Instance-Aware Contour Learning for Vectorized Building Extraction From Remote Sensing Imagery

Xingliang Huang, Kaiqiang Chen, *Member, IEEE*, Zhirui Wang, *Member, IEEE*, and Xian Sun, *Senior Member, IEEE*

***Abstract*—Extraction of vectorized building instances from remote sensing images has made significant progress in recent years. The extraction of vectorized building maps enables rapid, large-scale updates to geospatial databases. Recently, the contour-based methods have achieved amazing success because their learning goals are more compatible with vectorized polygons. However, existing contour-based approaches achieve contour updates only by learning the local features at each vertex, which can lead to a contour smoothing problem. Besides, these methods achieve classification via features at the center point, which hinders the receptive field at the instance level. Our primary motivation is to overcome existing limitations introduced by local modeling during contour regression while simultaneously improving the instance classification performance. In this article, we utilize instance-level features to guide the contour learning process. An instance-aware contour regression (IACR) module is designed to update the local contour features via cross attention on the instance-level features. Furthermore, based on the IACR module, we propose a novel vectorized building extraction framework BuildingVec, which interacts between the contour regression branch and instance classification branch through a well-designed cascade architecture. We also build a vectorized building dataset with fine-grained categories, named UBCv2 (vec), to benefit the study of vectorized extraction for fine-grained buildings. Experiments on the UBCv2 (vec) dataset demonstrate that BuildingVec achieves state-of-the-art performance compared to both mask-based methods and contour-based methods. When compared to other vectorized extraction methods on the Crowd AI dataset, BuildingVec also achieves a state-of-the-art performance with an AP$_{50}$ of 93.1%.**

***Index Terms*—Building extraction, building vectorization, remote sensing imagery.**

Manuscript received 28 May 2024; revised 24 June 2024; accepted 4 July 2024. Date of publication 9 July 2024; date of current version 24 July 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62331027 and Grant 62076241, and in part by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA0360300. *(Corresponding author: Kaiqiang Chen.)*

Xingliang Huang and Xian Sun are with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100190, China, also with the Key Laboratory of Network Information System Technology, Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China, and also with The University of Chinese Academy of Sciences and the School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: huangxingliang20@mails.ucas.ac.cn; sunxian@aircas.ac.cn).

Kaiqiang Chen and Zhirui Wang are with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100190, China (e-mail: chenkq@aircas.ac.cn; zhirui1990@126.com).

Digital Object Identifier 10.1109/JSTARS.2024.3425725

I. Introduction

Automatic building extraction provides important support for city mapping [1], [2], [3], [4], 3-D reconstruction [5], [6], and urban energy analysis [7], [8], [9]. Large-scale extraction of buildings using remote sensing images provides a low-cost solution to city-scale building mapping. Due to the portability and flexibility, representing buildings in polygonal format is widely adopted in geographic information system, and vectorized building extraction has become a very popular research direction [10], [11], [12], [13].

Vectorized building extraction methods can be divided into two main categories: mask-based segmentation and contour-based regression [13], [14], [15], [16], [17]. Some mask-based segmentation methods [10], [11] first segment the building areas at the pixel level, then utilize postprocessing to derive the instance-level building polygons. The authors in [10] and [11] achieve vectorized building results by adding auxiliary supervision based on the pixel-level segmentation framework. The main approach of this type of method is to add additional constraints specific to the geometric characteristics of the building during the training process to guide the segmentation model to learn masks with polygonal shapes (i.e., clear edges and angles). By introducing the concept of the frame field, the frame field learning [10] proposes to align the predicted frame field to building contours so that the model can learn more regularized masks. The extracted masks show precise building boundaries and corners, which can be postprocessed to derive polygons using the active skeleton model [18]. HiSup [11] imposes additional constraints on the edge lines of the learned masks by introducing the attracted field maps (AFMs) proposed in [12]. The building masks learned by HiSup have clear edge lines that can be converted into high-quality vectorized polygons. Other mask-based methods [19], [20], [21] follow the instance segmentation framework, which first generates box proposals using region proposal network (RPN) [22] and following box heads, then predicts the mask segmentations within these boxes. These methods achieve good performance on instance-level classification because the instance features are finely extracted by the RoIPool [23] or RoIAlign [19].

A major drawback of mask-based methods [10], [11] is that they heavily rely on cumbersome postprocessing to vectorize the extracted mask results. This can be addressed in the contour-based instance segmentation methods [14], [16], [24] since they achieve vectorized results in an end-to-end manner. As shown in

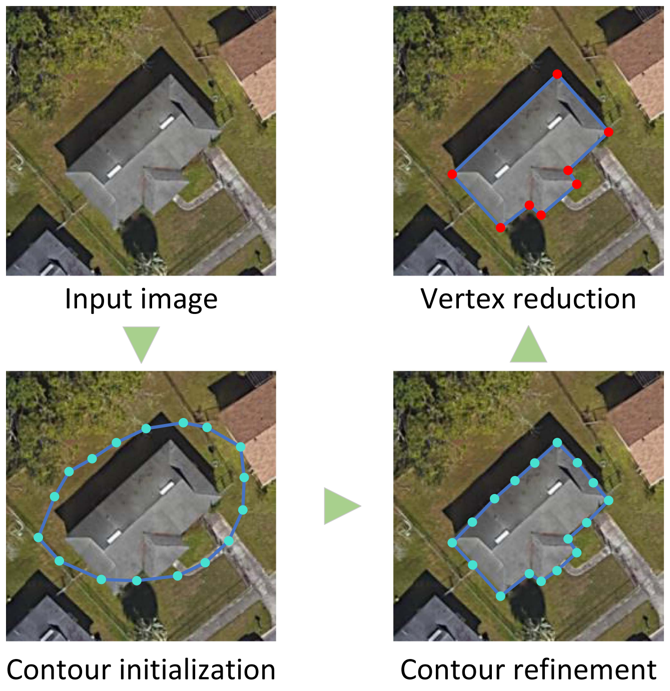

Fig. 1. General contour-based workflow of vectorized building extraction, including three essential stages, i.e., contour initialization, contour refinement and vertex reduction.

Fig. 1, contour-based methods achieve vectorized building extraction through three main steps: contour initialization of each individual building instance; contour refinement to map precise building boundaries; and vertex reduction to produce compact polygons. For example, BuildMapper [13] focuses on vectorized building extraction using contour learning. It refines the initial contours in three iterative stages and removes redundant vertices in the last stage. Most contour-based methods [13], [16], [17], [24], [25] replace the bounding box branch (four points) in one-stage object detection frameworks [26], [27] with a contour prediction branch (a sequence of points). These methods achieve very fast inference speed, but show inferior classification performance due to the lack of instance-level feature modeling.

Although current methods [13], [16], [17], [24] have achieved good results on vectorized building extraction, there are still some flaws that need to be further improved. As shown in Fig. 2, first, existing contour-based methods often result in smoothed contours due to local modeling strategies such as 1-D-convolution [17] and circular convolution [14], [16]. Such local modeling makes it challenging to generate structured polygonal vectors of buildings. Second, mask-based methods have insufficient ability to fit building polygons due to the gap between mask representation and polygon representation. Third, existing modeling approaches encounter difficulties in simultaneously balancing between instance-level representation (e.g., mask-based methods) and contour regression (contour-based methods).

Our primary motivation is to develop a contour-based approach that addresses the contour smoothing issue caused by local modeling while simultaneously enhancing instance-level representational modeling, thereby improving instance-level classification performance. The main idea of our method is illustrated in Fig. 2. In order to achieve high-quality contour regression, we propose a novel approach that injects instance-level features utilized in mask-based methods into the local vertex features during contour learning. Specifically, we use instance-level contexts to guide the local vertex features during the contour learning process. An instance-aware contour regression (IACR) module is designed to update the local contour features via cross-attention on the instance-level features. Furthermore, based on the IACR module, we propose a novel vectorized building extraction framework BuildingVec, which interacts between the contour regression branch and instance classification branch through a well-designed cascade architecture.

It is important to note that the lack of fine-grained vectorized building dataset limits the development of fine-grained building extraction algorithms. Most of the existing building extraction datasets only provide annotations in the form of masks, which are not vectorized. And there are also a few datasets designed for vectorized building extraction. However, there is no vectorized dataset that provides fine-grained categories of building classification. The UBCv2 dataset [28] provides fine-grained masks for buildings. We propose a vectorized fine-grained building extraction dataset based on the UBCv2 dataset, which we call UBCv2 (vec). This dataset allows for the validation of vectorized building extraction algorithms as well as building classification methods.

We conduct plenty of experiments on UBCv2 (vec) [28] and Crowd AI dataset [29], and the results show that the proposed BuildingVec method, has both promising classification and building contour regression performance.

The main contributions of this article are summarized as follows.

1) We propose an IACR module to improve the contour quality via the integration of instance-level contexts and vertex-wise local features.
2) We propose a novel framework BuildingVec, which achieves the vectorized building extraction and fine-grained classification through a well-designed cascade architecture.
3) To address the lack of fine-grained vectorized building extraction dataset, we construct a new vector extraction dataset with fine-grained categories called UBCv2 (vec).

II. Related Work

In this section, we briefly review the related works focusing on building extraction, including pixel-level segmentation, instance-level segmentation, and contour-based methods.

