FULLY AUTOMATED SCAN-TO-BIM VIA POINT CLOUD INSTANCE SEGMENTATION

D. Campagnolo[1], E. Camuffo*[2], U. Michieli[2], P. Borin[3], S. Milani[2], A. Giordano[1]*

[1] Department of Civil Environmental and Architectural Engineering; University of Padua
[2] Department of Information Engineering; University of Padua
[3] Department of Civil, Environmental, Architectural Engineering and Mathematics; University of Brescia

ABSTRACT

Digital reconstruction through Building Information Models (BIM) is a valuable methodology for documenting and analyzing existing buildings. Its pipeline starts with geometric acquisition. (*e.g.*, via photogrammetry or laser scanning) for accurate point cloud collection. However, the acquired data are noisy and unstructured, and the creation of a semantically-meaningful BIM representation requires a huge computational effort, as well as expensive and time-consuming human annotations. In this paper, we propose a fully automated scan-to-BIM pipeline. The approach relies on: (i) our dataset (HePIC), acquired from two large buildings and annotated at a point-wise semantic level based on existent BIM models; (ii) a novel *ad hoc* deep network (BIM-Net++) for semantic segmentation, whose output is then processed to extract instance information necessary to recreate BIM objects; (iii) novel model pre-training and class re-weighting to eliminate the need for a large amount of labeled data and human intervention.

***Index Terms*—** Point Clouds, BIM, Semantic Instance Segmentation, Scan-to-BIM, Few Shot Learning.

1. INTRODUCTION

Digital reconstruction of existing buildings through Building Information Models (BIMs) is becoming a popular method to document and analyze heritage and existing buildings using reality acquisition techniques. The inherent nature and purposes of BIM approaches require the combination of fast and accurate modeling of the 3D scene (by means of point cloud modeling with photogrammetry and/or laser scanning), together with some data organization algorithms. Indeed, the acquired 3D points lack an explicit structure, and labeling process is hardly automated and time-consuming.

Scan-to-BIM is the specific process of documenting existing buildings by recognizing objects present in the scene, modeling their geometry, and defining their mutual relationships [1]. Scan-to-BIM is concluded by typical multi-faceted class-type-entity, spatial, and functional BIM organization; this setting works as a base for any building restoration design. For this task, instance-level segmentation and geometry reconstruction represent most of the process complexity. Existing methods have been developed to tackle this issue using machine learning techniques. A first method [2] proposes to use Region Growing algorithms on voxelized input clouds for surface extraction. Points are classified as surfaces belonging to 3 classes, *i.e.*, *walls*, *ceilings*, and *floors*. Surfaces are further analyzed in order to identify the openings present. The output of the algorithm is a model based on semantically classified surfaces, which is considered to be the basis for subsequent work generating a volumetric model. In [3], authors automatically identify elements with Random Forests starting from a series of planar primitives obtained from clouds pre-segmentation, and apply class-specific reconstruction algorithms to create BIM objects. In [4] a voxelization-based methodology is developed with skeleton extraction algorithms to derive a 3D graph of the building. In [5] a semi-automatic approach is proposed, using Random Forests for segmentation and visual programming for geometry generation. Although most of these methods show good results in Scan-to-BIM, they are usually derived from the combination of multiple separate modules, manually tuned and combined. Current approaches are still far from a fully-automated pipeline that requires reasonable computational complexity and reduces the need for human intervention. In this paper, we propose BIM-Net, a deep learning architecture inserted in a fully automated scan-to-BIM pipeline that accurately recognizes objects inside the scanned point clouds and labels samples through semantic segmentation.

Semantic Segmentation is the task of providing dense predictions of input data $\mathbf{x} \in \mathbb{R}^3$, assigning labels $\hat{y}$ to each point according to a predefined set of classes [6]. Popular approaches are based on input voxelization [7, 8], raw point cloud processing [9, 10] or both [11, 7], and recent techniques have also resort to transformers [12]. The task has been successfully applied to architectural data, despite being more popular in autonomous driving [12, 13, 14, 15]. Examples of architectural semantic segmentation benchmarks and labeled architectural datasets exist but are limited in the literature. S3DIS [16] is one of the largest examples, composed of 271 everyday indoor scenes labeled with 13 classes; Arch [17] is a dataset of heritage buildings, composed of 17 scenes annotated with 9

* Denotes equal contribution.

