This paper has been accepted for publication in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2024
DOI:10.1109/tpami.2024.3388371

Fast Building Instance Proxy Reconstruction for Large Urban Scenes

Jianwei Guo, Haobo Qin, Yinchang Zhou, Xin Chen, Liangliang Nan, Hui Huang

Abstract—Digitalization of large-scale urban scenes (in particular buildings) has been a long-standing open problem, which attributes to the challenges in data acquisition, such as incomplete scene coverage, lack of semantics, low efficiency, and low reliability in path planning. In this paper, we address these challenges in urban building reconstruction from aerial images, and we propose an effective workflow and a few novel algorithms for efficient 3D building instance proxy reconstruction for large urban scenes. Specifically, we propose a novel learning-based approach to instance segmentation of urban buildings from aerial images followed by a voting-based algorithm to fuse the multi-view instance information to a sparse point cloud (reconstructed using a standard Structure from Motion pipeline). Our method enables effective instance segmentation of the building instances from the point cloud. We also introduce a layer-based surface reconstruction method dedicated to the 3D reconstruction of building proxies from extremely sparse point clouds. Extensive experiments on both synthetic and real-world aerial images of large urban scenes have demonstrated the effectiveness of our approach. The generated scene proxy models can already provide a promising 3D surface representation of the buildings in large urban scenes, and when applied to aerial path planning, the instance-enhanced building proxy models can significantly improve data completeness and accuracy, yielding highly detailed 3D building models.

Index Terms—urban scene reconstruction; photogrammetry; instance segmentation; aerial path planning; surface reconstruction

1 Introduction

Digitizing large-scale urban scenes is of great interest in computer vision and computer graphics communities [1], [2], [3], [4], as a 3D representation of urban scenes is essential for various real-world applications, such as urban planning, navigation, and environmental simulations. Compared to expensive vehicle-mounted or airborne LiDAR-based data acquisition approaches, aerial-based photogrammetry sensing using unmanned aerial vehicles (UAVs) provides a more affordable and flexible way to capture detailed geometry of complex urban scenes [5], [6], [7], [8].

The mainstream UAV-based aerial imaging methods typically follow a coarse-to-fine paradigm that requires two flight passes. The first pass captures an unknown scene quickly using a pre-defined pattern and generates a conservative approximation of the scene geometry that is referred to as a *scene proxy*. Such a coarse model is then used for aerial path planning in the second pass in which the flights of image acquisition are performed along an optimized trajectory to produce a more complete and better reconstruction. Previous works strive to improve the second pass on aerial path planning [8], [9], [10], [11], [12], while less attention has been paid to the first pass on 3D scene proxy generation. In practice, scene proxies are generated based on either simple extrusion of building footprints [13] or surface mesh reconstruction using dense point clouds [8], [9], [10], which has limitations such as low geometric accuracy, long capture process, and high demands of on-site computing power. Recently, [11] compute 2.5D proxies by detecting shadows from satellite images. This method relies heavily on satellite images with noticeable shadows and flat scene grounds, which has limited accuracy in practice. We argue that generating more accurate and tightly enclosed 3D scene proxies would improve the quality of planned aerial paths.

In this paper, we aim to address the open problem of high-quality building instance proxy generation from multi-view aerial images. This is a great challenge due to three reasons: (i) First, existing image-based 3D reconstruction workflows can robustly recover camera poses and a sparse 3D point cloud via *Structure from Motion* (SfM). It is also possible to generate a dense point cloud by using *Multiple View Stereo* (MVS) for better proxy reconstruction. However, the MVS step has high computational demands, especially for large urban scenes, which limits the scalability of previous methods [8]. In this work, we resort to using only sparse SfM data for efficient proxy generation. (ii) Second, the sparsity, incompleteness, noise, and outliers in the data pose great challenges to proxy reconstruction. For example, high-rise buildings are typically captured by only a few hundred points, and low buildings are often partially occluded by nearby buildings and trees. Previous point-based [14], [15] or primitive-based [16], [17] surface reconstruction methods require dense and complete point clouds as input, and thus cannot recover faithful structures from such corrupted data.

- *Jianwei Guo and Yinchang Zhou are with MAIS, Institute of Automation, Chinese Academy of Sciences, Beijing, China. (Jianwei Guo and Haobo Qin are with equal contributions)*
- *Haobo Qin is with University of Chinese Academy of Sciences, Beijing, China, and Shenzhen University, Shenzhen, China.*
- *Xin Chen is with Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen, China.*
- *Liangliang Nan is with Delft University of Technology, Netherlands.*
- *Hui Huang is the corresponding author (Email: hhzhiyan@gmail.com) with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China.*

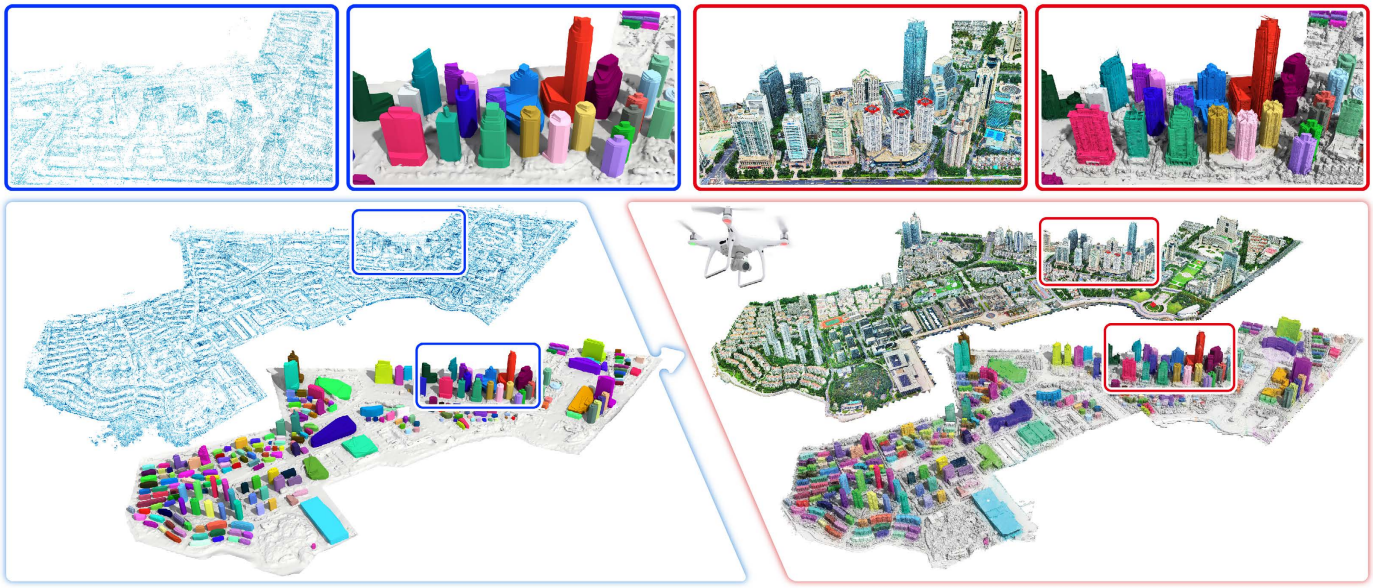

Fig. 1: The reconstruction of a large (2.9 km^2) downtown central scene. Left: the coarse input point cloud computed from 7,000 images and the building instance proxies reconstructed using our method (130 seconds on a commodity desktop computer). Right: the final detailed 3D model of the entire scene with building instance information inherited from the proxies, which is reconstructed (via *ContextCapture* running 10 days on a high-end server cluster) from 50,305 images acquired based on our proxy-derived drone aerial path.

(iii) Lastly, building instance information of the scene is lacking. Such information is crucial for distinguishing nearby buildings to improve safety in data acquisition and enable fine-grained path planning to capture finer building details. Moreover, such information also promotes 3D models for a wider range of practical applications.

To address the above-mentioned challenges, we propose a novel workflow that enables efficient 3D building proxy reconstruction at the instance level, suitable for large-scale urban scenes. Due to the extreme sparseness and incompleteness, the value of SfM points is often considered low and has been overlooked in past research. In this work, we re-examine its value because the SfM points still retain key features of building structures. Our main finding is that the sparse inputs are already sufficient for high-quality proxy reconstruction with high efficiency, which is made possible by our building instance segmentation and layer-based building proxy reconstruction methods. To obtain building instances, we introduce a neural network dedicated to generating building instance masks from aerial images. The multi-view instance masks are then fused with the sparse point cloud, and the building instances are obtained by exploiting the cross-modality information. With the building instance information, we propose a new layer-based surface extraction method to obtain a watertight and manifold mesh for each building, yielding an instance-enriched 3D model of the entire scene. The obtained 3D building proxies can already provide a lightweight surface representation of the scene. In particular, they enable more reliable and fine-grained aerial path planning for image acquisition toward urban reconstruction at a higher level of detail.

The main contributions of our work include:

1) a novel workflow for efficient 3D building proxy reconstruction at the instance level for large-scale urban scenes from aerial images, which enables fine-grained aerial path planning to recover finer details of urban buildings.
2) *InstFormer*, a novel neural network that extracts building instance masks from aerial images, and a voting-based multi-view instance fusion algorithm that exploits cross-modality information for effective building instance segmentation in sparse and noisy point clouds.
3) a layer-based building proxy reconstruction algorithm that can generate lightweight surface models of urban buildings from extremely sparse point clouds.
4) Two benchmark datasets for urban scene segmentation and reconstruction, respectively. The first one is for instance segmentation of buildings, which contains 720 aerial images captured from four cities with varying flight altitudes, and with buildings manually annotated. The other one is a synthetic benchmark with three large-scale virtual scenes dedicated to comprehensive evaluations of flight planning and 3D urban reconstruction.

2 Related Work

2.1 3D Geometric Proxy Generation

3D proxy reconstruction aims at automatically creating 3D coarse models from images or point clouds. One category of methods obtains a simplified representation of buildings by leveraging mesh decimation approaches. Such methods typically reconstruct a dense mesh model from the input points [18] followed by a simplification process exploiting geometric cues, such as quadric error metric [19], planar proxies [20], [21], [22], [23], [24], or general quadric surfaces [25], [26]. Another line of work attempts to detect geometric primitives from point clouds [27] and assemble

primitive shapes into a coarse polyhedron. For example, Chauve *et al.* [28] propose an adaptive decomposition of 3D space induced by planar primitives and form a watertight polygonal mesh by Delaunay triangulation. Lin *et al.* [29] fit parametric building blocks to the input LiDAR data for building reconstruction. Monszpart *et al.* [30] explore relation-based primitive fitting to provide compact and simplified representations of urban scenes. Nan and Wonka [17] and Kelly *et al.* [31] propose optimization approaches based on integer programming to approximate the geometry of the buildings. Bauchet and Lafarge [32] design a kinetic data structure for partitioning the space into convex polyhedra, from which the underlying surface mesh can be extracted with a min-cut formulation. Alternative methods are also proposed to reconstruct 2.5D building models [33], [34] including roofs [35] from LiDAR input or images. These methods require dense and relatively high-quality 3D point clouds as input, which are usually not feasible or expensive to satisfy with practical data acquisition systems, and thus they cannot guarantee plausible results due to the requirement of detecting a complete set of primitives from such corrupted data with severe missing structures.

