The Visual Computer (2022) 38:2303–2313
https://doi.org/10.1007/s00371-021-02112-7

ORIGINAL ARTICLE

BEACon: a boundary embedded attentional convolution network for point cloud instance segmentation

Tianrui Liu[1] · Yiyu Cai[2] · Jianmin Zheng[3] · Nadia Magnenat Thalmann[3]

Accepted: 15 March 2021 / Published online: 8 April 2021

Abstract
Motivated by how humans perceive geometry and color to recognize objects, we propose a boundary embedded attentional convolution (BEACon) network for point cloud instance segmentation. At the core of BEACon, we introduce the attentional weight in the convolution layer to adjust the neighboring features, with the weight being adapted to the relationship between geometry and color changes. As a result, BEACon makes use of both geometry and color information, takes instance boundary as an important feature, and thus learns a more discriminative feature representation in the neighborhood. Experimental results show that BEACon outperforms the state-of-the-art by a large margin. Ablation studies are also provided to prove the large benefit of incorporating both geometry and color into attention weight for instance segmentation.

Keywords 3D point cloud · Instance segmentation · Attentional convolution network

1 Introduction

Point cloud is a collection of points with spatial coordinates and possibly additional features such as color or intensity. Visualizing point cloud in a scene provides intuitive and accurate information of 3D space. Modern technologies such as 3D imaging, photogrammetry, and SLAM can produce colored point cloud. The applications of point cloud cover a large variety of fields, from augmented reality, autonomous navigation, to Scan-to-BIM in construction.

✉ Tianrui Liu
tianrui001@e.ntu.edu.sg

Yiyu Cai
myycai@ntu.edu.sg

Jianmin Zheng
asjmzheng@ntu.edu.sg

Nadia Magnenat Thalmann
nadiathalmann@ntu.edu.sg

1 Institute for Media Innovation, Interdisciplinary Graduate School, Nanyang Technological University, Singapore, Singapore

2 School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore, Singapore

3 School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore

One basic task in point cloud processing is segmentation that partitions the point cloud into groups, each of which exhibits certain homogeneous characteristics. In particular, semantic segmentation groups the points with similar semantics, and instance segmentation further divides the points into object instances. For either segmentation task, the key to the problem is feature learning. On the one hand, extracting the discriminative features from point cloud is vital to the following clustering and segmentation processes. While searching-based algorithm [1] showed good performance on clustering data with distinctive features, 3D point cloud in its original space is not so easily separable. For low dimension and structured data such as 2D image, the convolutional network is well known for its ability to extract features for various tasks [2]. This trend is extending to the 3D field as many recent semantic segmentation works [3–7] have shown promising results using the concept of convolution on an unordered point set. On the other hand, not all features are useful for segmentation, and feature selection is a well-studied topic [8]. In convolution networks, feature selection is accomplished by the attentional layer, which is used in multiple recent literature [9–11] to tackle the isotropic nature of convolutional kernels. However, the effectiveness of attentional convolution on instance segmentation remains unexplored. In addition, the weight matrix in previous works only depends on spatial or propagated feature difference. Although this may seem sufficient for semantic task, the relationship between color and geometry is more important when delineating an instance boundary.

Intuitively, people delineate the instance boundary by paying more attention to geometry or color in different circumstances, e.g. , we distinguish two instances of walls based on verticality, and a white board from the wall mainly based on color of the frame. To model the attention mechanism that is adaptive to the instance boundary, we propose BEACon network. We first define a generalized version of point set convolution and demonstrate how the design choice of BEACon fits into those definitions. For each layer, the boundary is represented as differences in multiple feature spaces including geometry and color spaces, and the attentional weight is generated by feeding boundary information to a set of multilayer perceptron (MLP). After the instance embedding is obtained, we use the Cut-Pursuit algorithm [12] for clustering and further design a vicinity merging algorithm specifically for the large indoor spaces dataset. Experiments on S3DIS [13] dataset show a significant improvement on instance tasks among the most recent works. We also test BEACon on PartNet [14] dataset to demonstrate its effectiveness on part instance segmentation. To summarize, our main contributions are as follows:

- We propose a network that incorporates boundary embedded attention mechanism for instance segmentation.
- We explicitly model the influence of both geometry and color changes on attentional weight. Experiment results prove its benefit, especially for instance segmentation.

2 Related work

This section briefly reviews the prior arts on semantic and instance segmentation of point clouds. As more related to our work, we will focus on deep learning approaches that directly process unordered point set. Other methods include volumetric approach [15–18] which requires voxelization of the input data, and multi-view approach [19].

2.1 Semantic segmentation

PointNet [20] pioneers direct MLP on unordered point set. However, it does not take into consideration the spatial context around the vicinity. To capture a larger spatial context, the related approaches can be divided into 3 categories: point-based, graph-based and CNN-based.

2.1.1 Point-based approach

Several methods use neighborhood context, recurrent neural network (RNN) or kernel to aggregate local information. Ye et al. [21] develop pointwise pyramid pooling to capture the spatial context at different scales, and the across-block relationship is explored with RNN. ShapeContextNet [22] uses kernels to extract the local features and train the shape context using the self-attention network. EdgeConv is proposed in DGCNN [23] for feature propagation. The graph in each layer is constructed dynamically in feature space, allowing point clouds being grouped even over long distances. However, the above-mentioned methods aggregate information on all the input points through each layer. Much information is overlapping, and the network becomes unnecessarily huge.

2.1.2 Graph-based approach

This approach incorporates the graph convolutional neural network into proposed network structures. The superpoint graph [24] first partitions the whole scene into small patches based on geometric features and then applies graph convolutional neural network to predict the semantic label of each patch. In its following work [25], the manually selected geometric features are replaced by a lightweight network called local point embedder. PGCNet [26] takes a similar approach by partitioning the scene into planar surfaces, followed by patch graph construction to produce semantic output. Wang et al. [27] transfer the local neighborhood point set into the spectral domain, and the structural information is encoded in the graph topology.

2.1.3 CNN-based approach

Different from the 2D image, 3D data do not have a regular grid-like partition scheme, and different design choices can be made for kernel weight and kernel shape. On the one hand, kernel function can be regarded as a weight matrix, and the weight is defined based on features in the neighborhood. Liu et al. [9] propose relation shape convolution, in which the kernel function is mapped with an MLP based on surrounding spatial features. PointWeb [11] adds an adaptive feature adjustment (AFA) module on top of PointNet++ [4] to adjust the learning based on feature difference. Wang et al. introduce GACNet [10], and the kernel function is obtained with MLP on the difference between spatial and propagated features. Li et al. [28] use distance norm as kernel function and propose adversarial network as a guide to refine the segmentation result.