We thank Matteo Ciprian for his contribution to the initial project kickoff. This work is partially supported by SYCURI Project, funded by University of Padua, in the Program 'World Class Research Infrastructure'.

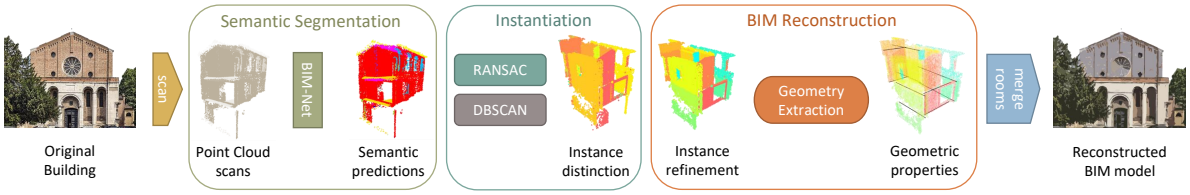

Fig. 1: Representation of the whole Scan-to-BIM pipeline.

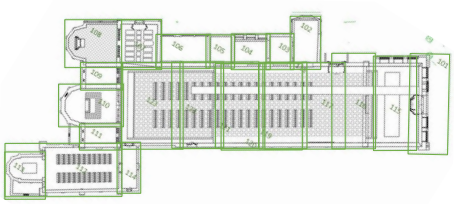

Fig. 2: *Church* floor plan. Green boxes denote sampled rooms.

classes and acquired with photogrammetry. This scarcity of benchmarks motivates our choice to propose a novel Heritage Point cloud Instance Collection (HePIC), annotated from two large-scale buildings at a point-wise semantic and instance level. Moreover, the strong imbalance of point cloud datasets biases segmentation network towards frequent samples [18]: small objects like *stairs* and *doors* have very few points with respect to classes like *walls* and are hardly recognized by standard architectures. For this reason, previous Scan-to-BIM approaches limited BIM reconstruction to the most frequent classes, while our approach reconstructs all the 8 classes in the dataset. Indeed, our lightweight semantic segmentation architecture (BIM-Net++) outperforms heavy networks in dealing with few training data, *i.e.*, few-shot learning.

Few-shot semantic segmentation has been tackled in several ways. In [19] an attention-aware mechanism is developed to build multi-prototypes for class aggregation. In [6, 20], regularization strategies and weighting schemes are used to obtain balanced per-class results. In this paper, we propose *ad hoc* model pre-training and class re-weighting schemes to eliminate the need for a large amount of labeled data and human intervention, upgrading our BIM-Net to BIM-Net++.

To summarize our contribution, we propose: (i) HePIC, a dataset with semantic and instance annotations; (ii) BIM-Net, a novel *ad hoc* deep network for semantic segmentation to extract instances information and recreate BIM objects; (iii) BIM-Net++, an improved version of BIM-Net equipped with novel model pre-training and class re-weighting schemes.

2. METHODOLOGY

Our Scan-to-BIM pipeline (Fig. 1) starts with point cloud acquisition and consists of three main components: (1) BIM-Net to recognize the semantics of architectural elements present in each scene, (2) a module for recognizing single instances of each class, and (3) a module for reconstructing the geometry and the relationships between elements. Note that in this process, each class needs specific algorithms for extracting

Table 1: Basic statistics about HePIC dataset.