2.2 Aerial Path Planning for Urban Scene Reconstruction

The goal of aerial path planning is to obtain high-quality trajectories for reliable and efficient data acquisition using UAVs, to enable high-quality 3D reconstruction of large-scale scenes. This challenging task is closely related to image-based scene reconstruction [36] and view selection (*e.g., Next-Best-View* planning [37], [38]). In contrast to the conservative fixed-height trajectories of zigzag patterns used by commercial software for oblique photography (*e.g.,* DJI-Terra* and PIX4D†), current research interests focus on maximizing scene coverage and meanwhile minimizing the trajectory length. Roberts *et al.* [9] and Hepp *et al.* [10] design novel scene coverage models and employ submodularity to select candidate views. Koch *et al.* [39] further utilize semantic information obtained using neural networks to optimize flight paths, where the semantics of the proxy model are used to define free and occupied airspace. This method relies on a semi-automatic approach to extract target objects and is limited to small-scale scenes. Smith *et al.* [8] introduce reconstructability heuristics into view optimization and develop novel optimization approaches to maximize reconstruction quality. Zhang *et al.* [13] jointly optimize the view selection and path planning in a single step by considering both scene reconstruction and path quality.

While paying less attention to scene proxy generation, most previous path planning methods [6], [8], [9], [10] reconstruct an MVS dense mesh to estimate the scene geometry and flyable airspace. Although they can efficiently obtain dense points from down-sampled images [6] or only a small set of images [8], MVS is still the most time-consuming process in these pipelines. The high computational cost hinders its usage in processing large-scale scenes containing densely populated buildings. Different from previous methods, our work explores the value of the sparse SfM points together with prior knowledge about urban buildings. We propose a novel workflow for reliable building proxy generation from such often overlooked data and demonstrate its applicability in aerial path planning for large-scale urban reconstruction.

∗. https://www.dji.com/dji-terra
†. https://www.pix4d.com

2.3 Instance Segmentation

To ensure safety and improve flexibility in data acquisition using UAVs, we propose to exploit building instance information of the scene. A large volume of methods has been proposed for the task of 2D instance segmentation [40], which can be roughly categorized into proposal-based and proposal-free methods. Proposal-based methods consider a top-down strategy that generates region proposals based on object detection and then predicts an instance mask within each proposal. According to the number of stages required for object positioning and mask generation, these methods can be further divided into single-stage (*e.g.,* YOLACT [41]), two-stage (*e.g.,* Mask-RCNN [42], mask scoring R-CNN [43], PANet [44]), and multi-stage (*e.g.,* Cascade R-CNN [45], Hybrid Task Cascade [46], SCNet [47]) methods. On the contrary, proposal-free methods work in a bottom-up manner. They first produce per-pixel predictions and then group pixels into object instances ([48], [49], [50], [51], [52]). Our approach is multi-stage based on the Cascade architecture [45], and we further improve the performance of the multi-stage detectors by strengthening the correlation between different tasks (*i.e.,* classification, detection, and segmentation) using the global context information. Although some recent work uses dual-pathway transformers for building extraction from remote sensing images [53], [54], they only address semantic segmentation by solving a binary classification problem.

Benefiting from point set learning neural networks such as PointNet++ [55], many 3D instance segmentation methods have been proposed [56], [57], [58], [59]. These methods can already achieve encouraging results on small-scale indoor scenes, but it is still challenging to extend them to outdoor scenes due to the lack of annotated 3D instance segmentation datasets. Chen *et al.* [52] introduce a multi-view instance segmentation framework for 3D buildings. They perform 2D roof instance segmentation on the height-enhanced multi-view images for better performance. This method relies on the input dense mesh models to generate usable height maps. In our work, the mesh models are not available, and we thus directly use sparse point clouds for 3D instance segmentation. Recently, several direct 3D instance segmentation methods (such as HAIS [60], SoftGroup [61] and Mask3D [62]) have improved the instance segmentation performance on a large-scale outdoor dataset, STPLS3D [63]. However, these learning-based methods rely on dense points that are not available for our low-altitude UAV images. Actually, they cannot extract sufficient point features from sparse point clouds. A comparison between SfM sparse point clouds and the mainstream available datasets is shown in Supplementary Materials. In this paper, we take advantage of the matured 2D instance segmentation and 2D-3D correspondences to achieve precise 3D instance segmentation of urban buildings from sparse points.

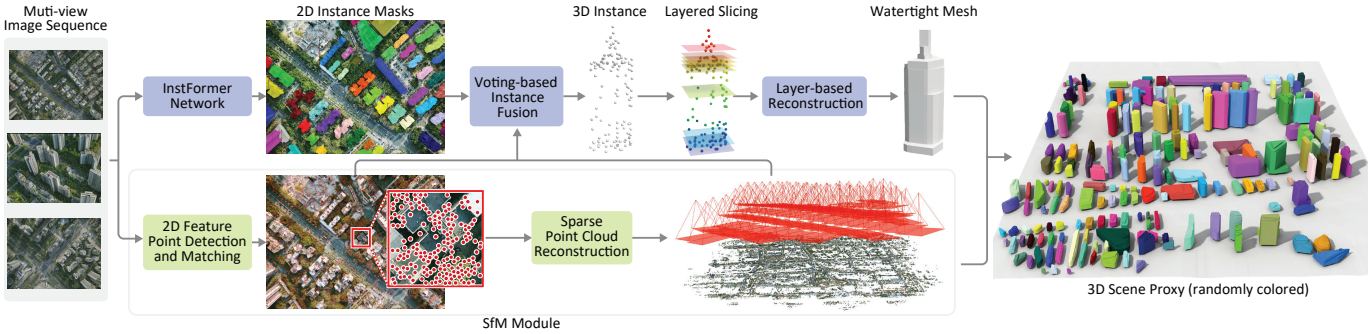

Fig. 2: Algorithm overview. Given a set of aerial images, we first propose an instance segmentation neural network, the *InstFormer*, to predict per-building instance masks in each view. Meanwhile, we generate a sparse point cloud from the input images using the SfM technique. Then by combining the 2D instance masks and the 2D-3D correspondences, we present a *voting-based multi-view instance fusion* algorithm to segment all 3D building instances. Finally, a novel *layer-based surface reconstruction* method is proposed to generate a 3D scene proxy with all semantically parsed buildings.

3 Overview

The goal of this work is to enable effective and efficient 3D proxy generation for individual buildings of a scene from a set of aerial images, which further allows reliable aerial path planning for detailed urban reconstruction. The input images are captured by a drone equipped with RGB cameras. We perform the initial aerial capture using a simple pre-defined overhead trajectory pattern [9].

An overview of our algorithm is outlined in Fig. 2, which consists of two novel modules: 3D building instance segmentation (Sec. 4) and layer-based proxy generation (Sec. 5). Given the input aerial images, we first propose a new instance segmentation neural network, called *InstFormer*, to predict per-building instance masks in multi-view images. Considering the dense distribution of buildings and occlusion caused by nearby buildings and trees, our instance segmentation is performed on the nadir images only, where the building roofs are fully visible and can be reliably segmented without ambiguity. We recover camera poses and generate a sparse point cloud from the input images by using SfM, where the correspondences between the feature points of the images and the reconstructed 3D points are also obtained. Based on the roof instance masks and the 2D-3D correspondences, we present a *voting-based multi-view instance fusion* mechanism that filters out over-segmented and invalid instances. The remaining masks are then projected back into the 3D space to segment the entire buildings. Since the SfM point cloud is typically sparse and has large missing regions (see Fig. 2), it is problematic to generate a faithful mesh from such data using existing reconstruction methods such as Poisson surface reconstruction [14]. To this end, we introduce an efficient *layer-based proxy reconstruction* algorithm that exploits structure priors of buildings to extract a volume mesh from such corrupted data. Following that, we obtain a manifold and watertight proxy model of each building by removing its interior redundant faces.

4 3D Building Instance Segmentation

Building instance segmentation serves as the foundation for reliable aerial path planning and semantic-aware 3D scene reconstruction. For point clouds with plausible density and completeness (e.g., MVS point clouds), it is straightforward to apply a 3D object detector [64], [65] or a 3D instance segmentation method [56], [57] to directly extract the 3D building instances. In our work, the sparse point clouds generated by SfM suffer from missing data, high levels of noise, and outliers, which hinder the direct application of these methods to robustly detect or segment 3D building instances. In this work, we take advantage of the fast development in image-based instance segmentation and 2D-3D correspondences (though a limited number) to achieve precise 3D instance segmentation of urban buildings from sparse point clouds.

4.1 2D Building Instance Segmentation

Although the existing 2D instance segmentation networks [42], [43], [45] have excellent performance on datasets such as MS-COCO [66], they are difficult to generalize directly to urban scenes due to large variations in the sizes and densities of building instances in a scene. We propose a novel instance segmentation neural network, called *InstFormer*, to produce accurate segmentation masks of dense buildings. We observe that the buildings in oblique images are more likely to be occluded by the nearby buildings or trees, while they typically do not overlap in nadir images. This motivates us to perform instance segmentation using nadir images. To train and evaluate our neural network, we create a new dataset by collecting real-world aerial images and annotating all the building instances in these images (see Sec. 6.1).

Fig. 3 summarizes the network architecture of InstFormer that predicts accurate instance masks for buildings at the pixel level. Since an input aerial image is of high resolution, we split the image into multiple overlapping blocks before feeding it into the neural network. To handle the unbalanced distribution of sample categories and avoid over-fitting, InstFormer adopts a 3-stage cascade structure comprising of three Box branches (*i.e.*, the *Bounding Box Head* in Fig. 3). The Box branches in the first two stages are responsible for gradually outputting coarse bounding boxes of buildings, and the counterpart in the last stage refines the

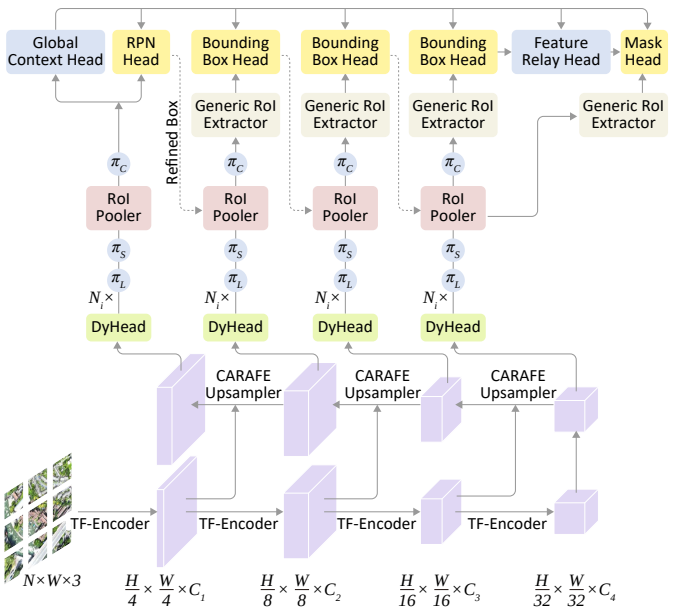

Fig. 3: Network architecture of *InstFormer* for dense and multi-scale building instance segmentation. The network consists of three key modules: *Backbone, Neck, and Head*. The *TF-Encoder* adopts a Vision Transformer as the backbone to extract the feature pyramid. The Neck module, including a *CARAFE Upsampler* [67] and a *DyHead*, is then applied to fuse and enhance the extracted features. In the Head module, we use a 3-stage cascade structure comprising of three *Bounding Box Heads* and one *Mask Head* to locate and generate instance masks in a coarse-to-fine manner.

box predictions and generates the instance masks. In the following, we briefly describe the key modules of InstFormer, and a more detailed illustration of the network architecture as well as the advantages of each module are provided in the Supplementary Materials.

