planning and obstacle avoidance algorithms require high accuracy of point cloud data in unmanned aerial vehicles, even a small amount of information loss may cause traffic accidents.

In contrast to the aforementioned methods, in this article, we make full use of the structural characteristics of point clouds to remove the spatial redundancy. In addition, we utilize the registration method to remove the time redundancy of a point cloud sequence. Compared with image- and octree-based point cloud compression techniques, the proposed method shows better performance.

B. Contributions

In this article, we propose a novel compression architecture for large-scale point cloud sequences for mobile robots. The major contributions are as follows.

1) Learning from the HEVC algorithm, we propose a novel compression architecture for a point cloud sequence, in which the data are divided into *I* frames and *P* frames, and several techniques are exploited to remove the temporal and spatial redundancies.
2) Inspired by the depth modeling mode (DMM) technique in 3-D-HEVC, an efficient intraframe prediction method is proposed based on point cloud clustering.
3) Considering the structure characteristics of the point cloud, an interframe prediction technique is developed using point cloud registration.
4) To promote the algorithm performance, several neater methods, such as contour map coding, snake-like prediction, and recurrent data coding, are exploited.

C. Organization

The rest of this article is structured as follows. In Section II, we discuss related works. In Section III, we give an overview of the point cloud coding framework. The intracoding method and the intercoding method are presented in Sections IV and V, respectively. Experimental results are shown in Section VI. Finally, this article is concluded with the discussion and possible future research directions in Section VII.

II. Related Work

Over the past decade, scholarly works on point cloud compression have been extensive. Taking the characteristics of the point cloud as a major consideration, the methods can be roughly classified into two categories: structured and unstructured point cloud compression. For each category, there are various kinds of point clouds.

A. Structured Point Cloud Compression

Structured point clouds can be converted into a 2-D panorama range image; therefore, image or video coding methods have been studied compressing the point clouds.

1) LiDAR Data From Mobile Robots: Vehicle-borne LiDAR point cloud data require high real-time performance; however, compared with the point cloud data in the survey and draw field, these data are relatively sparse. Tu *et al.* [10] proposed a method of compressing raw point cloud data using image compression methods. They convert the raw point cloud data into range images and use various image/video compression algorithms to reduce the volume of the data. Liu *et al.* [11] exploited a distance predictor to predict the forthcoming point using the information of previous points. In addition, they use the JPEG2000 standard to compress the color information. As image-based methods do not fully utilize the 3-D characteristics of point cloud data, Tu *et al.* [12] proposed a new compression method using location and orientation information from SLAM. Experimental results demonstrate that the SLAM-based method outperforms image compression-based methods.

2) LiDAR Data From the Survey and Draw Field: Houshiar and N*ü*chter [13] used conventional image-based coding methods to compress 3-D point clouds. By converting the point cloud into panorama images, they encode the range, reflectance, and color value for each point. Ahn *et al.* [14] proposed an adaptive range image coding algorithm for the geometry compression of large-scale 3-D point clouds by predicting the radial distance of each pixel using previously encoded neighbors. Experimental results show that their method obtains better compression performance compared with traditional image or video coding techniques.

3) RGB-D Data From Kinect Sensor: Wang *et al.* [15] designed a 3-D image warping-based depth video compression (IW-DVC) method to compress the depth image captured by RGB-D sensors. The proposed method combines ego-motion estimation with 3-D image warping techniques. Experiment results show that their method attains a high compression ratio without sacrificing depth image quality. By exploiting spatial and temporal redundancy within the point data, Kammerl *et al.* [16] proposed a novel lossy compression approach for point cloud streams. They compare the octree data structures of consecutive point clouds and encode their structural differences. Experimental results show that their method achieves a strong compression performance of a ratio, with 14 at 1-mm coordinate precision.

B. Unstructured Point Cloud Compression

1) Point Cloud Map Built by LiDAR: Elseberg *et al.* [9] proposed a novel structure of octree to store and compress 3-D data without loss of precision. Fan *et al.* [17] proposed a point cloud compression method based on hierarchical point clustering. Their method consists of two stages: level of detail (LOD) hierarchy construction and LOD hierarchy encoding. Experimental results show that their algorithm achieves not only generic topology applicability but also good rate–distortion performance at low bit rates. Motivated by image and video coding techniques, Cohen *et al.* [18] used the 3-D

block-based prediction and transform coding to compress the point cloud. They explore both modified shape-adaptive DCT and 3-D graph transform methods to code the residual data. Experimental results show that their method achieves good compression performance.

2) Point Cloud Map Built by RGB-D Sensor: Morell *et al.* [19] proposed a 3-D lossy compression system based on plane extraction. They represent the points of each scene plane as a Delaunay triangulation and a set of points/area information. Their method can be customized to achieve different data compression or accuracy ratios. Navarrete *et al.* [20] proposed a 3-D compression and decompression method, which allows the use of the compressed data for a registration process. Experimental results show that their method obtains significantly better compression rates with negligible errors for most applications.

3) Immersive 3-D Human Body: Mekuria *et al.* [21] proposed a hybrid point cloud compression architecture for 3-D tele-immersive and virtual reality applications. They combine typical octree-based point cloud compression schemes with hybrid schemes common in video coding. Compared to available point cloud codecs, their method achieves a higher rate–distortion performance. Thanou *et al.* [22] proposed a graph-based compression scheme of dynamic 3-D point cloud sequences. In their method, the time-varying geometry of these sequences is represented by a set of graphs. They use motion estimation to remove the temporal redundancy. Queiroz and Chou [23] proposed a transform coding method for point clouds using a Gaussian process model. They explore four models to represent the covariance function of the Gaussian process. Experimental results demonstrate that their method outperforms both the ID-GFT and AR-GFT methods.

C. Summary and Analysis

Generally, the aforementioned algorithms can significantly reduce the point cloud data size and are used for various applications, such as virtual reality, scanning of historical artifacts, and 3-D printing. However, few researchers have studied the coding method for a point cloud sequence captured by LiDAR mounted on mobile robots. As LiDAR data have the characteristics of being large scale, with uneven distribution and a long detection range, using the aforementioned methods to compress the LiDAR data is inefficient; there are both temporal and spatial redundancies in point cloud sequences. Fortunately, we can learn from their coding techniques, such as prediction [18], clustering [17], and registration [15]. In this article, we propose an efficient coding scheme for LiDAR point cloud data based on clustering and registration.

III. Overview of the Point Cloud Coding Framwork

We address the problem of compression of large-scale structured point cloud sequences from 3-D LiDARs mounted on mobile robots. The proposed compression method exploits both temporal and spatial redundancies in the point cloud sequence. A point cloud sequence is divided into intraframe and interframe. Learning from Mekuria's method [21] and the HEVC coding architecture [24], we propose a novel compression architecture, especially for a LiDAR point cloud sequence, as shown in Fig. 1.

Fig. 2 shows an example of the order arrangement of the frames, including time priority form and compress rate priority form. A point cloud sequence is divided into intraframes (I) and interpredict frames (P). An I-frame is compressed by exploiting the spatial redundancies, whereas a P-frame is compressed by removing the temporal redundancies. The compression of an interframe requires referencing its former encoded/decoded intraframe, which will be used to exploit the temporal redundancies. The very first frame is encoded using intraframe encoding as it does not have any frame as a reference. The I-frame is followed by a P-frame, as the P-frame has the ability to make use of its front I-frame.

A. Outlier Removal Filter

In practice, LiDAR sensors produce a large number of outliers during the range measurement. The outlier will decrease the algorithm efficiency with a large computational cost. In order to reduce the influence of outliers, a filter named RadiusOutlierRemoval is utilized [25]. RadiusOutlierRemoval filter is performed by removing outliers in its input cloud that does not have at least n number of neighbors within a certain range d. d is calculated by the following:

$$d = 2\pi \times r_{\max} \times \frac{H_{\text{resolution}}}{360^\circ} \tag{1}$$

where $r_{\max}$ represents the maximal measurement range of 120 m and $H_{\text{resolution}}$ denotes the horizontal angular resolution 0.18°. In our experiment, we set n to 2 and d to 0.38 m.

B. Convert to Range Image

We choose the KITTI dataset to perform our experiments [26]. The KITTI dataset is captured by a Velodyne HDL-64E rotating 3-D laser scanner covering 26.8° vertical and 360° horizontal fields of view with 64 beams and 0.18° angular resolution. The data are structured and can be converted to a range image through coordinate system transformation.

C. Clustering

In a frame of the point cloud, there are many spatial redundancies of points belonging to the same object. To remove the redundancies, a clustering method is utilized, preparing for intraprediction.