Class	# items			total # points		
	Church	*Castle*	Tot	*Church*	*Castle*	Tot
unassigned	23	66	89	216132	296201	512333
beams	147	1809	1956	6393	415069	421462
columns	18	238	256	18951	74588	93539
doors	27	265	292	13962	133257	147219
floors	83	314	397	146516	745314	891830
roofs	218	535	753	564371	632383	1196754
stairs	16	104	120	2394	77089	79483
walls	417	1141	1558	1229115	4229447	5458562
windows	189	81	270	62056	46512	108568
Tot	1250	4441	**5691**	2299983	6609767	**8909750**

different properties necessary for BIM object creation (*e.g.*, location lines for *walls* or location points for *doors*). Different geometric properties rule object hierarchies and relationships in different classes. Similarly, the class determines the selection of the most appropriate instance segmentation algorithm. From this perspective, an accurate semantic segmentation is essential to ensure good results in reconstruction, especially in presence of few labeled data. In the following paragraphs, we describe each component of the pipeline separately.

Dataset Acquisition. Our dataset was acquired via terrestrial laser scanning, supported by photogrammetry for roofing systems. Internal and external acquisitions were performed on two large buildings: a *Castle* and a *Church*[1]. Point labeling for dataset creation was performed using BIM models manually generated. *Castle* and *Church* point clouds were split into $K = 91$ $(68 + 23)$ rooms, each composed of 100k points with both semantic and instance labels, automatically assigned by comparing the point cloud with the BIM model obtained from it (Fig. 2). Eight classes were considered: *walls*, *floors*, *roofs*, *doors*, *windows*, *beams*, *columns*, *stairs*. Any point not belonging to one of these classes was marked as *unassigned*. Basic statistics about HePIC dataset are reported in Tab. 1.

Semantic Segmentation. Automatic point-wise labeling is performed independently on each sampled room, via semantic segmentation. Labels are given according to *semantics*, while a distinction among *instances* is performed successively. BIM-Net, an *ad hoc* voxel-based convolutional semantic segmentation network, was designed for this purpose. This choice is motivated by the fact that state-of-the-art architectures for point clouds are memory-requiring and very slow to be trained. A smaller network, sized with the dataset, with proper loss

[1] *Church* has been created by Maria Rosaria Cundari coordinated by prof. A. Giordano and NEOS s.r.l., *Castle* has been created by LaserPlan+. Codebase available at `https://github.com/LTTM/Scan-to-BIM`.

Fig. 3: Representation of BIM-Net architecture.

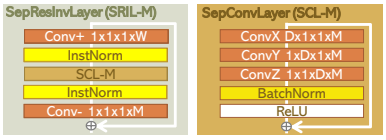

Fig. 4: BIM-Net modules.

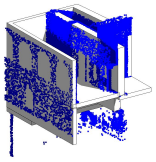

Fig. 5: BIM *walls*.

function can instead obtain good performance while limiting the memory footprint and training time. In particular, BIM-Net is a simple 4-layer convolutional architecture. Each layer is composed of a separable convolution layer (SCL) of size $M = [16, 32, 64, 128]$ respectively. SCL is based on 3D separable convolutions with filters of size $D = 7$, which lighten the computational burden. A Separable Residual Inverse Layer (SRIL) and an Instance Normalization block are added before each SCL to prevent instance-specific mean and covariance shift, simplifying the learning process. SRIL is based on Inverted Residual Block of [21], with input size W. The overall architecture is shown in Figs. 3 and 4. BIM-Net computes point clouds as voxels, with a given granularity (for HePIC, we used $96 \times 96 \times 96$ blocks), as in general architectural point clouds have regular, squared structures. Few-shot learning is finally addressed by BIM-Net++, adopting specific loss and regularization methods, such as *weighting* and *progressive weighting*. The final objective has been defined as: $\mathcal{L} = \frac{\gamma}{\sqrt{n}} \mathcal{L}_{ce} + \gamma_{inv} \mathcal{L}_{cwce}$, where n is each class frequency in the dataset, γ grows linearly with steps (from 0 to 1) and γ_{inv} decreases linearly with steps (from 1 to 0).