Overview of InstFormer. Instance segmentation typically involves three sub-tasks: detecting, classifying, and segmenting objects. Therefore, InstFormer adopts a hybrid task cascade (HTC) architecture. First, multiple tasks such as detection, mask prediction, and semantic segmentation are combined at each stage to form a joint multi-stage processing pipeline, allowing each stage to benefit from the other tasks. Second, contextual information goes through an extra branch for stuff segmentation, and a directional path is added to allow information to direct flow across stages. Overall, the HTC architecture effectively improves the flow of information not only across stages but also between tasks.

In our implementation, InstFormer consists of three key modules: *Backbone, Neck, and Head*. Given an input image block, we first utilize the Pyramid Vision Transformer [68] as the backbone to extract the *feature pyramid* (FP, see the *TF-Encoder* layer in Fig. 3), which generates high-resolution feature maps for images with dense and varying-scale instances. To further increase the receptive field to aggregate contextual information, a Neck module (including a *Upsampler* and a *DyHead*) based on the self-attention mechanism is applied to efficiently fuse and enhance the FP. After that, the enhanced features are fed into a *Pooler* layer to obtain fixed-size feature maps, which are further sent to a *Generic RoI Extractor* to extract the regions of interest (ROIs). Then the ROIs extracted in each stage are sent to the corresponding *Bounding Box* heads to predict the final bounding boxes. At the same time, we also use the *Global Context* (glbctx) head combined with a *Feature Relay* (FR) head to strengthen the correlation between classification, detection, and segmentation tasks. Finally, a *Mask* head uniformly processes the output of the FR and glbctx heads and generates accurate instance masks. To sum up, the proposed InstFormer can be mathematically formulated as follows:

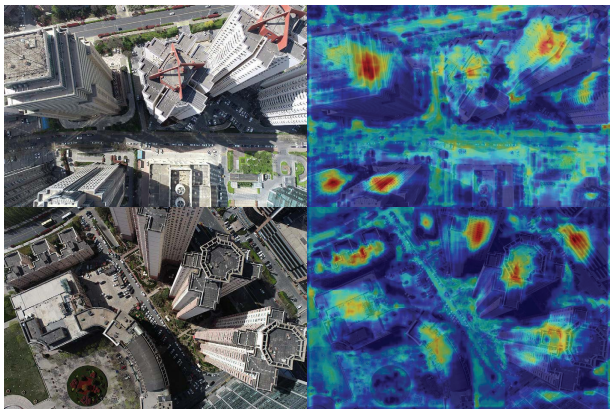

Fig. 4: Visualization of class activation maps. The building class score is mapped back to the previous transformer layer to emphasize the distinctive areas of buildings. Regions with high response values are shown in red, while regions with medium and low response values are indicated in green and blue, respectively.

$$\boldsymbol{x}_t^{\text{box}} := \mathcal{R}\left(\boldsymbol{x}, \boldsymbol{b}_{t-1}\right), \quad \boldsymbol{b}_t := \mathcal{B}_t\left(\boldsymbol{x}_t^{\text{box}}\right), \tag{1}$$

$$\boldsymbol{x}_t^{\text{mask}} := \mathcal{R}\left(\boldsymbol{x}, \boldsymbol{b}_t\right), \quad \boldsymbol{m}_t := \mathcal{M}_t\left(\mathcal{F}\left(\boldsymbol{x}_t^{\text{mask}}, \boldsymbol{m}_{1:t-1}\right)\right), \tag{2}$$

where $\boldsymbol{x}$ is the feature map extracted from the backbone. At stage t, we use a region-wise pooling operator $\mathcal{R}$ to extract the ROI-wise box features $\boldsymbol{x}_t^{\text{box}}$ based on the feature map $\boldsymbol{x}$ and the bounding box $\boldsymbol{b}_{t-1}$ predicted in stage $t-1$. Meanwhile, the mask features $\boldsymbol{x}_t^{\text{mask}}$ can be obtained by pooling $\boldsymbol{x}$ and $\boldsymbol{b}_t$. The prediction boxes $\boldsymbol{b}_t$ and masks $\boldsymbol{m}_t$ are learned from the *Bounding Box* head $\mathcal{B}_t$ and *Mask* head $\mathcal{M}_t$, respectively. $\mathcal{F}$ is a feature fusion operator and $\boldsymbol{m}_{1:t-1}$ represents the accumulated mask features taken from stage 1 to $t-1$. In Fig. 4, we use the class activation maps (CAMs) to get the informative regions used by our InstFormer to identify the category of buildings. The high-response areas are buildings, and the low-response areas are backgrounds, indicating that we can represent the buildings well and make discriminative localization.

Loss functions. Since there are only two concerned categories (*i.e.*, buildings and background), we adopt the cross-entropy loss $\mathcal{L}_t^{\text{cls}}$ for the binary classification. To make the bounding box location more accurate, we use a Complete-IoU (CIoU) loss [69] as regression loss function $\mathcal{L}_t^{\text{reg}}$. Another cross-entropy loss $\mathcal{L}_t^{\text{mask}}$ is used to predict the instance mask. Moreover, we also utilize the loss term $\mathcal{L}^{\text{glbctx}}$ introduced in SCNet [47] to obtain effective global contextual features and output multi-labels, so that we can use these

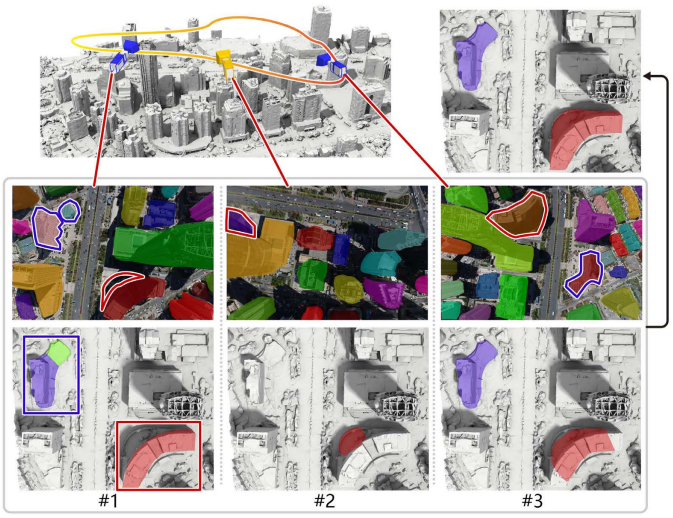

Fig. 5: Illustration of the voting-based instance fusion. In the first image (denoted by #1), the building in the light purple box is incorrectly separated and in all the three images, the red box only covers a small part of one building, leading to inaccuracy or under-segmentation. By fusing the masks from multi-view images based on a voting mechanism, an accurate 3D segmentation is achieved.

features later to perform more accurate multi-view instance fusion. Finally, we perform end-to-end multi-task training by minimizing the total loss function as follows:

$$\mathcal{L} = \sum_{t=1}^{3} \alpha_t \left(\mathcal{L}_t^{\text{cls}} + \mathcal{L}_t^{\text{reg}} \right) + \beta \mathcal{L}^{\text{mask}} + \gamma \mathcal{L}^{\text{glbctx}}. \quad (3)$$

The hyperparameter vector $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3]$ are the weights of classification and regression losses corresponding to each stage. The hyperparameter β is the weight of the mask loss. To maintain the consistency of IoU distribution between training and inference samples, we set $\beta = \sum_{t=1}^{3} \alpha_t$ to avoid over-fitting. Finally, γ corresponds to the loss weight of the global contextual feature, which is set to $\gamma = 3$ by default. The stage loss weights are set to $\boldsymbol{\alpha} = [1, 0.5, 0.25]$.

4.2 Voting-based Multi-view Instance Fusion

The InstFormer network outputs a set of building masks from multi-view images, where multiple instance masks may correspond to the same building. To separate different buildings in a 3D point cloud, one has to identify the masks belonging to the same building and then correlate them with their counterparts in the 3D point cloud. Establishing the correspondences between multiple masks is not straightforward due to the segmentation errors in 2D images. For example, one building correctly segmented in a few images could be separated in other views; or there are false positive buildings. A few such examples are given in Fig. 5.

To reliably fuse the error-prone instance masks from multiple views, we propose a voting-based approach to filter out the over-segmented and false positive masks. Let $\mathcal{D}^I = \left\{ \text{LIM}_k^I \right\}_{k=1}^{N}$ denote the set of all detected building instances in an image I, where LIM_k^I is referred to as the k-th *local instance mask* in image I. With the mapping $\mathcal{T}$ (provided by the SfM system) between the 2D feature points in the images and the sparse point cloud, we can obtain a set of 3D points $\mathcal{P}_{Ik}$ that correspond to the 2D feature points located in LIM_k^I. Besides, for the current image I, we retrieve a set of neighboring images $\mathcal{I}(I)$ according to the mapping $\mathcal{T}$, and in $\mathcal{I}(I)$ each neighboring image contributes to the reconstruction of any subset of points in $\mathcal{P}_{Ik}$. Next, each image $J \in \mathcal{I}(I)$ casts a vote $v\left(J||I_k\right)$ for each LIM_k^I in image I by checking the visibility of the 3D points $\mathcal{P}_{Ik}$ in image J. Specifically, we look into the number of instance masks from which all points in $\mathcal{P}_{Ik}$ are visible to determine if the vote is valid. The following three cases are considered for scoring:

- $v\left(J||I_k\right) = 0$. This is not a valid vote, which usually happens in two cases: (1) Image J contributes the reconstruction of $\mathcal{P}_{Ik}$ but only a small part of the building has been captured by image J and thus the building was not detected by our InstFormer; (2) The instance segmentation network recognizes non-building objects located in LIM_k^I as buildings, while no error occurs in image J.
- $v\left(J||I_k\right) = 1$, which means the local instance masks in both image I and J belong to the same building. This is the desired case and is thus considered a valid vote.
- $v\left(J||I_k\right) \geq 2$. This is also not a valid vote. It indicates that the 3D points $\mathcal{P}_{Ik}$ are segmented into multiple different instances in image J. This happens when image J provides a correct segmentation but the local instance mask LIM_k^I in image I is under-segmented, or the segmentation in image I is correct but the corresponding building in image J is over-segmented.

To determine whether a local instance mask in image I is valid, we use all images in $\mathcal{I}(I)$ to vote for LIM_k^I and obtain a voting vector $\mathcal{V} = \{v\left(J_1||I_k\right), v\left(J_2||I_k\right), \cdots, v\left(J_N||I_k\right)\}$. We return the result with the maximal scores: $v\left(I_k\right) \leftarrow \textbf{Mode}\left\{ v\left(J_i||I_k\right)|_{i=1}^{N} \right\}$, and further determine if the current instance is correctly segmented by:

$$\textbf{Flag}\left(\text{LIM}_k^I\right) = \begin{cases} \texttt{True}, \text{ if } v\left(I_k\right) = 1; \\ \texttt{False}, \text{ else.} \end{cases} \quad (4)$$

For image I, we first discard the invalid local instance masks. Then for each correct mask, we record in image J a list of valid local instance masks that generate a valid vote, i.e., $v\left(J||I_k\right) = 1$. After processing all images similarly, the correspondences between local instance masks in different images have been established. Thus, a group of local instance masks belonging to the same building instance are collected. The 3D instance segmentation is then achieved by combining the 3D points corresponding to each local instance mask in each group.

5 Layer-wise Proxy Generation

After 3D instance segmentation, we obtain a single point cloud for each building. As shown in Fig. 6 (a), the point cloud of the building is severely under-sampled and noisy, and in particular, important structures such as large parts of the facade are commonly missing. To handle such corrupted data, we propose a new slicing-based surface reconstruction method based on the fact that urban buildings typically have piecewise constant profiles along the vertical direction.

