IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 13, 2020, accepted July 25, 2020, date of publication July 29, 2020, date of current version August 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3012613

Convolution on Rotation-Invariant and Multi-Scale Feature Graph for 3D Point Set Segmentation

TAKAHIKO FURUYA[1], XU HANG[2], RYUTAROU OHBUCHI[1], (Member, IEEE), AND JINLIANG YAO[2]

[1]Department of Computer Science and Engineering, University of Yamanashi, Kofu 400-8511, Japan
[2]School of Computer Science, Hangzhou Dianzi University, Hangzhou 310000, China

Corresponding author: Takahiko Furuya (takahikof@yamanashi.ac.jp)

ABSTRACT Invariance against rotation of 3D objects is one of the essential properties for 3D shape analysis. Recently proposed algorithms have achieved rotationally invariant 3D point set analysis by using inherently rotation-invariant 3D shape features, i.e., distances and angles among 3D points, as input to Deep Neural Networks (DNNs). The DNNs capture spatial hierarchy and context among the geometric features to produce accurate analytical results. In this article, we delve further into the DNN-based approach to rotation-invariant and highly accurate 3D point set analysis. In particular, we focus our attention on segmentation of 3D point sets, which is one of the most challenging among 3D point set analysis tasks. We propose a novel DNN for 3D point set segmentation called *Rotation-invariant and Multi-scale feature Graph convolutional neural network*, or *RMGnet*. Our RMGnet is more flexible than the previous methods as it accepts as input any handcrafted 3D shape features having rotation invariance. In addition, to accurately segment 3D point sets composed of parts having various sizes, we randomize scales at which handcrafted features are extracted and perform multi-resolution analysis of the features by using the DNN. Experimental evaluation demonstrates high segmentation accuracy as well as rotation invariance of the proposed RMGnet.

INDEX TERMS 3D point set, 3D shape analysis, computer vision, deep learning, segmentation.

I. INTRODUCTION

Technology for 3D shape analysis has made a remarkable progress due in large part to advances in deep learning techniques and prevalence of 3D shape acquisition devices. In particular, many recent studies [1] focus on analyzing 3D shapes represented as 3D point sets for their comparison, classification, segmentation, or generation. PointNet [2] and its descendants [3]–[7] employ Deep Neural Networks (DNNs) that process raw 3D point set data in an end-to-end manner. These end-to-end DNNs for 3D point set take as their input 3D coordinates of the points and extract a semantic feature from the input for accurate 3D shape analysis.

However, these DNNs have a significant weakness; they do not have invariance against rotation of 3D point sets since coordinates of the 3D points co-vary with rotation in a 3D space. Invariance against 3D rotation is a property essential to many scenarios of 3D shape processing. For example, recognizing real-world 3D objects regardless of their orientations is required for reliable and secure autonomous vehicles and robots. Or, 3D CAD models often have different orientations due to varying definitions of coordinate axes among 3D CAD software. Rotation-invariance is necessary to accurately analyze such 3D shape models having various orientations.

Recently, deep learning-based 3D point set analysis algorithms that are invariant against 3D rotation have been proposed [8]–[13]. Most of these algorithms employ low-level geometric features that are not affected by rotation, for example, histogram of distances and/or angles among 3D points. These low-level features are processed by the DNN composed of fully-connected layers or graph convolution layers [14]. These studies experimentally demonstrated that their algorithms are robust against rotation of 3D shapes in such tasks as retrieval, classification, and segmentation of 3D point sets.

While invariance against 3D rotation has been achieved, there still remain two open questions.

First, the best choice of rotation-invariant features best-suited for a given DNN is not clear. Among the

The associate editor coordinating the review of this manuscript and approving it for publication was Jiju Poovvancheri.

previous studies, [10]–[13] adopt distances and/or angles computed from pairs of the 3D points as the input feature to the DNN. However, 3D shape analysis literature includes rotation-invariant handcrafted local features that describe local geometry of 3D point set by using methods other than distances and/or angles [15], [16]. Most of these handcrafted local features were theoretically and practically shown to be rotation-invariant.

Second, what is the effective approach to analyze 3D point sets which often consist of multiple parts having various relative size? To recognize both small and large parts of an object (e.g., the engine and the fuselage of an airplane), multi-resolution analysis is often employed. The existing methods perform multi-resolution analysis by using multi-scale input feature representation [11]–[13] or hierarchical convolution on feature maps having different spatial resolutions [10], [13]. Despite these previous attempts, approaches to multi-resolution analysis of 3D point sets have not yet been explored sufficiently.

In this article, we further the deep learning-based approach for rotation-invariant and accurate 3D point set processing. This article especially focuses on segmentation, or per-point classification, of 3D point sets, which is known to be quite challenging among the tasks of 3D point set analysis [1].

To accurately segment 3D point sets regardless of their orientations, we propose a novel DNN called *Rotation-invariant and Multi-scale feature Graph convolutional neural network* (*RMGnet*). Fig. 1 illustrates an overview of the proposed algorithm. RMGnet has novelty in both its input representation and network architecture. Input to RMGnet is a graph representation called Rotation-invariant and Multi-scale feature Graph (RMG). As the name suggests, RMG encodes 3D geometry of the 3D shape at diverse scales by using local 3D geometric feature having rotation-invariance. Scale, or radius, of each local region is selected randomly to effectively characterize partial shapes having diverse sizes. In terms of network architecture, we design a multi-resolution graph convolutional neural network (i.e., RMGnet), which is inspired by Graph U-Net [17]. RMGnet analyzes the input feature graph at multiple spatial resolutions by alternately and repeatedly applying graph convolution operation and graph pooling/unpooling operation.

Advantages of the proposed algorithm over the existing ones [10]–[13] are twofold. Firstly, our input graph representation (i.e., RMG) can be computed by using any one of, or a combination of, the rotation-invariant, handcrafted local 3D shape features proposed in the literature [15], [16]. This means that our RMGnet is more flexible than the existing DNNs that were designed to process a specific feature such as a histogram of distances and/or angles among points. Secondly, RMG can encode 3D geometry of 3D shapes at highly variable scales by randomly selecting radius of local regions. We expect multi-resolution analysis of RMG leads to accurate segmentation of parts having diverse scales. Some of the previous studies [11]–[13] also employed multi-scale input feature representation. However, their input features were