A. *Pixel-Level Segmentation*

Pixel-level building extraction methods are mainly based on semantic segmentation. A popular solution for vectorized building extraction is to add auxiliary constraints to the optimization objectives to achieve the structured polygonal contours of buildings. A topography-aware loss is proposed in [30] to enhance the preservation of building boundary by deep convolutional neural

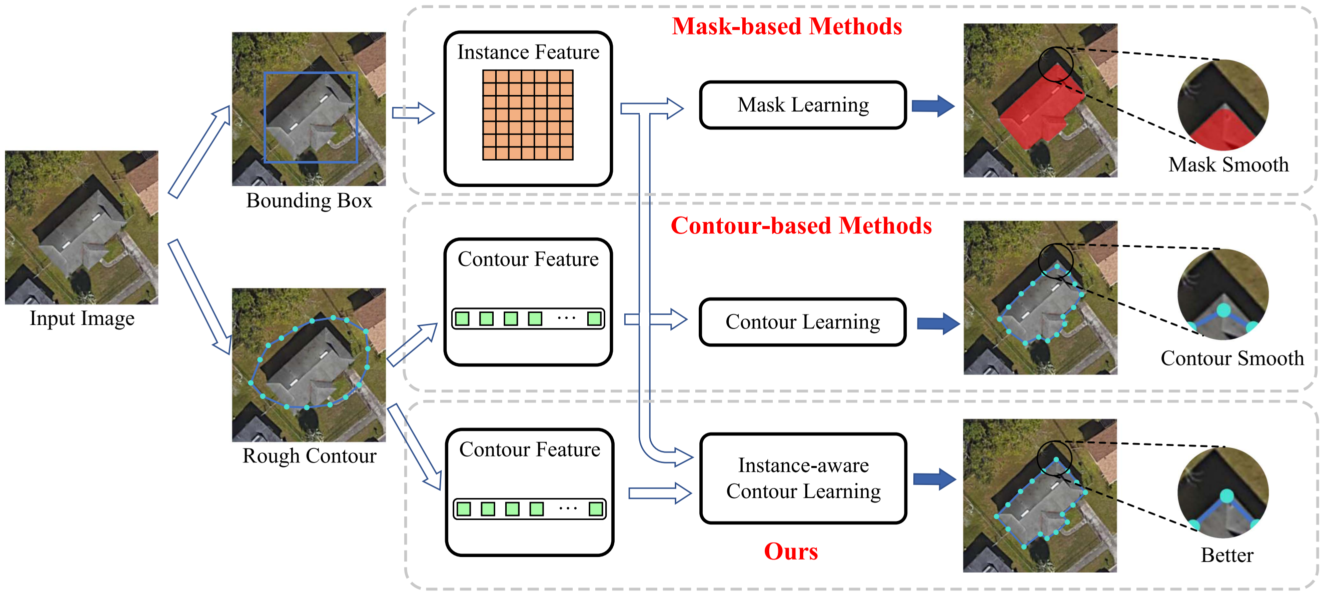

Fig. 2. Comparisons between mask-based instance segmentation methods, contour-based methods, and our proposed methods.

networks (CNNs). The frame field learning [10] proposes a frame field-based pixel-level constraint to align the building contours with the frame field, and obtains well-structured building masks using multitask learning. HiSup [11] accomplishes polygonal mapping of buildings with the help of AFMs [12] in the line segment detection. Although the aforementioned methods based on pixel-level segmentation have achieved promising building segmentation, they still struggle to address the challenge of separating individual building instances and optimization of cumbersome hyperparameter settings.

B. Instance-Level Segmentation

Instance segmentation shows great advantages in detecting and segmenting individual instances by referring to the paradigm of object detection. Currently, prevailing instance segmentation methods are mainly divided into one-stage and two-stage methods.

For single-stage methods, SOLO [31] and SOLOv2 [32] learns the instance masks directly according to the location and size of instances and does not depend on the bounding boxes. In another one-stage method YOLACT [33], the instance masks are calculated by weighting several prototype masks using coefficients, which achieves efficient parameter sharing across different instances.

Two-stage methods are designed with the “detect and segment” paradigm, which segment every instance using the features within the detected bounding box. Mask R-CNN [19] predicts the instance masks using a mask head, which is parallel to the bounding box regression branch. Instance features are first obtained by pooling methods such as RoIAlign [19] or RoIPooling [23], then the instance masks are learned by fully convolutional networks. PANet [34] improves the performance of the segmentation by adding a bottom-up path to the feature pyramid network (FPN). Cascade Mask R-CNN [20] is designed with multiple detection heads to refine the bounding boxes and masks of the instances in a cascade way, resulting in high-quality detection results. The HTC [21] further improves the performance by designing an interleaved cascade path and adding information flow among different mask prediction heads.

The building boundaries predicted by instance segmentation algorithms are often irregular and far from the vectorized annotation. Further postprocessing is required to derive the contours and eliminate redundant nodes. The Douglas–Peucker (DP) algorithm [35] is a popular method to simplify polygonal contours derived from building masks. In our proposed method, a corner classification head is added at the last contour regression head to select corner nodes. Such a method is fully end-to-end trainable and more robust than manually designed postprocessing.

C. Contour-Based Methods

A few researchers have considered the instance segmentation as a contour regression problem. For contour regression [13], [16], [17], [24], [25], the mask is interpreted as a sequence of vertices of the polygonal contour. Such methods have a great advantage in modeling the vectorized building polygons since the learning objective is more consistent with the manually annotated building polygons. It also avoids redundant raster-to-vector transformation and empirical postprocessing in the mask-based methods [10], [11].

The traditional contour-based methods [18], [36] extract the instance contours using an active contour model, which optimizes an empirically designed energy function. With the development of deep neural networks, CNNs are incorporated with the active contour model [37], [38] to improve the quality of generated contours. Another kind of methods [39], [40], [41], [42], [43] adopt recurrent neural networks (RNNs) to predict the vertex sequence of buildings in an autoregressive way. However, they usually miss the corner vertices, resulting in rough contours.

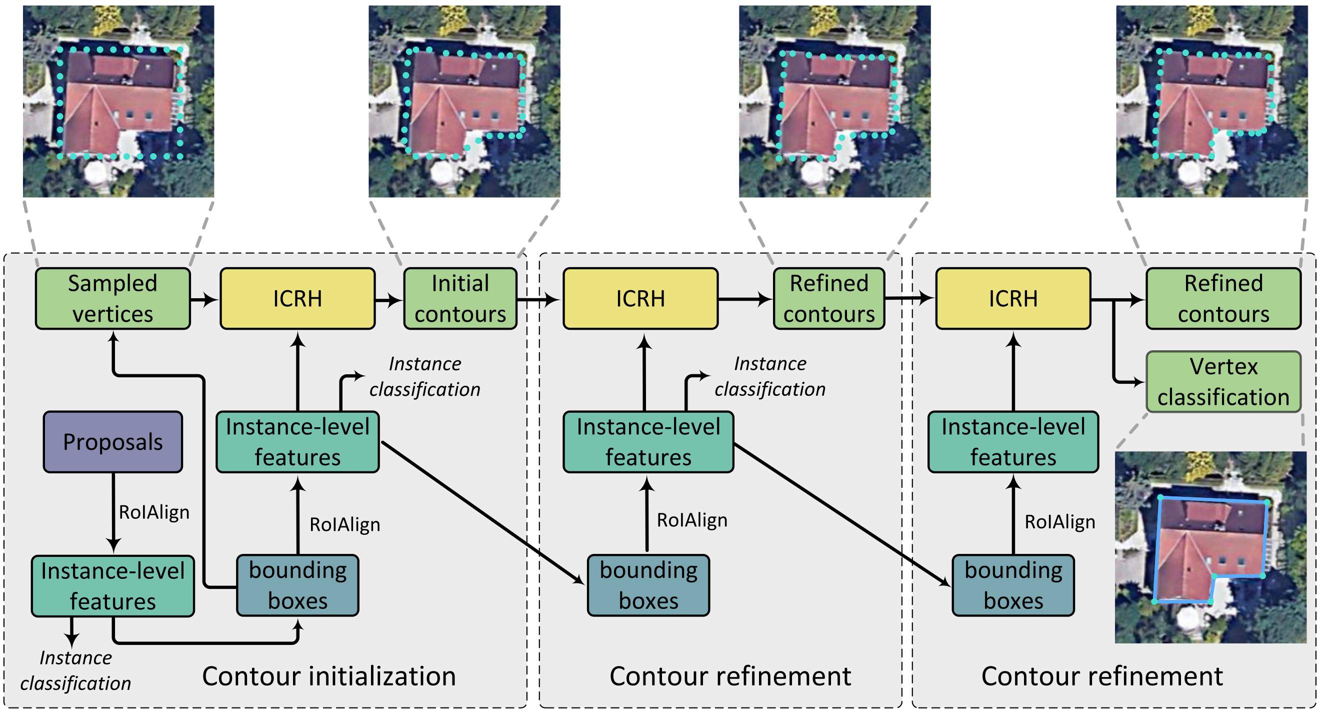

Fig. 3. Overview framework of BuildingVec. It produces the initial contours using the vertices sampled from the bounding boxes. Then, the IACR module refines the contours using both the instance-level features derived from the bounding boxes and vertex features derived from the vertex coordinates. In final, a vertex classification branch predicts the probability scores for each vertex to determine if it is a corner. In this framework, the contour branch and the bounding box branch are jointly optimized in a cascading manner.

Recently, CNN-based methods have become the most popular solution for contour regression problem. PolarMask [24] and PolarMask++ [25] encode the instance contour as a sequence of polar coordinates at multiple directions with fixed angular intervals. Those polar coordinates can be efficiently converted back to the instance mask. However, it can only roughly mimic contours because complex contours may cross a single direction multiple times. LSNet [44] solves this problem by directly predicting uniformly sampled contours using a fixed number of offsets. More recent contour-based methods are proposed, such as Curve GCN [15], Deep Snake [16], DANCE [14], and E2EC [17]. These methods have been improving the quality of contours for recent years.

Contour-based methods have been applied in extracting the buildings in remote sensing images. BuildMapper [13] is proposed to extract vectorized building contours in an end-to-end framework. PolyBuilding [45] directly decodes all building polygons using an deformable transformer decoder [46]. They have achieved promising vectorized contours in remote sensing images. However, BuildMapper predicts the classification score from the point feature in the center position, which lacks global perceptive field of the building instance. And the vertex regression ability is limited by local operators such as 1-D convolution. Besides, the contour refinement of different stages is still dependent in these methods. We introduce a bounding box branch parallel to the contour regression branch. The contours and bounding boxes are jointly refined by the instance-level features extracted by the boxes. Such an operation allows instance-level perception for the classification and contour refinement branch, respectively. And we add information flow among different contour refinement stages to allow efficient contour learning.

III. Algorithm Design

In this section, we first introduce the overview framework of the BuildingVec in Section III-A. Then, a novel IACR module is introduced in Section III-B. Next, three contour learning stages including contour initialization (see Section III-C), contour refinement (see Section III-D), and vertex reduction (see Section III-E) are described. In final, we introduce the information flow and loss design in Sections III-F and III-G, respectively.