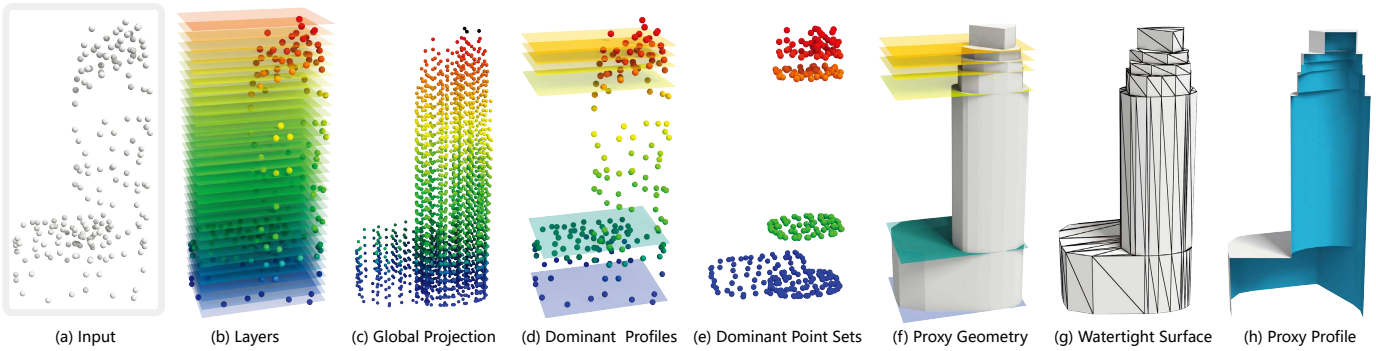

Fig. 6: Illustration of the proposed slicing-based proxy generation. Our algorithm takes a sparse point cloud (without normal information) as input (a), which is uniformly sliced into a sequence of layers along the vertical direction (b). We generate a global projected point set for each layer by projecting the 3D points from all its above layers onto it (c). According to point and area gains (*i.e.*, G_{point} and G_{area} in Eq. 5), we extract a set of dominant structural profiles, whose supporting planes are visualized in (d), and their corresponding global projected point sets are shown in (e). We then extrude the dominant structural profiles to form the proxy geometry (f), where the colored planes denote the presence of interior faces. Finally, an incremental surface extraction is conducted to obtain a watertight mesh, for which a profile of the proxy is shown in (h).

Terminology. The input point cloud of a building is first uniformly sliced into a sequence of raw slabs along the vertical direction, see Fig. 6 (b) for an example. Each slicing plane is called a *layer*, and the space between two adjacent layers is called a *layer space*. Each layer is also associated with two entities: the *local projected point set* formed by projecting the corresponding 3D points in the above adjacent layer space onto it; and a *global projected point set* formed by projecting the 3D points in all above layer spaces onto it. From the global projected point set, we extract the 2D convex hull denoted as a *potential structural profile*. Here we use the convex hull instead of α-shape because the input point cloud is too sparse that α-shape will generate an incomplete set of faces to reveal the actual profile of the building. In contrast, a convex hull can create an extra safety buffer when the reconstructed 3D building is used for deriving UAV trajectories. Next, we extract a set of *dominant structural profiles* (Fig. 6 (d)), each of which has significant structural differences from its neighboring ones. The dominant structural profiles together provide sufficient information to characterize the shape of the building.

Proxy reconstruction. The layers $(\mathcal{F}_1, \cdots, \mathcal{F}_K)$ are sorted from the top to bottom. Let $\mathcal{P}_i$ denote the local projected point set of the i-th layer and N_K the number of 3D points in the k-th layer space $\mathcal{S}_k$, we have $\mathbf{PSP}_k = \mathbf{Conv}\left(\bigcup_{i=1}^{k} \mathcal{P}_i\right)$, where **Conv** denotes the operation of extracting a convex hull. Similarly, $\mathbf{PSP}_{k-1}$ and $\mathbf{PSP}_{k+1}$ denote the potential structural profiles corresponding to the upper and lower layers of $\mathcal{S}_k$, respectively.

From the above definition, the shape of $\mathbf{PSP}_K$ (i.e., the potential structural profile at the bottom) constitutes the 2D building footprint. We then search the other dominant structural profiles ($\mathbf{DSP}_i$) from the top to bottom. We identify $\mathbf{DSP}_1$ (i.e., the top dominant structural profile) as the first potential structural profile whose surface area is larger than a threshold θ, where $\theta = 0.2 \cdot \mathbf{Area}(\mathbf{PSP}_K)$ by default. For each potential structural profile $\mathbf{PSP}_j$, we compute its structure difference with the previously identified dominant structural profile $\mathbf{DSP}_i$. A new dominant structural profile can be identified only if the structure difference is sufficiently large. Specifically, in the local projected point set of layer $\mathcal{F}_j$, we count the number (N_{out}) of points that are located outside of the shape of $\mathbf{PSP}_{j-1}$. To be robust against noise, we also compute the area difference between $\mathbf{PSP}_j$

ALGORITHM 1: Layer-wise Proxy Reconstruction

Input: Sparse point cloud $\mathcal{P}$ of a building and the number of layers K
Output: Building proxy geometry

Initialize the potential structural profile set $\bigcup_{j=1}^{K} \mathbf{PSP}_j$;
Initialize the dominant structural profile set $\mathbf{DSP} \leftarrow \varnothing$;
Compute the total area of 2D building footprint:

$$\mathbf{Area}_{total} \leftarrow \mathbf{Area}\left(\mathbf{PSP}_K\right)$$

foreach $\mathbf{PSP}_j$ **in** $\bigcup_{j=1}^{K} \mathbf{PSP}_j$ **do**
  if $\mathbf{Area}_j \geq 0.2 \cdot \mathbf{Area}_{total}$ **then**
    $m \leftarrow j$;
    $\mathbf{DSP}_1 \leftarrow \mathbf{PSP}_m$;
    $k \leftarrow 2$;
    break;
  end
end
foreach $\mathbf{PSP}_j$ **in** $\bigcup_{j=m+1}^{K-1} \mathbf{PSP}_j$ **do**
  According to Eq. 5, compute 3D points number gain $G_{point}^{(j)}$;
  if $G_{point}^{(j)} \geq \eta$ **then**
    According to Eq. 5, compute the area gain $G_{area}^{(j)}$;
    if $G_{area}^{(j)} \geq \varphi$ **then**
      $\mathbf{DSP}_{k-1} \leftarrow \mathbf{PSP}_{j-1}$;
      $\mathbf{DSP}_k \leftarrow \mathbf{PSP}_j$;
      $k \leftarrow k + 1$;
    end
  end
  continue;
end
Extrude polyhedral cells from **DSP** to assemble the proxy geometry.

and the previously identified dominant structural profile $\mathbf{DSP}_i$. Then $\mathbf{PSP}_j$ is said to be a dominant structural profile if it satisfies the following two criteria:

$$G_{point}^{(j)} = \frac{N_{out}}{N_{j-1}} \geq \eta, \quad G_{area}^{(j)} = \frac{\mathbf{Area}(\mathbf{PSP}_j)}{\mathbf{Area}(\mathbf{DSP}_i)} \geq \varphi, \quad (5)$$

where the default thresholds are set to $\eta = 0.2$ and $\varphi = 1.1$. We identify all dominant structural profiles by iteratively processing all potential structural profiles from the top to bottom. See Algorithm 1 for more details. Please note that when a new dominant structural profile ($\mathbf{PSP}_j$) is identified, the position and shape of the previous dominant structural profile should also be updated. Thus, we use $\mathbf{PSP}_{j-1}$ to replace the previous dominant structural profile. The reason is that although the shape of $\mathbf{PSP}_{j-1}$ is similar to the previous dominant structural profile, $\mathbf{PSP}_{j-1}$ has a larger convex hull which is more suitable for satisfying the safety requirement.

Finally, we extrude each dominant structural profile from bottom to up until it touches the upper dominant structural profile to obtain a convex polyhedral cell. These polyhedral cells are then stacked together to form the complete volume of the proxy geometry.

Surface extraction. Now we have reconstructed a proxy geometry by assembling multiple polyhedral cells, which can already be directly used for visualization. Such models contain many interior faces due to the direct stacking process (see Fig. 6 (f) for an illustration), making them unsuitable for aerial path planning, because existing aerial path planning algorithms [8], [11], [12] require to sample the outer surface of a building to compute reconstructability.

Based on the fact that the surface area of the dominant structural profiles is monotonously increasing from the top to bottom, we perform incremental surface extraction to ensure only the outer surface of a building is obtained. Specifically, when we extrude a dominant structural profile (that is a 2D convex hull), we exclude part of its top faces that are within the 2D projection of its immediate upper dominant structural profile (except for the top-most one) and its complete bottom (except for the bottom-most one). Finally, the restricted Delaunay triangulation method is used to triangulate newly added areas to generate a watertight mesh model.

Non-buildings reconstruction. Compared with urban buildings that span a wide range of height, non-building objects (*e.g.*, ground and trees) are less critical for aerial path planning because UAVs typically fly above a certain altitude to avoid collisions. In this work, we reconstruct the overall proxy of non-buildings by adopting a bilinear interpolation method given its higher efficiency compared to the Poisson reconstruction method. Specifically, we first project the non-building points to the ground plane and build a 2D grid with a one-meter resolution to uniformly sample the projected area. Then we lift each vertex of the grid to a height value interpolated from its adjacent 3D points. This way, a 2.5D mesh is efficiently created to approximate the non-buildings. The proxies of both buildings and non-buildings together allow safe aerial path planning.

TABLE 1: Statistics on the nine test scenes from the proxy and scene reconstruction dataset. #Img_proxy and #Img_final denote the numbers of images for proxy and final reconstruction, respectively. #3DInst is the number of building instance proxies in each scene. #SfM_pts is the average number of SfM points per building instance.

Scene	Data Type	#Img_proxy	#3DInst	#SfM_pts	#Img_final	Area (m^2)
AK-1 (Fig. 8)	Synthetic	95	30	354	2581	40000
JPN-1 (Supplementary Materials)	Synthetic	106	17	966	938	15912
CT-1 (Supplementary Materials)	Synthetic	82	32	336	2036	18721
Downtown (Fig. 1)	Real	7,000	255	625	50305	2959358
Residence-1 (Fig. 7)	Real	766	126	479	7835	471390
Residence-2 (Fig. 7)	Real	198	196	380	24765	800842
Campus (Fig. 9)	Real	1539	12	5713	5391	209812
Polytech (Supplementary Materials)	Real	500	1	7134	1230	11162
SI-PARK (Supplementary Materials)	Real	1019	14	676	4285	497754

6 Experimental Results

In this section, a set of experiments were conducted on both synthetic and real urban scenes of different scales to validate the proposed approach. After introducing our new datasets, we start with inspecting the performance of InstFormer against state-of-the-art instance segmentation methods. The effectiveness of our proxy reconstruction for path planning is then qualitatively and quantitatively evaluated by comparing it with several 3D proxy generation approaches. Finally, we apply our approach to capture real-world scenes to achieve high-quality 3D reconstructions. All experiments were conducted on a desktop computer equipped with an Intel i7-7700k processor with 4.2 GHz and 32 GB RAM. We implemented InstFormer in PyTorch based on MMDetection toolbox [70]. Offline training of InstFormer was on two NVIDIA GeForce RTX-A6000 (48GB memory) GPUs, and AdamW was selected as the optimizer. We trained it for 30 epochs on two datasets (our proposed dataset and the Mapping Challenge [71]), with training times of 12 hours and 72 hours, respectively.