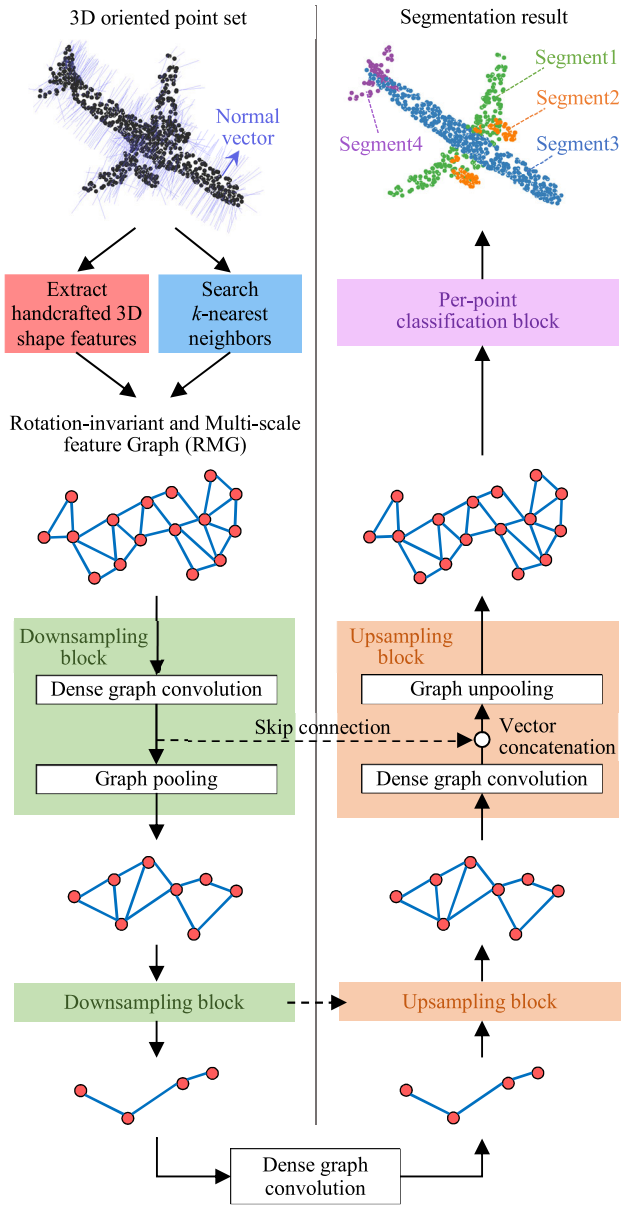

FIGURE 1. Overview of RMGnet. RMGnet in this figure has three resolution levels. Input 3D point set is represented as a graph by using rotation-invariant and multi-scale 3D geometric features. The graph is processed by an encoder-decoder graph convolutional DNN to produce orientation-agnostic segmentation of the input 3D shape.

computed at only two scales, that are, small scale defined by k-nearest neighbor search with fixed, small k and global scale that covers entire 3D shape.

Experimental evaluation demonstrates that the proposed RMGnet yields segmentation accuracy higher than the previous methods while satisfying rotation-invariance requirement. We also show that the high accuracy of RMGnet changes only marginally across the six handcrafted features we have tested as its input.

Contributions of this article can be summarized as follows;

(1) Proposition of RMGnet: A novel method designed for rotation-invariant and accurate segmentation of 3D point

sets. Multi-scale strategy is employed both for input feature representation and DNN architecture.
(2) Evaluation of RMGnet: Extensive and in-depth experiments using a standard benchmark dataset for 3D point set segmentation to verify efficacy of our approach.

The rest of this article is organized as follows. In the next section, we review related studies. Section III describes the proposed algorithm, followed by its experimental evaluation in Section IV. Section V summarizes this article.

II. RELATED WORK

A. END-TO-END DNN FOR 3D POINT SET ANALYSIS

PointNet [2] is the first-of-a-kind DNN that processes a set of 3D points representing a 3D shape in an end-to-end manner. 3D point sets are irregular and unordered data. To handle such data, PointNet first extracts a feature of each 3D point independently and then aggregates the features of all the 3D points to a single global feature to obtain invariance to permutation of the points.

To better capture hierarchical structure of the 3D shapes, various convolution operations on the 3D point set have been proposed. PointNet++ [3] groups neighboring 3D points to describe local structure of the 3D shape. PointCNN [4] orders 3D points in a local region by using χ-transformation and applies 1D convolution to the ordered points. DGCNN [5] and SpiderCNN [6] extract hierarchical and semantic shape features by using the convolution operations called EdgeConv and SpiderConv, respectively. DensePoint [7] combines graph convolution [14] and dense connection [18] to describe context among parts of the 3D shape at multiple resolution levels. More recent studies propose highly flexible and robust convolution operations on 3D point set. For example, graph convolution using Deformable Kernel [35] or Fuzzy Spherical Kernel [36] allows the DNN to gain robustness against translation, scaling, or density change of 3D point sets. Attention mechanisms on 3D point set [37]–[39] can adapt receptive fields of graph convolution to input 3D shape. These studies demonstrated that their DNNs achieve accurate classification and segmentation of 3D point sets. Also, efficient convolution operations on 3D point set [40], [41] have been proposed for large-scale point cloud analysis.

Nevertheless, the end-to-end DNNs mentioned above are not invariant against rotation about an arbitrary axis in the 3D space. Therefore, accuracy of these DNNs tend to suffer when orientations of the input 3D objects are different between training and inference. In contrast, our approach is unaffected by rotation of 3D point sets since each 3D point is described by using a handcrafted 3D shape feature having rotation invariance.

B. ROTATION-INVARIANT 3D POINT SET ANALYSIS

1) HANDCRAFTED FEATURE

For robust comparison or registration of 3D point sets, a number of rotation-invariant, local geometric features have been devised [15], [16]. There are roughly two approaches to achieve rotation invariance. The first approach employs geometric feature that is inherently invariant against rotation in the 3D space. PFH [19], PPF [20], and LSF [21] are the examples of such handcrafted local features. They were designed for oriented 3D point set, in which each point is associated with its orientation or a normal vector. These local features are formed as histograms of distances and angles computed from every pair of oriented points within a local region. These features are thus often termed ''point-pair feature''. The second approach normalizes orientation of 3D points in each local region by orthogonally transforming the points to the coordinate system called Local Reference Frame (LRF). Spin Image [22], SHOT [23], RoPS [24], and POD [25] are the examples of local features that adopt the second approach. These features are computed by spatially partitioning the LRF and then characterizing distribution of 3D points within each division.

Our proposed method may employ any one of these handcrafted local features mentioned above as its input feature representation having rotation invariance.

2) DEEP LEARNING

Following the success of PointNet, deep learning-based 3D point set analysis methods having rotation invariance have been proposed. Xiao *et al.* [26] proposed to align orientation of an entire 3D shape by using Principal Component Analysis prior to input to a DNN. PPF-FoldNet proposed by Deng *et al.* [27] learns rotation invariant local feature for 3D shape matching by autoencoding the point-pair features. DLAN by Furuya and Ohbuchi [8] refines and aggregates a set of POD features by using a PointNet-like DNN for rotation invariant 3D shape retrieval. Chen *et al.* [9] applied EdgeConv to a graph whose node is associated with a point-pair feature computed in a local region. Note, however, that these algorithms [8], [9], [26], [27] are not designed for segmentation of 3D point sets.

More recent studies [10]–[13] proposed DNNs with rotation invariance that are applicable to multiple tasks including segmentation of 3D point sets. All of these studies employ the point-pair feature to achieve rotation invariance although there exist small differences in feature computation process. RI-Conv proposed by Zhang *et al.* [10] computes distances and angles among a 3D point and its neighbors. While RI-Conv is inherently rotation-invariant, its segmentation accuracy is not satisfactory probably because its input feature is computed at a single, small scale.