Instantiation. Plane extraction algorithms have been used to perform instance segmentation on planar elements (*i.e.*, *walls*, *floors*, and *roofs*) and clustering algorithms for other elements (*i.e.*, *columns*, *beams*, *doors*, *windows*, and *stairs*). For planar elements, we used RANSAC algorithm from Point Cloud Library (PCL) [22]. The algorithm is enforced by computing cosine similarity (between planes of the same instance) and point-plane distance (between each plane and instance centroid), in order to group points belonging to planes with the same normal and distance lower than a threshold. For non-planar elements, we use the DBSCAN algorithm, to cluster points of each instance. In this case, instance boundaries are well-defined and no additional refinements are required.

BIM Reconstruction. BIM entity reconstruction requires geometry reconstruction algorithms specific to each class of objects. Consequently, meaningful parameters and relationships between elements must be defined. Classes are grouped by BIM object creation method: curves for *walls*, *columns*, *beams* and *stairs*, polylines and polycurves for *floors* and roofs, points for *doors* and *windows*. Then, these elements are used

Algorithm 1: *Walls* geometric features extraction.

Input: $\{X_k\}_{k=1}^{K}$ point clouds, $\{\hat{Y}_k\}_{k=1}^{K}$ pred. semantic labels
Output: $\{\mathcal{P}_t\}_{t=1}^{T}$ parameters for each *walls* instance $t \in [T]$
foreach *point set* $\mathcal{X} = X_k|_{k=walls}$ **do**
  while *outliers in* $\mathcal{X}$ **do**
    // *get interior and exterior planes*
    Plane $P_1 \leftarrow$ RANSAC($\mathcal{X}$)
    Centroid $\mathbf{c}_1 \leftarrow$ Avg(P_1 inliers)
    Normal $\mathbf{n}_1 \leftarrow P_1$ normal
    Plane $P_2 \leftarrow$ RANSAC($\mathcal{X} \setminus P_1$ inliers; $\mathbf{n}_1$)
    // *compute wall width and height.*
    // $\mathbf{x}_z$ *is the z coordinate of inliers*
    width, height $\leftarrow$ dist$_{L2}(P_1, P_2)$; $\mathbf{x}_{z,max} - \mathbf{x}_{z,min}$
    // *get principal plane*
    $\mathbf{c}_0 \leftarrow \mathbf{c}_1 + (\text{width}/2) \cdot \mathbf{n}_1$
    Plane $P_0 \leftarrow$ Plane($\mathbf{c}_0$,$\mathbf{n}_1$)
    // $P_\perp$ *is set with origin on* $\mathbf{c}_0$ *and y axis parallel to global z axis*
    Plane $P_\perp \leftarrow$ Plane normal to P_0
    // *get axis endpoints via signed point-plane distance*
    $\mathbf{x}_{start} \leftarrow \max \text{dist}(\mathbf{x}; P_\perp), \mathbf{x} \in \mathcal{X}$
    $\mathbf{x}_{end} \leftarrow \min \text{dist}(\mathbf{x}; P_\perp), \mathbf{x} \in \mathcal{X}$
    $\mathcal{P}_t \leftarrow \{\text{width}, \text{height}, \mathbf{x}_{start}, \mathbf{x}_{end}\}$

Table 2: Comparison with other state-of-the-art architectures.

	5k steps			25k steps		
model	**PA%**	**PP%**	**IoU%**	**PA%**	**PP%**	**IoU%**
SegCloud [8]	17.6	24.7	13.2	17.6	24.7	13.2
Cylinder3D [7]	21.0	23.2	14.2	21.0	23.2	14.2
RandLA-Net [9]	35.4	49.3	27.8	35.6	56.2	28.8
PVCNN [11]	40.7	45.3	32.9	43.3	48.1	34.9
BIM-Net	44.0	**54.4**	37.3	47.1	**58.9**	40.6
BIM-Net++	**56.2**	49.4	**40.9**	**59.1**	53.0	**43.7**

to aggregate points and derive the appropriate BIM representation. For example for class *walls*, geometric parameters used for standard BIM creation are *thickness*, *height* and *axis*. The procedure to extract *walls* geometric properties is reported in Alg. 1, and the reconstructed BIM is shown in Fig. 5, where blue points represent point set $\mathcal{X}$ and white volumes represent the reconstructed wall model. For *floors* and *roofs*, a similar procedure is used, implementing PCL *ConcaveHull* algorithm of [22] for extracting bounding polycurve. Similar algorithm has also been applied to other classes. BIM reconstruction was finally performed through Dynamo visual programming [23].