A. Overview

The proposed framework for building extraction is illustrated in Fig. 3. The contour branch and the bounding box branch are jointly optimized in a cascade manner. First, the initial bounding boxes are predicted according to the instance-level features obtained from the RPN [22]. Then, rough contours are initialized from the uniformly sampled points from boxes. An IACR head predicts the initial contours using the vertex features and the instance-level features obtained by RoIAlign [19]. Next, more accurate instance-level features can be obtained from the refined bounding boxes, which generate instance-level features for prediction in the next stage. In the contour refinement stage, the IACR module takes the vertex features obtained by contour coordinates in the previous stage as input, and predicts an offset for each vertex. The instance-level features are also used to guide this refine process. Finally, a vertex classification head is added

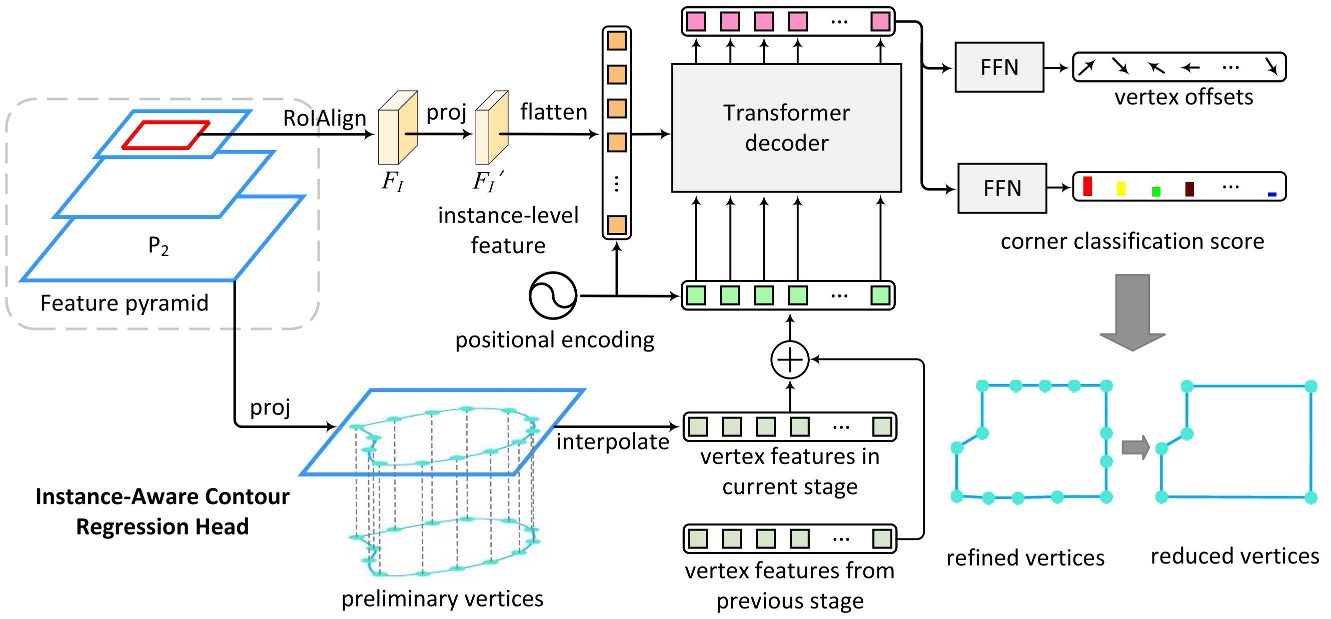

Fig. 4. Detailed process of the IACR module. The vertex features are derived by interpolating on the bottom feature map of FPN. If vertex features from the previous stage are available, the vertex features are fused by adding the current and previous vertex features. The instance-level features are obtained by the instance bounding boxes through RoIAlign operation, and then be flattened to a sequential features. PEs are added to both vertex features and instance-level features. Then, a transformer decoder predicts the revolved features of the vertex features by interacting vertex features and instance-level features. Finally, two FFNs predict the vertex offsets and corner classification scores, respectively. The contour vertices are refined by adding the predicted offsets with their previous coordinates. In the last stage, interpolated vertices are reduced by the predicted corner classification score.

to distinguish corner vertices in the final stage. The simplified polygons are generated by removing noncorner vertices using a threshold. The BuildingVec framework is fully end-to-end trainable and can be optimized without postprocessing.

B. *Instance-Aware Contour Regression (IACR)*

We propose an IACR module to learn the high-quality contours using the vertex features as well as the instance-level features. The main motivation of this module is that contour features are often distributed at the boundary regions with limited representability. Instance-level features provide a global perspective on the instance, which is crucial for contour learning, especially for contour initialization. To fully couple the vertex features and the instance-level features, a decoder-only structure based on the transformer is designed to learn the contour refinement offsets.

As shown in Fig. 4, the IACR module learns the refined contour according to the preliminary contour and instance-level features. Specifically, given a preliminary contour C and a corresponding bounding box B, the contour C can be denoted as a sequence of points on the image: $\{(x_i, y_i)|i = 1, 2, \dots, N\}$. According to the box, the RoIAlign extracts the instance-level features $F_I \in \mathbb{R}^{M\times M\times C_p}$ from the feature pyramid encoded by the FPN [47], where M is the fixed spatial size and C_p is the output dimension of the FPN. The feature pyramid consists of feature maps with multiple strides, which are denoted as $P_i \in \mathbb{R}^{\frac{H}{s_i}\times\frac{W}{s_i}\times C_p}$, where H and W are the height and width of the input image, s_i is the stride, and i indicates the order of the level. The vertex features are obtained on the P_2 with s_2, which is equal to 4 pixels.

A decoder-only structure based on the transformer [48] is designed to combine the instance-level features and vertex features. First, the instance-level features are flattened to a 1-D sequence, denoted as $F_I' \in \mathbb{R}^{MM\times C_p}$. We use a linear layer to project the dimension of the 1-D-sequence F_I' from C_p to C_v, reducing the computational cost in the following transformer decoder. In this article, C_v is empirically set to 128. Then, we also down-dimension P_2 to C_v to derive the vertex features. Specifically, the vertex features are computed by interpolating on the P_2 using the vertex coordinates. The feature sequence is denoted as $F_v \in \mathbb{R}^{N\times C_v}$, where N is the number of vertices per building. Next, the transformer takes the instance-level feature sequence F_I' and the vertex feature sequence F_v as inputs and produces the refined vertex feature sequence F_r. A learnable positional encoding (PE) is added to F_I' and F_v before they are fed into the Transformer decoder. Finally, a feed-forward network (FFN) predicts the vertex offsets of the preliminary vertices, denoted as $\{(\Delta x_i, \Delta y_i)|i = 1, 2, \dots, N\} \in \mathbb{R}^{N\times 2}$. The final refined vertices are given by $\{(x_i', y_i') = (x_i + \Delta x_i, y_i + \Delta y_i)|i = 1, 2, \dots, N\}$. For the final stage, an additional FFN is employed to predict the vertex corner classification score, which is introduced in Section III-E.

C. *Contour Initialization*

In the contour initialization stage, the bounding boxes are predicted according to proposals generated by the RPN in the same manner as Faster R-CNN [22]. The instance-level features are computed using RoIAlign [19] according to the bounding boxes. Horizontal bounding boxes are used to identify all

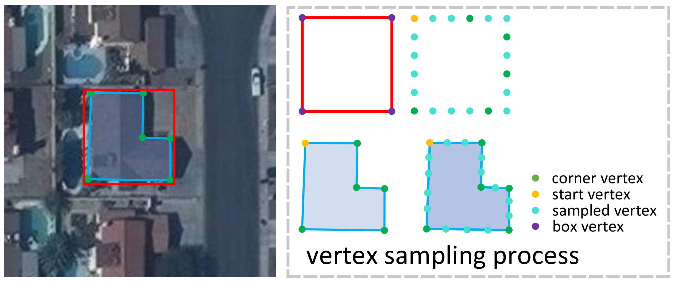

Fig. 5. Example of the vertex sampling process. Only one building instance is illustrated.

buildings uniformly, including irregularly shapes or rotated bounding boxes. To create a preliminary contour for the initialization stage, we first sample the bounding boxes uniformly, treating them as polygons with four vertices. Fig. 5 shows an example of sampling uniform vertices from a box. Given a bounding box B with a size of height h and width w, N vertices are sampled uniformly, starting from the upper left corner in a clockwise direction. The sampling stride between two adjacent vertices can be computed as follows:

$$s_{\text{sample}} = \frac{2(h+w)}{N}. \tag{1}$$

We denote the sampled vertices from bounding boxes as $\{(x_i^{\text{sample}}, y_i^{\text{sample}})|i = 1, 2, \ldots, N\}$.

The initial contours are generated based on the sampled vertices from bounding boxes. Specifically, using the instance-level features derived from bounding boxes and vertex features derived from sampled vertices, the ICRA predicts the offsets for the initial contours, which are denoted as $\{\Delta_i = (\Delta x_i, \Delta y_i)|i = 1, 2, \ldots, N\}$.

The supervision of vertices in all stages is calculated from the ground truth polygons. Specifically, vectorized building dataset provides polygonal annotations instead of masks. As the number of vertices in the annotated polygons varies, sampling of the polygon annotations is also essential. Fig. 5 shows an example of sampling on the building contour. Generally, the number of vertices in ground truth polygons differs from the sampled vertices N. Given a polygon with k number of vertices, we only need to sample $N - k$ vertices because we have preserved all corner ones. The number of vertices sampled for each side of a polygon is assigned proportionally based on the side length, following the principle of integer assignment. The start vertex for the sampled polygon is still the vertex in the upper left corner, and the final polygon is arranged clockwise. The sampled polygons from the ground truth are denoted as $\{(x_i^{gt}, y_i^{gt})|i = 1, 2, \ldots, N\}$.

To achieve scale invariance for buildings of varying sizes during the coordinate regression at the initialization stage, we normalize the target offsets by the size of the bounding boxes as follows:

$$t_i^x = \frac{x_i^{\text{gt}} - x_i^{\text{sample}}}{w_i}, t_i^y = \frac{y_i^{\text{gt}} - y_i^{\text{sample}}}{h_i}. \tag{2}$$

The target for all vertices can be described as $\{t_i = (t_i^x, t_i^y), i = 1, 2, \ldots, N\}$. The loss calculation for contour initialization can be defined as

$$L_{\text{init}} = \frac{1}{N}\sum_{i=1}^{N} l_1\left(t_i - \Delta_i\right) \tag{3}$$

where l_1 indicates the smooth L_1 loss proposed in [23].

D. Contour Refinement

In the contour refinement stage, the IACR takes into the vertex features in the previous stage and instance-level features computed in the current stage. Similar to the contour initialization process, given the contour $\{p_i^{\text{pre}} = (x_i^{\text{pre}}, y_i^{\text{pre}})|i = 1, 2, \ldots, N\}$ from the previous stage, it predicts the offsets $\{\Delta_i = (\Delta x_i, \Delta y_i|i = 1, 2, \ldots, N\}$. The main distinction is that the target offsets are encoded using absolute positions, instead of normalization by the bounding boxes. Since the initialized contours have already been close to the ground truth, such an operation will bring more accurate regression. Therefore, the loss for refinement stage can be described as follows:

$$L_{\text{refine}} = \frac{1}{N}\sum_{i=1}^{N} l_1\left(\Delta_i + p_i^{\text{pre}} - p_i^{\text{gt}}\right) \tag{4}$$

where p_i^{gt} denotes $\{(x_i^{\text{gt}}, y_i^{\text{gt}})|i = 1, 2, \ldots, N\}$.