To make the input feature more expressive, LGR-Net by Zhao *et al.* [12] and Rotation Invariant Framework by Li *et al.* [13] adopted ''2-scale'' feature that combines local feature and global feature. The 2-scale feature of LGR-Net is computed by using two independent DNN branches, one for local feature extraction and another for global feature extraction. Rotation Invariant Framework describes the input 3D point sets by using the 2-scale point pair features computed in local as well as global coordinate systems.

In contrast to the algorithms above, our RMGnet accepts, as its input, arbitrary local feature including, for example, the point-pair feature. Moreover, randomizing scale of the input local features allows effective characterization of 3D shapes consisting of various sized parts.

III. PROPOSED ALGORITHM

A. OVERVIEW OF RMGNET

In this section, we elaborate the proposed algorithm RMGnet, which is tailored to rotation-invariant segmentation of 3D point sets. Fig. 1 depicts an overview of RMGnet. Our new input representation, called RMG, encodes geometry of the 3D point set by using rotation-invariant and multi-scale handcrafted 3D shape features. RMG is a sparse directed graph created by connecting neighboring 3D points. Each node, or each 3D point, of the graph is associated with rotationally invariant 3D shape feature vector computed at random spatial scale. The input feature vector per node can be computed by using any one of the existing 3D shape descriptors with rotation invariance, or a combination of them. RMG is processed by the DNN called RMGnet whose architecture is inspired by Graph U-Net [17]. RMGnet is an encoder-decoder graph convolutional DNN with skip connections. At each resolution level of RMGnet, we apply multi-layer graph convolution with dense connection [7] to effectively encode hierarchy and context of the 3D shape features.

We also adopt several common techniques for deep learning for accurate segmentation. We employ data augmentation of training 3D shapes to improve the generalization ability of RMGnet. In addition, to cope with class imbalance problem in the training dataset, we equalize the distribution of 3D object category contained in a mini-batch. Furthermore, to boost segmentation accuracy at the inference stage, we employ single model ensemble, or voting, strategy using test-time data augmentation.

B. INPUT GRAPH REPRESENTATION

We assume that each 3D shape is represented as a set of N oriented 3D points where each 3D point is associated with its normal vector. N is fixed at 1,024 in our experiments. The oriented 3D point set is preprocessed to normalize its position and scale. Specifically, the point set is first translated so that its gravity center corresponds to the origin of the 3D space. The translated point set is then scaled to be inscribed in a sphere of radius 1.

Each node of RMG corresponds to one of N oriented points. A node i is connected to the other node j if the node i is included in a set of k-nearest neighbors of the node j in the 3D Euclidean space. We use $k = 16$, which is the neighborhood size close to the ones employed in the previous studies [5]–[7]. Each node, or point, is then associated with a rotation-invariant feature vector extracted from a Sphere-Of-Interest (SOI) including the point and its ε-ball neighbors. Radius ε of the SOI controls scale of feature extraction. To effectively describe 3D partial shapes having diverse scales, ε for each node is randomly and independently selected from a uniform distribution $U(0.1, 0.5)$. Note that the values of ε are not used as input to RMGnet since they are used only to determine the radii of SOIs. Obviously, k-nearest neighbors and ε-ball neighbors of the 3D points are unchanged by rotation in the 3D space. In addition, each SOI is encoded by one of the rotation invariant geometric features listed below. Using such a feature makes the RMG invariant against rotation of 3D point sets.

To describe the SOIs, we use either of the six handcrafted 3D shape features having rotation invariance. The first two features, PFH and LSF, are inherently rotation-invariant.

1) PFH [19]

For each pair of 3D points within the SOI, three angles are computed in a Darboux frame defined by the two 3D points and their normal vectors. The set of three angles extracted from every point pair in the SOI are accumulated in a joint histogram to form a 125-dimensional PFH vector.

2) LSF [21]

LSF is a point-pair feature similar to PFH. LSF employs distance between two points in addition to the three angles in the Darboux frame. These four quantities are accumulated in a joint histogram to generate a 625-dimensional LSF feature.

The remaining four features, i.e., SHOT, Spin Image, RoPS, and POD, achieve rotation invariance by normalizing orientation of the SOI. In this article, we use Local Reference Frame called EM (EM-LRF) [28] for robust orientation normalization. EM-LRF is computed as follows. One of three bases of the LRF is given by the normal vector of the central point of the SOI. The eigenvectors of the SOI is then computed by eigen-decomposing the covariance matrix of the 3D points within the SOI. The second basis is obtained by projecting the eigenvector of the SOI associated with the largest eigenvalue onto the tangent plane of the normal vector. The third basis is computed as outer product of the first and the second bases.

3) SHOT [23]

The LRF is spatially divided by spherical grids whose partitions are computed along the radial, azimuth, and elevation axes. Each division is described by a histogram of angles between the normal vector of the central point in the SOI and the normal vectors within the division. SHOT is a 320-dimensional vector.

4) SPIN IMAGE (SI) [22]

3D points within the SOI are converted to a 2D image by projecting the 3D points onto a cylindrical coordinate system. Pixel values of the 2D image are determined based on frequencies of 3D points that are projected to the pixels. The 2D image is then flattened to form a 153 dimensional SI feature vector.

5) RoPS [24]

3D points within the SOI are projected to three planes, each of which is defined by two of the three bases of the LRF. The three 2D images are then described by using moments and entropy of the pixel values to form a 135-dimensional RoPS feature vector.

6) POD [25]

The LRF is spatially partitioned by using 3D regular grid. Oriented points within each division of the grid are described by using their density, gravity center, and covariance of normal vectors. Different from the original POD that uses $4 \times 2 \times 1$ regular grids, we use $4 \times 4 \times 4$ regular grids to obtain 640-dimensional POD feature vectors.

Since each 3D shape feature mentioned above describes different aspect of 3D geometry within the SOI, combining multiple different handcrafted features could potentially enhance expressive power of RMG. When we combine two (or optionally more) features extracted from the same SOI, these features are fused by vector concatenation. The concatenated vector is then assigned to the central point of the SOI, or the node.

C. DNN ARCHITECTURE

RMGnet is an encoder-decoder graph convolutional DNN that analyzes the input graph, or RMG, at multiple spatial resolution levels. In the example shown in Fig. 1, the number of resolution levels is set to 3. In the experimental section, we will evaluate influence that the number of resolution levels has on segmentation accuracy. RMGnet consists of three building blocks, that are, downsampling block, upsampling block, and per-point classification block.

