AUTOMATED TRAINING DATA CREATION FOR SEMANTIC SEGMENTATION OF 3D POINT CLOUDS

Sam De Geyter[1,2], Maarten Bassier[1] and Maarten Vergauwen[1]

[1] Dept. of Civil Engineering, TC Construction – Geomatics, KU Leuven – Faculty of Engineering Technology, Ghent, Belgium
(sam.degeyter, maarten.bassier, maarten.vergauwen)@kuleuven.be

[2] MEET HET BV, Mariakerke, Belgium

Commission V, WG V/7

KEY WORDS: Scan-to-BIM, 3D semantic segmentation, 3D training data, Point cloud, LIDAR

ABSTRACT:

The creation of as-built Building Information Modelling (BIM) models currently is mostly manual which makes it time consuming and error prone. A crucial step that remains to be automated is the interpretation of the point clouds and the modelling of the BIM geometry. Research has shown that despite the advancements in semantic segmentation, the Deep Learning (DL) networks that are used in the interpretation do not achieve the necessary accuracy for market adoption. One of the main reasons is a lack of sufficient and representative labelled data to train these models. In this work, the possibility to use already conducted Scan-to-BIM projects to automatically generate highly needed training data in the form of labelled point clouds is investigated. More specifically, a pipeline is presented that uses real-world point clouds and their corresponding manually created BIM models. In doing so, realistic and representative training data is created. The presented paper is focussed on the semantic segmentation of 6 common structure BIM classes, representing the main structure of a building. The experiments show that the pipeline successfully creates new training data for a recent DL network.

1. INTRODUCTION

When creating as-built Building Information Modelling (BIM) models, the common starting point is an accurate 3D point cloud captured by a laser scanner. The process of creating a BIM from point cloud data is typically referred to as Scan-to-BIM. However, the process is mostly manual where an experienced and expensive modeller uses the point cloud data as a background layer to model the BIM geometry. This manual effort makes the process very time consuming, error prone, and expensive. Consequentially it slows down the BIM adoption in the market. One of the key steps, which is in desperate need of automation, is the interpretation of the 3D point cloud data. A labelled point cloud will help a modeller to speed up the modelling and will undoubtedly support further automation of proper BIM object detection and reconstruction.

Recent works in the field of semantic segmentation on 3D point clouds, with in particular RandLA-Net (Hu et al., 2020), have tackled the problem of processing large datasets at once and have achieved an overall accuracy on the S3DIS benchmark (6-fold cross validation) of 88%. However, these state-of-the-art models suffer to preform equally well on real-world data and therefore, are not ready for market adoption. The largest limitation of most state-of-the-art Deep Learning (DL) methods for 3D data is the small amount of labelled training data. The fact that these are data driven methods makes this general lack of sufficient and representative training data to initialize these models a major problem. This also makes it extremely difficult to sufficiently train these models for real market applications. Especially in situations where high accuracies are vital.

To tackle the lack of labelled training data, this work introduces a workflow to generate labelled point clouds from existing Scan-to-BIM processes. In this workflow, manually created BIM models are used to label every point in the point cloud with a set of predefined, frequently occurring BIM classes including floors, ceilings, walls, beams, columns and clutter. Using this approach, a set of labelled point clouds of buildings and scenes representative to the actual applications of Scan-to-BIM projects is created in an automated and time-efficient manner. This allows to enhance the current state-of-the-art models by using data of projects already targeted for Scan-to-BIM and will make them better suited for real-world applications. Additionally, by providing a larger variety of scenes, the models become more robust to segmenting different kinds of scenes, such as residential, commercial and industrial scenes.

In summary, the contributions of this work are:

- A literature study of training data creation methods for semantic segmentation of 3D point clouds.
- An automated pipeline for training data creation from as-Built BIM models and real-world point clouds
- An analysis of the impact of this additional real-world datasets on a state-of-the-art DL model.