E. Vertex Reduction

In the final stage, an additional FFN is employed to predict the vertex corner classification score. The corner classification score is supervised by the ground truth label using cross-entropy loss:

$$L_{\text{cnr}} = \frac{1}{N}\sum_{i=1}^{N}\left(-c_i^{\text{gt}}\log c_i - \left(1 - c_i^{\text{gt}}\right)\log(1 - c_i)\right) \tag{5}$$

where $c_i^{\text{gt}} \in [0, 1]$ denotes the ground-truth corner label and c_i denotes the predicted corner classification score. During inference, a corner threshold S_{cnr} is used to remove redundant vertices. In this article, we set S_{cnr} as 0.1.

F. Information Flow Among Vector Features

We find that the vertex features of all stages are independent. This dependence presents challenges for the optimization process since it cannot directly propagate the gradients from later stages to the parameters in prior stages. Fig. 6 depicts the comparison of with/without information flow in vertex features across various stages. In existing contour-based methods, the features used to learn vertex offsets are extracted independently from the backbone feature map. The connection between two adjacent stages is only at the coordinate addition. We add a feature addition branch in parallel to the coordinate addition branch by adding the vertex features from the previous stage to the vertex features in the current stage. This operation allows the refinement process to be linked across all stages at the feature level, not just at the coordinate level.

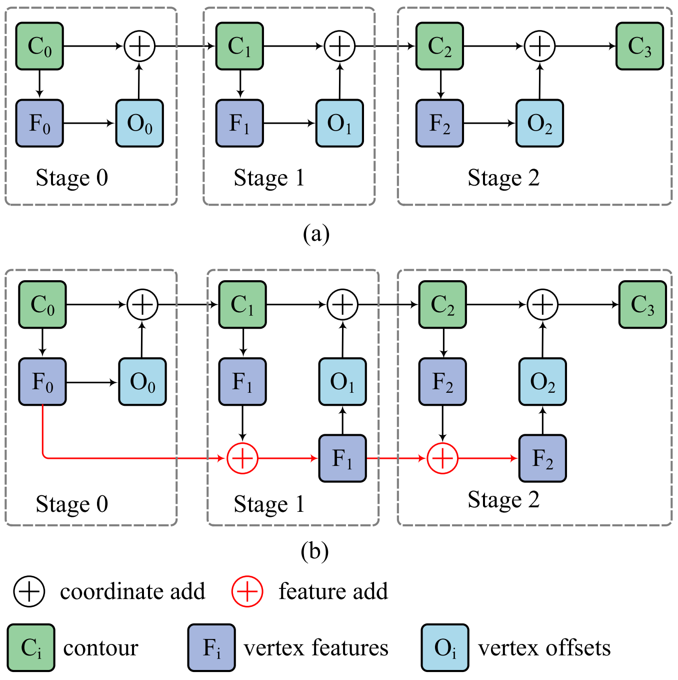

Fig. 6. Information flow in vertex features across all stages. The red arrows indicate the information flow in vertex features. (a) Without information flow in vertex features. (b) With information flow (red arrow) in vertex features.

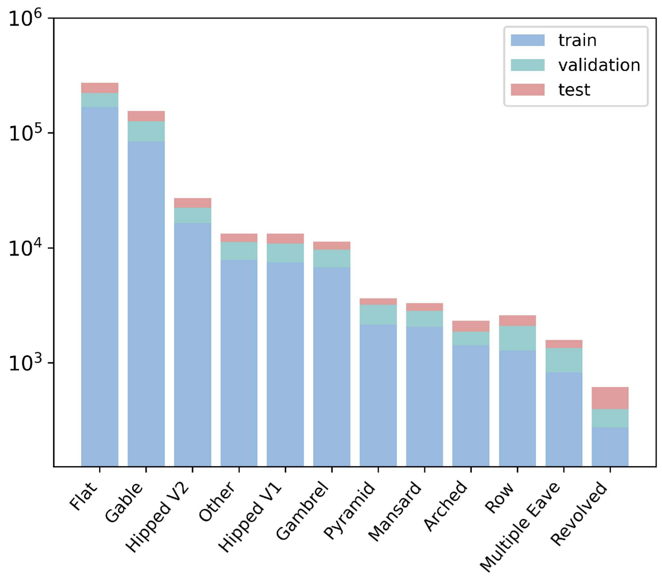

Fig. 7. Quantitative distribution of building instances per category in the UBCv2 (vec) dataset. The categories are defined according to the rooftop structure, e.g., flag, gable, hipped, and others.

G. Training Losses

Besides losses for the contour initialization and contour refinement, the bounding boxes and building classification scores are also jointly optimized in the whole framework. We adopt the smooth L_1 loss [23] for the bounding box learning, denoted as L_{box}. And the cross entropy loss is adopted as the classification loss for every building instance

$$L_{\text{cls}} = \frac{1}{C} \sum_{i=0}^{C-1} (-y_i \log p_i - (1 - y_i) \log(1 - p_i)) \tag{6}$$

where $y = [y_0, y_1, \ldots, y_{C-1}]$ denotes the one-hot representation of a category label. If it belongs to the ith class, then $y_i = 1$ and $y_j = 0$ for $j \neq i$. And y_i denotes the probability that the instance belongs to the ith class as predicted by the model. C denotes the number of classes.

During training, all the modules, including the bounding box head, classification head, and contour prediction head, can be jointly optimized using the following formulation:

$$L = \lambda_{\text{box}} L_{\text{box}} + \lambda_{\text{cls}} L_{\text{cls}} + \lambda_{\text{init}} L_{\text{init}} + \lambda_{\text{refine}} L_{\text{refine}} \tag{7}$$

where $\lambda_{\text{box}}, \lambda_{\text{cls}}, \lambda_{\text{init}}$, and λ_{refine} are weights to balance the losses, set to 1.0, 1.0, 20, and 0.02, respectively.

IV. Experiments

A. Datasets

It is important to note that the lack of fine-grained vectorized building dataset limits the development of fine-grained building extraction algorithms. Most of the existing building extraction datasets only provide annotations in the form of masks, which are not vectorized. And there are also a few datasets designed for vectorized building extraction. However, there is no vectorized dataset that provides fine-grained categories of building classification. The UBCv2 dataset [28] provides fine-grained masks for buildings. We propose a vectorized fine-grained building extraction dataset based on the UBCv2 dataset, which we call UBCv2 (vec). This dataset allows for the validation of vectorized building extraction algorithm as well as building classification methods.

We evaluate the performance of BuildingVec and state-of-the-art baselines on two datasets, i.e., the UBCv2 (vec) dataset and Crowd AI mapping challenge dataset [29]. We first introduce the details of the UBCv2 (vec) dataset, because UBCv2 (vec) is a vectorized dataset with requirements for the fine-grained building classification, which is an important motivation of the proposed method. Then, we introduce the Crowd AI dataset, which is commonly adopted in other state-of-the-art methods focusing on vectorized building extraction.

1) UBCv2 (vec) Building Extraction Dataset: The UBCv2 dataset [28] is a large-scale dataset for building extraction that requires the prediction of fine-grained rooftop categories. It is a novel dataset that focuses on building classification in remote sensing images. Buildings in the UBCv2 dataset are categorized into 12 classes according to the roof structures, i.e., flat, gable, hipped, pyramid, etc. However, the building instances are only provided in the format of masks, which do not satisfy the requirements of vectorized building extraction. Thus, we propose a vectorized version of the UBCv2 dataset, referred to as UBCv2 (vec) dataset, by recovering the corner vertices in the vectorized polygons. The number of instances per category in the UBCv2 (vec) dataset is given in Fig. 7. The long-tail distribution in UBCv2 (vec) dataset poses a significant challenge for the classification capability of algorithms. Overall, there are 508 277 vectorized building instances annotated from 10 460 remote sensing images in the UBCv2 (vec) dataset. These images are divided into 6112 for training, 2255 for validation, and 2093 for testing. The vectorized dataset is built and organized into shapefiles, which can be readily converted into MS-COCO [49] format. The UBCv2 (vec) dataset is publicly available online[1] for download.

2) Crowd AI Dataset: The Crowd AI dataset [29] is a building extraction dataset comprising large amount of samples, but those samples lack diversity and category labels. Considering this dataset is a commonly used dataset and has been reported by several related building extraction studies. We also evaluate our model on this dataset. The crowd AI dataset contains 280 741 images for training and 60 317 images for testing. It provides images with RGB bands, which has a size of 300 $\times$ 300 pixels. The building instances are annotated in the MS-COCO [49] format, and represented in polygons, which can be used to evaluate the performance of vectorized building extraction models.

B. Evaluation Metrics

In this article, the standard COCO [49] metrics are used to measure the quality of predicted results. To compare with both mask-based methods and contour-based methods, we choose the average precision (AP) of the mask and boundary as the main metrics, denoted as AP^{mask} and $\text{AP}^{\text{boundary}}$, respectively. The proportion of true positives correctly detected by the model is also quantified by the average recall (AR) metric. Specifically, AP^{mask} is measured by the intersection of union (IoU) between the predicted masks and ground-truth masks. In terms of boundary-level metric, the IoU between the predicted boundaries and ground-truth boundaries is measured by a specific width as described in [50]. If the IoU between a prediction and a ground truth is greater than a specific threshold (i.e., 0.5), the prediction can be treated as a true positive. Both the AP^{mask} and $\text{AP}^{\text{boundary}}$ are computed by the ratio of true positives and the sum of true positives and false positives over ten IoU thresholds ranging from 0.5 to 0.95, with a step of 0.05. For the overall AP of all classes, mAP is calculated as follows:

$$\text{mAP} = \frac{1}{C}\sum_{i=1}^{C}\text{AP}_i \quad (8)$$

where C denotes the number of classes and i denotes the class index.

In addition to evaluating the performance of the models, several metrics are employed to assess the efficiency of the compared models. These include floating point operation (FLOP), frames per second (FPS), and parameters. The FLOP denotes the number of floating-point operations and is understood to be the computational cost. It can be used to measure the complexity of models. The FPS indicates the number of frames per second that can be processed by the model during inference. Furthermore, the number of parameters included in the model is reported.