1) DOWNSAMPLING BLOCK

The downsampling block transforms the input graph to the lower resolution graph with higher semantic features. This is done by dense graph convolution and graph pooling. We employ densely-connected graph convolution [7] to capture context among the node features on RMG. Fig. 2 depicts processing pipeline of our dense graph convolution. Each node feature on the input graph is processed by three graph convolution functions ψ connected in tandem. Output node feature is obtained by concatenating the three feature vectors yielded from each convolution. Our graph convolution function ψ on node i and its feature $\mathbf{f}_i$ is defined by (1).

$$\psi\left(i\right) := \phi_2\left(\phi_1\left(\mathbf{f}_i\right) \oplus \rho\left(\left\{\phi_1\left(\mathbf{f}_j\right), \forall j \in kNN\left(i\right)\right\}\right)\right) \quad (1)$$

In (1), $\phi\left(\mathbf{f}\right) = \sigma\left(\mathbf{f} \cdot \mathbf{W} + \mathbf{b}\right)$ is a fully-connected layer parameterized by the projection matrix $\mathbf{W}$ and the bias vector $\mathbf{b}$. We use batch normalization [29] followed by ReLU as the activation function σ. The number of neurons for ϕ is fixed at 128 at all the resolution levels of RMGnet. kNN(i) is a set of k-nearest neighbors of node i. ρ is feature aggregation by max pooling. $\oplus$ denotes vector concatenation.

After dense graph convolution, the number of nodes on RMG is halved by graph pooling. We employ gPool operation proposed by Gao and Ji [17]. gPool computes inner products among the n node features on the graph and the parameter vector. gPool then picks $n/2$ nodes whose inner products are larger than the other $n/2$ nodes. The $n/2$ nodes having smaller inner products are discarded. The parameter vector is initialized randomly and tuned via DNN training. To augment connectivity among the subsampled nodes, the adjacency matrix of the subsampled nodes are raised to its second power as with [17].

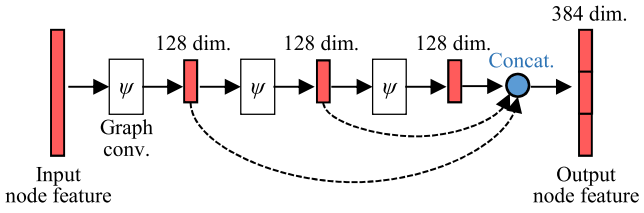

FIGURE 2. Processing pipeline of our dense graph convolution. Graph convolution ψ is applied to the input node feature (and its neighboring features) three times. Output vectors from each convolution are concatenated to form the multi-scale, contextual node feature.

2) UPSAMPLING BLOCK

The upsampling block transforms the input graph having low resolution to the graph with higher resolution by doubling the number of nodes. As with the downsampling block, the features of the input graph are first transformed by the dense graph convolution. The convolved node features are then fused with the node features passed from the downsampling block via the skip connection [42]. As shown in Fig. 1, we employ vector concatenation to fuse the two convolved node features computed in the downsampling block and the upsampling block at the same resolution level. Feature fusion using skip connection is beneficial not only to mitigate the vanishing gradient problem that DNN training often faces, but also to exploit both spatial and semantic features for accurate segmentation.

After the feature fusion, the graph is upsampled by using gUnpool operation [17]. gUnpool restores the nodes and their edges discarded by gPool operation at the same resolution level of the encoder.

3) PER-POINT CLASSIFICATION BLOCK

The feature graph generated from the upsampling block at the highest resolution level is further processed by the per-point classification block to predict per-point segment labels. To consider whole 3D shape in segmentation, each node feature is concatenated with its global feature, which is obtained by max-pooling all the node features of the graph. After vector concatenation, a segment label for each node is predicted via three fully-connected layers followed by a softmax function. The number of neurons for the fully-connected layers are 256, 128, and 50, where 50 corresponds to the number of segment labels.

D. TRAINING AND INFERENCE

1) TRAINING

The parameters of RMGnet are initialized by using the algorithm proposed by He *et al.* [30]. We use cross entropy as the loss function. The loss function is minimized by using Adam [31] with initial learning rate 0.0001. DNN training is iterated for 300 epochs.

Each minibatch contains 16 oriented 3D point sets. The training dataset used in the experiments consists of 3D object categories whose sizes are highly imbalanced. To mitigate negative effect due to the category imbalance, we perform balanced minibatch sampling that equalizes frequency of object categories contained in a minibatch. Since the number of object categories is equal to the minibatch size (i.e., 16) in our experiments, we randomly choose one 3D shape per object category to create a minibatch.

To diversify training 3D shape, we perform online data augmentation. Specifically, when a 3D point set is chosen as a training sample in a minibatch, data augmentation is applied with a probability of 0.8. We apply random anisotropic scaling followed by random noise addition to the training sample. Three orthogonal axes for anisotropic scaling is determined randomly in the 3D Euclidean space. Scaling factor for each axis is randomly selected from a uniform distribution $U(0.9, 1.1)$. Random noises, sampled from a normal distribution $N(0, 0.01)$, are added both to coordinates and normal vectors of the 3D points. After adding the random noise, each normal vector is rescaled to unit length. RMG is then computed by using the augmented 3D point set.

2) INFERENCE

To boost segmentation accuracy at the inference stage, we perform single model ensemble, or voting. We feed a testing 3D point set into RMGnet 10 times and take an average of the 10 prediction vectors yielded from the softmax layer to compute segment labels for the 3D points. To obtain diverse input RMGs from one 3D point set, the data augmentation used in training is also applied to each input for inference. The SOI radius ε of each point is recomputed per input after the augmentation is applied.

IV. EXPERIMENTS AND RESULTS

A. EXPERIMENTAL SETUP

1) DATASET

We use ShapeNetPart [32], which is the *de facto* standard benchmark dataset for 3D point set segmentation. We use the pre-release version of ShapeNetPart dataset that consists of the training set having 13,998 3D point sets and the testing set including 2,874 3D point sets. RMGnet is trained by using the training set and segmentation accuracy is evaluated by using the testing set. These 3D point sets are classified into 16 categories of rigid objects such as airplane, chair, and car. Each 3D point is annotated with one of 50 segment labels, for example, wing, seat, and wheel.

To evaluate rotation invariance, we adopt two evaluation scenarios as used in the previous work [10], [12], [13]. In the first scenario, denoted "z/SO3", each training 3D shape is randomly rotated about its upright ("z") axis while each testing 3D shape is rotated about an axis randomly determined in the 3D space ("SO3"). Rotation angle is chosen randomly. In the second scenario, "SO3/SO3", both training and testing 3D shapes are rotated about randomly selected axes.

2) ACCURACY MEASURE

We use Intersection over Union (IoU) for 3D point sets [2] as a numerical measure of segmentation accuracy. Instance-level IoU (I-IoU) is calculated by averaging IoU values for all the 2,874 testing 3D shapes. Category-level IoU (C-IoU) is computed by averaging 16 IoU values, each of which is a mean IoU for the object category. C-IoU is used for comparison with the previous methods.

To verify the effectiveness of our multi-scale approach, segmentation accuracy is also evaluated with respect to size of the segments. For each segment category, we calculate an average proportion P of the segments to the whole shapes. The proportion is calculated by dividing the number of 3D points comprising the segment by the total number of points (i.e., 1,024). According to the value of P, each segment category is classified into one of three groups, that are, small-size group (SS), medium-size group (MS), and large-size group (LS). SS, MS, and LS contain segment categories whose P are in ranges (0, 0.33], (0.33, 0.67], and (0.67, 1), respectively. IoU values are averaged in each group to obtain SS-IoU, MS-IoU, and LS-IoU.