D. Intraprediction

Inspired by DMMs in 3-D-HEVC, we develop an efficient intraprediction method according to the clustering results. The prediction method can make full use of the spatial structure characteristics of point clouds and remove the spatial redundancy.

E. Contour Map Coding

The segmentation is partly stored as a contour map. In this article, we use sliding windows to code the contour information with integers [27].

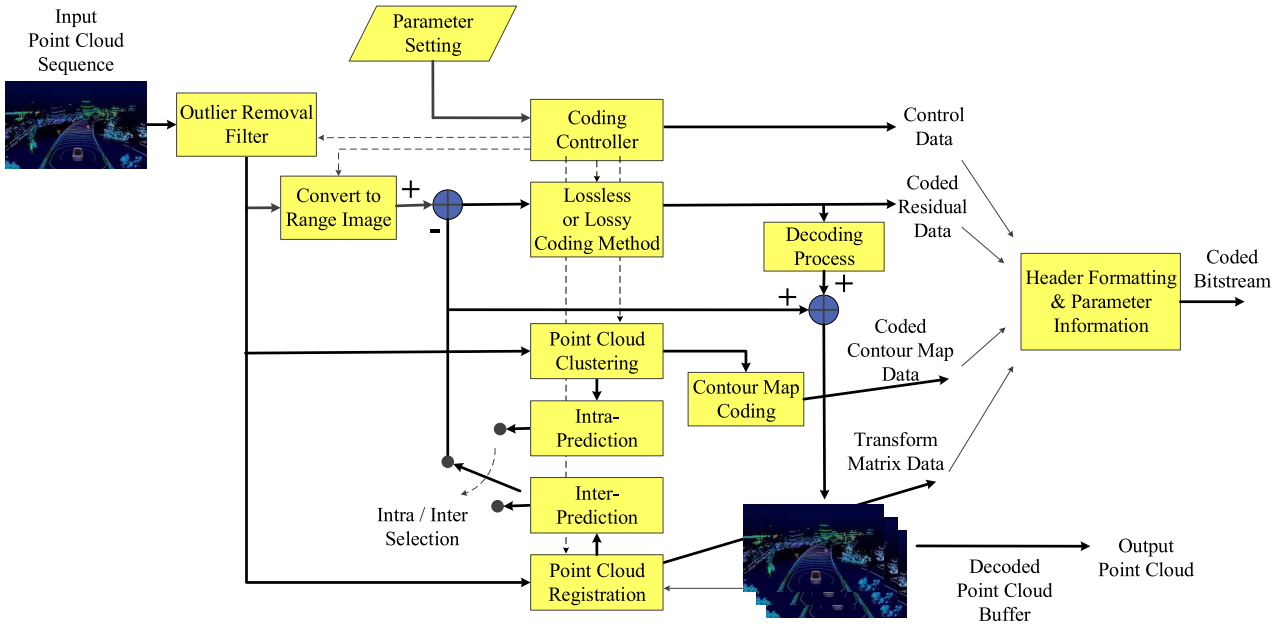


Fig. 1. Architecture of the proposed point cloud sequence compression scheme.

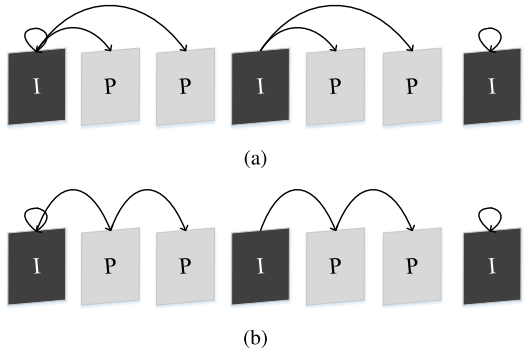


Fig. 2. Order of the intraframe and interframe. (a) Time priority. (b) Compress rate priority.

F. Interprediction

Temporally successive point cloud sequences share a wide range of similar structures. Motion estimation is a key to remove the temporal redundancies. An interprediction method is proposed based on point cloud registration.

G. Residual Data Coding Method

In order to obtain optimal coding performance, both the lossless and lossy schemes are explored to code the intraresidual and interresidual data.

H. Coder Controller

The coder uses a prespecified codec setting, which includes the parameter configuration for 1) and 3), the resolution ratio setting for 2), the registration technique for 6), and the coding method for 7).

I. Header Formatting and Parameter Information

The parameter information and intraencoded and interencoded data are organized in a predefined order and form the coded bitstream.

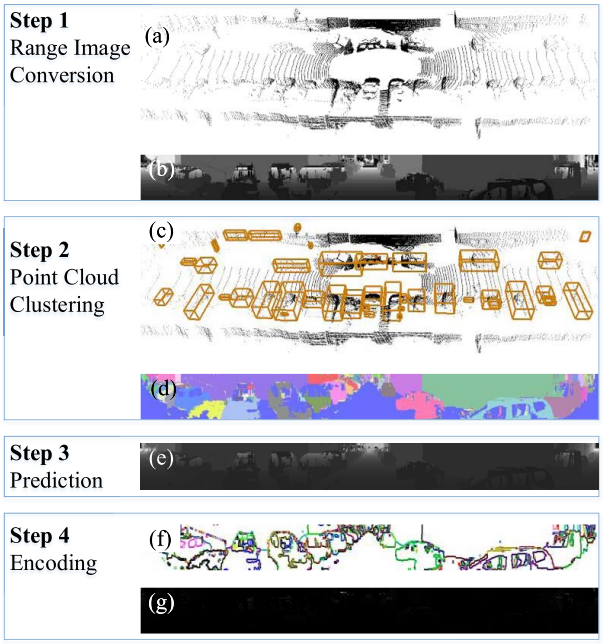


Fig. 3. Overall workflow of the intracoding method, best viewed in color. (a) Input point cloud from Velodyne HDL-64E. (b) We convert the point cloud to a range image. (c) Segmentation result is shown in the range image. (d) Segmentation result is shown in the point cloud, where each object is surrounded by a bounding box. (e) Prediction result is shown in the range image, where the object clusters are predicted using the average value and the ground is predicted using a plane. (f) We extract the contour map using the cluster result. (g) Difference between the real range image and the prediction result is calculated as the residual data.

IV. Intracoding Method

Inspired by the depth coding method in 3-D-HEVC, we propose an intraprediction technique for the point cloud. The intrapoint cloud compression method includes four main techniques: range image conversion, point cloud clustering, prediction, and residual data coding. Each technique will be detailed in this section. Fig. 3 shows the overall workflow

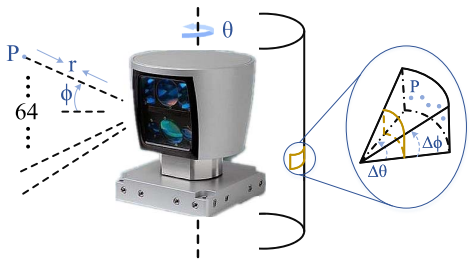


Fig. 4. Rotating LiDAR sensor, a Velodyne HDL-64E.

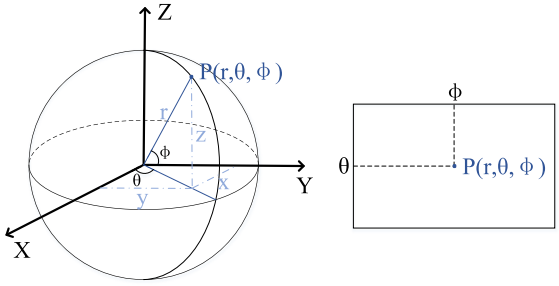


Fig. 5. Converting points from a spherical coordinate to a range image. Each point in the spherical coordinate system corresponds to a pixel in the range image.

of the proposed intracoding methodology. The first step is to covert the point cloud data into range images. In the second step, a clustering process is executed based on the range image to segment the point cloud into ground and main objects. The next step is the prediction of the range image based on the segmentation result. Finally, the contour map and residuals between the prediction and the real range image are computed and coded by either lossless or lossy coding methods.