[1][Online]. Available: https://github.com/AICyberTeam/UBC-dataset/tree/UBCv2

C. Implementation Details

We implement different resize strategies for the two datasets. For the UBCv2 (vec) dataset, multiscale training and random flipping are implemented. Specifically, training images are randomly resized to scales ranging from 800 $\times$ 800 pixels to 1344 $\times$ 1344 pixels. And all test images are resized to 1024 $\times$ 1024 pixels. For the Crowd AI dataset, all images, including training and testing, are rescaled to 320 $\times$ 320 pixels. During training, images of both datasets are randomly rescaled horizontally and vertically.

We implement BuildingVec using the MMDetection [51] toolbox as it has integrated many baselines. All experiments are conducted on 4 $\times$ NVIDIA A40 48-GB graphics processing unit (GPU) with a batch size of 2 on every GPU. The AdamW optimizer is used to optimize the entire model. The learning rate is set to 0.0001 with a weight decay of 0.0001. For both datasets, we train BuildingVec for 24 epochs and decay the learning rate by 0.1 at the 18th epoch. In order to assess the running efficiency metrics of the models, we conducted tests on a single A40 GPU with an input image size of 1024 $\times$ 1024 pixels for all models.

D. Results

In this section, we first compare BuildingVec with other related state-of-the-art methods focusing on building extraction on the UBCv2 (vec) dataset and Crowd AI dataset. Then, we conduct our ablation study on the UBCv2 (vec) dataset.

1) Quantitative Comparison With State-of-The-Art Methods: We first report results on the UBCv2 (vec) dataset. We compare related methods including mask-based methods (i.e., SOLOv2 [32], QueryInst [52], Mask R-CNN [19], Cascade Mask R-CNN [20], and HTC [21]) and contour-based methods (i.e., PolarMask [24], Deep Snake [16], and E2EC [17]) for mAP, AP_{50}, and AP per category in both mask- and boundary level. In experiments, all models adopt ResNet-50 [53] with FPN [47] as the backbone network.

The quantitative results evaluated by AP^{mask} and $\text{AP}^{\text{boundary}}$ are given in Tables I and II, respectively. BuildingVec achieves highest AP^{mask} performance on most classes, with a score of 21.2 % mAP^{mask} and 35.5 % $\text{AP}^{\text{mask}}_{50}$, which outperforms the second-best method HTC by 1.3% in mAP^{mask} and 4.1 % in $\text{AP}^{\text{mask}}_{50}$. The HTC [21] is a cascade method that learns the bounding boxes and masks in a hybrid manner. It has been validated on various datasets as state-of-the-art instance segmentation method for a while. However, it cannot achieve reliable results on building extraction dataset, because mask-based methods have limitations in representing building polygons with regular shapes. BuildingVec is able to achieve this gain because the idea of contour-based learning allows direct regression on key vertices of polygons. Compared to one-stage instance segmentation method SOLOv2 [32], BuildingVec outperforms it by 7.1% in mAP^{mask} and 9.9% in $\text{AP}^{\text{mask}}_{50}$. SOLOv2 has achieved a good tradeoff between accuracy and efficiency as described in [32]. However, it exhibits a significant performance drop on the UBCv2 (vec) dataset. A potential reason for this phenomenon is the dense distribution of buildings in street blocks. SOLOv2 shows worse performance on these samples, particularly in

TABLE I
COMPARISON OF STATE-OF-THE-ART METHODS FOR BOTH MASK-BASED AND CONTOUR-BASED BRANCHES

Method	mAP^{mask}	AP_{50}^{mask}	FL	GA	GM	RO	ME	H1	H2	MA	PM	AR	RE	OT	FLOPs	FPS	Params
Mask-based:																	
SOLOv2 [32]	14.1	24.6	22.4	20.7	26.3	5.5	10.4	9.0	37.2	3.4	6.4	12.8	7.2	**8.4**	247.73 G	12.2	46.05 M
QueryInst	14.9	24.8	23.9	20.4	26.6	3.7	23.7	9.3	37.0	6.5	6.8	12.7	4.1	6.1	1115.32 G	11.4	172.31 M
Mask R-CNN [19]	17.8	29.1	27.7	26.4	28.7	4.8	26.6	10.9	41.8	4.8	12.3	14.4	9.8	7.1	263.04 G	19.7	43.81 M
Cascade Mask R-CNN [20]	19.1	30.6	29.3	27.1	**31.1**	5.8	29.7	12.2	42.9	6.1	11.2	**19.1**	8.2	7.6	394.23 G	13.9	76.84 M
HTC	19.9	31.4	29.7	27.4	30.3	**10.4**	31.0	11.8	43.6	6.7	12.0	18.8	**10.3**	8.1	394.81 G	13.5	76.97 M
Contour-based:																	
Deep Snake [16]	11.3	20.6	22.3	20.6	22.0	4.4	7.7	6.1	30.6	3.2	2.5	6.1	4.4	5.8	369.66 G	—	41.75 M
PolarMask [24]	14.2	24.5	22.0	23.7	25.4	4.2	12.0	10.2	35.9	7.4	8.4	12.3	3.3	5.5	510.96 G	—	34.31 M
E2EC [17]	14.3	23.6	25.5	23.3	27.0	2.5	9.0	8.1	34.9	5.5	4.6	17.3	8.6	5.7	**194.82 G**	**37.1**	**29.77 M**
DANCE [14]	14.8	24.4	27.3	26.3	28.9	5.1	9.4	10.3	39.1	4.4	4.9	12.6	1.8	7.2	208.94 G	—	44.26 M
BuildingVec	**21.2**	**35.5**	**30.0**	**30.0**	29.4	7.6	**40.4**	**17.8**	**44.7**	**7.8**	**14.2**	18.4	10.2	7.3	487.79 G	13.4	74.18 M

The class-wise instance segmentation results are evaluated using mask AP (%) on UBCv2 (vec) test set. The abbreviations of each fine-grained roof class is: FL—Flat, GA—Fable, GM—Gambrel, RO—Row, ME—Multiple Eave, H1—Hipped V1, H2—Hipped V2, MA— Mansard, PY—Pyramid, AR—Arched, RE—Revolved, and OT-Other. The highest score is bolded.

TABLE II
COMPARISON OF STATE-OF-THE-ART METHODS FOR BOTH MASK-BASED AND CONTOUR-BASED BRANCHES

Method	$\text{mAP}^{boundary}$	$\text{AP}_{50}^{boundary}$	FL	GA	GM	RO	ME	H1	H2	MA	PM	AR	RE	OT	FLOPs	FPS	Params
Mask-based:																	
SOLOv2 [32]	13.0	23.8	20.3	20.2	26.2	2.7	10.2	9.0	36.4	3.2	6.4	9.5	5.6	6.9	247.73 G	12.2	46.05 M
QueryInst	14.3	24.4	22.6	20.2	26.5	2.2	23.6	9.2	36.5	6.3	6.8	10.5	3.5	5.4	1115.32 G	11.4	172.31 M
Mask R-CNN [19]	16.6	28.4	25.4	26.0	28.6	2.3	26.4	10.9	41.2	4.4	12.1	10.5	7.2	6.0	263.04 G	19.7	43.81 M
Cascade Mask R-CNN [20]	17.9	30.0	26.9	26.5	**30.9**	3.8	29.6	12.1	42.3	5.8	10.9	**14.7**	6.4	6.7	394.23 G	13.9	76.84 M
HTC	18.5	30.7	27.3	26.8	30.1	**7.6**	30.9	11.6	42.9	6.4	11.7	14.0	7.5	**7.0**	394.81 G	13.5	76.97 M
Contour-based:																	
Deep Snake [16]	10.3	19.8	20.2	19.9	21.7	2.5	7.4	6.0	29.8	2.9	2.4	2.8	3.6	4.6	369.66 G	—	41.75 M
PolarMask [24]	13.2	23.7	19.6	23.1	25.2	2.0	11.8	10.2	35.1	7.0	8.3	8.6	2.4	4.5	510.96 G	—	34.31 M
E2EC [17]	13.6	23.1	23.8	23.0	26.9	1.6	8.9	8.1	34.2	5.4	4.5	14.1	7.5	5.1	**194.82 G**	**37.1**	**29.77 M**
DANCE [14]	13.9	24.0	25.1	25.8	28.7	3.0	9.1	10.2	38.5	4.1	4.7	9.2	1.6	6.4	208.94 G	—	44.26 M
BuildingVec	**19.9**	**34.8**	**27.5**	**29.3**	29.1	5.4	**40.0**	**17.6**	**43.6**	**7.4**	**14.0**	14.0	**8.2**	6.0	487.79 G	13.4	74.18 M

The class-wise instance segmentation results are evaluated using boundary AP (%) on UBCv2 (vec) test set. The highest score is bolded.

remote sensing images. In terms of boundary-based metrics given in Table II, BuildingVec also achieves competitive results compared to other methods. It achieves highest $\text{AP}^{\text{boundary}}$ performance on most classes, with a score of 19.9 % $\text{mAP}^{\text{boundary}}$ and 34.8 % $\text{AP}_{50}^{\text{boundary}}$, which outperforms the second-best method HTC by 1.4 % in $\text{mAP}^{\text{boundary}}$ and 4.1 % in $\text{AP}_{50}^{\text{boundary}}$. Compared to SOLOv2, BuildingVec outperforms it by 6.9% in $\text{mAP}^{\text{boundary}}$ and 11.0 % in $\text{AP}_{50}^{\text{boundary}}$.

As given in Table I, BuildingVec outperforms other state-of-the-art contour-based methods on the UBCv2 (vec) test set. BuildingVec substantially outperforms contour-based methods such as Deep Snake [16], PolarMask [24], E2EC [17], and DANCE [14]. Deep Snake is a classic contour-based method, which learns the offsets of initial contours using circular convolutions. PolarMask treats the contour of an object as a sequence of points encoded by polar coordinates. The contour ground truth is encoded by uniformly sampling (i.e., 36 points) around the circumference. E2EC is also a multistage contour learning method that adopts 1-D-convolution to learn the deformation of contour points. DANCE utilizes the bounding boxes of the instances to initialize contours and achieves good performance on many datasets. BuildingVec outperforms DANCE by 6.4 % in mAP^{mask}, which is a great gap. In terms of boundary-based metrics given in Table II, BuildingVec also achieves competitive results compared to other methods. It also outperforms DANCE by 6.0 % in $\text{mAP}^{\text{boundary}}$. This illustrates the effectiveness of BuildingVec's instance-aware contour learning framework.