6.1 Datasets

Aerial instance segmentation dataset. For the training and evaluation of *InstFormer*, we have created a new dataset consisting of 720 nadir images from four cities captured with varying flight altitudes, and all building instances in these images have been manually annotated by eight students of computer science, using the annotation tool of LabelMe [72]. A few annotated images from the building instance segmentation dataset are shown in the Supplementary Materials. Unlike existing instance segmentation datasets (e.g., COCO, PASCAL VOC) that all target general objects, we focus on multi-view images of buildings captured by drones for 3D urban building reconstruction, where the photos capture both roofs and facades with high resolution. Besides, building scales vary largely, and different buildings overlap from different perspectives. These characteristics pose quite new challenges to instance segmentation.

Proxy and scene reconstruction dataset. For a comprehensive quantitative evaluation, we first test our method on synthetic scenes with ground-truth geometry, which allows a quantitative assessment of the reconstruction performance. Although there exist several virtual scenes that have been created in the previous work [8], [10], [11], the covered scenes are small in size and contain only a few (smaller than 10) buildings in each scene. In this work, we introduce a new synthetic benchmark with three larger-scale virtual scenes

TABLE 2: Quantitative comparison with state-of-the-art 2D instance segmentation methods on the proposed dataset.

Methods	Object Detection				Instance Segmentation			
	AP	AP_{50}	AP_{75}	AP_L	AP	AP_{50}	AP_{75}	AP_L
Mask R-CNN	48.9	70.9	54.7	50.4	48.3	71.4	54.8	49.9
Swin Transformer	47.5	70.6	52.1	48.8	47.1	72.1	52.4	48.5
ConvNext-V2	49.1	70.8	54.5	51.6	48.6	71.0	54.5	50.0
Mask2Former	54.1	76.2	60.4	55.7	52.5	76.8	58.4	53.6
Cascade Mask R-CNN	48.9	69.3	54.9	50.4	47.2	69.5	54.1	48.7
DetectoRS	53.6	75.0	58.7	55.2	51.7	75.8	58.8	53.4
SCNet	51.8	74.6	57.8	53.3	50.6	75.8	57.5	52.2
InstFormer	**54.9**	**77.0**	**61.8**	**56.5**	**53.1**	**77.9**	**59.7**	**54.8**

TABLE 3: Quantitative comparison with state-of-the-art 2D instance segmentation methods on the Mapping Challenge dataset [71].

Methods	Object Detection				Instance Segmentation			
	AP	AP_{50}	AP_{75}	AP_L	AP	AP_{50}	AP_{75}	AP_L
Mask R-CNN	74.7	94.7	86.2	89.9	68.2	94.6	82.9	80.7
Swin Transformer	78.1	94.9	90.6	88.5	74.4	94.9	89.1	84.1
ConvNext-V2	83.0	95.9	92.7	95.2	79.1	95.9	91.7	90.0
Mask2Former	83.2	96.1	92.8	94.7	78.4	96.1	91.5	89.5
Cascade Mask R-CNN	80.3	95.9	91.7	92.6	75.0	95.9	90.2	86.4
DetectoRS	81.0	95.7	91.7	94.0	77.0	95.7	90.4	88.7
SCNet	74.0	94.7	85.5	88.9	69.7	94.2	83.3	84.0
InstFormer	**84.1**	**96.9**	**93.8**	**95.8**	**80.5**	**96.8**	**92.8**	**90.9**

containing dozens of buildings (see Tab. 1), rich geometric details, and realistic appearances. For each scene, we use the Unreal Engine‡ and the physical engine of Airsim§ to simulate the drones to capture the scene and generate highly realistic images. Tab. 1 reports the detailed statistics of our new dataset. Please refer to the Supplementary Materials and video for the visualization of the three virtual scenes.

We also evaluate our method on a dataset of six real urban scenes. All images were captured using a DJI Phantom 4 RTK, which is a single-camera drone with a focal length of 24mm. The images for the proxy reconstruction were captured using the aerial paths generated by [9]. After the reconstruction of the 3D scene proxies using our proposed method, we used [11] to generate the optimized aerial paths for the second-pass image capturing. Tab. 1 reports the statistics of this dataset. Note that we use ContextCapture to produce SfM sparse point clouds and MVS reconstruction given its high efficiency. However, we do not rely on any specific package. The open-source packages, such as COLMAP, VisualSFM, and PMVS, can also be used.

6.2 Evaluation on Instance Segmentation

Comparison on 2D instance segmentation. The performance of InstFormer is first thoroughly evaluated by quantitative comparisons with state-of-the-art instance segmentation methods including Cascade models (Cascade Mask R-CNN [45], DetectoRS [73], SCNet [47]) and none-cascade models (Mask R-CNN [42], Mask2Former [74], Swin Transformer [75], ConvNeXt-V2 [76]). We retrain and test all these models on our new aerial instance segmentation dataset. The evaluation metric is the standard average precision calculated using mask Intersection-over-Union (IoU). It measures the precision between predictions and ground-truth annotations in a range of IoU thresholds, *e.g.*, AP_{50} and AP_{75} denoting the scores with IoU thresholds of 50% and 75%, while AP indicating the average score with IoU thresholds from 50% to 95% with a step size of 5%. Since buildings are relatively large objects in the scene, we also compute AP_L for evaluating the average precision of large instances. Tab. 2 reports the quantitative results of these segmentation methods. The comparison shows that the proposed InstFormer achieves the best performance for both building detection and instance segmentation, indicating the superiority of our model on large-scale building instance segmentation.

We have also evaluated all of the methods using another public large-scale dataset from Mapping Challenge [71], aiming to detect buildings from high-resolution satellite imagery in different urban settings. This dataset consists of a training set of 280,741 images (300×300 pixels), a validation set of 60,317 images, and a test set of 60,697 images. Tab. 3 shows the quantitative prediction results, and it can be seen that Instformer continues to achieve the best performance. The comparison in terms of AP_L shows the perception capacity of Instformer for large buildings. Meanwhile, the best value of AP indicates the superiority of our model in multi-scale building instance segmentation. In terms of object detection, Instformer consistently achieves the best AP, the key evaluation metric in the Mapping Challenge. Moreover, Instformer also performs the best in terms of AP_L in object detection, which further proves its advantage over other alternatives in detecting large buildings.

Instance segmentation of dense reconstruction. It is difficult to quantitatively evaluate instance segmentation directly in 3D due to the lack of ground truth. Thus, we carried out a visual inspection by transferring the instance segmentation result on sparse points to the dense mesh obtained in the final reconstruction. Specifically, we extract the building footprint by first projecting all of the sparse points to the ground and constructing the α-shape from the projection points. Then for each point in the dense mesh, we find the closest instance in the sparse cloud and with its projection point located within the corresponding building footprint. A KD-tree is used to accelerate the process of querying the closest point. Fig. 1 and Fig. 7 show our instance segmentation results on three large urban scenes, in which both building proxies and dense meshes are accurately segmented.

6.3 Evaluation on Synthetic Scenes

Experimental setup. We conduct synthetic experiments in a virtual environment that is built on Unreal Engine and Airsim simulator. In the initial pre-acquisition phase, a UAV equipped with a single camera performs high-altitude flight, capturing the ground surface with a vertical perspective for full coverage imaging (with an 80% along-track overlap and a 70% across-track overlap). This step results in nadir aerial images. Subsequently, we compute a sparse point cloud which is used to construct a proxy model. We use the path planning algorithm proposed by [11] for evaluation, which first generates a rich initial view set according to uniformly sampled points on the surface of the proxy model. Then a Max-Min optimization is proposed to select a minimal set of viewpoints that maximize the reconstructability under the same number of viewpoints. An effective flight path is

‡. https://www.unrealengine.com/
§. https://microsoft.github.io/AirSim/

TABLE 4: Quantitative evaluation and comparison on proxy generation and final 3D reconstruction on three synthetic datasets. For the final reconstruction, *Error* and scene coverage (measured by *Completeness*) are recorded. $\#Images$ denotes the number of pre-captured photos for proxy reconstruction. The first- and second-place performances are highlighted using bold and italic fonts, respectively. LWPxy represents our layer-wise proxy.

Scene	#Images	SFM (min)	Method	Proxy Recon. (min)	Proxy Face Num.	Proxy Size (Mb)	Error↓ 90% (cm)	Error↓ 95% (cm)	Comp.↑ 2 cm (%)	Comp.↑ 5 cm (%)	Comp.↑ 10 cm (%)
AK-1	95	2.7	Coarse	0.87	18189	0.65	8.43	18.14	29.41	57.49	*70.25*
			Convex Hull	0.95	20101	0.66	8.77	20.99	29.54	57.65	70.20
			CSF	2.33	49737	1.86	*8.16*	*18.00*	29.52	57.23	69.51
			MVS Dense	15	503729	40.4	8.44	18.66	*29.84*	*57.74*	**70.39**
			LWPxy	1.16	20990	0.73	**7.88**	**16.06**	**30.12**	**57.87**	70.12
JPN-1	106	3.08	Coarse	0.7	9392	0.33	7.76	20.35	28.59	47.58	*56.02*
			Convex Hull	0.77	10724	0.38	*6.68*	*17.89*	28.94	47.71	55.85
			CSF	1.83	46357	1.73	7.05	18.24	28.50	47.14	55.45
			MVS Dense	11.5	343195	27.7	7.15	18.81	*28.97*	*47.79*	56.02
			LWPxy	0.92	11652	0.43	**6.46**	**17.81**	**29.14**	**47.98**	**56.12**
CT-1	82	2.4	Coarse	1.08	8833	0.31	3.54	12.36	33.73	51.88	57.01
			Convex Hull	1.15	11059	0.39	*2.98*	8.26	*33.89*	52.11	57.00
			CSF	2.58	35659	1.30	2.99	**7.14**	33.21	51.56	56.63
			MVS Dense	12.92	255001	18.9	3.34	10.86	**34.14**	*52.16*	*57.08*
			LWPxy	1.28	12509	0.42	**2.90**	*7.24*	33.83	**52.23**	**57.16**

Fig. 7: Instance segmentation results on the *Residence-1* (top) and *Residence-2* (bottom) real scenes. The three rows of each scene demonstrate the building instance proxies, the final textured mesh of the scene, and the scene mesh with highlighted building instances, respectively. The building instances are randomly colored.

TABLE 5: Ablation study on the effect of instance segmentation and InstFormer. We compare the performance of our InstFormer-based pipeline with two baseline methods. No-seg: directly slice and layer the entire scene without instance segmentation. Clustering: segment the buildings based on 3D clustering.

Scene	Method	Acc.↓ 90% (cm)	Acc.↓ 95% (cm)	Comp.↑ 2 cm (%)	Comp.↑ 5 cm (%)	Comp.↑ 10 cm (%)
AK-1	LWPxy	**7.88**	**16.06**	**30.12**	**57.87**	70.12
	Clustering	8.93	21.69	29.88	57.80	70.27
	No-seg	8.27	17.98	29.84	57.86	**70.34**
JPN-1	LWPxy	**6.46**	17.81	**29.14**	**47.98**	**56.12**
	Clustering	8.25	19.83	27.25	46.29	54.75
	No-seg	7.13	**17.58**	28.17	47.27	55.65
CT-1	LWPxy	**2.90**	7.24	**33.83**	**52.23**	**57.16**
	Clustering	3.02	**7.10**	32.75	51.25	56.18
	No-seg	3.48	8.89	32.53	51.32	56.61

created, which passes through all selected viewpoints, to guide the second fine-grained image collection. Finally, the collected images are fed into ContextCapture for detailed reconstruction.