3) COMPETITORS

We compare our RMGnet against nine algorithms for 3D point set segmentation. PointNet [2], PointNet++ [3], PointCNN [4], DGCNN [5], SpiderCNN [6], and DensePoint [7] take as their input raw 3D point sets. RIConv [10], Rotation Invariant Framework (RI-Framework) [13], and LRG-Net [12] accept rotation invariant feature as input to their respective DNN.

To make clear handcrafted 3D shape feature used for RMG computation, we prefix the name of shape feature to RMG-net. For example, POD-RMGnet denotes that POD feature is used to describe SOIs of 3D point sets.

4) IMPLEMENTATION AND HARDWARE CONFIGURATION

We implemented our DNN by using Python with TensorFlow library [33]. Handcrafted feature extraction part was implemented by using C++. The codes of SHOT, SI, and RoPS were borrowed from Point Cloud Library [34] while PFH, LSF, and POD were implemented by ourselves. We ran our code on a PC having an Intel Core i7 6700 CPU, an Nvidia GeForce GTX 1080Ti GPU, and 64GB RAM. Computational cost of RMGnet is evaluated in the next section.

B. EXPERIMENTAL RESULTS

1) COMPARISON WITH EXISTING ALGORITHMS

Table 1 compares segmentation accuracies, measured in C-IoU, of the algorithms under z/SO3 and SO3/SO3 scenarios. Accuracies of PointNet and its successors suffer especially under z/SO3 scenario, where the DNNs need to segment 3D point sets having orientations unseen during training.

TABLE 1. Comparison of segmentation accuracy (C-IoU [%]).

Algorithms	Rotation-invariant?	z/SO3	SO3/SO3
PointNet [2]	No	37.8	74.4
PointNet++ [3]	No	48.2	76.7
PointCNN [4]	No	34.7	71.4
DGCNN [5]	No	37.4	73.3
SpiderCNN [6]	No	42.9	72.3
DensePoint [7]	No	41.3	74.3
RIConv [10]	Yes	75.3	75.5
RI-Framework [13]	Yes	79.2	79.4
LGR-Net [12]	Yes	80.0	80.1
PFH-RMGnet	Yes	80.0	79.8
LSF-RMGnet	Yes	**81.4**	**81.3**
SHOT-RMGnet	Yes	81.2	81.0
SI-RMGnet	Yes	80.5	80.6
RoPS-RMGnet	Yes	80.5	80.7
POD-RMGnet	Yes	**81.5**	**81.4**

On the other hand, RIConv, RI-Framework, LRG-Net, and the proposed RMGnet have invariance against rotation of 3D objects since they produce almost the same segmentation accuracies in z/SO3 and SO3/SO3. Among these rotation invariant algorithms, our POD-RMGnet yields the best segmentation accuracy, i.e., 81.5% for z/SO3 and 81.4% for SO3/SO3.

Table 2 and Table 3 show per-category mean IoU for z/SO3 and SO3/SO3, respectively. Average of the per-category IoUs in Table 2 and Table 3 correspond to C-IoUs in Table 1. In many object categories, our RMGnet outperforms the existing algorithms. There are small differences of the IoU values in Table 2 and Table 3. We speculate that these differences stem from randomness that resides in RMG generation and DNN training.

Fig. 3 qualitatively compares segmentation accuracy of RIConv [10] and POD-RMGnet. Both algorithms succeed in detecting large parts of 3D objects. Compared to RIConv algorithm that uses single-scale feature, our multi-scale approach has an advantage in segmenting small parts such as wings of a rocket or handlebars of a motorbike.

Table 4 compares computational cost of the three segmentation algorithms, that are, PointNet++, RIConv, and POD-RMGnet. We used the implementations of PointNet++ and RIConv provided by the authors for comparison. Among the three methods, our RMGnet takes the longest time both for training and inference. The high temporal cost of RMGnet stems in large part from handcrafted feature extraction. In our implementation, feature extraction from SOIs is computed by using CPU. RMGnet would be accelerated if feature extraction could be processed on GPU. In terms of memory footprint of GPU, POD-RMGnet has the lowest memory consumption among the three methods. This is because all the graph convolution layers of RMGnet have relatively small number of neurons, i.e., 128.

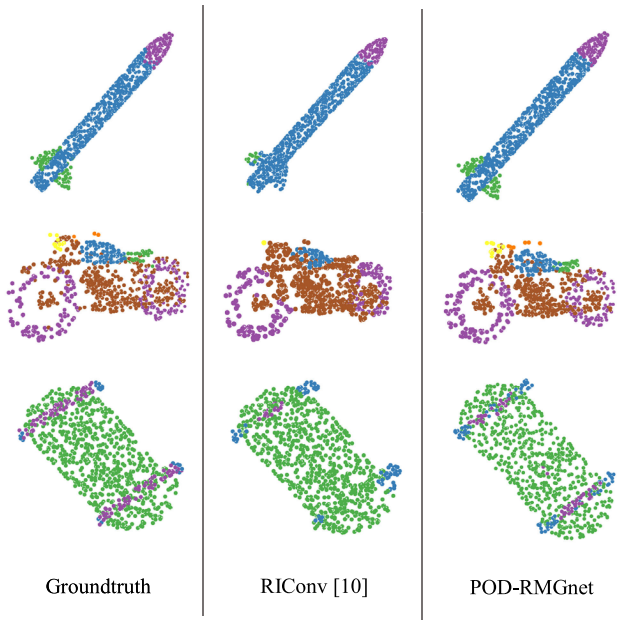

FIGURE 3. Qualitative comparison with RIConv algorithm that has invariance against rotation. Our RMGnet is advantageous especially in detecting small parts of 3D objects.

2) COMPARISON OF INPUT FEATURES

In Table 1, all the six variants of RMGnet produce similarly high C-IoU values around 80%. It is worth noting that such high segmentation accuracy can be obtained not only by the point pair features commonly adopted by the previous studies but also by the LRF-based 3D shape features. These results suggest that our approach, i.e., forming and analyzing rotation invariant and multi-scale feature graphs, is reasonable for rotationally invariant 3D point set segmentation.

As shown in Table 1, POD-RMGnet performs the best among the six variants of our method. We suspect that POD feature can describe more information about 3D geometry than the other handcrafted features. That is, the POD feature vector contains (nearly) raw voxel representation of the local 3D shape since POD computes densities of 3D points within the divisions created by regular grids. RMGnet thus can learn to extract highly semantic features useful for segmentation from POD features. In contrast, the other handcrafted features have less 3D geometric information as they encode the 3D local shape by using lower dimensional representations such as 2D images or histograms of scalar statistics.

We also evaluate effectiveness of combining different handcrafted features. In this experiment, we combine two features out of three features that perform the best in Table 1, i.e., POD, LSF, and SHOT. As shown in Table 5, combining

TABLE 2. Per-category mean IoU [%] under z/SO3 scenario.