2) Qualitative Comparison With State-of-the-Art Methods on the UBCv2 (vec) Dataset: Fig. 8 shows the mask results of BuildingVec and mask-based methods on the UBCv2 (vec) test images. BuildingVec can predict regular polygons, which are essential for vectorized building extraction. Specifically, in the first row of Fig. 8, BuildingVec achieved accurate predictions for buildings densely arranged with an inclined rectangular distribution. However, other mask-based methods (e.g., SOLOv2 [32] and HTC [21]) achieve relatively rough masks. In the bottom row (white rectangle) of Fig. 8, BuildingVec produces more robust building masks while maintaining accurate fine-grained categorization of buildings. The visualized masks, which are converted using contour predictions of BuildingVec, demonstrate its compatibility with tasks that require mask predictions.

Fig. 9 shows the contour results of contour-based methods and BuildingVec on the UBCv2 (vec) test images. Contour-based methods show good performance on densely distributed buildings (first row in Fig. 9). However, DeepSnake [16] and E2EC [17] both show worse classification performance than BuildingVec (e.g., the gable roof in the first row of Fig. 9). For other scenarios, BuildingVec shows promising contour results with less false positives. E2EC [17] also shows good contour results in many examples, but it has limitations in predicting the categories of building roofs. BuildingVec is able to learn the

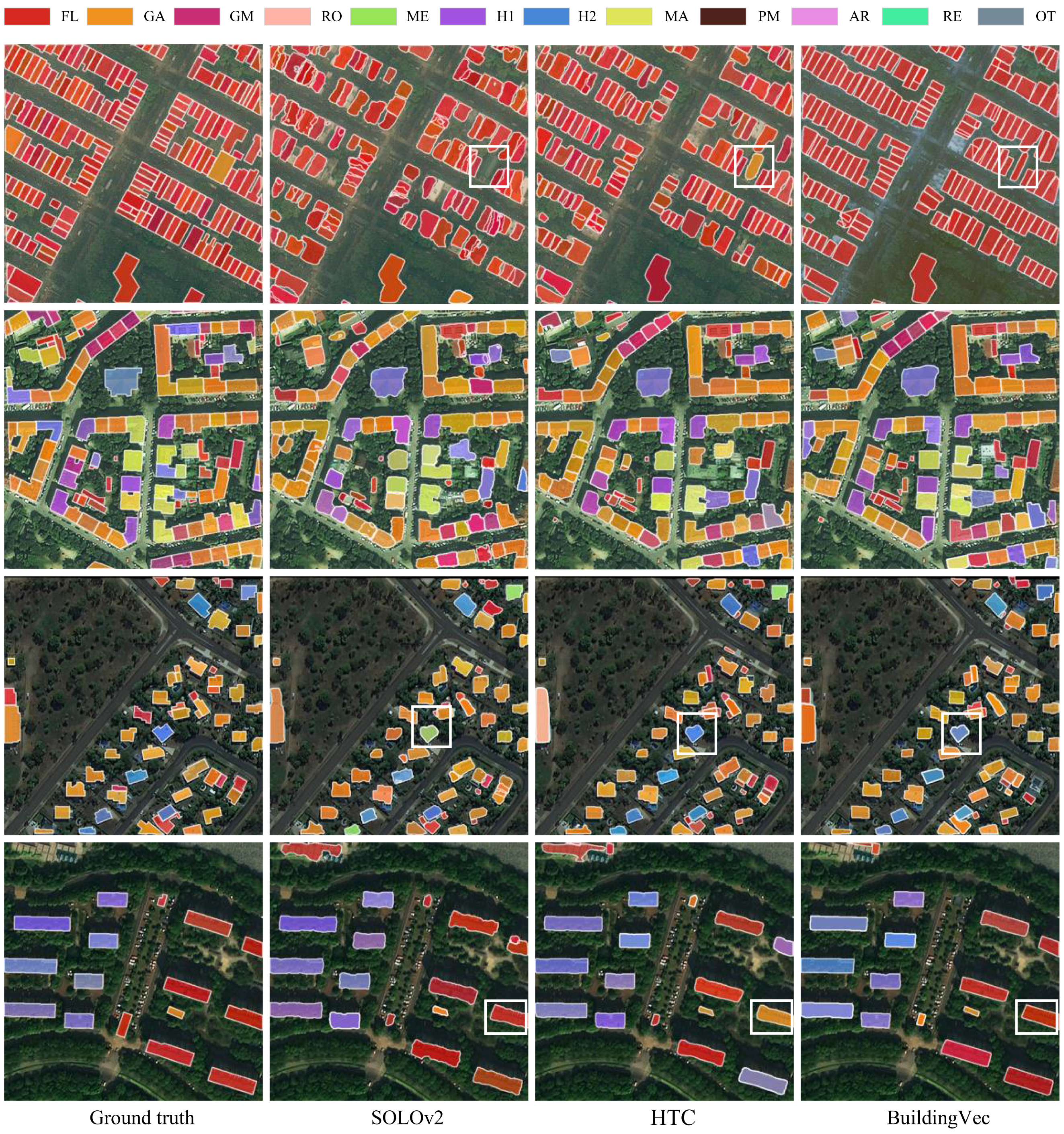

Fig. 8. Visualized results of BuildingVec and state-of-the-art mask-based methods on the UBCv2 (vec) test images.

regularized contours of the buildings while maintaining a strong classification performance on the roofs. This is made possible through our use of instance-level features in the cascade learning of BuildingVec framework.

In addition, Table I presents the running efficiency of the compared models. It can be observed that BuildingVec maintains a similar FPS to models such as HTC [21], while exhibiting a minimal increase in computational cost. E2EC [17], as a real-time one-stage detection algorithm, offers a significant advantage in terms of running speed and other aspects.

3) Quantitative Comparison With State-of-the-Art Methods on the Crowd AI Dataset: Table III gives the performance of BuildingVec and other state-of-the-art methods focusing on vectorized building extraction. We report the AP and AR metrics of these methods evaluated on masks. BuildingVec shows a state-of-the-art performance on the Crowd AI test set. For mask-based methods (HTC-DP [54] and Frame Field Learning [10]), BuildingVec outperforms them by 25.8% and 8.9% in mAP, respectively. For RNN-based methods (Building Outline Delineation [42] and PolyMapper [43]), BuildingVec

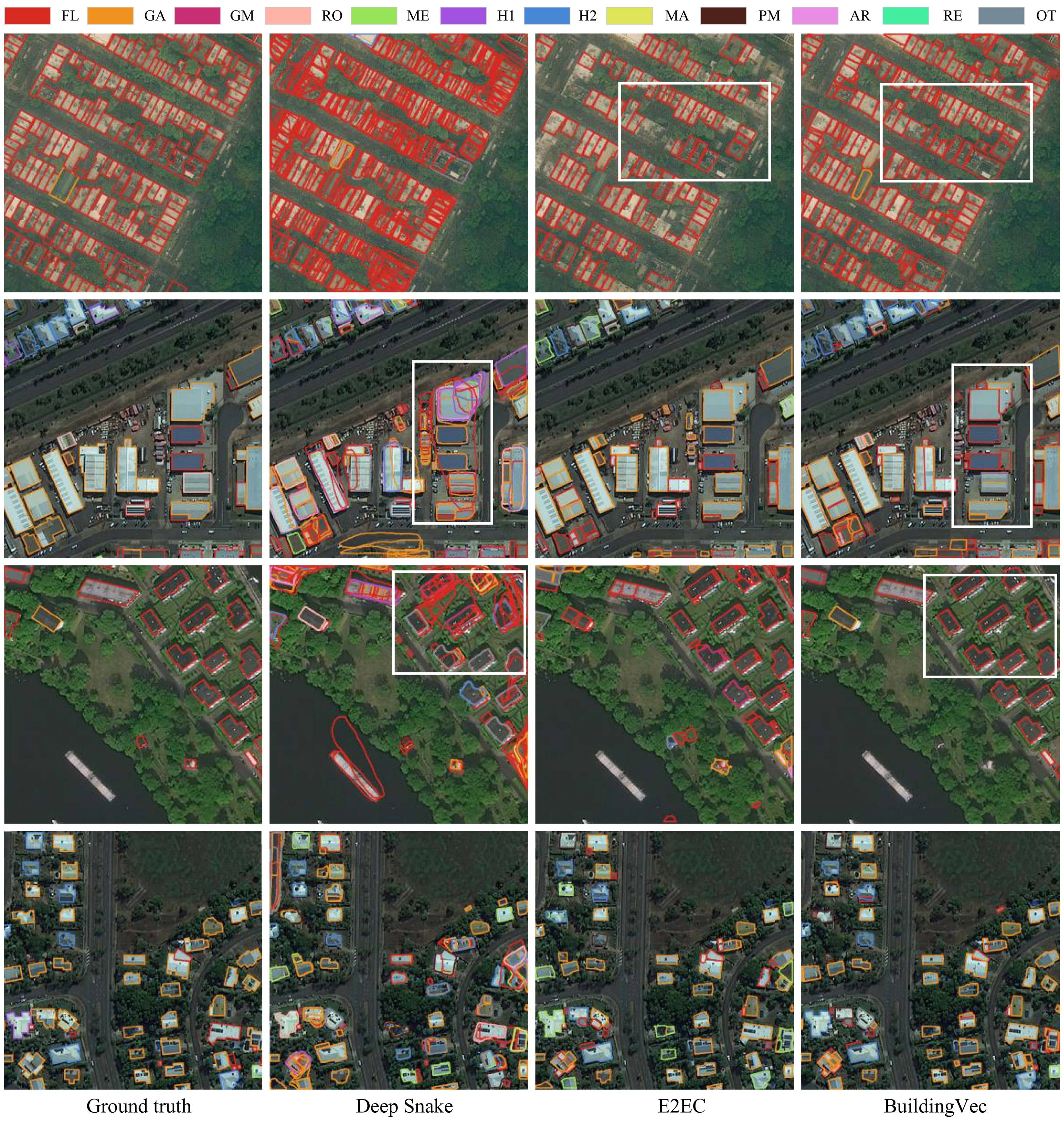

Fig. 9. Visualized results of BuildingVec and state-of-the-art contour-based methods on the UBCv2 (vec) test images.

outperforms them by 22.8% and 14.5% in mAP, respectively. For contour-based methods (BuildMapper [13]), BuildingVec outperforms it by 6.3% and 3.0% in mAP and AP_{50}, respectively. These quantitative results indicate that BuildingVec is capable of generating more accurate and refined building polygons, accompanied by enhanced vectorization performance.