Evaluation metrics. To quantitatively evaluate the scene proxy reconstruction, we follow the metrics that are commonly used in previous path planning methods [8], [11], [13]. The proxy model is indirectly evaluated by comparing the quality of the final reconstructed detail model after the second flight, because the proxy model affects path planning which in turn determines the quality of the final reconstruction.

Specifically, we use the point-level metrics introduced by [8] to compute the reconstruction quality: *Error* and *Completeness*. *Error* measures the geometric accuracy of the reconstruction. It is computed as the average distance between the vertices of the ground-truth model and the reconstruction. Considering noise and outliers, this metric is evaluated on the majority of the points, i.e., 90% and 95% of the points that have a distance smaller than x centimeters. A smaller *Error* value indicates a higher accuracy. *Completeness* measures the coverage of the ground truth by the reconstructed model. We compute the minimal distance between the points on the ground truth to their closest points on the reconstruction, and *Completeness* is then defined as the percentages of the distances smaller than a threshold. A

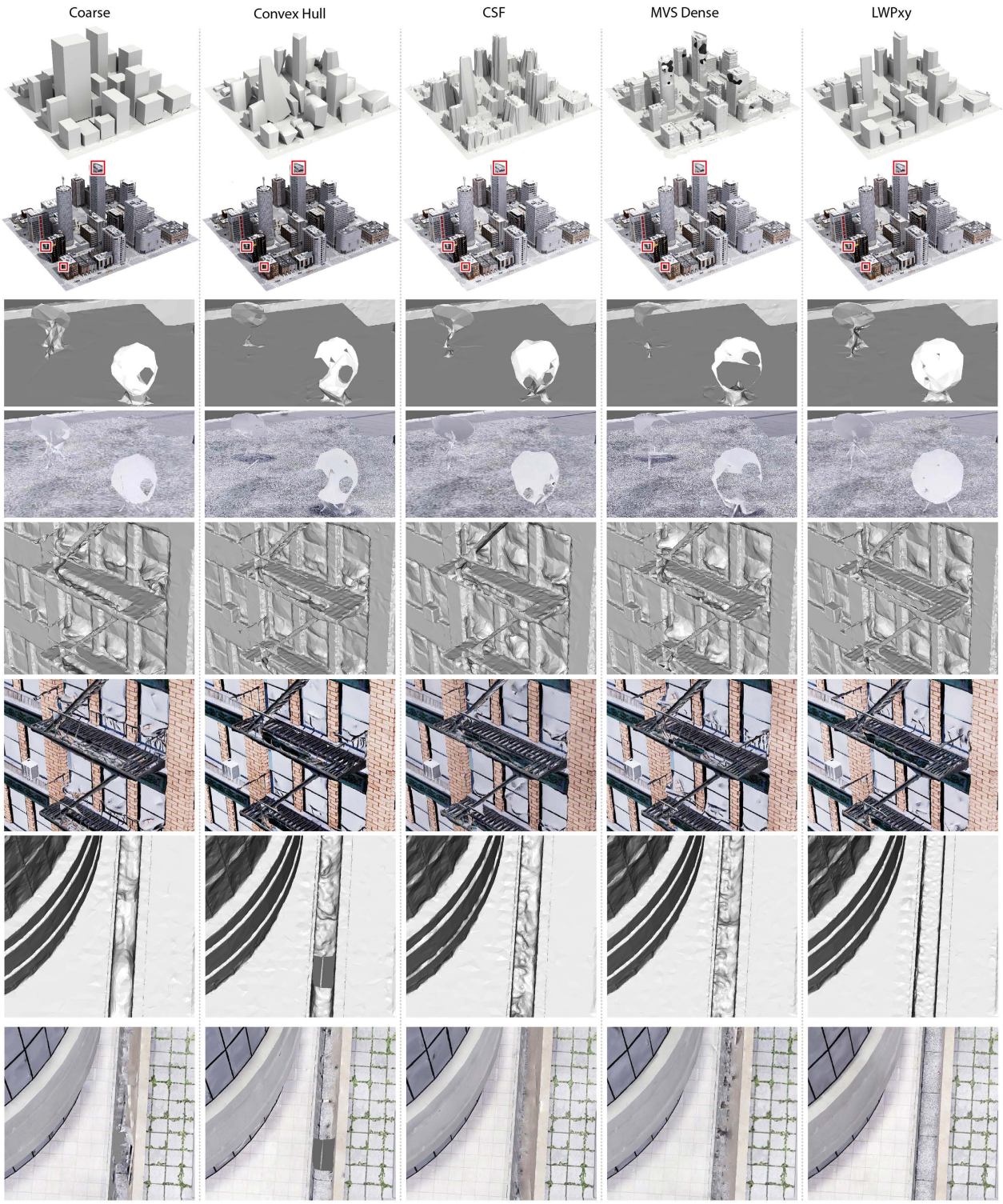

Fig. 8: Visual comparison on the virtual scene *AK-1*, for evaluating different scene proxies for aerial path planning. The top row shows the generated proxies from the sparse input (please refer to Figure 6 in the supplementary materials for the input sparse points), and then the proxy geometries are used for path planning to collect more images to generate the final reconstruction results (demonstrated in the second row). The blown-up views in the rest of the figure show the details of the 3D reconstructions.

larger *Completeness* value indicates higher completeness.

Comparisons. We compare the quality of our layer-wise proxy (denoted as *LWPxy*) against those generated by several proxy alternatives, including the *coarse* proxy by finding the bounding box of the point cloud of each segmented building, *3D convex hull* of each building, and a surface approximation approach as the intermediate proxy based on cloth simulation filtering (*CSF*) [77]. Note that both

TABLE 6: Quantitative comparison of proxy generation and final 3D reconstruction (*i.e.*, , the reconstruction quality, *Error*, and scene coverage, *Completeness*) of paths produced by using different proxies on the real scene *Polytech*.

Scene	#Images	SFM (min)	Method	Proxy Recon. (min)	Proxy Face Num.	Proxy Size (Mb)	Error↓ 90% (cm)	Error↓ 95% (cm)	Comp.↑ 2 cm (%)	Comp.↑ 5 cm (%)	Comp.↑ 10 cm (%)
Polytech	500	34.15	Coarse	2.84	70434	2.17	63.97	111.34	*15.67*	44.92	58.82
			Convex Hull	2.95	66705	2.65	*63.65*	108.74	9.83	36.73	57.81
			CSF	2.33	122216	4.91	67.93	111.46	12.68	46.36	62.27
			MVS Dense	487	6102497	111	**63.12**	**105.81**	15.23	*49.15*	*64.12*
			LWPxy	3.08	61735	2.5	65.32	*106.02*	**15.96**	**49.77**	**64.17**

coarse and *3D convex hull* models are automatically built based on our instance segmentation for a fair comparison. Moreover, we adopt the *MVS dense* mesh reconstructed by the commercial software, ContextCapture, as a fine proxy.

We conduct the comparison using the three virtual scenes. Specifically, we first generate building proxies using each competing method and then derive a flight trajectory using the same aerial path planning algorithm [11]. The visualization results of path planning for different proxy models can be found in the Supplementary Materials. Finally, the drone is flown along the generated path to capture images to reconstruct a fine-grained 3D scene model. To ensure a fair comparison between different proxies, we follow the protocol proposed by [8] and constrain the number of planning viewpoints to be as consistent as possible, resulting in a similar total number of final captured photos. This way, the path lengths and acquisition times for different methods are also similar. By fixing the number of images, we can isolate the effect of camera position on reconstruction quality, where the camera position is related to the proxy model.

Tab. 4 reports the quantitative evaluation results of proxy generation and final reconstruction. We can see that the efficiency of our proxy generation method and the face number in the reconstructed proxy are comparable to the simple *coarse* and *3D convex hull* approaches, while *CSF* and *MVS dense* take much longer time and result in larger models. In terms of final reconstruction quality, our approach generally attains higher scores than other alternatives, which can be observed from the *Error* and *Completeness* measures, indicating its superiority to better capture scene geometries including the details. We can also observe that *MVS dense* does not demonstrate the best results. This is because the point cloud reconstructed from only nadir images has serious incompleteness (especially near the vertical surfaces), resulting in proxy models with large holes and inaccuracies (see Fig. 8). A visual comparison of the proxies and final scene reconstructions associated with the five methods is demonstrated in Fig. 8. Compared to the fine proxy of *MVS dense*, our proxy is more compact yet still achieves comparable or even better reconstructions. From these comparisons, we can conclude that with a more accurate proxy geometry, our approach can achieve better accuracy and completeness, leading to better-detailed reconstruction.

Effect of InstFormer. As has been shown, instance segmentation plays a crucial role in the proposed urban reconstruction pipeline. To evaluate the efficacy of instance segmentation, we have implemented two baseline approaches based on directly slicing the scene to generate a scene proxy. The first baseline is denoted by No-seg which slices and layers the entire scene directly without instance segmentation. For the projected points of each layer, we calculate a polygon contour using the 2D α-shape algorithm. The polygon contours of all the layers stacked together approximate the coarse geometry of a building, serving as the scene proxy model. The second baseline approach to achieving the segmentation of buildings is based on clustering. First, we use the CSF method [77] to roughly segment the buildings from the ground. The building points are then projected to the ground plane, and we use DBSCAN [78] to cluster the points into building instances. After that, the same slicing algorithm is applied to generate the scene proxy. We have applied these methods to three virtual scenes, and the results are demonstrated in Tab. 5. From the results, we can see that though these alternative approaches can also achieve a rough approximation of the scene, their final reconstruction quality is much lower than our results based on InstFormer.

6.4 Evaluation on Real Scene Reconstruction

In this section, we discuss the results of our method applied to real scenes and conduct a comparison with baselines.

Quantitative evaluation. We plan aerial paths based on the proxy models generated using our method and other alternatives for a real scene called *Polytech* (visualized in the Supplementary Materials). Then we capture the scene using the derived aerial paths for high-quality building reconstruction, and we compare the results using different proxy generation methods. The *Polytech* scene contains a single complex building, for which a high-quality complete point cloud is acquired using a LiDAR scanner with a ranging accuracy of 2mm. From the laser scan, a ground-truth mesh is reconstructed for the quantitative evaluation of the methods. Tab. 6 summarizes the evaluation results. Similar to the results on the synthetic scenes, our approach achieves the best performance in terms of *Completeness*. The performance regarding the *Error* metric is also generally comparable to that of *MVS dense*. The overall comparison demonstrates that our approach can efficiently generate lightweight proxy models and yield an accurate and complete final scene reconstruction.