3. RESULTS

An experimental evaluation has been performed on BIM-Net and HePIC dataset, split into training and test sets. 15 rooms ($\approx$ 1.5M points) from *Castle* and 4 rooms ($\approx$ 400k points) from *Church* have been selected for testing and the rest for training (disjoint with the test ones). An NVIDIA GeForce RTX 3090Ti GPU was employed for all the experiments. The training of

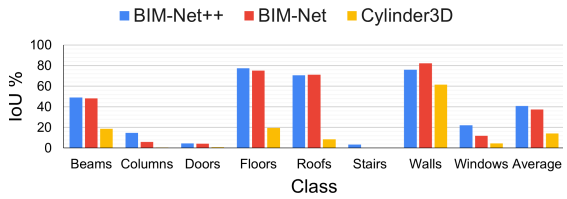

Fig. 6: Per-class mIoU on 25k steps. The last entry denotes the mIoU, averaged over all the classes.

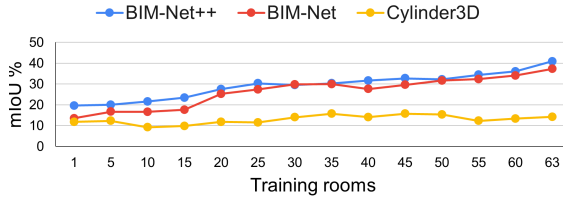

Fig. 7: mIoU vs the number of rooms in the training set. Rooms 1-50 are from *Castle*, 51-63 are from *Church*.

BIM-Net was optimized with Adam, with the learning rate polynomially decreasing from 10^{-3} to 10^{-5} and weight decay 10^{-5}. Batch size was set to 8. The training was performed on 5k steps, but we also tested the final models with longer training (25k steps) for comparison. The results deriving from this longer training do not show a great variation, especially on the state-of-the-art baselines, as the dataset counts very few labeled samples. Tab. 2 shows the final segmentation performance of our architecture on HePIC dataset, compared with others. Being our dataset composed of a few labeled point clouds, state-of-the-art networks (with large number of parameters) tend to overfit the training data, while BIM-Net obtains better performance with reduced training time (Fig. 6). This result is clearly visible also in Fig. 7, where our BIM-Net (red) outperforms Cylinder3D (yellow) by a large margin. Additionally, BIM-Net takes about $1/5$ of training time with respect to Cylinder3D [7]. BIM-Net shows to be suitable for our dataset, but also to achieve state-of-the-art performance on popular point cloud datasets (Tab. 3). Pretraining the network on a similar dataset like S3DIS [16], further improves the performance; this happens both when considering the full set of classes, and when restricting evaluation to the most frequent ones (*i.e.*, *walls*, *roofs*, *floors*). Experiments on the restricted set of classes were performed to show that the network obtains most of the performance on these classes. Indeed, results on BIM-Net trained on 3 classes are comparable with results on these 3 classes, when training on 8 (fifth row in Tab. 3). BIM-Net outperforms baselines in terms of mean Intersection over Union (mIoU) [25], Point Accuracy (PA) [26], and Point Precision (PP) [27], but still lacks accuracy on rare classes (*e.g.*, *stairs*, *doors*, *columns*). The introduction of regularization methods, like *progressive weighting* and an additional *class-wise cross-entropy* (cwce) loss, helps the network improve the overall per-class results. A detailed ablation on progressive weighting schemes is reported in Tab. 4. The last row represents our final BIM-Net++ model. BIM-Net++ obtains the best results also on few-shot learning (Fig. 7). Indeed, when training the network on a few rooms only ($\approx$1 to 20), BIM-Net++ outperforms both BIM-Net and Cylinder3D. Its improvement is minor when the room number increases. Cylinder3D (used with voxels partitions, instead of cylindrical to suit better architectural data), performs poorly in both cases, being suitable only for large-scale datasets [7]. Finally, instance clustering, applied on top of previous semantics, obtains 56.0% mAcc and 0.608 similarity score [28], which results sufficiently accurate for BIM reconstruction, as we show for *walls* in Fig. 5.