4) Ablation Study: To validate the effectiveness of the proposed IACR module and other methods, comprehensive ablation experiments are conducted on the UBCv2 (vec) dataset. Limited by the computing resources and time, we train models for 12 epochs in this section. All results are validated on the test set.

a) Validation on the effectiveness of IACR module: To validate the effectiveness of our IACR design, we compare the proposed IACR module with other local modeling methods (e.g., MLP and 1-D-convolution) . We also compare the performance with and without the instance-level guidance in the IACR module. Specifically, the MLPs employ the same input feature dimensions as the IACR module, which are all 128. It comprises three layers of feedforward neural networks,

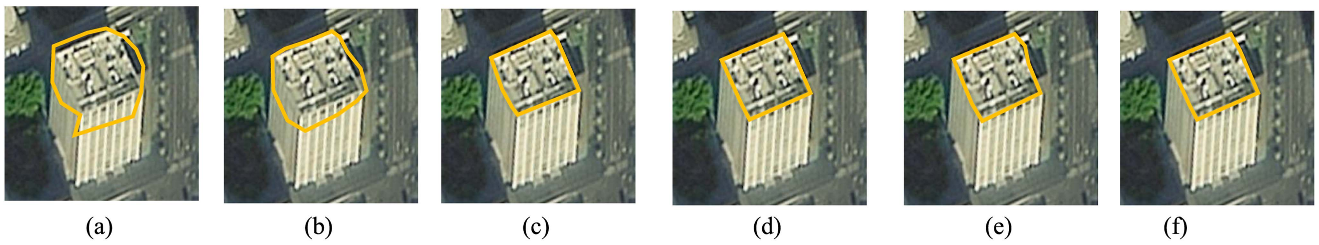

Fig. 10. One visualized example detected by variants with different number of contour refinement stages. (a) Stage = 1. (b) Stage = 2. (c) Stage = 3. (d) Stage = 4. (e) Stage = 5. (f) Stage = 6.

TABLE III
QUANTITATIVE RESULTS OF BUILDINGVEC AND OTHER STATE-OF-THE-ART BUILDING EXTRACTION METHODS EVALUATED USING MASK AP ON THE CROWDAI DATASET

Method	mAP	AP_{50}	AP_{75}	AR	AR_{50}	AR_{75}
HTC-DP [54]	44.4	-	-	-	-	-
BOD [42]	47.4	-	-	-	-	-
PolyMapper [43]	55.7	86.0	65.1			
FFL [10]	61.3	87.4	70.6	64.5	89.2	73.4
PolyWorld	63.3	88.6	70.5	**75.4**	93.5	83.1
BuildMapper [13]	63.9	90.1	75.0	-	-	-
BuildingVec	**70.2**	**93.1**	**82.7**	74.7	**94.6**	**85.0**

The highest score is bolded.

TABLE IV
ABALATION STUDY ON THE MODELING METHODS OF CONTOUR REGRESSION

Regression method	mAP^{mask}	AP^{mask}_{50}	$mAP^{boundary}$	$AP^{boundary}_{50}$
MLP	15.5	30.5	13.8	28.9
1D-Conv	17.1	32.0	15.6	30.5
IACR w/o IA	17.2	31.3	15.8	30.3
IACR w/ IA	**18.8**	**32.8**	**17.5**	**31.8**

The highest score is bolded.

TABLE V
ABALATION STUDY OF DIFFERENT REFINEMENT STAGES (FROM 1 TO 6)

Stages	mAP^{mask}	AP^{mask}_{50}	$mAP^{boundary}$	$AP^{boundary}_{50}$
1	18.5	32.5	16.9	31.6
2	18.2	31.7	16.9	30.8
3	**18.8**	**32.8**	**17.5**	**31.8**
4	18.8	32.4	17.4	31.6
5	18.5	32.3	17.2	31.4
6	18.7	32.5	17.3	31.7

The highest score is bolded.

with hidden layer dimensions of 512. As given in Table IV, MLP achieves worse performance than 1-D-convolution. This is because the receptive field of the MLP is limited to a single vertex and cannot perceive features of neighboring nodes. Since the proposed IACR module has global modeling capability, it outperforms the 1-D-convolution method that only performs local convolution. The IACR module with the cross attention on the instance features achieves better performance (1.6 % mAP^{mask}) than that without instance-aware ability. This further demonstrates the effectiveness of the proposed instance-aware module.

TABLE VI
ABALATION STUDY ON THE VERTEX REDUCTION THRESHOLD SCNR

$Scnr$	mAP^{mask}	AP^{mask}_{50}	$mAP^{boundary}$	$AP^{boundary}_{50}$
0.05	21.2	35.5	19.9	34.7
0.1	**21.2**	**35.5**	**19.9**	**34.8**
0.2	20.8	35.2	19.5	34.4
0.3	17.2	30.7	16.0	29.8

The highest score is bolded.

b) Ablation study on the number of contour refinement stages: We performed ablation experiments on the stages of the contour refinement process. As shown in Table V, we train six models with the number of contour refinement stages from 1 to 6. As the number of stages increases, the performance of the model peaks at a stage number of 3. Further increase in the number of stages for contour refinement makes it difficult to improve the performance of the model any further. Fig. 10 shows an instance detected by models with different stages. We observe that good results can already be achieved using three-stage contour refinements.

c) Ablation study on the vertex sampling threshold S_{cnr}: We examine the impact of varying vertex reduction thresholds S_{cnr} on the results. As illustrated in Table VI, the thresholds $S_{\text{cnr}} = \{0.05, 0.1, 0.2, 0.3\}$ are employed for vertex reduction, respectively. The model's performance peaks at 0.1 as the threshold increases. This suggests that the elimination of redundant vertices from the predicted polygon results may yield more regular building results. As the threshold S_{cnr} increases, some valid vertices are eliminated, resulting in a reduction in the performance of the model. This pattern can also be observed in Fig. 11. In Fig. 11(d), some of the results change from quadrilaterals to triangles. This indicates that the appropriate threshold value is crucial for determining the degree of vectorization of the extracted results.

d) Ablation study on the PE: Table VII displays the ablation experiments of the PE, primarily in the first and second rows, as well as the third and last rows. By incorporating the position encoding, the model's performance can be greatly improved by 5.1% and 7.3% in mAP^{mask}, respectively. This is due to the Transformer's self-attention mechanism, which is characterized by permutation invariance. Without PE, the model is not able to learn the sequential relationship among the vertices of the contour, and can only learn the coordinate offsets roughly.

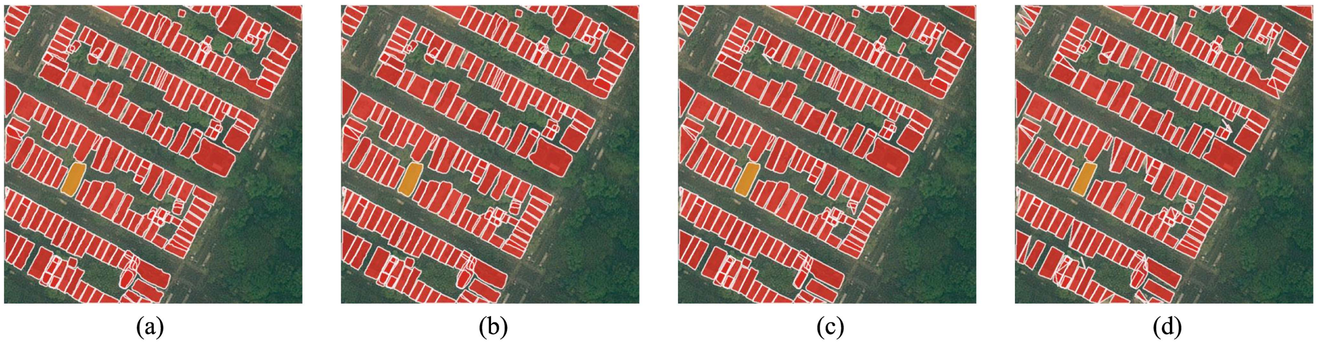

Fig. 11. Visualized results under different vertex reduction threshold S_{cnr}. (a) $S_{\text{cnr}} = 0.05$. (b) $S_{\text{cnr}} = 0.1$. (c) $S_{\text{cnr}} = 0.2$. (d) $S_{\text{cnr}} = 0.3$.

TABLE VII
ABALATION STUDY OF INFORMATION FLOW AND PE ON UBCv2 (VEC) TEST SET

Methods	PE	information flow	mAP^{mask}	AP^{mask}_{50}	FL	GA	GM	RO	ME	H1	H2	MA	PM	AR	RE	OT
BuildingVec			10.8	25.5	14.3	16.7	11.2	3.0	24.2	6.9	22.7	4.8	4.9	14.7	2.9	5.0
BuildingVec	✓		15.9	29.0	22.7	21.9	17.7	3.4	32.2	10.3	35.5	7.3	9.6	18.4	6.8	6.6
BuildingVec		✓	11.5	27.0	13.8	16.6	10.8	2.7	25.3	7.4	22.7	5.8	6.0	16.9	5.7	5.1
BuildingVec	✓	✓	18.8	32.5	26.9	26.2	25.1	5.6	35.7	12.7	41.0	7.7	12.0	20.5	7.0	7.7

The class-wise instance segmentation results are evaluated using APmask (%). The PE denotes positional encoding.

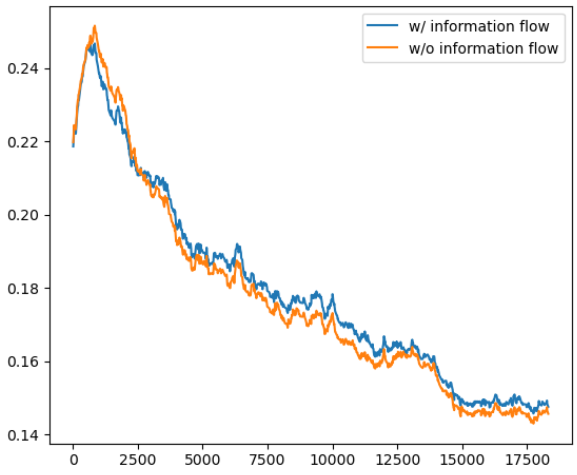

Fig. 12. Loss curve for models trained with and without the information flow on vector features.

e) Validation on the effectiveness of information flow: Table VII illustrates the ablation experiments of the information flow in vector features, primarily in the first and third rows, and second and last rows. By incorporating the information flow among vector features among different stages, the model's performance is improved by 0.7% and 2.9% in mAP^{mask}, respectively. Fig. 12 also demonstrates that the incorporation of information flow has a beneficial effect on the model's loss optimization. These results indicate that the additional branch connecting the vertex features at different stages can improve the learning efficiency of the contour refinement process.