Visual comparison and detail recovery. Next, we evaluate the performance of our method on real-world reconstruction using several large outdoor scenes. Fig. 1 and Fig. 7 show the results of our proxies and the final textured models of three scenes of different scales. To better understand the superiority of our proxy generation approach, we compare the final reconstruction results to those obtained from other proxy reconstruction methods. The reconstructed models on the *Campus* scene are shown in Fig. 9, where we also make a comparison of the details of the reconstructed models. The zoomed-in views reveal that our approach can recover more

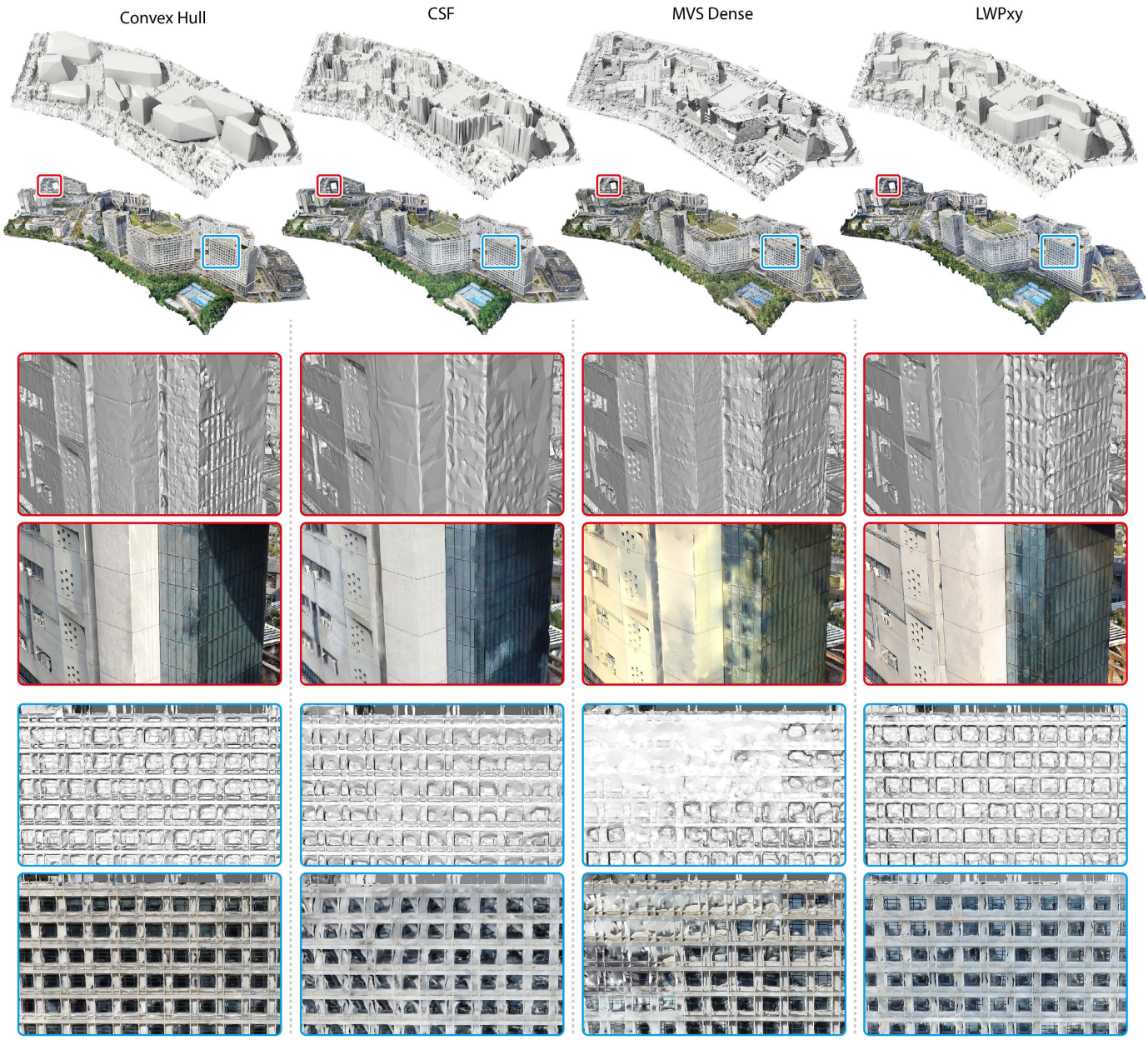

Fig. 9: Visual comparison on the real scene *Campus*, for evaluating the effect of different proxy generation methods on the performance of the final reconstruction. Please zoom in to the blown-up views to see the details of the 3D reconstructions.

geometric details. The visual comparisons on the real scenes of *Polytech* and *SI-PARK* are provided in the Supplementary Materials and video.

Flexible capture by instance planning. With the instance segmentation of all buildings, we can plan a more accurate and complete path for each building, which enables fine-grained and flexible data capture to obtain a more accurate reconstruction. To achieve that, we have tested two different strategies for aerial path planning using the same algorithm [11]:

- *Plan_single*: Generate a single flight trajectory for each building based on the instance information.
- *Plan_all*: Generate an optimized trajectory for the entire scene containing all buildings.

Fig. 10 demonstrates the derived aerial paths and the final reconstruction results of the *Campus* scene. With the strategy of *Plan_all*, all buildings are captured at the same time but on different days, and some buildings are not fully covered. Therefore, the reconstructed model easily contains noticeable visual artifacts, which can be observed from the finer-level details in Fig. 10. In contrast, with the guidance of instance information, fine-grained path planning can be performed for each building. Besides, viewpoints focusing on the same building can be integrated into a flight path, which enables more efficient capture of important building details.

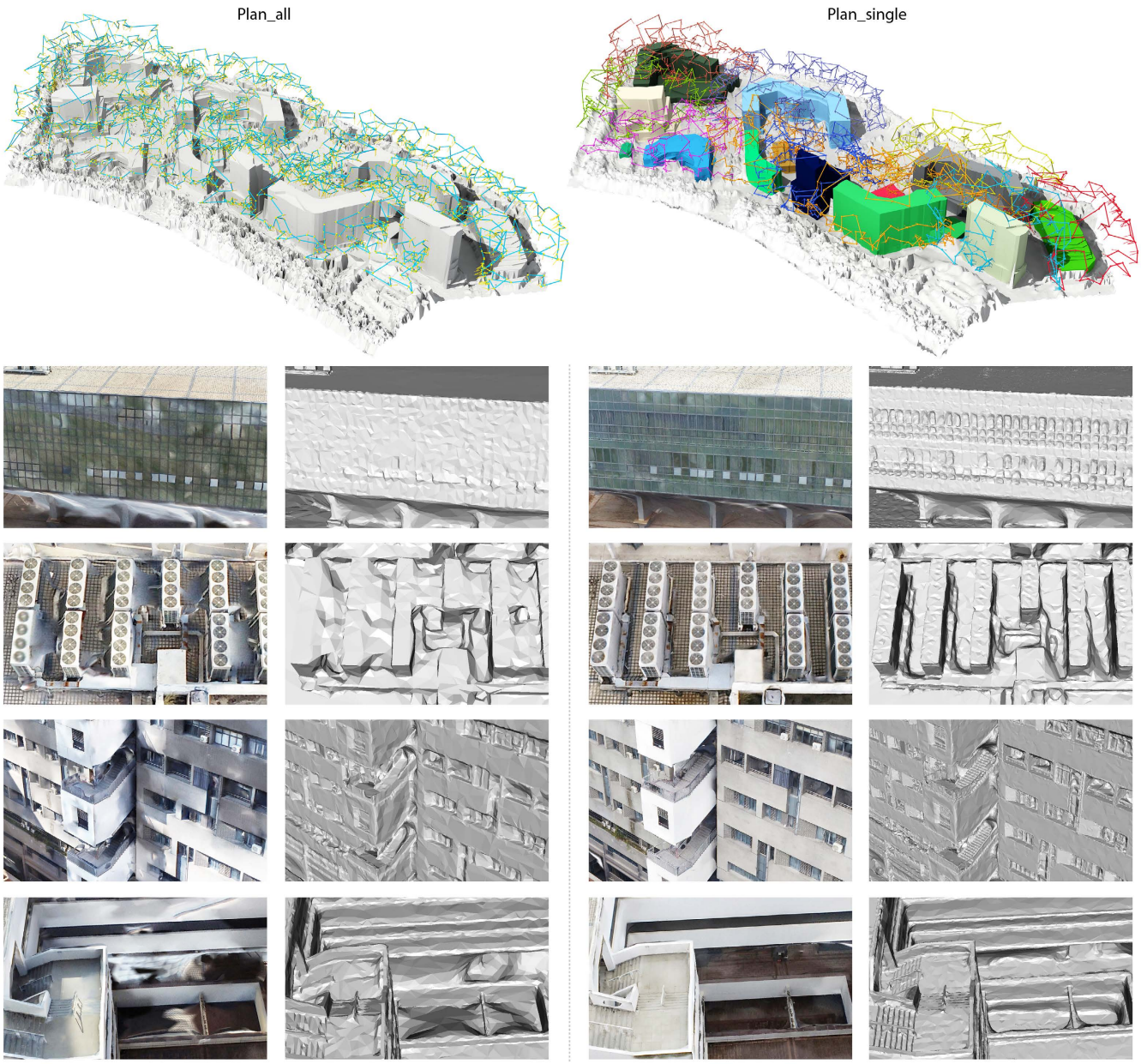

Fig. 10: Visual comparison of the final reconstruction results using two different aerial path planning strategies. The instance information enables fine-grained aerial path planning and the generation of consistent texture maps.

Fig. 11: Our proxy generation method does not deliver the exact geometry of slanted roofs (left) and buildings with a wider top than the bottom (right).

6.5 Limitations

Our layer-wise proxy reconstruction method is designed based on the assumption that building geometry is described by an arrangement of stacked prisms, making it particularly suitable for the majority of high-rise buildings in reality. However, for buildings with slanted roofs, our method can only approximate them with flat tops. In rare cases where buildings have a top wider than the bottom, our method will also fail. Fig. 11 shows our reconstruction results on two such examples. Though the reconstructed proxy does not deliver the exact geometry of the roofs, it still conveys the overall building geometry, and it is sufficient for planning high-quality aerial paths to ensure image capturing with better coverage and at a finer level of detail in another pass of data acquisition.

7 Conclusion and Future Work

We have presented a novel workflow and two algorithms for efficient and effective 3D building instance proxy reconstruction for large urban scenes. Our workflow attempts

high-quality urban reconstruction from a different perspective, *i.e.*, by generating high-quality 3D building proxies rather than pure aerial path optimization. Extensive experiments on several large urban scenarios have validated the effectiveness and practicality of the proposed workflow and its main modules. Specifically, the building proxies generated using our method can better express the scene geometry compared to previous methods, and they are particularly suitable for generating finer-grained aerial paths to further improve the accuracy and enrich the geometric detail of architectural models.

Our work reveals that improving the quality of building proxies provides a straightforward way to address several challenges in UAV data acquisition for high-quality urban reconstruction, such as incomplete scene coverage, lack of semantics, low efficiency, and low reliability of path planning. In future work, we plan to extend this idea to other common urban objects, such as trees and bridges, to allow the creation of semantic-rich detailed 3D models of scenes.

ACKNOWLEDGMENTS

We sincerely thank the anonymous editor and reviewers for their valuable and constructive comments. This work was supported in parts by the NSFC (U2001206, U21B2023, 62172416, U22B2034), DEGP Innovation Team (2022KCXTD025), Shenzhen Science and Technology Program (KQTD20210811090044003, RCJC20200714114435012, JCYJ20210324120213036), and Youth Innovation Promotion Association of the Chinese Academy of Sciences (2022131).

REFERENCES

[1] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. Van Gool, and W. Purgathofer, "A survey of urban reconstruction," *Comp. Graph. Forum*, vol. 32, no. 6, pp. 146–177, 2013.

[2] J. Xiao and Y. Furukawa, "Reconstructing the world's museums," *Int. J. Comput. Vis.*, vol. 110, pp. 243–258, 2014.

[3] C. Häne, C. Zach, A. Cohen, and M. Pollefeys, "Dense semantic 3d reconstruction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 9, pp. 1730–1743, 2017.

[4] C. Poullis, "Large-scale urban reconstruction with tensor clustering and global boundary refinement," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 5, pp. 1132–1145, 2020.

[5] C. Mostegel, M. Rumpler, F. Fraundorfer, and H. Bischof, "Uav-based autonomous image acquisition with multi-view stereo quality assurance by confidence prediction," in *CVPR Workshops*, 2016, pp. 1–10.