A. Range Image Conversion

We use the publicly available KITTI dataset to evaluate the performance of our proposed algorithm. The KITTI dataset is captured by a Velodyne HDL-64E rotating 3-D laser scanner, as shown in Fig. 4. At each roll angle, the system emits laser pulses to measure the radial distances from the scanner center to objects. The 3-D geometry of points is represented using the spherical coordinate system. As shown in Fig. 5, a detected point P can be expressed by $P(r, \theta, \phi)$, where θ, ϕ, and r represent the polar angle, azimuthal angle, and radial distance, respectively. The corresponding Cartesian coordinates converted from the spherical coordinates can be calculated using Formula (2). The scanner covers 26.9° vertical and 360° horizontal fields of view with 64 beams and a 0.18° azimuthal angular resolution. We losslessly project the 3-D point cloud onto a spherical image by calculating the Euclidean distance per point. The results are shown in Fig. 3(a) and (b)

$$\begin{aligned} x &= r \cdot \cos\phi \cdot \cos\theta \\ y &= r \cdot \cos\phi \cdot \sin\theta \\ z &= r \cdot \sin\phi. \end{aligned} \tag{2}$$

B. Point Cloud Clustering

Due to the requirement for segmentation accuracy and real time, we use Bogoslavskyi's method to perform the point cloud clustering [28]. The clustering method utilizes the spatial geometric relations of objects, consisting of ground removing and object clustering [29].

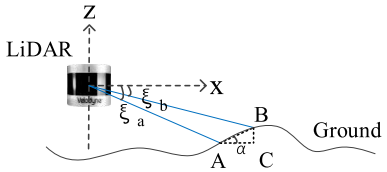


Fig. 6. Schematic of the ground extraction algorithm.

1) Ground Removing: The aforementioned range image is utilized to perform the point cloud clustering. First, we calculate the angle between two adjacent points in the vertical direction with the xoy plane, represented by α. Fig. 6 shows an illustration of the angle. Points A and B are derived from two neighboring rows $r-1$ and r of the range image, represented by $R_{r-1,c}$ and $R_{r,c}$, respectively. With *a priori* angle knowledge of vertically consecutive individual laser beams, the angle α can be calculated using trigonometric rules as follows:

$$\begin{aligned} \alpha &= \arctan(|\mathrm{BC}|, |\mathrm{ac}|) = \arctan(\Delta z, \Delta x) \\ \Delta z &= |R_{r-1,c}\sin\zeta_\alpha - R_{r,c}\sin\zeta_\beta| \\ \Delta x &= |R_{r-1,c}\cos\zeta_\alpha - R_{r,c}\cos\zeta_\beta| \end{aligned} \tag{3}$$

where ζ_α and ζ_β denote the vertical angles of the laser beams corresponding to rows $r-1$ and r, respectively.

As shown in Fig. 6, if points A and B belong to the ground, the angle α is very small. If points A and B do not belong to the ground, the angle α is particularly large. We define a threshold in advance to extract points belonging to the ground by comparing the α value with it. In the experiment, the threshold is set to 10°.

2) Object Clustering: After removing the ground points from the depth map, the remaining data are used for clustering. The key of the point cloud clustering algorithm is to determine which points come from the same object. The schematic of the clustering algorithm is shown in Fig. 7. Points A and B represent two random points in space, measured by the OA and OB laser beams emitted by the LiDAR. The angle between the laser beam OA and AB is indicated by β. As can be seen from the diagram, if two points, A and B, belong to the same object, then A and B are close, and the angle β is close to 90°. If two points, A and B, come from different objects, then A and B are farther away and the angle β is close to 0° or 180°. We can judge whether the two points belong to the same object according to the β value. β is defined as follows:

$$\beta = \arctan\frac{|\mathrm{BH}|}{|\mathrm{HA}|} = \arctan\frac{d_2\sin\alpha}{d_1 - d_2\cos\alpha} \tag{4}$$

where d_1 and d_2 represent the distance between OA and OB, respectively. Angle α is the angle between two beams and can be obtained by LiDAR sensor parameters. We define a threshold interval for angle β and determine whether points A and B belong to the same object by calculating. In the experiment, the threshold is set to 75°.

Through the above steps, we can get the clustering result, as shown in Fig. 3(c) and (d). The clustering method can

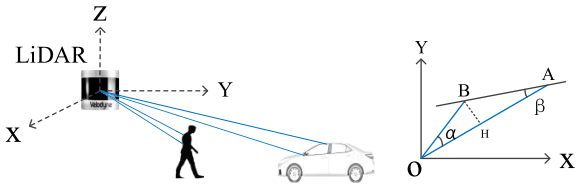


Fig. 7. Schematic of the clustering method. Left: example scene with two objects, a pedestrian and a car. Right: suppose that the LiDAR sensor is located at point O. Lines OA and OB represent two laser beams. Points A and B generate a line to estimate the surface of an object. Point H is the intersection point of the vertical line from point B to OA and OA. We judge whether points A and B belong to the same object according to the β angle. If $\beta > \theta$, where θ is a predefined threshold, we consider that the two points belong to the same object.

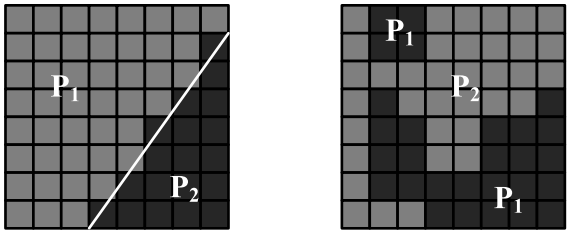


Fig. 8. DMMs. (a) Wedgelet partitioning. (b) Contour partitioning.

segment the ground and objects in the range image, which paves the way for the following intraprediction.

C. Intraprediction

1) Region Prediction Inspired by DMM: The proposed intraprediction method is inspired by the DMM technique adopted in 3-D-HEVC [30]. The 3-D-HEVC is designed for encoding multiview video and depth data that are captured by RGB-D sensors [31]. The range image of the LiDAR is similar to the depth image in the 3-D video; however, each point in the range image needs more bits to represent the distance.

The DMM consists of two partitioning techniques: wedgelet partitioning and contour partitioning. In a wedgelet partition, a straight line divides the depth block, resulting in two regions entirely connected, represented by P_1 and P_2, as shown in Fig. 8(a). In contrast, for a contour partition, the two regions can be shaped arbitrarily and may consist of multiple disconnected parts, as demonstrated by regions P_1 and P_2 in Fig. 8(b).

Each partition method divides a coding unit into two regions, P_1 and P_2, and uses a constant partition value (CPV) to predict each region. With the aim of obtaining the optimal partition mode, each possible segmentation pattern is evaluated using a rate–distortion optimization (RDO) metric [30]. The rate–distortion cost is expressed as follows:

$$\mathrm{RD_{cost}} = \mathrm{SSE} + \lambda \times \mathrm{Bit} \tag{5}$$

where SSE denotes the sum of the error between the original block and the reconstruction block, λ is the Lagrangian multiplier, and Bit represents the coding bit rate. The optimal partition mode and its corresponding index are stored on both sides of the encoder and the decoder.

Algorithm 1 Plane Fitting Algorithm With RANSAC

Require:
The 3-D point cloud: $point\_list$;
The tolerance threshold of distance t between the chosen plane and the other points;
The maximum probable number of points belonging to the same plane: $forseeable\_support$;

Ensure:
The best plane parameter: $bestPlane$.

$i = 0$; $bestSupport = 0$; $bestStd = \propto$
$\varepsilon = 1 - forseeable\_support/length(point\_list)$
$N = round(log(1-\alpha)/log(1-(1-\varepsilon)^3))$
while $i \leq N$ **do**
  j = pick 3 points randomly among $(point\_list)$
  $pl = pts2plane(j)$
  $dis = dis2plane(pl, point\_list)$
  $s = find(abs(dis) \leq t)$
  $st = Standard\_deviation(s)$
  if length(s) > $bestSupport$ or (length(s) = $bestSupport$ and $st < bestStd$) **then**
    $bestSupport = length(s)$
    $bestPlane = pl$
    $bestStd = st$
  end if
  $i = i + 1$
end while

Inspired by the DMM adopted in 3-D-HEVC, we develop a similar prediction technique for point cloud depth data. According to the clustering result, the range image is split into various segmentations, as shown in Fig. 3(d). Except for the ground points, each segment is predicted using the average value of all points belonging to the cluster. Therefore, we can obtain the predicted map for the object regions.

2) Ground Prediction With the RANSAC Method: For the ground points, we use random sample consensus (RANSAC) to fit an ideal plane [32]. In Cartesian coordinates, a plane can be expressed as follows:

$$d = \mathrm{ax} + \mathrm{by} + \mathrm{cz} \tag{6}$$

where (a, b, c) denotes the normal vector n and d represents the distance from the origin to the plane. RANSAC randomly selects three points from the dataset, calculates the parameters of the corresponding plane, and then tries to enlarge the plane according to the given threshold. The plane fitting algorithm with RANSAC is described in Algorithm 1.