V. Conclusion

In this article, we have proposed BuildingVec, which is an instance-aware vectorized building extraction framework from remote sensing imagery. A high-performance IACR module is proposed to achieve contour feature learning via cross-attention on the instance-level features. Based on the IACR module, the proposed BuildingVec framework focuses on improving the classification capability and vertex learning efficiency by interacting between the contour regression branch and instance classification branch through a well-designed cascade architecture. There is no cumbersome postprocessing required to eliminate redundant vertices in the proposed framework. To comprehensively validate the performance of BuildingVec, we build a vectorized building dataset, named UBCv2 (vec), by editing the UBCv2 dataset, which is a large-scale building dataset with fine-grained categorical information. The proposed method offers novel insights into end-to-end vectorized extraction, while also incorporating instance-level categorization representation.

References

[1] T. Esch et al., "Breaking new ground in mapping human settlements from space—The global urban footprint," *ISPRS J. Photogramm. Remote Sens.*, vol. 134, pp. 30–42, 2017.

[2] Y. Tian et al., "Fully-weighted HGNN: Learning efficient non-local relations with hypergraph in aerial imagery," *ISPRS J. Photogramm. Remote Sens.*, vol. 191, pp. 263–276, 2022.

[3] Z. Zhang et al., "Vectorized rooftop area data for 90 cities in China," *Sci. Data*, vol. 9, p. 66, 2022. [Online]. Available: https://www.nature.com/articles/s41597-022-01168-x

[4] Y. Tian, X. Huang, R. Niu, H. Yu, P. Wang, and X. Sun, "Hypertron: Explicit social-temporal hypergraph framework for multi-agent forecasting," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, 2022, pp. 1356–1362.

[5] W.-B. Wu et al., "A first Chinese building height estimate at 10 m resolution (CNBH-10 m) using multi-source Earth observations and machine learning," *Remote Sens. Environ.*, vol. 291, 2023, Art. no. 113578.

[6] F. Biljecki, K. Arroyo Ohori, H. Ledoux, R. Peters, and J. Stoter, "Population estimation using a 3D city model: A multi-scale country-wide study in The Netherlands," *PLoS One*, vol. 11, no. 6, Jun. 2016, Art. no. e0156808.

[7] Z. Zhang et al., "Carbon mitigation potential afforded by rooftop photovoltaic in China," *Nat. Commun.*, vol. 14, no. 1, 2023, Art. no. 2347.

[8] J.-M. Bahu, A. Koch, E. Kremers, and S. M. Murshed, "Towards a 3D spatial urban energy modelling approach," *Int. J. 3-D Inf. Model.*, vol. 3, no. 3, pp. 1–16, 2014.

[9] L. Wiginton, H. T. Nguyen, and J. M. Pearce, "Quantifying rooftop solar photovoltaic potential for regional renewable energy policy," *Comput. Environ. Urban Syst.*, vol. 34, no. 4, pp. 345–357, 2010.

[10] N. Girard, D. Smirnov, J. Solomon, and Y. Tarabalka, "Polygonal building extraction by frame field learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 5891–5900.

[11] B. Xu, J. Xu, N. Xue, and G.-S. Xia, "HiSup: Accurate polygonal mapping of buildings in satellite imagery with hierarchical supervision," *ISPRS J. Photogramm. Remote Sens.*, vol. 198, pp. 284–296, 2023.

[12] N. Xue et al., "Learning regional attraction for line segment detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 6, pp. 1998–2013, Jun. 2021.

[13] S. Wei, T. Zhang, S. Ji, M. Luo, and J. Gong, "BuildMapper: A fully learnable framework for vectorized building contour extraction," *ISPRS J. Photogramm. Remote Sens.*, vol. 197, pp. 87–104, 2023.

[14] Z. Liu, J. H. Liew, X. Chen, and J. Feng, "Dance: A deep attentive contour model for efficient instance segmentation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2021, pp. 345–354.

[15] H. Ling, J. Gao, A. Kar, W. Chen, and S. Fidler, "Fast interactive object annotation with curve-GCN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5257–5266.

[16] S. Peng, W. Jiang, H. Pi, X. Li, H. Bao, and X. Zhou, "Deep snake for real-time instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8533–8542.

[17] T. Zhang, S. Wei, and S. Ji, "E2EC: An end-to-end contour-based method for high-quality high-speed instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 4443–4452.

[18] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comput. Vis.*, vol. 1, no. 4, pp. 321–331, 1988.

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2961–2969.

[20] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6154–6162.

[21] K. Chen et al., "Hybrid task cascade for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4974–4983.

[22] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Adv. Neural Inf. Process. Syst.*, 2015, vol. 28.

[23] R. Girshick, "Fast R-CNN," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 1440–1448.

[24] E. Xie et al., "PolarMask: Single shot instance segmentation with polar representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12193–12202.

[25] E. Xie, W. Wang, M. Ding, R. Zhang, and P. Luo, "PolarMask: Enhanced polar representation for single-shot instance segmentation and beyond," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5385–5400, Sep. 2022.

[26] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9627–9636.

[27] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6569–6578.

[28] X. Huang et al., "Urban building classification (UBC) V2—A benchmark for global building detection and fine-grained classification from satellite imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5620116.

[29] S. P. Mohanty et al., "Deep learning for understanding satellite imagery: An experimental survey," *Front. Artif. Intell.*, vol. 3, 2020, Art. no. 534696.

[30] Y. Wu, L. Xu, Y. Chen, A. Wong, and D. A. Clausi, "TAL: Topography-aware multi-resolution fusion learning for enhanced building footprint extraction," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, 2022, Art. no. 6506305.

[31] X. Wang, T. Kong, C. Shen, Y. Jiang, and L. Li, "SOLO: Segmenting objects by locations," in *Proc. Eur. Conf. Comput. Vis*, 2020, pp. 649–665.

[32] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "SOLOv2: Dynamic and fast instance segmentation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 17721–17732.

[33] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "YOLACT: Real-time instance segmentation," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 9157–9166.

[34] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8759–8768.

[35] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica Int. J. Geographic Inf. Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.

[36] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 266–277, Feb. 2001.

[37] D. Marcos et al., "Learning deep structured active contours end-to-end," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8877–8885.

[38] A. Hatamizadeh, D. Sengupta, and D. Terzopoulos, "End-to-end deep convolutional active contours for image segmentation," 2019, *arXiv:1909.13359*.

[39] W. Huang, Z. Liu, H. Tang, and J. Ge, "Sequentially delineation of rooftops with holes from VHR aerial images using a convolutional recurrent neural network," *Remote Sens.*, vol. 13, no. 21, 2021, Art. no. 4271.

[40] W. Huang, H. Tang, and P. Xu, "OEC-RNN: Object-oriented delineation of rooftops with edges and corners using the recurrent neural network from the aerial images," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2021, Art. no. 5604912.

[41] W. Zhao, C. Persello, and A. Stein, "Building outline delineation: From aerial images to polygons with an improved end-to-end learning framework," *ISPRS J. Photogramm. Remote Sens.*, vol. 175, pp. 119–131, 2021.

[42] Z. Liu, H. Tang, and W. Huang, "Building outline delineation from VHR remote sensing images using the convolutional recurrent neural network embedded with line segment information," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 4705713.

[43] Z. Li, J. D. Wegner, and A. Lucchi, "Topological map extraction from overhead images," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1715–1724.

[44] K. Duan, L. Xie, H. Qi, S. Bai, Q. Huang, and Q. Tian, "Location-sensitive visual recognition with cross-IOU loss," 2021, *arXiv:2104.04899*.

[45] Y. Hu, Z. Wang, Z. Huang, and Y. Liu, "PolyBuilding: Polygon transformer for building extraction," *ISPRS J. Photogramm. Remote Sens.*, vol. 199, pp. 15–27, 2023.

[46] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," 2020, *arXiv:2010.04159*.

[47] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2117–2125.

[48] A. Vaswani et al., "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[49] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[50] B. Cheng, R. Girshick, P. Dollár, A. C. Berg, and A. Kirillov, "Boundary IoU: Improving object-centric image segmentation evaluation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15329–15337.

[51] K. Chen et al., "MMDetection: Open MMLab detection toolbox and benchmark," 2019, *arXiv:1906.07155*.

[52] Y. Fang et al., "Instances as queries," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 6910–6919.

[53] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[54] W. Zhao, C. Persello, and A. Stein, "Building instance segmentation and boundary regularization from high-resolution remote sensing images," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2020, pp. 3916–3919.

Xingliang Huang received the B.Sc. degree in communication engineering from Chongqing University, Chongqing, China, in 2020. He is currently working toward the Ph.D. degree in signal and information processing with the University of Chinese Academy of Sciences, Beijing, China, and with Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing.

His research interests include computer vision, pattern recognition, and remote sensing image processing, especially on building extraction.

Kaiqiang Chen (Member, IEEE) received the Ph.D. degree in signal and information processing from the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing, China, in 2019.

He was a Visiting Scholar with the Karlsruhe Institute of Technology, Karlsruhe, Germany, from 2017 to 2018. He is currently an Assistant Researcher with the Aerospace Information Research Institute, Chinese Academy of Sciences. His research interests include computer vision and remote sensing image analysis.

Zhirui Wang (Member, IEEE) received the B.Sc. degree in information and communication engineering from the Harbin Institute of Technology, Harbin, China, in 2013, and the Ph.D. degree in electronics and communications engineering from Tsinghua University, Beijing, China, in 2018.

He is currently an Assistant Researcher with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing. His research interests include the intelligent interpretation of remote sensing images, such as terrain classification, and target detection and recognition.

Xian Sun (Senior Member, IEEE) received the B.Sc. degree from the Beijing University of Aeronautics and Astronautics, Beijing, China, in 2004, the M.Sc. degree from the Institute of Electronics, Chinese Academy of Sciences, Beijing, and Ph.D. degree in signal and information processing from the Institute of Electronics, Chinese Academy of Sciences, Beijing, in 2009.

He is currently a Professor with the Aerospace Information Research Institute, Chinese Academy of Sciences. His research interests include computer vision, geospatial data mining, and remote sensing image understanding.