Algorithms	aero	bag	cap	car	chair	earph.	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate	table
PointNet [2]	40.4	48.1	46.3	24.5	45.1	39.4	29.2	42.6	52.7	36.7	21.2	55.0	29.7	26.6	32.1	35.8
PointNet++ [3]	51.3	66.0	50.8	25.2	66.7	27.7	29.7	65.6	59.7	70.1	17.2	67.3	49.9	23.4	43.8	57.6
PointCNN [4]	21.8	52.0	52.1	23.6	29.4	18.2	40.7	36.9	51.1	33.1	18.9	48.0	23.0	27.7	38.6	39.9
DGCNN [5]	37.0	50.2	38.5	24.1	43.9	32.3	23.7	48.6	54.8	28.7	17.8	74.4	25.2	24.1	43.1	32.3
SpiderCNN [6]	48.8	47.9	41.0	25.1	59.8	23.0	28.5	49.5	45.0	83.6	20.9	55.1	41.7	36.5	39.2	41.2
DensePoint [7]	37.6	56.9	57.8	23.6	28.6	41.8	39.2	43.5	51.1	39.9	23.2	74.7	36.9	45.3	37.0	24.3
RIConv [10]	80.6	80.0	70.8	68.8	86.8	70.3	87.3	84.7	77.8	80.6	57.4	91.2	71.5	52.3	66.5	78.4
RI-Framework [13]	81.4	**82.3**	86.3	75.3	88.5	72.8	90.3	82.1	81.3	81.9	67.5	92.6	75.5	54.8	75.1	78.9
LGR-Net [12]	81.5	80.5	81.4	75.5	87.4	72.6	88.7	83.4	**83.1**	86.8	66.2	92.9	76.8	62.9	**80.0**	80.0
PFH-RMGnet	81.8	77.1	85.4	73.4	88.4	75.6	90.1	82.7	80.0	**88.2**	68.1	92.3	78.2	61.8	**78.0**	79.7
LSF-RMGnet	**82.4**	**81.4**	86.1	75.4	88.0	**82.8**	91.3	83.5	80.0	**88.4**	70.9	92.5	79.1	**63.1**	77.3	80.1
SHOT-RMGnet	82.1	81.3	**88.6**	74.8	88.1	**80.4**	91.1	84.9	79.8	83.5	**71.9**	93.6	78.9	62.4	76.6	**80.5**
SI-RMGnet	81.2	81.0	**88.0**	75.6	**88.7**	76.9	90.6	84.1	81.0	86.0	69.1	91.4	77.6	60.1	76.3	80.4
RoPS-RMGnet	81.6	80.7	85.6	**75.7**	88.6	77.1	**91.7**	**85.2**	**82.4**	80.2	71.2	**93.8**	**80.0**	60.7	73.8	79.3
POD-RMGnet	**82.6**	80.9	86.8	**76.6**	**89.0**	80.2	**91.8**	**85.3**	80.6	83.4	**71.5**	**94.7**	**79.9**	**64.0**	77.2	**80.5**

TABLE 3. Per-category mean IoU [%] under SO3/SO3 scenario.

Algorithms	aero	bag	cap	car	chair	earph.	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate	table
PointNet [2]	81.6	68.7	74.0	70.3	87.6	68.5	88.9	80.0	74.9	83.6	56.5	77.6	75.2	53.9	69.4	79.9
PointNet++ [3]	79.5	71.6	**87.7**	70.7	88.8	64.9	88.8	78.1	79.2	**94.9**	54.3	92.0	76.4	50.3	68.4	**81.0**
PointCNN [4]	78.0	80.1	78.2	68.2	81.2	70.2	82.0	70.6	68.9	80.8	48.6	77.3	63.2	50.6	63.2	**82.0**
DGCNN [5]	77.7	71.8	77.7	55.2	87.3	68.7	88.7	**85.5**	81.8	81.3	36.2	86.0	77.3	51.6	65.3	80.2
SpiderCNN [6]	74.3	72.4	72.6	58.4	82.0	68.5	87.8	81.3	71.3	**94.5**	45.7	88.1	**83.4**	50.5	60.8	78.3
DensePoint [7]	77.1	76.0	80.6	61.7	85.7	73.2	87.7	80.1	75.2	83.0	49.9	90.1	70.4	48.4	70.1	79.2
RIConv [10]	80.6	80.2	70.7	68.8	86.8	70.4	87.2	84.3	78.0	80.1	57.3	91.2	71.3	52.1	66.6	78.5
RI-Framework [13]	81.4	**84.5**	85.1	75.0	88.2	72.4	90.7	84.4	80.3	84.0	68.8	92.6	76.1	52.1	74.1	80.0
LGR-Net [12]	81.7	78.1	82.5	75.1	87.6	74.5	89.4	**86.1**	**83.0**	86.4	65.3	92.6	75.2	**64.1**	**79.8**	80.5
PFH-RMGnet	80.9	76.5	**87.1**	73.7	87.9	76.7	90.1	82.3	80.5	86.2	66.1	93.2	76.8	**63.5**	75.9	79.8
LSF-RMGnet	**82.1**	79.0	86.6	**75.6**	88.8	76.0	91.2	82.8	81.5	88.3	**72.5**	**95.3**	80.8	61.4	**78.4**	80.8
SHOT-RMGnet	81.3	**83.0**	85.9	74.9	**88.9**	76.3	90.5	83.2	81.3	88.6	71.0	92.9	79.6	63.2	75.7	80.4
SI-RMGnet	81.7	79.5	85.5	74.5	88.4	**79.9**	**91.4**	83.6	**82.0**	83.8	68.6	**94.0**	80.2	60.8	75.1	80.1
RoPS-RMGnet	81.2	82.1	87.0	**75.6**	88.6	**79.8**	91.3	83.9	81.5	79.5	70.6	93.6	80.5	61.1	75.2	79.5
POD-RMGnet	**82.4**	81.0	85.7	**76.9**	**89.7**	79.7	**91.5**	84.1	81.9	84.7	**72.6**	93.8	**81.9**	61.4	77.5	79.5

TABLE 4. Comparison of computational cost.

Algorithms	Training time for 300 epochs	Inference time per minibatch	GPU memory footprint
PointNet++ [3]	13 hours	0.05 sec.	10.8 GBytes
RIConv [10]	23 hours	0.13 sec.	9.2 GBytes
POD-RMGnet	29 hours	0.35 sec.	6.9 GBytes

TABLE 5. Effectiveness of feature combination (z/SO3 scenario).

Input to RMGnet	C-IoU	I-IoU
POD only	81.5	83.4
LSF only	81.4	82.9
SHOT only	81.2	82.9
POD and LSF	**81.8**	**83.9**
POD and SHOT	81.7	83.8
LSF and SHOT	81.3	83.6

two features improves either or both of C-IoU and I-IoU. Combining POD and LSF is the most effective in Table 5 probably because they describe the SOI by using totally different approaches, that are, distance/angle among point pairs and 3D geometry in the LRF.

3) IN-DEPTH EVALUATION OF RMGnet

In this subsection we evaluate efficacy of multi-scale feature extraction for computing RMGs, multi-resolution analysis by RMGnet, and several techniques for effective training and inference. In the following experiments, POD-RMGnet is evaluated under z/SO3 scenario.