On the other hand, the kernel can be modeled with locations, and its location can be fixed in place or trainable. Lei et al. [29] adopt a fixed spherical bin kernel to extract the local features. Komarichev et al. [30] transfer the 3D kernel into 2D by projecting the points onto an annular ring which is normal to local geometry. While the point locations of all the above kernels are pre-defined, KPConv [31] generalizes the idea of point convolution and models the kernel point with trainable locations. It also emphasizes the radius neighborhood rather than the kNN.

2.2 Instance segmentation

SGPN [32] is the first proposed framework to solve this problem with a similarity matrix and a confidence map. ASIS [33] explores the mutual aid between the semantic-instance tasks and proposes semantic-aware instance segmentation and instance-fused semantic segmentation. JSIS3D [34] also emphasizes the joint relationship that is modeled by multi-value conditional random fields. 3D-SIS [35] and 3D-BoNet [36] take a different approach by predicting bounding box of each instance and subsequently predict point mask to obtain the segmentation result. MTML [37] applies a multi-task learning strategy, predicting both instance embedding and point offset in 3D space.

Our BEACon network is inspired by the recent success of the attentional convolution for semantic segmentation. However, we realize the relationship between geometry and color plays a more important role when delineating an instance boundary. Thus, we aim to improve the instance segmentation by designing the attentional weight with the embedded boundary information.

3 Our method

This section presents the methods to construct the BEACon network. Section 3.1 generalizes the idea of point set convolution, which provides a guideline of designing B-Conv layer in Sect. 3.2. Section 3.3 details the network structure and the loss function. Section 3.4 explains the necessity behind the vicinity merging algorithm, which is designed for large indoor spaces datasets such as S3DIS.

3.1 Generalization of point set convolution

Given a point cloud with point sets $\mathcal{P} \in \mathbb{R}^{N\times 3}$ and corresponding feature sets $\mathcal{F} \in \mathbb{R}^{N\times C}$, the general point convolution of $\mathcal{F}$ by a kernel g at a point $x \in \mathbb{R}^3$ is defined in KPConv [31] as:

$$(\mathcal{F} * g)(x) = \sum_{x_i \in \mathcal{N}_x,\, f_i \in \mathcal{N}_f} g(x_i - x) f_i \tag{1}$$

where $\mathcal{N}_x \in \mathcal{P}$ ($\mathcal{N}_f \in \mathcal{F}$) is the neighboring point set (feature set) defined around the query point x. However, the kernel function can be generalized to take the difference between features as well. In addition, the input feature for a particular layer can be processed by feature mapping function before the convolution operation, denoted as $\mathcal{M}(\cdot)$.

$$(\mathcal{F} * g)(x) = \sum_{x_i \in \mathcal{N}_x,\, f_i \in \mathcal{N}_f} g(x_i - x, f_i - f) \cdot \mathcal{M}(f_i) \tag{2}$$

Note that the aggregation function for convolution is a summation. This aggregation function can be more general and replaced by other functions such as max pooling. In image processing, the image will have fewer feature elements because of stride operation. In point set convolution, a similar approach is accomplished by sampling query points $\mathcal{P}_q$ from the point set $\mathcal{P}$. Common sampling methods include inverse density sampling [38], furthest point sampling [3,4], and grid down-sampling [31,39].

$$\mathcal{P}_q = \mathcal{S}(\mathcal{P}) \tag{3}$$
$$(\mathcal{F} * g)(x) = \mathcal{A}\big(g(x_i - x, f_i - f) \cdot \mathcal{M}(f_i)\big) \tag{4}$$

The generalized point set convolution can be represented as in Eqs. 3 and 4, where $\mathcal{S}(\cdot)$ is the sampling function, $\mathcal{A}(\cdot)$ denotes the aggregation function, $x_i \in \mathcal{N}_x$, $f_i \in \mathcal{N}_f$, and $x \in \mathcal{P}_q$.

The novelty of our proposed BEACon network is based on the above generalization. We interpret the kernel g as attentional layer, which fuses both local geometry $(x_i - x)$ and color $(f_i - f)$ information, and the convolution operation is achieved by furthest point sampling as our sampling function $\mathcal{S}(\cdot)$.

3.2 B-Conv layer

Embedding boundary information into an attentional matrix is the core operation in BEACon layers, termed B-Conv layer. This information can guide the network to learn more discriminative local features in the neighborhood, as we experimentally verify this claim in Sect. 4.4. In classic image processing, an edge is commonly computed on the basis of gradient of the nearby pixels. Multiple criteria can be used to produce the binary label [40]. However, for point cloud the gradient of either geometry or color alone cannot guarantee the boundary of the desired instance. Rather, the relative relationship between geometry and color difference describes the instance boundary and can provide more clues to the attentional matrix.

With these considerations, we formally define the instance boundary as differences in four spaces: 3D space, color space, geo-feature space and propagated feature space. The difference in 3D space transforms the neighborhood area into a local coordinate system around the query point, while the differences in color space and geo-feature space provide other similarity measures between neighboring points, as shown