According to the fitting plane and LiDAR parameters, the virtual ground points can be obtained. The difference between the real ground points and the virtual ground points will be calculated as the residual data

$$R_{\mathrm{residual\_ground}} = R_{\mathrm{real\_ground}} - P_{\mathrm{predict\_ground}} \tag{7}$$

where $R_{\mathrm{residual\_ground}}$ represents the residual data and $R_{\mathrm{real\_ground}}$ and $P_{\mathrm{predict\_ground}}$ represent the real and virtual ground points, respectively. The residual data and plane parameters will be encoded. The residual data value is nearly zero, and therefore,

we can use very few bits to encode them. During the decoding process, the ground points will be recovered.

3) Prediction Result: Through the above steps, we obtained the predict data for both the object regions and the ground regions. The difference between the true range data and predict data is calculated to obtain the residual data, as shown in Fig. 3(g). The pixel value of the residual data is nearly zero. Compared with the original image, the entropy of the residual image is smaller and needs fewer bits to encode. In this way, the spatial redundancy within a frame of the point cloud is removed.

D. Coding Method

1) Contour Map Encoding: As shown in Fig. 3(f) and (g), in order to reconstruct the depth image, we also need to encode the contour map. The contour map includes two important pieces of information: the contour shape and the value of each region. In order to encode the contour map, we extract the boundary pixels of it. The proposed compression scheme for the segment map works as follows.

a) Boundary Extraction and Coding: The boundary map is extracted first where a pixel (x, y) is 1 if the pixels in $(x + 1, y)$ or $(x, y + 1)$ belong to a different segment and 0 otherwise. After this, we divide the contour map into 4×4 pixel macroblocks. Each block is assigned an integer within a certain interval $[0, 2^n)$.

b) Per-Region Value Encoding: So far, we have concentrated on transforming the boundary of the segment image. In addition, each region's value is equally important. The average depth value a_k for each segmented region R_k is computed and stored in laser scanning order. The predicted value of each of the regions is then sorted on a 1-D array and an index is associated with each mask (denoted by i).

c) Exceptions: The entire contour map can be reconstructed by the information provided in the boundary encoding part. However, there are some single pixels between two borders, which will be treated as a border at the first step. Therefore, an array is built individually to store the location of each of these pixels and their respective value.

d) Coding Method: All of the intermediate value is lossless compressed using the arithmetic coder. Lossless coding is needed since the amount of data is not large and the masks contain the key information about edge positions.

e) Reconstruction: Reconstruction is the inverse process of coding. First, the boundary map is reconstructed according to the encoding data. Then, we fill the data for each segment.

2) Residual Data Compression Method: Several lossless encoding schemes are considered to compress the intrapredict residual data, which includes BZip2 [33], Brotli [34], LZMA [35], PPMd [36], LZ4 [37], Zstandard [38], LZ5 [39], Lizard [40], and Deflate [41]. In addition, we also explore the traditional lossy image coding methods to compress the residual data, such as JPEG [42] and JPEG2000 [7].

V. Interframe Coding Method

A schematic of the LiDAR coordinate system changing with time in the point cloud sequence is shown in Fig. 9. Each frame of the point cloud has its own coordinate system, and the LiDAR center is used as the origin of the coordinates. The time interval between the two adjacent point clouds is very short. In this period, the LiDAR moves only a small distance. As shown in Fig. 10(a) and (b), the adjacent frames in the point cloud sequence have a large range of similar structures. A lot of redundancies exist in the temporal dimensions. It is desirable to develop an interprediction method to remove the time redundancies within the point cloud sequence.

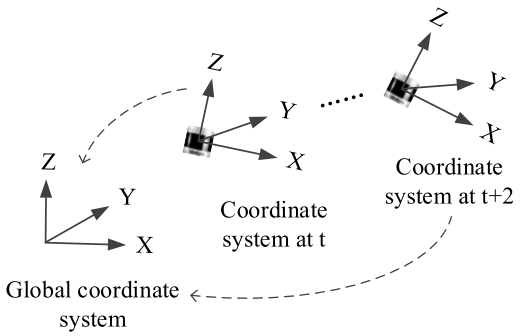


Fig. 9. Schematic of the LiDAR coordinate system changing with time.

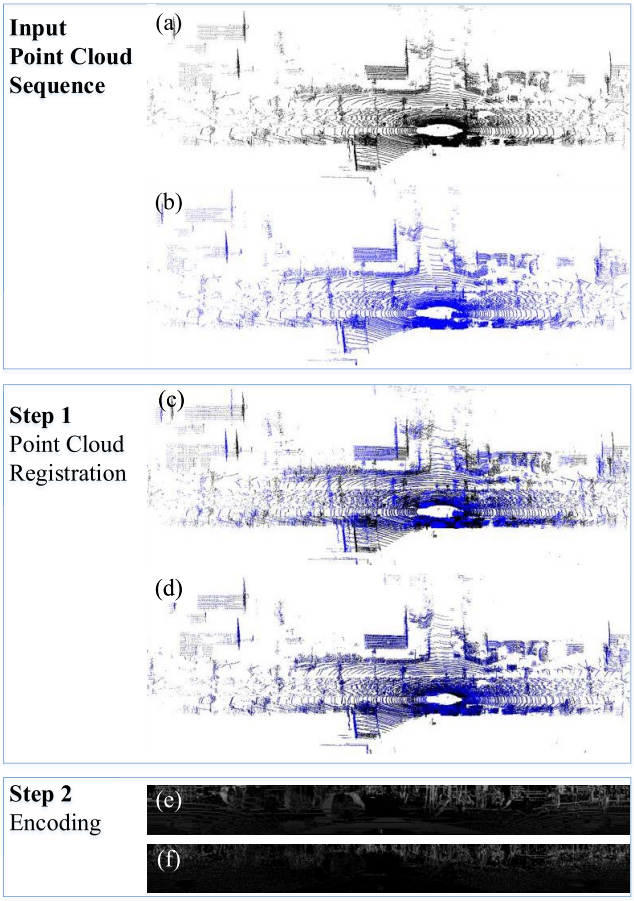


Fig. 10. Intercoding method. (a) Point cloud at $t - 1$. (b) Point cloud at t. (c) Point clouds at the same coordinate. (d) Point cloud after registration. (e) Residual map without snake prediction. (f) Residual map after snake prediction.

A. Limitations of Using the Video Interprediction Method

The interframe prediction method makes use of temporal redundancy by finding the motion vectors of different contents

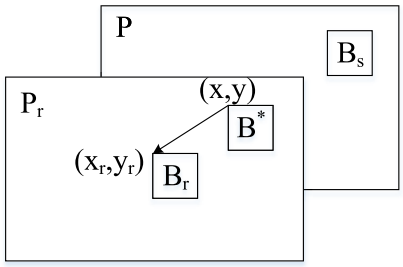


Fig. 11. Interprediction method in HEVC.

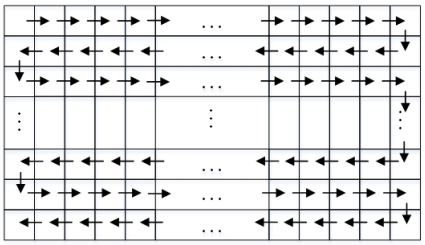


Fig. 12. Snake prediction for the residual data.

between frames. The motion vector is usually estimated per block, through a search of best matching between the source block B_s in current frame P and the reference block B_r in reference frame P_r, as shown in Fig. 11. The coordinates of B^* and B_s are the same locations in each frame image. The coordinates of B_r and B_s are (X_r, Y_r) and (X, Y), respectively. The motion vector (MV) is defined as $(X-X_r, Y-Y_r)$. Motion compensation is the process of obtaining the estimated value of the current frame according to the MV. We only need to encode residual values and MVs during the coding procedure.

Theoretically, the interframe prediction technology of video can be used for the depth map sequence. However, this method is performed based on the assumption that the color information of each pixel on the same object surface is close to that of others even though the object or camera is in motion. Therefore, this 2-D block-based motion estimation method is obviously unsuitable for the depth map converted from the point cloud as the pixel in the depth map represents the distance from the LiDAR to objects in the scene when the LiDAR or object is moving. It is inefficient to apply the color-based prediction method to the depth map representing distance information. The interprediction method for the point cloud sequence should utilize the spatial structure characteristics of point clouds.