Table 6 shows influence that the scale of 3D shape feature extraction has on segmentation accuracy. We compare our multi-scale feature extraction against single-scale feature extraction with a fixed SOI radius ε. In Table 6, our multi-scale feature outperforms the single-scale features in most of the IoU measures. The highest SS-IoU, MS-IoU, and LS-IoU are produced by the multi-scale feature. This result indicates randomizing feature scale has a positive impact on segmenting parts having diverse sizes. In particular, accuracy improvements are salient in segmenting small and medium parts, which would be more difficult to detect than large parts.

TABLE 6. Effectiveness of multi-scale feature extraction.

	Radius ε of SOI	C-IoU	I-IoU	SS-IoU	MS-IoU	LS-IoU
Single-scale	Fixed at 0.1	78.8	79.9	65.6	83.9	90.1
	Fixed at 0.2	80.1	82.4	67.4	84.9	91.2
	Fixed at 0.3	80.9	83.2	68.7	85.8	91.2
	Fixed at 0.5	80.8	**83.4**	68.3	86.2	91.3
Multi-scale	Randomly chosen from $U(0.1, 0.5)$	**81.5**	**83.4**	**69.6**	**86.6**	**91.5**

Table 7 shows the relationship between the number of resolution levels, that controls the depth of RMGnet, and segmentation accuracy. Peak of C-IoU appears at around three to four resolution levels. Interestingly, even with a single resolution, our RMGnet produces C-IoU of 80.2% that is higher than those of the existing algorithms listed in Table 1. This is probably because our input representation, i.e., RMG, characterizes the input 3D shapes at various scales using randomized SOI radii. Increasing the number of resolution levels to more than five slightly lowers segmentation accuracy since training deeper neural networks becomes more difficult.

TABLE 7. Number of resolution levels and segmentation accuracy.

# of resolution levels	C-IoU	I-IoU	SS-IoU	MS-IoU	LS-IoU
1	80.2	82.8	66.8	87.1	91.2
2	81.3	83.6	68.8	86.6	91.3
3	**81.5**	83.4	**69.6**	86.6	91.5
4	**81.5**	**83.7**	69.2	86.2	**91.7**
5	81.1	**83.7**	68.2	**87.3**	91.6
6	80.8	83.6	68.0	86.9	91.6

To further validate the design parameters of the proposed algorithm, we investigate an impact of DNN architecture on segmentation accuracy. To this end, the architecture of RMGnet described in Section III-C is replaced with the previously proposed DNNs for point set segmentation, that are, PointNet++ [3], DensePoint [7], and Graph U-Net [17]. POD feature is used as input to these DNNs. Table 8 shows that the proposed RMGnet performs the best among the four different DNN architectures. Compared to RMGnet, Graph U-Net may have less expressive power since it does not have the dense connection among graph convolution layers. DensePoint shows lower accuracy than RMGnet despite the existence of the densely-connected convolution layers. We speculate that DensePoint, as well as PointNet++, outputs fuzzy, or smoothed, segmentation results due to feature propagation layers [3] that interpolate node features by their weighted average.

TABLE 8. Comparison of DNN architecture.

Algorithms	C-IoU	I-IoU	SS-IoU	MS-IoU	LS-IoU
POD-RMGnet	**81.5**	**83.4**	**69.6**	**86.6**	**91.5**
POD-PointNet++	78.4	81.1	64.6	83.4	89.9
POD-DensePoint	78.5	81.2	64.1	84.4	90.0
POD-GraphUnet	75.4	78.5	60.5	79.2	89.8

Table 9 demonstrates the effectiveness of the techniques we adopted, that are, data augmentation, balanced minibatch sampling, and voting. When balanced minibatch sampling is disabled, a minibatch is formed by randomly and uniformly sampling 3D point sets from the training dataset ignoring their object categories. Among the three techniques, data augmentation and voting have positive impact on improving both C-IoU and I-IoU. Balanced minibatch sampling, on the other hand, improves C-IoU while it slightly decreases I-IoU. Training the DNN using category-imbalanced minibatches would better I-IoU, whose computation is also affected by imbalance of object categories in the testing dataset.

TABLE 9. Ablation study of RMGnet (DA: data augmentation, BMS: balanced minibatch sampling).

DA	BMS	Voting	C-IoU	I-IoU
Yes	Yes	Yes	**81.5**	83.4
No	Yes	Yes	80.7	83.3
Yes	No	Yes	80.3	**83.9**
Yes	Yes	No	79.3	81.7
No	No	No	78.4	82.4

Fig. 4 exemplifies segmentation results yielded by POD-RMGnet. The proposed method outputs segmentation results quite similar to the groundtruth regardless of the orientations of the input 3D point sets. Our RMGnet succeeds in segmenting not only large parts (e.g., the fuselage of the airplane or the barrel of the gun), but also small parts (e.g., engines of the airplane or a trigger of the gun).

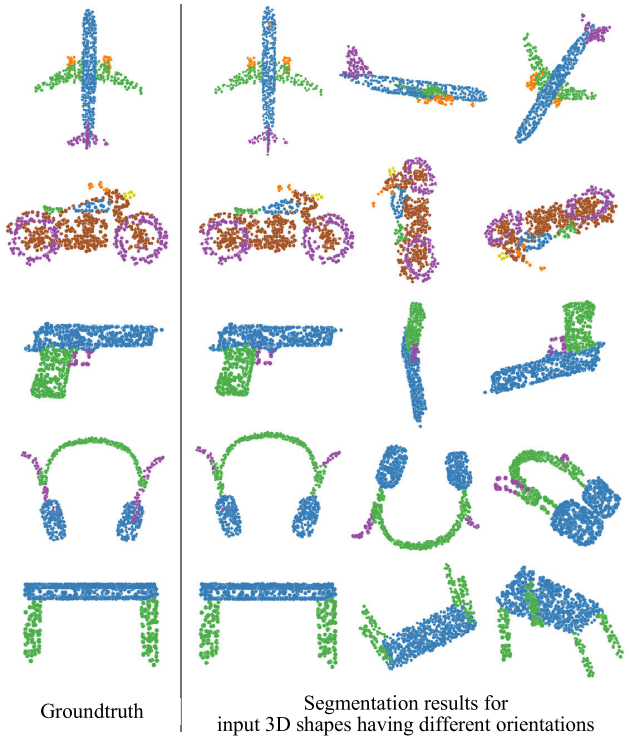

FIGURE 4. Examples of segmentation results produced by POD-RMGnet. The proposed algorithm yields accurate segmentations that are almost identical to the ground truth regardless of orientations of the input 3D shapes.

V. CONCLUSION AND FUTURE WORK

In this article, we proposed the novel algorithm, i.e., RMGnet, for accurate and rotation-invariant segmentation of 3D shapes represented by 3D point sets. Previous algorithms achieved rotation invariance by tightly coupling input feature with the networks architecture. Our RMGnet, on the other hand, is more flexible as it could work in conjunction with almost

any kind of handcrafted 3D shape features having rotation invariance. In addition, multi-scale extraction of the handcrafted features and multi-resolution analysis by the graph convolutional DNN enable segmentation of the 3D shapes consisting of parts having diverse scales. Experimental evaluation demonstrated that RMGnet produces segmentation accuracy higher than the state-of-the-art algorithms. Also, we quantitatively and qualitatively evaluated rotation invariance of the proposed RMGnet.