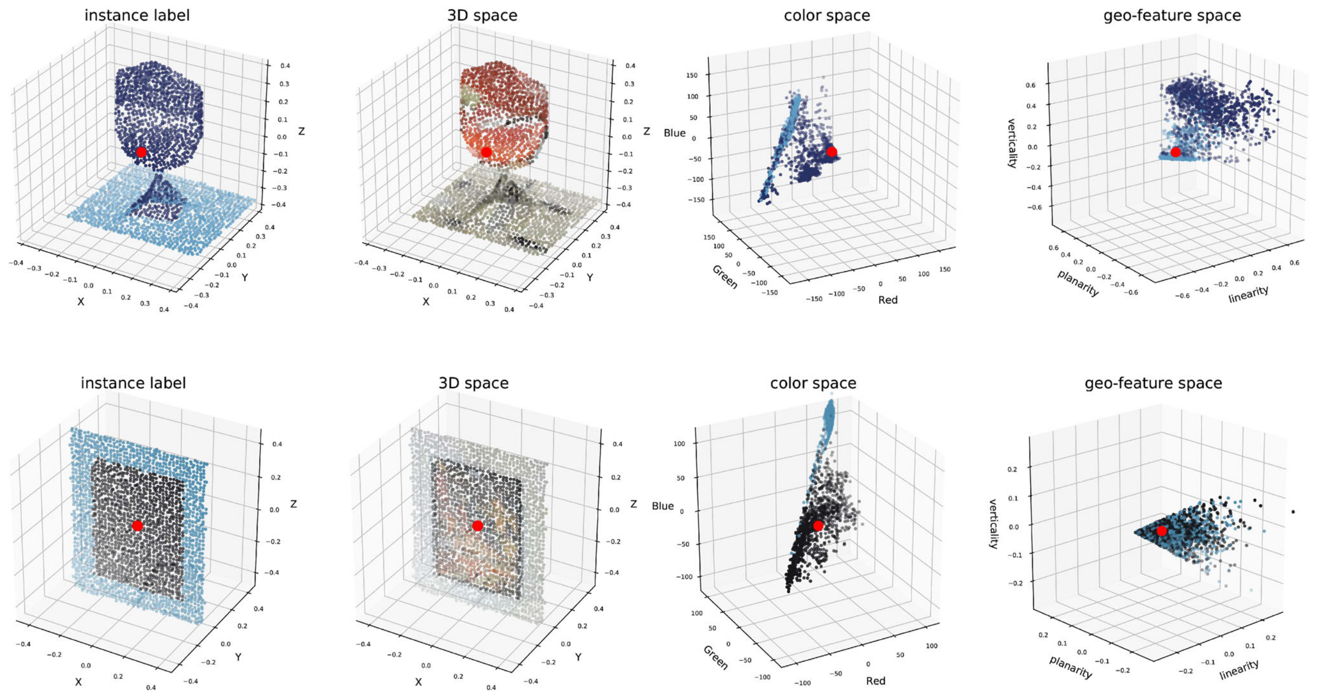


Fig. 1 Visualization of the query point (red dot at the origin) and ΔXYZ, ΔRGB, ΔF_{geo} in 3D, color and geo-feature spaces (scattering is omitted). It can be observed that a picture on the wall can only be separated in color space. To some extent, BEACon learns the "shapes" in all of those spaces and generates the attentional weight by exploring their inter-relationship

in Fig. 1. Since the propagated feature has a better describability of a larger spatial context, a fourth propagated feature space is added if the layer is not an input layer. Intuitively, more attention should be given to nearer points in 3D space, but the network can also adjust its attention based on feature distribution in all the other three spaces.

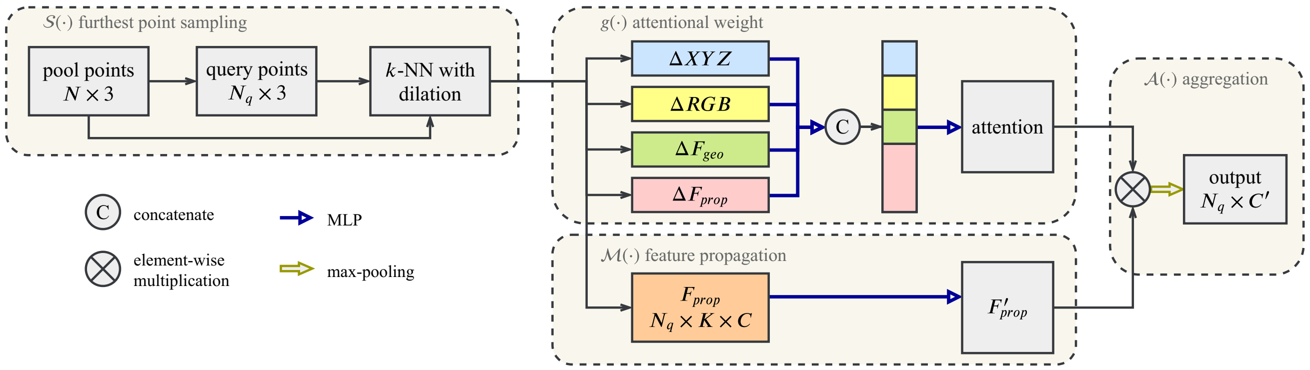
The design of the B-Conv layer can be explained with the generalized point set convolution, as illustrated in Fig. 2. More specifically, $\mathcal{S}(\cdot)$ is furthest point sampling, which is applied to extract the query points with shape $N_q \times 3$ from pool points with shape $N \times 3$. We adopt kNN to search a fixed number of neighbors near query points with a pre-defined dilation rate D. The layer departs from here to generate attentional weight and propagated feature. To calculate the differences, the query point feature is subtracted from neighboring features in four spaces. The kernel function $g(\cdot)$ embeds the instance boundary in two steps. The first step uses 4 MLPs and extracts high level features unique to 4 difference spaces, respectively. After concatenation, the second step uses another MLP to explore the inter-relationship between those features and generate the attentional weight with dimension $N_p \times K \times C'$. To generate the propagated feature, we simply feed the gathered neighborhood feature to an MLP, as $\mathcal{M}(\cdot) = MLP(\cdot)$.

The attentional weight is multiplied element-wise with the propagated feature. Although aggregation function is defined as summation in convolution operation, we experimentally show that $\mathcal{A}(\cdot)$ as max pooling function can learn more discriminative features in the neighborhood.

Relationship to Prior Works As attentional convolution network, BEACon shares similar traits with recent works and is inspired mainly from GACNet [10] and PointWeb [11]. However, there are key differences: (1) BEACon only processes downsampled query points, while graph pooling happens at the end of layer operation in GACNet. (2) During kNN search, BEACon applies the dilation rate to increase the layer receptive field. (3) GACNet learns a weighted average to sum up neighboring features, PointWeb learns a bias to adjust the neighboring features, while BEACon learns a weight matrix to scale the neighboring features. (4) Both GACNet and PointWeb use feature difference as input to attention matrix, while BEACon decouples the difference into separate feature spaces and explicitly models the influence of geometry and color on attentional weight.