B. Proposed Interprediction Method

Learning from the prediction concept designed for video, we propose a registration-based interprediction method for a point cloud sequence. Registration of two point clouds is to find the rotation and the translation that maximizes the overlap between the two clouds [43]. For instance, suppose that P_r and P_c are two sets of points. We aim to find the transformation T^* that minimizes the distances between the corresponding points in the two scenes

$$T^* = \operatorname{argmin} \sum_{C} \left(p_i^c - T \oplus p_j^r\right)^T \Omega_{\mathrm{ij}} \left(p_i^c - T \oplus p_j^r\right) \tag{8}$$

where T represents the current estimate of the transformation that maps P_r in the reference frame of P_c, c and Ω_{ij} are information matrix that consider the noise statistics of the sensor, C is a set of correspondences between points in the two clouds, and $\oplus$ is the standard composition operator that applies the transformation T to the point P_c.

In this article, two adjacent point clouds are matched by using the iterative closest point (ICP) algorithm. Thus, the rotation and translation matrix can be obtained. The two frames of the point cloud are transformed into the same coordinate system using (8)

$$\begin{bmatrix} x_g \\ y_g \\ z_g \end{bmatrix} = R_{\mathrm{yaw}} \times R_{\mathrm{pitch}} \times R_{\mathrm{roll}} \times \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix} + \begin{bmatrix} C_x \\ C_y \\ C_z \end{bmatrix} \tag{9}$$

where (x, y, z) represents the coordinates after transformation, (x, y, z) denotes current coordinates, C_x, C_y, and C_z denote the translation matrix of the coordinates, and R_{yaw}, R_{pitch}, and R_{roll} represent the rotation matrix of the yaw, pitch, and roll angle, repsectively.

When the two coordinates of the point cloud data are unified, the difference between the real point cloud P_{true} and the predicted one P_{predict} is calculated as residual data P_{residual}

$$P_{\mathrm{residual}} = P_{\mathrm{true}} - P_{\mathrm{predict}}. \tag{10}$$

As the predicted data are very close to the real data, the residual data value is nearly zero, as shown in Fig. 10(e). We only need to encode the residual data with very few bits. As a result, the temporal redundancy is removed.

C. Snake-Like Prediction for Interresidual Data

In order to further remove the redundancy of the residual data, a snake-like prediction scheme is performed, which is operated by comparing the current depth pixel with its reference pixel, as shown in Fig. 12. If a pixel is at the end of a row, it is regarded as the reference pixel of the pixel in the same column in the following. The difference between the value of the pixel and its reference pixel is obtained for further encoding, as shown in Fig. 10(f).

In the coding process, we only need to encode the rotation matrix, translation matrix, and residual data of two point clouds. Similar to the processing for intraresidual data, we perform both lossless and loss compression schemes for the interresidual data.

VI. Experimental Results

A. Experimental Conditions and Evaluation Metrics

To verify the performance of the proposed algorithm, we implement it in C++ with some operations in the point cloud library (PCL). The experiments are performed on a desktop computer with an i5-6300HQ 2.3 GHz CPU with a 4-GB memory. Four representative point cloud sequences provided by the KITTI dataset are chosen for testing. The four sequences consist of four scenes: campus, city, road, and residential. The experiments test 100 frames for each

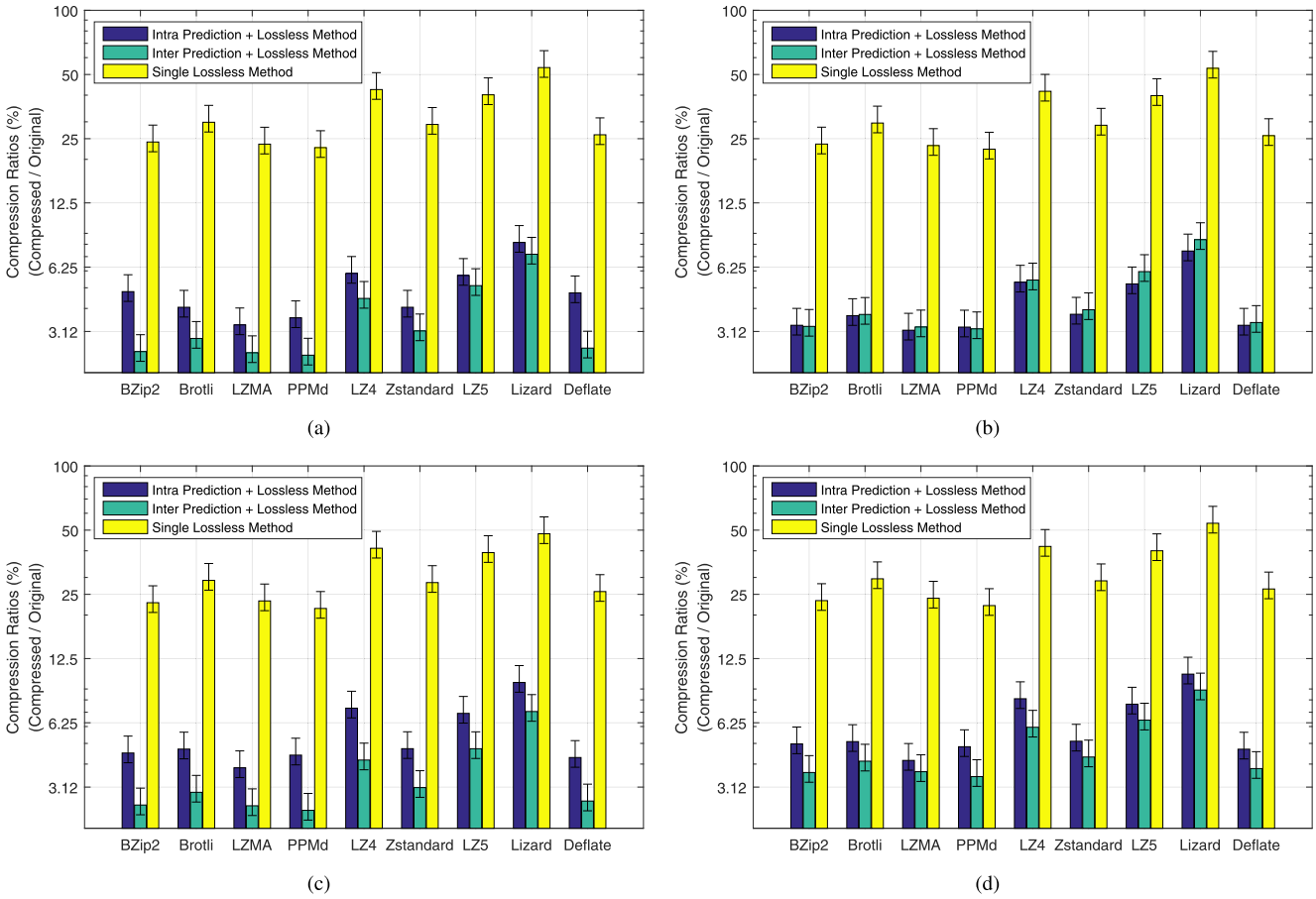


Fig. 13. Compression ratios of different lossless compression methods. (a) Campus. (b) City. (c) Road. (d) Residential.

sequence. In particular, in the intercoding process, we use the former frame of the current point cloud as the reference frame.

Coding efficiency is measured by compression rate and root-mean-square error (RMSE). Compression rate presents the ratio between the compressed data size and the original size. The lower the value, the better the performance

$$\text{Ratio} = \frac{\text{Compressed}_{\text{size}}}{\text{Original}_{\text{size}}} \times 100\% \tag{11}$$

where Ratio represents the compression rate and $\text{Original}_{\text{size}}$ and $\text{Compressed}_{\text{size}}$ denote the sizes of the point cloud data before compression and after compression, respectively.

RMSE represents the root-mean-squared error between the original 3-D points and the reconstructed ones, which reflects the quality of reconstruction. RMSE is only used to evaluate the lossy coding performance. If the reconstruction point cloud is closer to the original point cloud, the value of the RMSE will be smaller. Given a $m \times n$ range image corresponding to a point cloud, the RMSE is defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2} \tag{12}$$

where I and K represent the original range image and reconstruction range image, respectively.

B. Lossless Compression Results

1) Comparison With Lossless Compression Methods: To verify the efficiency of the intraprediction and interprediction techniques on eliminating the time and space redundancy, we use several lossless coding schemes to encode the intraprediction and interprediction residual data, including BZip2 [33], Brotli [34], LZMA [35], PPMd [36], LZ4 [37], Zstandard [38], LZ5 [39], Lizard [40], and Deflate [41]. For comparison, the range image converted from the LiDAR point clouds is also directly encoded with these lossless coding schemes without performing intraprediction or interprediction. A compression rate is utilized to evaluate the coding performance. Experimental results are shown in Fig. 13. It can be observed that compared with stand-alone lossless methods, the combination of the proposed prediction methods with lossless coding schemes achieves a smaller compression ratio. In our method, the smaller the compression ratio, the better the performance.