Future work includes evaluation of RMGnet using more realistic data. The experiments conducted in this article uses 3D point sets derived from synthetic 3D shape data. Therefore, we intend to collect 3D point sets, for example, acquired by 3D range scanners and evaluate segmentation accuracy of RMGnet by using these 3D point sets.

ACKNOWLEDGMENT

(Takahiko Furuya and Xu Hang contributed equally to this work.)

REFERENCES

[1] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," 2019, *arXiv:1912.12033*. [Online]. Available: http://arxiv.org/abs/1912.12033

[2] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 652–660.

[3] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5105–5114.

[4] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on X-transformed points," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2018, pp. 820–830.

[5] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, Nov. 2019.

[6] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep learning on point sets with parameterized convolutional filters," *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 87–102.

[7] Y. Liu, B. Fan, G. Meng, J. Lu, S. Xiang, and C. Pan, "DensePoint: Learning densely contextual representation for efficient point cloud processing," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5239–5248.

[8] T. Furuya and R. Ohbuchi, "Deep aggregation of local 3D geometric features for 3D model retrieval," in *Proc. Brit. Mach. Vis. Conf.*, 2016, p. 121.

[9] C. Chen, G. Li, R. Xu, T. Chen, M. Wang, and L. Lin, "ClusterNet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4994–5002.

[10] Z. Zhang, B.-S. Hua, D. W. Rosen, and S.-K. Yeung, "Rotation invariant convolutions for 3D point clouds deep learning," in *Proc. Int. Conf. 3D Vis. (3DV)*, Sep. 2019, pp. 204–213.

[11] X. Sun, Z. Lian, and J. Xiao, "SRINet: Learning strictly rotation-invariant representations for point cloud classification and segmentation," in *Proc. 27th ACM Int. Conf. Multimedia*, Oct. 2019, pp. 980–988.

[12] C. Zhao, J. Yang, X. Xiong, A. Zhu, Z. Cao, and X. Li, "Rotation invariant point cloud classification: Where local geometry meets global topology," 2019, *arXiv:1911.00195*. [Online]. Available: http://arxiv.org/abs/1911.00195

[13] X. Li, R. Li, G. Chen, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "A rotation-invariant framework for deep point cloud analysis," 2020, *arXiv:2003.07238*. [Online]. Available: http://arxiv.org/abs/2003.07238

[14] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf. (ESWC)*, 2018, pp. 593–607.

[15] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. M. Kwok, "A comprehensive performance evaluation of 3D local feature descriptors," *Int. J. Comput. Vis.*, vol. 116, no. 1, pp. 66–89, Jan. 2016.

[16] J. Garstka and G. Peters, "Evaluation of local 3-D point cloud descriptors in terms of suitability for object classification," in *Proc. 13th Int. Conf. Informat. Control, Autom. Robot.*, Jul. 2016, pp. 540–547.

[17] H. Gao and S. Ji, "Graph U-nets," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 2083–2092.

[18] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.

[19] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 3212–3217.

[20] B. Drost, M. Ulrich, N. Navab, S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 998–1005.

[21] Y. Ohkita, Y. Ohishi, T. Furuya, and R. Ohbuchi, "Non-rigid 3D model retrieval using set of local statistical features," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, Jul. 2012, pp. 593–598.

[22] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3D scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 433–449, May 1999.

[23] F. Tombari, S. Salti, and L. D. Stefano, "Unique signatures of histograms for local surface description," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2010, pp. 356–369.

[24] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3D local surface description and object recognition," *Int. J. Comput. Vis.*, vol. 105, no. 1, pp. 63–86, Oct. 2013.

[25] T. Furuya and R. Ohbuchi, "Diffusion-on-Manifold aggregation of local features for shape-based 3D model retrieval," in *Proc. 5th ACM Int. Conf. Multimedia Retr. ICMR*, 2015, pp. 171–178.

[26] Z. Xiao, H. Lin, R. Li, H. Chao, and S. Ding, "Endowing deep 3D models with rotation invariance based on principal component analysis," 2019, *arXiv:1910.08901*. [Online]. Available: http://arxiv.org/abs/1910.08901

[27] H. Deng, T. Birdal, and S. Ilic, "PPF-FoldNet: Unsupervised learning of rotation invariant 3D local descriptors," *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 602–618.

[28] A. Petrelli and L. Di Stefano, "On the repeatability of the local reference frame for partial shape matching," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2244–2251.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 448–456.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1026–1034.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.

[32] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3D shape collections," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 1–12, Nov. 2016.

[33] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2016, pp. 265–283.

[34] 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.

[35] Z.-H. Lin, S.-Y. Huang, and Y.-C. F. Wang, "Convolution in the cloud: Learning deformable kernels in 3D graph convolution networks for point cloud analysis," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1800–1809.

[36] H. Lei, N. Akhtar, and A. Mian, "SegGCN: Efficient 3D point cloud segmentation with fuzzy spherical kernel," *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11611–11620.

[37] Y. Shi, H. Fang, J. Zhu, and Y. Fang, "Pairwise attention encoding for point cloud feature learning," in *Proc. Int. Conf. 3D Vis. (3DV)*, Sep. 2019, pp. 135–144.

[38] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10288–10297.

[39] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "PointASNL: Robust point clouds processing using nonlocal neural networks with adaptive sampling," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5589–5598.

[40] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11108–11117.

[41] Q. Xu, X. Sun, C.-Y. Wu, P. Wang, and U. Neumann, "Grid-GCN for fast and scalable point cloud learning," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5661–5670.

[42] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI*. Berlin, Germany: Springer, 2015, pp. 234–241. [Online]. Available: https://link.springer.com/book/10.1007/978-3-319-24574-4

TAKAHIKO FURUYA received the Ph.D. degree in information and systems engineering from the University of Yamanashi, Japan, in 2015. In 2015, he has become an Assistant Professor with the Graduate School of Medicine and Engineering, University of Yamanashi. His research interests include multimedia information retrieval, 3-D shape analysis, computer vision, and machine learning.

XU HANG was born in 1995. He received the bachelor's degree from the Ningbo Institute of Technology, Zhejiang University, China. He is currently pursuing the master's degree within a joint program between Hangzhou Dianzi University, China, and the University of Yamanashi, Japan. His research interests include deep learning and 3-D shape analysis.

RYUTAROU OHBUCHI (Member, IEEE) received the Ph.D. degree in computer science from the University of North Carolina at Chapel Hill, USA, in 1994. He joined IBM Research Tokyo, in 1994. He has become an Associate Professor with the Graduate School of Medicine and Engineering, University of Yamanashi, in 1999. He has become a Full Professor, in 2007. His research interests include multimedia information retrieval, 3-D shape analysis, and machine learning.

JINLIANG YAO received the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, China, in 2009. Since 2009, he has been employed at the Computer Science School, Hangzhou Dianzi University, where he currently works as an Associate Professor with the Cognitive Laboratory. His research interests include image retrieval, image processing, and pattern recognition.