3.3 BEACon network

The BEACon network consists of two parallel networks for semantic and instance segmentation (Fig. 3). Semantic branch and instance branch share the same encoder but have different decoders. At the end of the network, the semantic branch generates the semantic probability and the instance



$S(\cdot)$ furthest point sampling
pool points $N \times 3$
query points $N_q \times 3$
k-NN with dilation
$g(\cdot)$ attentional weight
ΔXYZ
ΔRGB
ΔF_{geo}
ΔF_{prop}
C
attention
$\mathcal{A}(\cdot)$ aggregation
output $N_q \times C'$
concatenate
MLP
element-wise multiplication
max-pooling
$\mathcal{M}(\cdot)$ feature propagation
F_{prop} $N_q \times K \times C$
F'_{prop}

Fig. 2 The architecture of B-Conv layer, illustrated in terms of generalized point set convolution

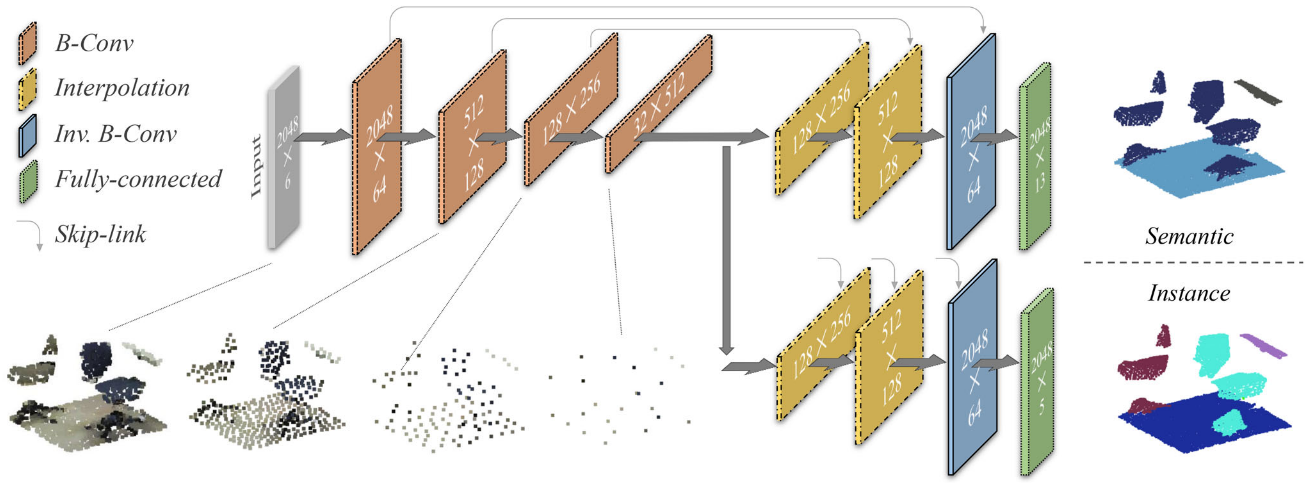


Fig. 3 The architecture of BEACon network for semantic segmentation (top) and instance segmentation (bottom). The number on the encoder layer indicates the size of the output matrix, e.g. , 128 points with 256 features after the third layer

branch generates the embedding of the input point clouds. The initial input feature for a point is composed of XYZ, RGB and geo-features including linearity, planarity, scattering, and verticality. To preserve the finer-scale features, we add skip-links between corresponding layers of the encoder and the decoder.

3.3.1 Interpolation layer

Decoder starts with interpolation layers to restore the scale of the original point cloud. We still use kNN to search for the neighboring points, but in this case, the number of query points is larger than the number of pool points. The interpolated point feature is a linear combination of nearest points, and the weight is calculated as the inverse of point distance.

3.3.2 Inverse B-Conv layer

The inverse B-Conv layer is an interpolation layer followed by the B-Conv layer. The skip-linked feature is concatenated with the interpolated feature as input to the B-Conv layer, and a new neighborhood searching is conducted before the standard operation of the B-Conv layer. To keep the model small and adjust the feature in the neighborhood at the finest scale, we only apply inverse B-Conv layer at the last convolution layer.

3.3.3 Loss functions

The output layer is defined with a simple classifier in mind, with several fully connected layers and dropout layers. During training, the losses are defined separately for the semantic and instance branch, and their sum is used to update the whole neural network.

The semantic branch is supervised by the classical cross-entropy loss. The instance branch, however, does not have a fixed number of labels during the runtime and is adopted with a class-agnostic instance embedding learning, similar to the one in [41]. The loss function can be formulated as follows:

$$L_{\text{ins}} = L_{\text{var}} + L_{\text{dist}} + \eta \cdot L_{\text{reg}} \tag{5}$$

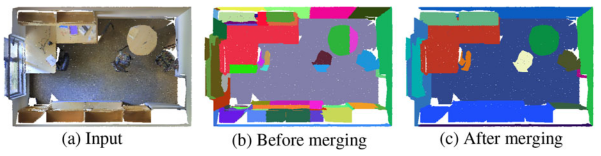


Fig. 4 The effect of vicinity merging. Generally, the connected instances are merged together if they belong to the same category

where L_{var} aims to pull the instance embedding toward its instance center, L_{dist} encourages separation between instance clusters, and L_{reg} is the regularization term. Each term can be further defined as follows:

$$L_{\text{var}} = \frac{1}{I}\sum_{i=1}^{I}\frac{1}{N_i}\sum_{j=1}^{N_i}[\|\mu_i - e_i\|_1 - \delta_v]_+^2 \tag{6}$$

$$L_{\text{dist}} = \frac{1}{I(I-1)}\sum_{\substack{i_A=1,\\ i_A\neq i_B}}^{I}\sum_{i_B=1}^{I}[2\delta_d - \|\mu_{i_A} - \mu_{i_B}\|_1]_+^2 \tag{7}$$

$$L_{\text{reg}} = \frac{1}{I}\sum_{i=1}^{I}\|\mu_i\|_1 \tag{8}$$

where I is the number of ground-truth instances, N_i is the number of points in instance i, μ_i is the mean embedding of instance i, $\|\cdot\|_1$ is the l_1 distance, e_j is the instance embedding of an input point, δ_v and δ_d are margins that define the attractive force and repulsive force, and $[x]_+ = max(0, x)$.

During the test time, we use the Cut-Pursuit algorithm [12] to cluster instance embedding for the entire room. The category of the instance is determined by the mode of the semantic label for that instance.

3.4 Vicinity merging

For a dataset consists of large indoor spaces, it is common to separate the space into smaller volumes. However, this introduces problems—an instance may be divided into multiple parts, and because of the separated geometry, the embedding becomes different even for the same instance. For example, the handle of the chair may be separated from the whole chair structure and can be classified as clutter. To navigate through this problem, we concatenate the predicted semantic label at the end of instance embedding before feeding into the Cut-Pursuit algorithm, making the embedding more consistent if they belong to the same category.

In addition, we propose a vicinity merging process specifically for large indoor spaces dataset. The effect of vicinity merging is shown in Fig. 4. The vicinity merging algorithm is based on a simple rule—if two instances are from the same semantic category and directly connect, they should be merged into one instance. For other special categories, we also use common knowledge to add more rules to the merging criteria. Planarity, for example, is an additional condition for merging wall instances. Note the proposed vicinity merging process is universal to a similar dataset and can be extended to other datasets in which the object instance is connected within itself and separable from others, like common daily objects. In other special cases, hand-crafted rules are needed to achieve better merging performance.