[6] R. Huang, D. Zou, R. Vaughan, and P. Tan, "Active image-based modeling with a toy drone," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6124–6131.

[7] Q. Kuang, J. Wu, J. Pan, and B. Zhou, "Real-time uav path planning for autonomous urban scene reconstruction," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 1156–1162.

[8] N. Smith, N. Moehrle, M. Goesele, and W. Heidrich, "Aerial path planning for urban scene reconstruction: A continuous optimization method and benchmark," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 37, no. 6, pp. 183:1–183:15, 2018.

[9] M. Roberts, D. Dey, A. Truong, S. Sinha, S. Shah, A. Kapoor, P. Hanrahan, and N. Joshi, "Submodular trajectory optimization for aerial 3d scanning," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 5324–5333.

[10] B. Hepp, M. Nießner, and O. Hilliges, "Plan3d: Viewpoint and trajectory optimization for aerial multi-view stereo reconstruction," *ACM Trans. Graph.*, vol. 38, no. 1, pp. 4:1–4:17, 2018.

[11] X. Zhou, K. Xie, K. Huang, Y. Liu, Y. Zhou, M. Gong, and H. Huang, "Offsite aerial path planning for efficient urban scene reconstruction," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 39, no. 6, pp. 192:1–192:16, 2020.

[12] Y. Liu, R. Cui, K. Xie, M. Gong, and H. Huang, "Aerial path planning for online real-time exploration and offline high-quality reconstruction of large-scale urban scenes," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 40, no. 6, pp. 226:1–226:16, 2021.

[13] H. Zhang, Y. Yao, K. Xie, C.-W. Fu, H. Zhang, and H. Huang, "Continuous aerial path planning for 3d urban scene reconstruction," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 40, no. 6, pp. 225:1–225:15, 2021.

[14] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 29:1–29:13, 2013.

[15] S. Fuhrmann and M. Goesele, "Floating scale surface reconstruction," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 33, no. 4, pp. 46:1–46:11, 2014.

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

[17] L. Nan and P. Wonka, "Polyfit: Polygonal surface reconstruction from point clouds," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2353–2361.

[18] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, G. Guennebaud, J. A. Levine, A. Sharf, and C. T. Silva, "A survey of surface reconstruction from point clouds," *Comp. Graph. Forum*, vol. 36, no. 1, pp. 301–329, 2017.

[19] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proc. ACM SIGGRAPH*, 1997, pp. 209–216.

[20] D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," in *Proc. ACM SIGGRAPH*, 2004, pp. 905–914.

[21] Y. Verdie, F. Lafarge, and P. Alliez, "LOD generation for urban scenes," *ACM Trans. Graph.*, vol. 34, no. 3, pp. 30:1–30:14, 2015.

[22] X. Gao, K. Wu, and Z. Pan, "Low-poly mesh generation for building models," in *Proc. ACM SIGGRAPH*, 2022, pp. 1–9.

[23] J. Guo, Y. Liu, X. Song, H. Liu, X. Zhang, and Z. Cheng, "Line-based 3d building abstraction and polygonal surface reconstruction from images," *IEEE Trans. on Vis. and Comp. Graph.*, 2022.

[24] B. Tan, N. Xue, T. Wu, and G.-S. Xia, "Nope-sac: Neural one-plane ransac for sparse-view planar 3d reconstruction," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–15, 2023.

[25] F. Lafarge, R. Keriven, M. Brédif, and H.-H. Vu, "A hybrid multiview stereo algorithm for modeling urban scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 5–17, 2013.

[26] L. Zhang, J. Guo, J. Xiao, X. Zhang, and D.-M. Yan, "Blending surface segmentation and editing for 3d models," *IEEE Trans. on Vis. and Comp. Graph.*, vol. 28, no. 8, pp. 2879–2894, 2020.

[27] A. Kaiser, J. A. Ybanez Zepeda, and T. Boubekeur, "A survey of simple geometric primitives detection methods for captured 3d data," *Comp. Graph. Forum*, vol. 38, no. 1, pp. 167–196, 2019.

[28] A.-L. Chauve, P. Labatut, and J.-P. Pons, "Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 1261–1268.

[29] H. Lin, J. Gao, Y. Zhou, G. Lu, M. Ye, C. Zhang, L. Liu, and R. Yang, "Semantic decomposition and reconstruction of residential scenes from lidar data," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 32, no. 4, pp. 66:1–66:10, 2013.

[30] A. Monszpart, N. Mellado, G. J. Brostow, and N. J. Mitra, "Rapter: rebuilding man-made scenes with regular arrangements of planes," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 34, no. 4, pp. 103:1–103:12, 2015.

[31] T. Kelly, J. Femiani, P. Wonka, and N. J. Mitra, "Bigsur: large-scale structured urban reconstruction," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 36, no. 6, pp. 204:1–204:16, 2017.

[32] J.-P. Bauchet and F. Lafarge, "Kinetic shape reconstruction," *ACM Trans. Graph.*, vol. 39, no. 5, pp. 156:1–156:14, 2020.

[33] Q.-Y. Zhou and U. Neumann, "2.5 d dual contouring: A robust approach to creating building models from aerial lidar point clouds," in *European Conference on Computer Vision (ECCV)*, 2010, pp. 115–128.

[34] I. Garcia-Dorado, I. Demir, and D. G. Aliaga, "Automatic urban modeling using volumetric reconstruction with surface graph cuts," *Computers & Graphics*, vol. 37, no. 7, pp. 896–910, 2013.

[35] J. Ren, B. Zhang, B. Wu, J. Huang, L. Fan, M. Ovsjanikov, and P. Wonka, "Intuitive and efficient roof modeling for reconstruction and synthesis," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 40, no. 6, pp. 249:1–249:17, 2021.

[36] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 36, no. 4, pp. 78:1–78:13, 2017.

[37] M. Reed and P. Allen, "Constraint-based sensor planning for scene modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1460–1467, 2000.

[38] O. A. M. Maldonado, S. Hadfield, N. Pugeault, and R. Bowden, "Next-best stereo: Extending next-best view optimisation for collaborative sensors," *Proc. BMVC*, pp. 1–12, 2016.

[39] T. Koch, M. Körner, and F. Fraundorfer, "Automatic and semantically-aware 3d uav flight planning for image-based 3d reconstruction," *Remote Sensing*, vol. 11, no. 13, p. 1550, 2019.

[40] W. Gu, S. Bai, and L. Kong, "A review on 2d instance segmentation based on deep neural networks," *Image and Vision Computing*, vol. 120, p. 104401, 2022.

[41] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact: Real-time instance segmentation," in *IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 9156–9165.

[42] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[43] Z. Huang, L. Huang, Y. Gong, C. Huang, and X. Wang, "Mask scoring r-cnn," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6402–6411.

[44] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8759–8768.

[45] Z. Cai and N. Vasconcelos, "Cascade r-cnn: High quality object detection and instance segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 5, pp. 1483–1498, 2021.

[46] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "Hybrid task cascade for instance segmentation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4969–4978.

[47] T. Vu, K. Haeyong, and C. D. Yoo, "Scnet: Training inference sample consistency for instance segmentation," in *AAAI Conference on Artificial Intelligence*, 2021, pp. 2701–2709.

[48] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully convolutional instance-aware semantic segmentation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2359–2367.

[49] X. Liang, L. Lin, Y. Wei, X. Shen, J. Yang, and S. Yan, "Proposal-free network for instance-level object segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2978–2991, 2017.

[50] S. Kong and C. C. Fowlkes, "Recurrent pixel embedding for instance grouping," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 9018–9028.

[51] H. Chen, K. Sun, Z. Tian, C. Shen, Y. Huang, and Y. Yan, "Blendmask: Top-down meets bottom-up for instance segmentation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8573–8581.

[52] J. Chen, Y. Xu, S. Lu, R. Liang, and L. Nan, "3-d instance segmentation of mvs buildings," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–14, 2022.

[53] K. Chen, Z. Zou, and Z. Shi, "Building extraction from remote sensing images with sparse token transformers," *Remote Sensing*, vol. 13, no. 21, 2021.

[54] L. Wang, S. Fang, X. Meng, and R. Li, "Building extraction with vision transformer," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–11, 2022.

[55] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[56] L. Yi, W. Zhao, H. Wang, M. Sung, and L. J. Guibas, "Gspn: Generative shape proposal network for 3d instance segmentation in point cloud," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3947–3956.

[57] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia, "Pointgroup: Dual-set point grouping for 3d instance segmentation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 4867–4876.

[58] W. Wang, R. Yu, Q. Huang, and U. Neumann, "Sgpn: Similarity group proposal network for 3d point cloud instance segmentation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2569–2578.

[59] J. Hou, A. Dai, and M. Nießner, "3d-sis: 3d semantic instance segmentation of rgb-d scans," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4421–4430.

[60] S. Chen, J. Fang, Q. Zhang, W. Liu, and X. Wang, "Hierarchical aggregation for 3d instance segmentation," in *IEEE International Conference on Computer Vision (ICCV)*, 2021, pp. 15 467–15 476.

[61] T. Vu, K. Kim, T. M. Luu, X. T. Nguyen, and C. D. Yoo, "Softgroup for 3d instance segmentation on 3d point clouds," in *CVPR*, 2022.

[62] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe, "Mask3D for 3D Semantic Instance Segmentation," in *International Conference on Robotics and Automation (ICRA)*, 2023.

[63] M. Chen, Q. Hu, Z. Yu, H. THOMAS, A. Feng, Y. Hou, K. McCullough, F. Ren, and L. Soibelman, "Stpls3d: A large-scale synthetic and real aerial photogrammetry 3d point cloud dataset," in *33rd British Machine Vision Conference BMVC*, 2022.

[64] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 770–779.

[65] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 11 784–11 793.

[66] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 740–755.

[67] J. Wang, K. Chen, R. Xu, Z. Liu, C. C. Loy, and D. Lin, "Carafe: Content-aware reassembly of features," in *IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 3007–3016.

[68] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pvtv2: Improved baselines with pyramid vision transformer," *Computational Visual Media*, vol. 8, no. 3, pp. 415–424, 2022.

[69] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," in *AAAI Conference on Artificial Intelligence*, 2020, pp. 12 993–13 000.

[70] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "MMDetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

[71] S. P. Mohanty, J. Czakon, K. A. Kaczmarek, A. Pyskir, P. Tarasiewicz, S. Kunwar, J. Rohrbach, D. Luo, M. Prasad, S. Fleer *et al.*, "Deep learning for understanding satellite imagery: An experimental survey," *Frontiers in Artificial Intelligence*, vol. 3, pp. 534 696:1–534 696:21, 2020.

[72] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: a database and web-based tool for image annotation," *Int. J. Comput. Vis.*, vol. 77, no. 1, pp. 157–173, 2008.

[73] S. Qiao, L.-C. Chen, and A. Yuille, "Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 10 213–10 224.

[74] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 1290–1299.

[75] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *IEEE International Conference on Computer Vision (ICCV)*, 2021, pp. 10 012–10 022.

[76] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie, "Convnext v2: Co-designing and scaling convnets with masked autoencoders," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 133–16 142.

[77] W. Zhang, J. Qi, P. Wan, H. Wang, D. Xie, X. Wang, and G. Yan, "An easy-to-use airborne lidar data filtering method based on cloth simulation," *Remote sensing*, vol. 8, no. 6, p. 501, 2016.

[78] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, 1996, pp. 226–231.