For intracoding results, the smallest compression ratio is 3.16%, achieved by the combination of the proposed method with the LZMA scheme for the point cloud of the city scene. The worst compression ratio is 10.56% with the combination of the proposed method with the Lizard scheme. The structure of the point cloud data in city scenes is single, which contains a large number of ground points. The structure of the contour map is simple. As a result, we can use a few bits to code the contour map. On the contrary, the point cloud of the

TABLE I
COMPRESSION RATE RESULTS COMPARED WITH THE OCTREE METHOD

Scene	Lossless Coding Method		Octree method		
	Intra-coding	Inter-coding	Distance Resolution		
			$1mm^3$	$5mm^3$	$1cm^3$
Campus	3.35	2.47	20.73	12.95	10.53
City	3.16	3.27	20.14	12.70	10.28
Road	3.84	2.54	19.69	12.01	9.60
Residiential	4.16	3.68	20.35	12.57	10.15
Average	**3.63**	**2.99**	**20.23**	**12.56**	**10.14**

residential scene is complex and the compression ratio is much higher. Lossless coding methods are designed by removing the coding redundancies of data without considering the characteristics of the point cloud structure; therefore, the compression rates for the point clouds of different scenes show very little difference. The above experimental results indicate that the clustering-based intraprediction algorithm can effectively remove the spatial redundancy of the point cloud data.

For intercoding results, it can be seen that the combination of the interprediction method with the PPMd algorithm has the smallest compression ratio of 2.41%. The worst compression ratio is 8.84% for the residential scene achieved by the combination of the proposed method with the Lizard scheme. Experiment results indicate that the interprediction method based on point cloud registration can effectively remove the time redundancy within the point cloud sequence. Compared with the intraframe coding algorithm, the interframe coding algorithm achieves better performance for the campus, person, and residential scene point clouds. As the point cloud data of two adjacent frames are similar, the residual data after interprediction are smaller. As a result, we can use fewer bits to encode the residual data. A special case is that for the city scene point cloud data, the compression ratio is higher compared to the intracoding method. The reason is that in city scenarios, cars equipped with LiDAR travel faster, and the structural differences between adjacent frames of the point cloud are larger, which results in a decrease in compression performance.

2) Comparion With Octree Methods: Table I shows the compression ratio results of the proposed lossless compression method compared with the octree method [21]. The encoding precision of the octree method is set to 1 mm^3, 5 mm^3, and 1 cm^3. For the proposed method, we choose the LZMA algorithm as a representative of the lossless compression algorithm. It can be observed that both the proposed intracoding and intercoding methods obtain a smaller compression rate than the octree method. Using the intracoding method, the size of the point cloud is compressed to 3.63% of its original size, whereas the intercoding method can reduce the size of the point cloud to 2.99% of its original size, on average. More importantly, the proposed method is lossless.

C. Lossy Coding Method

1) Recurrent Data Coding Method: Besides the lossless compression methods, traditional lossy image compression methods, such as JPEG and JPEG2000, are also explored to compress the residual data. As residual data are recorded in floating-point numbers, the traditional techniques can encode only integer pixel values. In order to decrease the distance error, we try to use a recurrent data coding method that treats the error after each step as the new input and applies the JPEG or JPEG2000 coding method again to compress the error. This idea is shown in Fig. 14. This approach introduces a tradeoff between distance error and compression efficiency. In our scheme, the data are encoded twice.

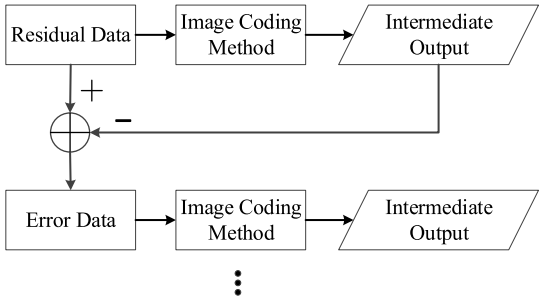


Fig. 14. Recurrent coding method.

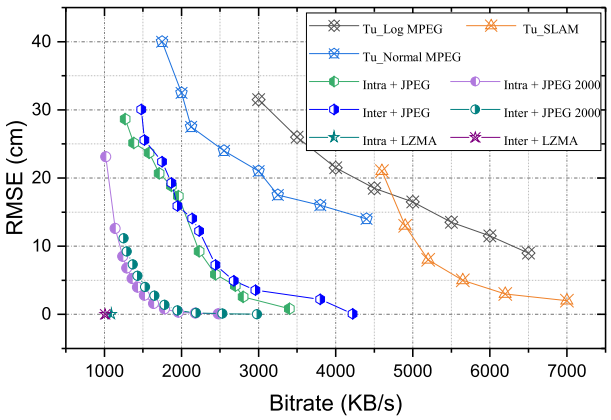
2) Comparison With Other Methods: For comparison, the original point cloud data are also compressed directly with the lossy methods without any processing. In addition, we also compare our experimental results with image- [7] and SLAM-based [9] point cloud compression methods. The two methods are also designed for vehicle point cloud compression. The experimental results are given in terms of RMSE–Bitrate curves of two scenarios of point cloud data. RMSE–Bitrate curves reflect the relationships between bit rate and RMSE. The smaller the bitrate and RMSE, the better the coding performance as it achieves a lower RMSE with a smaller bandwidth simultaneously.

Fig. 15 shows the results of our algorithm compared with other methods. It can be observed that both the intraprediction and interprediction methods have outstanding advantages in bitrate and RMSE compared to the curves without any processing. In addition, the proposed intracoding and intercoding methods also show better performance compared with Tu's methods [10], [12]. This is Tu's methods seldom perform preprocessing for the point cloud data, and there are still redundancies with the point cloud. Our proposed method needs a much lower bitrate in a small RMSE situation. Because JPEG2000 works better than JPEG, the combination of the proposed method with JPEG2000 has a better performance compared with the JPEG method.

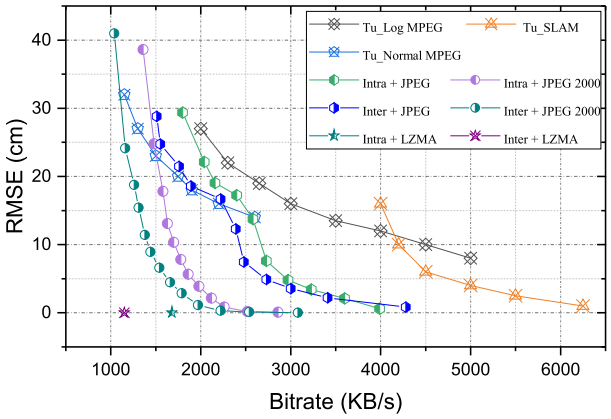
In order to make an intuitionistic performance comparison with the lossless method, we also plot the best lossless compression result in Fig. 15, shown with an asterisk. As the RMSE of the lossless method is zero, the cure of the lossless method is only a point. As we can see, compared with the JPEG2000 method, the lossless method also shows strong competitiveness. The above experimental results indicate that the proposed prediction method can largely remove spatial and temporal data redundancy within the point cloud sequence, achieving a high compression efficiency either by a lossless or lossy coding method.

D. Contribution of Each Step for Compression Efficiency

With the aim of evaluating the contribution of each step in the intracoding and intercoding process, we record the



(a)



(b)

Fig. 15. Comparison of RMSE–Bitrate curves of two scenarios with Tu's methods. (a) City scene. (b) Residential scene.

average changes of the data volumes of a frame of the point cloud during the compression. LZMA and JPEG2000 are selected as the representatives to encode the residual data. For JPEG2000, we set the parameter compression ratio as 2, which means that the size of the compressed data will be half of its original size. The average changes of frame data volumes during the intracoding and intercoding process are shown in Tables II and III, respectively.

As we can see, in the intracoding process, after the conversion, the data size is reduced to nearly a quarter of its original size. The intraprediction efficiency is related to the complexity of the point cloud. The data volume can be reduced to half of the point cloud with simple structural complexity, such as with the city, and campus scenes. The last contribution comes from the coding part.