4 Experiments

4.1 Datasets

S3DIS [13] is the primary training source for our BEACon network. It contains 6 areas and 270 rooms, most of which are office room settings. Totally, 13 classes are introduced in this dataset. Each point has both semantic and instance annotations. To make the network less sensitive to scan noise, we first grid downsample the room point cloud with 0.02m. Geo-features are calculated based on the 20-nn search in the entire room. The room is then divided into 1.2m $\times$ 1.2m blocks with two strategies. For semantic branch, each block has an overlap of 0.8m, so each point is predicted three times and the predicted probabilities are averaged. For instance, branch the blocks are sampled in a non-overlapping fashion, so the entire room is predicted exactly once for instance embedding. Each block is further divided into batches with maximum points of 4096. After prediction, the per-point labels are back-projected to the full point set for evaluation purpose.

PartNet [14] consists of 573,585 part instances over 26,671 3D models covering 24 object categories. Semantic and instance annotations can be prepared for each category. The number of part instance per-object ranges from 2 to 220 with an average of 18, and each object consists of 10000 points. Similar to S3DIS, we calculate the geo-feature based on all the points in one object and randomly sample the points into 4 batches with 2500 points.

4.2 Implementation Details of BEACon Network

For the S3DIS dataset, each point is represented with a 10-dim feature vector, including 3D coordinates (XYZ), color (RGB) and geo-features. We define $\delta_v = 0.5$, $\delta_d = 1.5$ and $\eta = 0.001$ for loss function. To augment the dataset, a perturbation of $\pi/32$ on the z-axis and 0.001 scale variance in all directions are used. Adam optimizer is selected for the training with a base learning rate of 0.001 and a decay rate of 0.8 for every 5000 steps. The minimum learning rate is capped at 10^{-6}. The embedding dimension for instance segmentation is set to 5, and regularization strength for Cut-Pursuit is set to 3 with a 5-nn graph. During training, we randomly sample 2048 points from each batch and trained for 60 epochs with

Table 1 Semantic segmentation results on S3DIS, $_6$ is for sixfold cross-validation

Method	oAcc	mAcc	mIoU	$oAcc_6$	$mAcc_6$	$mIoU_6$
ASIS [33]	86.90	60.90	53.40	86.20	70.10	59.30
PointWeb [11]	86.97	66.64	60.28	87.31	76.19	66.73
HPEI [42]	87.18	68.30	61.85	88.20	76.26	67.83
GACNet [10]	87.79	–	62.85	–	–	–
SSP+SPG [25]	87.90	68.20	61.70	87.90	78.30	68.40
KPConv [31]	–	**72.80**	**67.10**	–	**79.10**	**70.60**
BEACon	**88.71**	68.65	62.66	**88.58**	75.88	66.89

The highest score in the category is indicated in bold

Table 2 Instance segmentation results on S3DIS, $_6$ is for sixfold cross-validation

Method	mCov	mWCov	mPrec	mRec	$mCov_6$	$mWCov_6$	$mPrec_6$	$mRec_6$
SGPN [32]	32.7	35.5	36.0	28.7	37.9	40.8	38.2	31.2
ASIS [33]	44.6	47.8	55.3	42.4	51.2	55.1	63.6	47.5
3D-BoNet [36]	–	–	57.5	40.2	–	–	**65.6**	47.6
BEACon	**60.94**	**62.97**	**60.06**	**56.73**	**64.03**	**66.38**	61.51	**62.38**

The highest score in the category is indicated in bold

batch size 4. During testing, we use all the available points in the block as input. For the PartNet dataset, we keep all the settings similar to S3DIS's.

4.2.1 Evaluation metrics

For evaluation of semantic prediction, the accuracy and IoU across all the categories are obtained, and mean accuracy (mAcc) and mean IoU are calculated by averaging the per-class accuracy and IoU. In addition, the overall accuracy (oAcc) is also calculated for all the predicted points.

For instance segmentation, the coverage and weighted coverage are evaluated, along with the precision and recall. Coverage is the average instance-wise IoU of prediction matched with ground-truth. The weighted coverage can be calculated by multiplying the ratio of current ground truth instance points and all the ground truth instances points. The precision and recall are defined with the threshold 0.5, and mean precision (mPrec) and mean recall (mRec) are obtained by averaging the per-category results.

4.2.2 Vicinity merging details

As mentioned in Sect. 3.4, only dataset such as S3DIS requires vicinity merging. For each category, we iterate through all the instances and merge the ones that are directly connected. We repeat this procedure until no instance can be merged anymore. For ceiling and floor, we merge all the instances unselectively. For walls, we first filter out the small instances and then use RANSAC to fit planes to the instances. The instances will only be merged if they belong to the same plane and directly connected. For chairs, instances that are directly connected or have intersections when projected to a horizontal plane will be merged.

4.3 S3DIS results

The evaluation metrics for S3DIS follow the fifthfold validation and sixfold cross-validation. We report our semantic results in Table 1. Although not specifically designed for semantic segmentation, our BEACon network has the competitive performance among the most recent works. We observe that by incorporating both geometry and color information into the attentional layer, the network can have a clearer boundary between object instance, which in turn better delineate the boundary between semantic classes as well. Instance results are shown in Table 2. BEACon network outperforms state-of-the-art methods by a large margin in all four metrics. Compared with ASIS, we achieved more than 15% improvement on mCov and mWConv, with 4.09% improvement on mean precision and 14.56% on mean recall.

We provide some qualitative results of our prediction as shown in Fig. 5. For semantic results, different colors are corresponding to different categories. For instance results, we randomly assign each instance with a color. The color does not have a meaning but as an indication of different instances. Most of the instances can be correctly recalled. However, one drawback of the vicinity merging algorithm is that it makes some classes indistinguishable between two objects which are directly connected.