Table 3: Different datasets and pretraining schemes with a variable number of classes on 5k steps.

model	dataset	# classes	pretraining	PA%	PP%	IoU%
BIM-Net	Arch [17]	9	×	26.0	39.8	18.4
BIM-Net	S3DIS [16]	13	×	71.7	76.5	59.5
BIM-Net	HePIC	3	×	90.2	89.1	81.3
BIM-Net	HePIC	3	S3DIS [16]	90.7	90.7	83.1
BIM-Net	HePIC	8 (3)	S3DIS [16]	88.4	85.0	76.1
BIM-Net	HePIC	8	×	42.1	**56.2**	35.7
BIM-Net	HePIC	8	Arch [17]	40.8	48.8	34.6
BIM-Net	HePIC	8	S3DIS [16]	**44.0**	54.4	**37.3**

Table 4: Regularization on BIM-Net pretrained on S3DIS using HePIC with 8 classes on 5k steps. w is the loss weight.

loss	w	γ	PA%	PP%	IoU%
$w\ \mathcal{L}_{ce}$	$1/\sqrt{n}$ [20]	×	51.5	48.7	39.3
$w\ \mathcal{L}_{ce}$	$1/n$ [9]	×	59.2	43.3	34.3
$w\ \mathcal{L}_{ce}$	$1/\log(n)$	×	43.6	54.4	37.3
$\mathcal{L}_{cwce}$	×	×	**59.6**	47.1	38.7
$\mathcal{L}_{ohem}$ [24]	×	×	44.9	**59.6**	38.0
$\gamma\ w\ \mathcal{L}_{ce}$	$1/\sqrt{n}$	0→1	51.6	47.1	38.7
$w\ \mathcal{L}_{ce} + \mathcal{L}_{cwce}$	$1/\sqrt{n}$	0→1	56.3	47.8	39.6
$w\ \mathcal{L}_{ce} + \mathcal{L}_{cwce}$	$1/\sqrt{n}$	×	54.4	46.7	38.6
$\gamma\ w\ \mathcal{L}_{ce} + \mathcal{L}_{cwce}$	$1/\sqrt{n}$	1→0	55.7	47.1	38.8
$\gamma\ w\ \mathcal{L}_{ce} + \gamma_{inv}\ \mathcal{L}_{cwce}$	$1/\sqrt{n}$	1→0	53.0	47.3	38.5
$\gamma\ w\ \mathcal{L}_{ce} + \gamma_{inv}\ \mathcal{L}_{cwce}$	$1/\sqrt{n}$	0→1	56.2	49.4	**40.9**

4. CONCLUSION

This paper proposes a pipeline for a fully automated Scan-to-BIM process, discussing the automatic recognition of architectural elements based on deep learning. Our HePIC dataset was developed to compensate for the scarcity of semantic- and instance- level labeled architectural benchmarks. Our BIM-Net was designed to obtain semantic segmentation labeling without human intervention which is then exploited to reconstruct BIM objects; further, its small size allows it to obtain better results on HePIC. BIM-Net was equipped with re-weighting schemes and regularization to obtain BIM-Net++. Our final model outperforms RandLA-Net and Cylinder3D by 14.9% and 29.5% mIoU, respectively. It also improves our base BIM-Net of 3.1% mIoU, reaching good accuracy also on rare classes. Finally, instantiation results are good enough for: (i) providing the overall spatial structure of each building or part of it; (ii) reconstructing the basic elements such as *walls*, *floors*, and *roofs* to create spatial containers; (iii) giving information to create hosted elements (*i.e.*, *windows*, *doors*).

5. REFERENCES

[1] P. Tang, D. Huber, B. Akinci, R. Lipman, and A. Lytle, "Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques," *Automation in Construction*, vol. 19, no. 7, 2010.