In the intercoding process, after interprediction, the scenes of the campus and person point clouds get less data volume, with 276 and 279 kB, respectively. On the contrary, the data volume for the city and residential point clouds is 376 and 362 kB, on average, respectively, after interprediction. This is because of the slow driving speed in the campus and people scenes when recording the point cloud data. Thus, we get less data after interprediction. The point cloud data for city scenarios are just the opposite. A fast driving speed reduces the structural similarity of adjacent point clouds, which results in the volume of data being large after interprediction.

TABLE II

AVERAGE CHANGES IN DATA VOLUMES DURING INTRACODING (KB)

Scene	Original Size	After Conversion	After Intra prediction	After coding	
				LZMA	JPEG2000
Campus	3208	835	449	105	201
City	3149	837	413	98	178
Road	3331	822	513	125	240
Residential	3222	823	537	132	212

TABLE III

AVERAGE CHANGES IN DATA VOLUMES DURING INTERCODING (KB)

Scene	Original Size	After Registration & Inter prediction	After coding	
			LZMA	JPEG2000
Campus	3208	276	78	84
City	3149	376	101	159
Road	3331	279	83	87
Residential	3222	362	116	179

TABLE IV

AVERAGE CODING TIME OF INTRACODING METHOD (S)

Coding method	Clustering	Intra prediction	Coding	Total
Intra & LZMA	0.17	0.07	0.33	0.57
Intra & JPEG2000	0.17	0.07	0.14	0.38

TABLE V

AVERAGE CODING TIME OF INTERCODING METHOD (S)

Coding method	Registration	Inter prediction	Coding	Total
Inter & LZMA	26.47	0.22	0.33	27.12
Inter & JPEG2000	26.47	0.22	0.15	26.84

It is noteworthy that, compared with the lossless method, the lossy JPEG2000 method shows few advantages or performs worse. This is because JPEG2000 is specially designed for image compression by removing the redundancies of the color information. However, the residual data are based on distance information. Moreover, the proposed prediction step has already removed the redundancy of the point cloud.

The steps of conversion, intraprediction or interprediction, and coding are all important techniques of the proposed intracoding or intercoding methods. Their contributions are not simply added together, rather their effects are multiplied. Thus, even a small improvement in a single step can make a big difference.

E. Speed Performance

The proposed intracoding method consists of four steps, namely, range image conversion, point cloud segmentation, intraprediction, and coding. The intercoding method includes point cloud registration, interprediction, and coding steps. We test the average speed performance of the proposed method with 100 frames. The average coding time of each step for the intracoding and intercoding process is presented in Tables IV and V, respectively. As we can see, the total intracoding time is 0.57 s for the lossless method (LZMA) and 0.38 s for the lossy method (JPEG2000). With hardware acceleration, the intracoding algorithm is expected to be used for real-time point cloud compression. Compared to intracoding, intercoding requires a lot of coding time, which is caused by the high complexity of the ICP algorithm.

TABLE VI
ENCODING AND DECODING TIME OF THE PROPOSED METHOD VERSUS OCTREE, TU-MPEG, AND TU-SLAM (S)

Time	Intra-frame	Inter-frame	Octree	Tu-MPEG	Tu-SLAM
Encoding	0.57	27.12	0.024	0.50	0.10
Decoding	0.02	0.04	0.009	0.02	0.03

Table VI shows the average encoding and decoding time of the proposed method compared with the octree, Tu-MPEG [10], and Tu-SLAM [12] methods. It can be seen that the octree and have low complexity. Compared with the Tu-MPEG algorithm, our intracoding method has similar algorithm complexity. The high complexity of the intercoding method is caused by the ICP point cloud registration method, which needs to be further optimized to reduce the complexity.

Currently, the algorithm cannot be run in real time, but the algorithm framework can be used offline. After the LiDAR data have collected, we can use this algorithm to encode the data to reduce the storage space. After compression, when transmitting these data to others through the Internet, the bandwidth will also be reduced. In future work, we will continue to optimize the point cloud coding architecture to improve the coding efficiency and reduce the complexity. We aim to find a balance between the algorithm complexity and the compression rate.

VII. CONCLUSION AND DISCUSSION

The huge and the increasing volume of point cloud data could prove an important bottleneck for transmission and storage, especially in tasks such as autonomous driving. In this article, we extended some of the concepts used in HEVC and 3-D-HEVC and introduced a novel coding scheme to compress point cloud sequences. The coding framework consists of intraprediction, interprediction, and residual data coding. Because the point cloud data of vehicle-mounted LiDAR are orderly, we transformed the point cloud data into a depth map. Different from texture video, a depth map is mainly characterized by sharp object edges and large homogeneous regions with nearly constant values. Inspired by DMM in 3-D-HEVC, we exploited a clustering-based intraprediction method to remove the spatial redundancy of the point cloud data. In the intercoding, the ICP algorithm is utilized to perform point cloud registration and obtain the rotation matrix and the translation matrix. By transforming two point clouds into the same coordinate system, this method can largely eliminate the temporal redundancy of the point cloud sequence. Moreover, both lossless and lossy methods were explored to encode the intraresidual and interresidual data. Experimental results demonstrated that the proposed method outperforms the other methods. Using the intracoding method, the size of the point cloud is compressed to 3.63% of its original size, whereas the intercoding method can reduce the size of the point cloud to 2.99% of its original, on average. It owes its performance to several novel techniques, such as clustering-based intraprediciton, registration-based interprediction, and snake prediction. More importantly, the improvements of a single technique for the compression performance in the intracoding or intercoding process are not simply added together, they are instead multiplied. To the best of our knowledge, our method is the first to extend the DMM technique in 3-D-HEVC and HEVC coding architecture to point cloud sequence compression. It utilizes both the advantages of the video coding algorithm and geometrical characteristics of the point clouds. Future study will concentrate on dense 3-D point cloud maps compression.

ACKNOWLEDGMENT

The authors would like to thank the editors and anonymous reviewers for the valuable comments.

REFERENCES

[1] K. Lenac, A. Kitanov, R. Cupec, and I. Petrović, "Fast planar surface 3D SLAM using LIDAR," *Robot. Auto. Syst.*, vol. 92, pp. 197–220, Jun. 2017.
[2] M. Liu, "Robotic online path planning on point cloud," *IEEE Trans. Cybern.*, vol. 46, no. 5, pp. 1217–1228, May 2016.
[3] H. Ren, W. Liu, and A. Lim, "Marker-based surgical instrument tracking using dual Kinect sensors," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 921–924, Jul. 2014.
[4] M. Weinmann, B. Jutzi, S. Hinz, and C. Mallet, "Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers," *ISPRS J. Photogramm. Remote Sens.*, vol. 105, pp. 286–304, Jul. 2015.
[5] D. Kim, S. Shin, and I. S. Kweon, "On-line initialization and extrinsic calibration of an inertial navigation system with a relative preintegration method on manifold," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 3, pp. 1272–1285, Jul. 2018.
[6] H. Ren *et al.*, "Computer-assisted transoral surgery with flexible robotics and navigation technologies: A review of recent progress and research challenges," *Crit. Rev. Biomed. Eng.*, vol. 41, nos. 4–5, pp. 365–391, 2013.
[7] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek, "An overview of JPEG-2000," in *Proc. DCC. Data Compress. Conf.*, 2000, pp. 523–541.
[8] M. Wang, K. N. Ngan, and H. Li, "An efficient frame-content based intra frame rate control for high efficiency video coding," *IEEE Signal Process. Lett.*, vol. 22, no. 7, pp. 896–900, Jul. 2015.
[9] J. Elseberg, D. Borrmann, and A. Nüchter, "One billion points in the cloud—An octree for efficient processing of 3D laser scans," *ISPRS J. Photogramm. Remote Sens.*, vol. 76, pp. 76–88, Feb. 2013.
[10] C. Tu, E. Takeuchi, C. Miyajima, and K. Takeda, "Compressing continuous point cloud data using image compression methods," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 1712–1719.
[11] X. Liu, Y. Wang, Q. Hu, and D. Yu, "A scan-line-based data compression approach for point clouds: Lossless and effective," in *Proc. 4th Int. Workshop Earth Observ. Remote Sens. Appl. (EORSA)*, Jul. 2016, pp. 270–274.
[12] C. Tu, E. Takeuchi, C. Miyajima, and K. Takeda, "Continuous point cloud data compression using SLAM based prediction," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1744–1751.
[13] H. Houshiar and A. Nüchter, "3D point cloud compression using conventional image compression for efficient data transmission," in *Proc. XXV Int. Conf. Inf., Commun. Autom. Technol. (ICAT)*, Oct. 2015, pp. 1–8.
[14] J.-K. Ahn, K.-Y. Lee, J.-Y. Sim, and C.-S. Kim, "Large-scale 3D point cloud compression using adaptive radial distance prediction in hybrid coordinate domains," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 3, pp. 422–434, Apr. 2015.
[15] X. Wang, Y. A. Sekercioglu, T. Drummond, E. Natalizio, I. Fantoni, and V. Fremont, "Fast depth video compression for mobile RGB-D sensors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 4, pp. 673–686, Apr. 2016.
[16] J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 778–785.
[17] Y. Fan, Y. Huang, and J. Peng, "Point cloud compression based on hierarchical point clustering," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, Oct. 2013, pp. 1–7.