Computational Time The BEACon network has 2.5 M parameters, which is 56% more parameters than ASIS (1.6 M). However, the Cut-Pursuit algorithm is much faster and processes the whole room point cloud at once. For input with

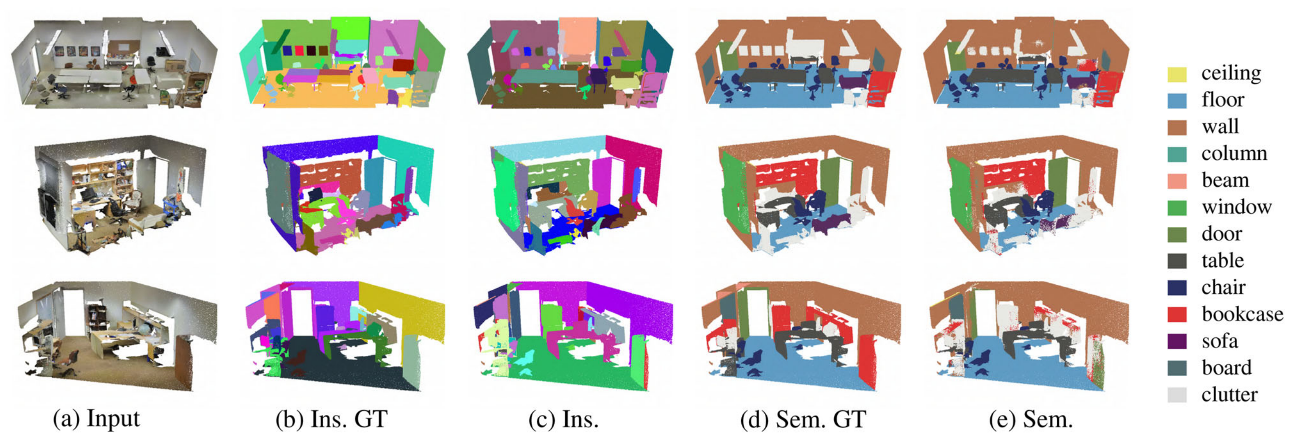


Fig. 5 Qualitative results on S3DIS dataset (Area 5)

Table 3 Computational time analysis run on RTX2080Ti to process point cloud with 4096 points

Method	Parameters	Network (ms)	Grouping (ms)	Merging (ms)	Total (ms)
ASIS [32]	1599090	55	187	11	253
BEACon	2516066	103	–	133	236
geometry only	2321650	100	–	133	233
color only	2321650	100	–	133	233
geo feature only	2321954	101	–	133	234

4096 points in office-39 (last row in Fig. 5), although the BEACon network inference time is 103 ms (55 ms for ASIS), the overall processing time is 236 ms, which is faster than ASIS (253 ms) as shown in Table 3. We also research the time complexity for making use of both geometry and color information. BEACon network inference time is about 3 ms more than the partial attention variant. However, the instance segmentation performance is significantly higher as discussed in Sect. 4.4.1. The network takes about 3–4 h to converge on a single RTX2080Ti.

4.4 Ablation study

We evaluate our design choices by removing or replacing certain components. The results are collectively reported in Table 4. Experiment ⑥ is equivalent to our BEACon approach.

4.4.1 Effect of attention mechanism

To show the effectiveness of our proposed attention mechanism, we specifically designed a baseline network without attention kernel, denoted as ① in Table 4. The attention module is removed, and the input of each layer is concatenated with ΔXYZ to provide localized information. In experiment ② to ④, we conduct studies on partial attention by only inputting single feature difference to generate attentional weight. Experiment ② resembles the attention mechanism commonly used in semantic segmentation, with spatial difference concatenated with propagated feature difference as input of the attentional weight. By embedding color and geometry difference as boundary information, BEACon (experiment ⑥) has an improvement of +1.04 mIoU over experiment ② and has a large performance gain on mPrec (+4.07) and mRec (+1.61). The results indicate that the geometry-color-based attention performs on par with its spatial-only counterpart for semantic segmentation, but can largely benefit the instance segmentation. We also test the weighted sum attention mechanism as in GACNet [10] in experiment ⑤. The weight matrix is normalized across the neighborhood, and the adjusted features are summed up. The results show that max-pooling performs best as the aggregation function.

We further visualize the effect of attention by comparing neighboring point attention around carefully selected query points in experiment ① and ⑥, as shown in Fig. 6. The attention is calculated as the histogram of neighborhood index after the max-pooling operation. In other words, a neighboring point with maximum attention would have most of its features remained after the aggregation function. We extract the result from layer 3, where each query point has 32 neighbors with a dilation rate $D = 2$. Compared to the network without attention, BEACon has a smaller attention spread when the query point is near the edge of the picture and has a larger spread at the center of the picture. While BEACon puts maximum attention on the points that have similar color

Table 4 Ablation studies on S3DIS dataset (Area 5)

id	$g(\cdot)$			$\mathcal{A}(\cdot)$		Clustering		Merging		mIoU	mPrec	mRec
	ΔXYZ	ΔRGB	ΔF_{geo}	Sum	Max	MS	CP	BM	VM			
1							✓		✓	45.59	43.36	40.74
2	✓				✓		✓		✓	61.62	55.99	55.12
3		✓			✓		✓		✓	56.27	51.18	49.93
4			✓		✓		✓		✓	54.93	58.04	50.11
5	✓	✓	✓	✓			✓		✓	60.30	59.14	49.53
6	✓	✓	✓		✓		✓		✓	**62.66**	**60.06**	**56.73**
7	✓	✓	✓		✓	✓		✓		–	57.44	49.24
8	✓	✓	✓		✓	✓			✓	–	57.98	45.69

The highest score in the category is indicated in bold

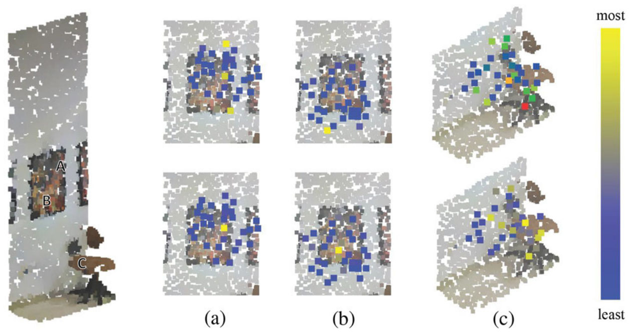


Fig. 6 Comparison between experiment ① (top) and experiment ⑥ on neighboring point attention. The point receiving most attention is colored yellow

and geometry as query point, ① tends to divert the attention randomly. When the query point is on the chair, BEACon puts most of the attention on the structure of the chair, while ① spreads its attention to the wall, causing the wrong feature being aggregated down the line.