[2] X. Xiong, A. Adan, B. Akinci, and D. Huber, "Automatic creation of semantically rich 3d building models from laser scanner data," *Automation in Construction*, vol. 31, 05 2013.

[3] M. Bassier and M. Vergauwen, "Clustering of wall geometry from unstructured point clouds using conditional random fields," *Remote Sensing*, vol. 11, no. 13, 2019.

[4] M. Deidda, A. Pala, and G. M. Sanna, "Modelling a cell tower using sfm: Automated detection of structural elements from skeleton extraction on a point cloud," *Arch. Photogramm. Remote Sens. Spatial Inf.*, vol. XLIII-B2-2020, 2020.

[5] V. Croce, M. G. Bevilacqua, G. Caroti, and A. Piemonte, "Connecting geometry and semantics via artificial intelligence: From 3d classification of heritage data to h-bim representations," *Arch. Photogramm. Remote Sens. Spatial Inf.*, vol. XLIII-B2-2021, 06 2021.

[6] E. Camuffo, D. Mari, and S. Milani, "Recent advancements in learning algorithms for point clouds: An updated overview," *Sensors*, 2022.

[7] X. Zhu, H. Zhou, T. Wang, F. Hong, Y. Ma, W. Li, H. Li, and D. Lin, "Cylindrical and asymmetrical 3d convolution networks for lidar segmentation," *CVPR*, 2021.

[8] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "Segcloud: Semantic segmentation of 3d point clouds," in *3DV*. IEEE, 2017.

[9] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Randla-net: Efficient semantic segmentation of large-scale point clouds," in *CVPR*, 2020.

[10] G. Qian, Y. Li, H. Peng, J.e Mai, H. Hammoud, M. Elhoseiny, and B. Ghanem, "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," in *NeurIPS*, 2022.

[11] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-voxel cnn for efficient 3d deep learning," in *NeurIPS*, 2019.

[12] H. Zhao, L. Jiang, J. Jia, P. HS Torr, and V. Koltun, "Point transformer," in *CVPR*, 2021.

[13] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia, *Stratified Transformer for 3D Point Cloud Segmentation*, 2022.

[14] G. Qian, Y. Li, H. Peng, J. Mai, H. A. A. K. Hammoud, M. Elhoseiny, and B. Ghanem, *PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies*, 06 2022.

[15] Y-Q. Yang, Y-X. Guo, J-Y. Xiong, Y. Liu, H. Pan, P-S. Wang, X. Tong, and B. Guo, *Swin3D: A Pretrained Transformer Backbone for 3D Indoor Scene Understanding*, 2023.

[16] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3D semantic parsing of large-scale indoor spaces," in *CVPR*, 2016.

[17] F. Matrone, A. Lingua, R. Pierdicca, E. S. Malinverni, M. Paolanti, E. Grilli, F. Remondino, A. Murtiyoso, and T. Landes, "A benchmark for large-scale heritage point cloud semantic segmentation," *Arch. Photogramm. Remote Sens. Spatial Inf.*, 2020.

[18] E. Camuffo, U. Michieli, and S. Milani, "Learning from mistakes: Self-regularizing hierarchical semantic representations in point cloud segmentation," *arXiv:2301.11145*, 2023.

[19] N. Zhao, T-S. Chua, and G. H. Lee, "Few-shot 3d point cloud semantic segmentation," in *CVPR*, 2021.

[20] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving," *arXiv: arXiv:2003.03653*, 2020.

[21] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, 06 2018.

[22] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *ICRA*, Shanghai, China, May 9-13 2011, IEEE.

[23] J. Graham, L. Smith, and I. Siegel, *Dynamo for Revit: An Introduction to Programming*, O'Reilly Media, 2016.

[24] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," 2016.

[25] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *CVPR*, 2015.

[26] M. Johnson-Roberson, C. Barto, R. Mehta, S. Nittur Sridhar, K. Rosaen, and R. Vasudevan, "Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks?," *ICRA*, 2016.

[27] A. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," *CVPR*, 06 2019.

[28] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, 1987.