[18] R. A. Cohen, D. Tian, and A. Vetro, "Point cloud attribute compression using 3-D intra prediction and shape-adaptive transforms," in *Proc. Data Compress. Conf. (DCC)*, Mar. 2016, pp. 141–150.

[19] V. Morell, S. Orts, M. Cazorla, and J. Garcia-Rodriguez, "Geometric 3D point cloud compression," *Pattern Recognit. Lett.*, vol. 50, pp. 55–62, Dec. 2014.

[20] J. Navarrete, D. Viejo, and M. Cazorla, "Compression and registration of 3D point clouds using GMMs," *Pattern Recognit. Lett.*, vol. 110, pp. 8–15, Jul. 2018.

[21] R. Mekuria, K. Blom, and P. Cesar, "Design, implementation, and evaluation of a point cloud codec for tele-immersive video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 4, pp. 828–842, Apr. 2017.

[22] D. Thanou, P. A. Chou, and P. Frossard, "Graph-based compression of dynamic 3D point cloud sequences," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1765–1778, Apr. 2016.

[23] R. L. de Queiroz and P. A. Chou, "Transform coding for point clouds using a Gaussian process model," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3507–3517, Jul. 2017.

[24] M. Wang, J. Xiong, L. Xu, W. Xie, K. N. Ngan, and J. Qin, "Rate constrained multiple-QP optimization for HEVC," *IEEE Trans. Multimedia*, vol. 22, no. 6, pp. 1395–1406, Jun. 2020.

[25] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 1–4.

[26] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

[27] B. Matejek *et al.*, "Compresso: Efficient compression of segmentation data for connectomics," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2017.

[28] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3D laser scans for online operation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 163–169.

[29] I. Bogoslavskyi and C. Stachniss, "Efficient online segmentation for sparse 3D laser scans," *PFG—J. Photogramm., Remote Sens. Geoinf. Sci.*, vol. 85, no. 1, pp. 41–52, Feb. 2017.

[30] G. J. Sullivan, J. M. Boyce, Y. Chen, J.-R. Ohm, C. A. Segall, and A. Vetro, "Standardized extensions of high efficiency video coding (HEVC)," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 1001–1016, Dec. 2013.

[31] G. Tech, Y. Chen, K. Muller, J.-R. Ohm, A. Vetro, and Y.-K. Wang, "Overview of the multiview and 3D extensions of high efficiency video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 35–49, Jan. 2016.

[32] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," *Comput. Graph. Forum*, vol. 26, no. 2, pp. 214–226, Jun. 2007.

[33] J. Gilchrist, "Parallel data compression with bzip2," in *Proc. 16th IASTED Int. Conf. Parallel Distrib. Comput. Syst.*, vol. 16, 2004, pp. 559–564.

[34] J. Alakuijala *et al.*, "Brotli: A general-purpose data compressor," *ACM Trans. Inf. Syst.*, vol. 37, no. 1, pp. 1–30, 2018.

[35] Z. Bin Tariq, N. Arshad, and M. Nabeel, "Enhanced LZMA and BZIP2 for improved energy data compression," in *Proc. 4th Int. Conf. Smart Cities Green ICT Syst.*, 2015, pp. 1–8.

[36] F. Behr, V. Fossum, M. Mitzenmacher, and D. Xiao, "Estimating and comparing entropies across written natural languages using PPM compression," in *Proc. Data Compress. Conf. (DCC)*, 2003, p. 416.

[37] W. Liu, F. Mei, C. Wang, M. O'Neill, and E. E. Swartzlander, "Data compression device based on modified LZ4 algorithm," *IEEE Trans. Consum. Electron.*, vol. 64, no. 1, pp. 110–117, Feb. 2018.

[38] X. Delaunay, A. Courtois, and F. Gouillon, "Evaluation of lossless and lossy algorithms for the compression of scientific datasets in netCDF-4 or HDF5 files," *Geoscientific Model Develop.*, vol. 12, no. 9, pp. 4099–4113, Sep. 2019.

[39] M. Mahoney. (2006). *Rationale for a Large Text Compression Benchmark*. Accessed: Aug. 20, 2006. [Online]. Available: http://cs.fit.edu/mmahoney/compression/rationale.html

[40] R. A. McLeod, R. D. Righetto, A. Stewart, and H. Stahlberg, "MRCZ—A file format for cryo-TEM data with fast compression," *J. Struct. Biol.*, vol. 201, no. 3, pp. 252–257, 2018.

[41] K. Ma and W. W.-W. Chen, "Method and apparatus for efficient hardware based deflate," U.S. Patent 7 307 552, Dec. 11, 2007.

[42] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Trans. Consum. Electron.*, vol. 38, no. 1, pp. xviii–xxxiv, Feb. 1992.

[43] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," *Proc. SPIE*, vol. 1611, pp. 586–606, Apr. 1992.

Xuebin Sun received the bachelor's degree from the Tianjin University of Technology, Tianjin, China, in 2011, and the master's and Ph.D. degrees from Tianjin University, Tianjin, in 2014 and 2018, respectively.

He is currently a Post-Doctoral Research Fellow with the Guangdong Key Laboratory of Intelligent Information Processing, College of Electrical and Information Engineering, Shenzhen University, Shenzhen, China. His research interests focus on video coding optimization, digital image processing, deep learning, and point cloud processing algorithms.

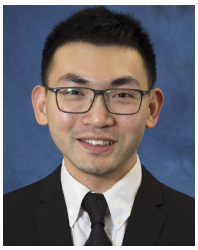
Yuxiang Sun (Member, IEEE) received the bachelor's degree from the Hefei University of Technology, Hefei, China, in 2009, the master's degree from the University of Science and Technology of China, Hefei, in 2012, and the Ph.D. degree from The Chinese University of Hong Kong, Hong Kong, in 2017.

He was a Research Associate with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong. He is currently a Research Assistant Professor with the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Hong Kong. His current research interests include autonomous driving, deep learning, robotics and autonomous systems, and semantic scene understanding.

Dr. Sun is a recipient of the Best Paper in Robotics Award at IEEE-ROBIO 2019 and the Best Student Paper Finalist Award at IEEE-ROBIO 2015.

Weixun Zuo received the bachelor's degree from Anhui University, Hefei, China, in 2016, and the master's degree from The Hong Kong University of Science and Technology, Hong Kong, in 2017.

He was a Research Assistant with the Robotics Institute, Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong. He is currently with Shenzhen Unity Drive Innovation Technology Company Ltd., Shenzhen, Guangdong, China. His current research interests include mobile robots, semantic segmentation, deep learning, and autonomous vehicles.



Shing Shin Cheng received the B.S. degree in mechanical engineering from Johns Hopkins University, Baltimore, MD, USA, in 2013, the M.S. degree in mechanical engineering from the University of Maryland, College Park, MD, in 2016, and the Ph.D. degree in robotics from the Georgia Institute of Technology, Atlanta, GA, USA, in 2018.

He is currently an Assistant Professor with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong. His work has been published in leading robotics and materials journals and conferences. His research interests include medical robotics, image guidance and navigation, and smart sensors and actuators.

Ming Liu (Senior Member, IEEE) received the B.A. degree in automation from Tongji University, Shanghai, China, in 2005. He is currently pursuing the Ph.D. degree with the Department of Mechanical and Process Engineering, ETH Zürich, Zurich, Switzerland, in 2013, under the supervision of Prof. R. Siegwart.

He is currently with the ECE Department, CSE Department, and Robotics Institute of Hong Kong University of Science and Technology, Hong Kong. His research interests include dynamic environment modeling, deep learning for robotics, 3-D mapping, machine learning, and visual control.

Dr. Liu won twice the Innoviation Contest Chunhui Cup Winning Award in 2012 and 2013. He won the Wu Weijun AI Award in 2016. He was the Program Chair of the IEEE-RCAR 2016, and the Program Chair of the International Robotics Conference in Foshan 2017. He is the Conference Chair of ICVS 2017.