4.4.2 Effect of clustering and merging algorithm

We compare our selected clustering algorithm Cut-Pursuit (CP) to another commonly used algorithm MeanShift (MS). One simple strategy is to directly replace Cut-Pursuit with MeanShift in our design. However, the computation complexity of MeanShift increase quadratically as the number of input points goes up. It is unpractical to processing the entire room at once using MeanShift. We therefore use MeanShift only inside each batch. It takes 55 minutes to test and evaluate the entire Area 5, which is 5 minutes longer than BEACon. We also tested the MeanShift with BlockMerging (BM) strategy that is used in [33]. BlockMerging requires the blocks to have overlap. Unlike Cut-Pursuit with vicinity merging (VM), we have to predict instance embedding 3 times in this case. The entire evaluation of Area 5 takes 110 minutes.

The advantage of Cut-Pursuit lies in its effect and speed. Compared to MeanShift which solves the clustering problem with a pure density-based approach, Cut-Pursuit models it as a global optimization of a graph-cut and has a smoother segmentation result. In addition, it can process the entire room at once in a short time. Thus, the Cut-Pursuit is chosen to cluster instance embedding in this research. For merging algorithm, one drawback of BlockMerging is that it requires an overlapped area between processing block and processed blocks. Vicinity merging does not have such a limitation.

4.5 PartNet results

We show the effectiveness of the BEACon network for part instance segmentation using the four largest categories in the PartNet dataset, following the evaluation protocol in GSPN [43] where the third level is used. Note the RGB value provided by PartNet is misleading, so we omit the color as input and embed the boundary information without color information. In fact, the point cloud of the shape is randomly sampled on the CAD model surface. On many models, outer surface and inner surface belong to the same category but have different color. We report our experiment results in Table 5. In PartNet [14], multiple network architectures are tested for the semantic task, and we report the highest score in the paper regardless of the methods used. For a network that simultaneously processes both tasks, BEACon has a close semantic score to the network that specifically designed for semantic segmentation. Our instance segmentation outperforms the best method in PartNet, with a maximum 25.02% improvement on the chair category. We also provide the qualitative results as shown in Fig. 7. With geometry embedded boundary information, BEACon can distinguish the instance based on small geometric differences, like the horizontal support under the table.

Table 5 Segmentation results on PartNet dataset

	Method	Chair	Lamp	Storage	Table
mIoU	PartNet [14]	**43.9**	**25.3**	**49.4**	**33.9**
	BEACon	41.97	25.04	48.15	32.69
mAP	SGPN [32]	19.4	14.6	14.4	21.5
	GSPN [43]	26.8	21.9	18.3	26.7
	PartNet [14]	29.0	18.7	27.5	23.9
	BEACon	**54.02**	**36.37**	**47.55**	**38.32**

The highest score in the category is indicated in bold

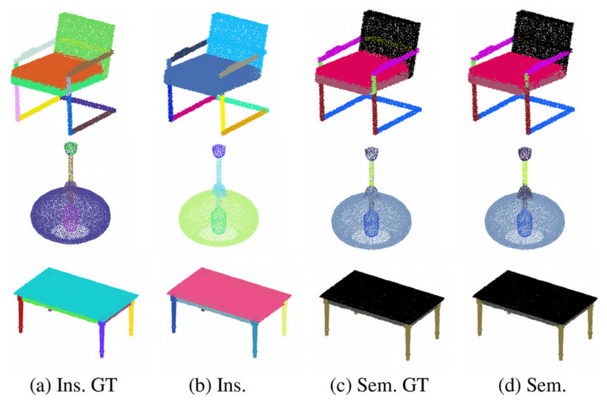


Fig. 7 Qualitative results on PartNet dataset

5 Conclusion

We have presented BEACon, a boundary embedded attentional convolution network for point cloud instance segmentation. We draw inspiration from human perception and model the attentional weight that adapts itself to the relationship between geometry and color difference. Experimental results show that our network can learn a more discriminative feature around the neighborhood and achieve better performance than the state-of-the-art on several benchmarks.

Acknowledgements This research is supported by the National Research Foundation, Singapore, under its International Research Centres in Singapore Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

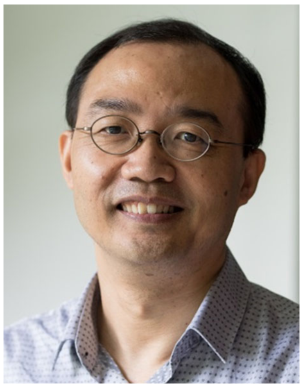
References

1. Haoran, L., Fazhi, H., Yilin, C.: Learning dynamic simultaneous clustering and classification via automatic differential evolution and firework algorithm. Appl. Soft Comput. J. **96**, 106593 (2020)
2. Zhang, S., He, F.: DRCDN: learning deep residual convolutional dehazing networks. Vis. Comput. **36**, 1797–1808 (2020)
3. Li, Y., Bu, R., Sun, M., Chen, B.: PointCNN: Convolution On X-Transformed Points. In: Advances in Neural Information Processing Systems (2018)
4. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems (2017)
5. Rethage, D., Wald, J., Sturm, J, Navab, N., Tombari, F.: Fully-convolutional point networks for large-scale point clouds. In: ECCV (2018)
6. Wu, W., Qi, Z., Fuxin, L.: PointConv: Deep convolutional networks on 3D point clouds. In: CVPR (2019)
7. Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y.: SpiderCNN: Deep learning on point sets with parameterized convolutional filters. In: ECCV (2018)
8. Li, H., He, F., Liang, Y., Quan, Q.: A dividing-based many-objective evolutionary algorithm for large-scale feature selection. Soft Comput. **24**, 6851–6870 (2020)
9. Liu, Y., Fan, B., Xiang, S., Pan, C.: Relation-shape convolutional neural network for point cloud analysis. In: CVPR (2019)
10. Wang, L., Huang, Y., Hou, Y., Zhang, S., Shan, J.: Graph attention convolution for point cloud semantic segmentation. In: CVPR (2019)
11. Zhao, H., Jiang, L., Fu, C.-W., Jia, J.: PointWeb: Enhancing local neighborhood features for point cloud processing. In: CVPR (2019)
12. Landrieu, L., Obozinski, G.: Cut Pursuit: fast algorithms to learn piecewise constant functions on general weighted graphs. SIAM J. Imaging Sci. Soc. Ind. Appl. Math. **10**(4), 1724–1766 (2017)
13. Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S.: 3D semantic parsing of large-scale indoor spaces. In: CVPR, pp. 1534–1543 (2016)
14. Mo, K., Zhu, S., Chang, A.X., Yi, L., Tripathi, S., Guibas, L.J., Su, H.: PartNet: A large-scale Benchmark for fine-grained and hierarchical part-level 3d object understanding. In: CVPR (2019)
15. Maturana, D., Scherer, S.: "VoxNet: A 3D convolutional neural network for real-time object recognition. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2015)
16. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3D ShapeNets: A deep representation for volumetric shapes. In: CVPR, vol. 07-12-June, pp. 1912–1920 (2015)
17. Graham, B., Engelcke, M., Van Der Maaten, L.: 3D semantic segmentation with submanifold sparse convolutional networks. In: CVPR (2018)
18. Choy, C., Gwak, J., Savarese, S.: 4D Spatio-temporal ConvNets: Minkowski convolutional neural networks. In: CVPR (2019)
19. Boulch, A., Guerry, J., Le Saux, B., Audebert, N.: SnapNet: 3D point cloud semantic labeling with 2D deep segmentation networks. Comput. Gr. **71**, 189–198 (2018)
20. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep learning on point sets for 3d classification and segmentation. In: CVPR, pp. 601–610 (2017)
21. Ye, X., Li, J., Huang, H., Du, L., Zhang, X.: 3D Recurrent neural networks with context fusion for point cloud semantic segmentation. In: ECCV, pp. 403–417 (2018)
22. Liu, S., Xie, S., Chen, Z., Tu, Z.: Attentional ShapeContextNet for point cloud recognition. In: CVPR, pp. 4606–4615 (2018)
23. Wang, Y., Bronstein, M.M., Solomon, J.M., Sun, Y., Liu, Z., Sarma, S.E.: Dynamic graph CNN for learning on point clouds. In: ACM Trans. Graph. 1, 1, Article, vol. 1, No. 1, p. 13 (2019)
24. Landrieu, L., Simonovsky, M.: Large-scale point cloud semantic segmentation with superpoint graphs. In: CVPR (2018)
25. Landrieu, L., Boussaha, M.: Point cloud oversegmentation with graph-structured deep metric learning. In: CVPR (2019)
26. Sun, Y., Miao, Y., Chen, J., Pajarola, R.: PGCNet: patch graph convolutional network for point cloud segmentation of indoor scenes. Vis. Comput. **36**, 2407–2418 (2020)
27. Wang, C., Samari, B., Siddiqi, K.: Local spectral graph convolution for point set feature learning. In: ECCV (2018)
28. Li, H., Sun, Z.: A structural-constraint 3D point clouds segmentation adversarial method. Vis. Comput. **37**, 325 (2020)

29. Lei, H., Akhtar, N., Mian, A.: Octree guided CNN with Spherical Kernels for 3D Point Clouds. In: CVPR (2019)
30. Komarichev, A., Zhong, Z., Hua, J.: A-CNN: Annularly convolutional neural networks on point clouds. In: CVPR (2019)
31. Thomas, H., Qi, C.R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L.J.: "KPConv: flexible and deformable convolution for point clouds. In: ICCV (2019)
32. Wang, W., Yu, R., Huang, Q., Neumann, U.: SGPN: Similarity group proposal network for 3D point cloud instance segmentation. In: CVPR (2018)
33. Wang, X., Liu, S., Shen, X., Shen, C., Jia, J.: Associatively segmenting instances and semantics in point clouds. In: CVPR (2019)
34. Pham, Q.-H., Thanh Nguyen, D., Hua Gemma Roig, B.-S., Yeung, S.-K.: JSIS3D: Joint semantic-instance segmentation of 3D point clouds with multi-task pointwise networks and multi-value conditional random fields. In: CVPR (2019)
35. Hou, J., Dai, A., Nießner, M.: 3D-SIS: 3D semantic instance segmentation of RGB-D scans. In: CVPR (2019)
36. Yang, B., Wang, J., Clark, R., Hu, Q., Wang, S., Markham, A., Trigoni, N.: Learning object bounding boxes for 3D instance segmentation on point clouds. In: NeurIPS (2019)
37. Lahoud, J., Ghanem, B., Pollefeys, M., Zurich, E., Oswald, M.R.: 3D instance segmentation via multi-task metric learning. In: ICCV (2019)
38. Groh, Fabian, Wieschollek, Patrick, Lensch, Hendrik P.A.: Flex-convolution million-scale point-cloud learning beyond grid-worlds. In: ACCV (2018)
39. Thomas, H., Deschaud, J.-E., Marcotegui, B., Goulette, F., Le Gall, Y.: Semantic classification of 3D point clouds with multi-scale spherical neighborhoods. In: International Conference on 3D Vision (3DV), pp. 390–398 (2018)
40. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **PAMI–8**(6), 679–698 (1986)
41. De Brabandere, B., Neven, D., Gool, L.V.: Semantic instance segmentation with a discriminative loss function. In: CVPR Workshop (2017)
42. Jiang, L., Zhao, H., Liu, S., Shen, X., Fu, C.-W., Jia, J.: Hierarchical point-edge interaction network for point cloud semantic segmentation. In: ICCV (2019)
43. Li, Y., Zhao, W., Wang, H., Sung, M., Guibas, L.: GSPN: generative shape proposal network for 3D instance segmentation in point cloud. In: CVPR (2019)

Tianrui Liu is currently pursuing his Ph.D. degree at Nanyang Technological University. He received the B.Eng degree in Aerospace Engineering from Nanyang Technological University, Singapore, in 2018. His research interests include computer graphics and computer vision, especially in point cloud processing and augmented reality.

Dr Yiyu Cai is from the School of Mechanical & Aerospace Engineering at Nanyang Technological University, Singapore. His research interests include virtual and augmented reality, robotization, and artificial intelligence and their applications in medicine, construction, and education.



Dr Jianmin Zheng is an associate professor at the School of Computer Science and Engineering at Nanyang Technological University, Singapore. He received his B.S. and Ph.D. degrees from Zhejiang University, China. His recent research focuses on T-spline technologies, intelligent geometric processing, AI for part design and 3D printing, reality computing, and visualization.

Professor Nadia Magnenat Thalmann is presently the director of the Institute for Media Innovation in NTU, Singapore, and the research director of MIRALab at the University of Geneva in Switzerland. She is also the CTO at Dex-Lab Pte Ltd in Singapore which focuses on artificial intelligence and robotics. She has pioneered research in virtual human technology as well as in early social robotics. With her team, she has published more than 700 scientific papers in top journals and conferences, mainly in Computer Graphics and human Face and Body Simulation. She has received several prestigious awards as the Humboldt Research Award and the Eurographics Career Award. She is a life member of the Swiss Academy of Engineering Sciences.