The remainder of this work is organised as follows. Section 2 discusses the relevant literature. In section 3 the methodology of the proposed training data creation pipeline is explained. Following, the experiments are presented in section 4. Finally, a discussion of the obtained results of this work and a conclusion will be provided in sections 5 and 6.

2. RELATED WORK

In this section, the related work concerning training data creation and recent advancements in DL techniques targeting the 3D semantic segmentation of point clouds is examined. The literature review in this work is strictly limited to semantic segmentation of point cloud data and the creation of the training data for this point cloud interpretation. The literature review will

start with an overview of recent semantic segmentation techniques in section 2.1 followed by other approaches of creating or augmenting training data in section 2.2.

2.1 Semantic segmentation

The main methods to interpret point cloud data within the field of Machine Learning (ML) are either based on predefined handcrafted features or on DL methods where the features are learned from the training process (Griffiths and Boehm, 2019). Due to recent advances in these DL methods on 2D data interpretation, for instance in object detection and semantic segmentation on images, and the availability of low-cost 3D acquisition devices, for example RGB-D cameras, the extension of 2D DL techniques to 3D has been widely researched (Ahmed et al., 2019). Nevertheless, the full implementation of DL techniques on 3D point cloud data is still facing significant challenges. A key challenge is the unstructured character of the point cloud data (Bello et al., 2020; Griffiths and Boehm, 2019). These data characteristics make deep learning on point cloud data challenging, especially for Convolutional Neural Networks (CNNs) which are based on convolutional operations and thus need an ordered input (Bello et al., 2020). In Griffiths and Boehm, four common approaches of processing this unorderd data are examined. Two approache focus on the creation of an ordered representation of the point cloud. A first approach translates the traditional 2D CNN to 3D by representing the point cloud data in a voxel grid. An example of a voxel based CNN is DeepNet (Hackel et al., 2017) which is based on VoxNet (Maturana and Scherer, 2015) and ShapeNet (Wu et al., 2015). The second approaches introduces the OctNet data representation which is a "hybrid grid-octree" data structure and allows CNNs to process the three dimensional data (Riegler et al., 2017).

The two other approaches are designed to directly process the unordered structure of the point cloud. One approach is an unsupervised learning approach, an example of this approach is FoldingNet (Yang et al., 2018). These unsupervised approaches typically use auto-encoders as learning structure. This type of networks learns to regenerate their input again as output (Griffiths and Boehm, 2019). On the other hand, there are a plethora of supervised ML networks tackling the task of semantic segmentation. Since PointNet (Qi et al., 2017a) different networks and CNN structures tackle the problem of deriving per point labels from irregular point cloud data. PointNet itself does not use a CNN to extract features but is built merely of fully connected layers. Where PointNet fails to generalize on more complex scenes and to identify fine structures, because it does not take local structures into account, its successor PointNet++ (Qi et al., 2017b) succeeds to extract local features on different scales. Besides this pioneering works, lots of efforts have been made to create DL networks to successfully semantically segment point cloud data. Some examples of recent networks are: PointSIFT (Jiang et al., 2018), PointCNN (Li et al., 2018), KPConv (Thomas et al., 2019), PointWeb (Zhao et al., 2019), ShellNet (Zhang et al., 2019) .

The downside of these point-based methods are their large memory and computational costs which makes it difficult to be used on large-scale point clouds directly (Hu et al., 2020). Some methods do allow the direct processing of large-scale point clouds. One of the most performant networks is RandLa-Net (Hu et al., 2020) which does not use time and memory expensive pre-processing steps or downsampling methods. RandLa-Net uses a random downsampling which is much faster. To compensate the loss of information due to the random downsampling they integrate a local feature aggregator. Currently, this state-of-the-art network achieves an Overall Accuracy (OA) of 88% on the Stanford 3D Indoor Set (S3DIS) (Armeni et al., 2016).

2.2 Training data

Unfortunately, DL methods remain data driven methods and thus require a lot of labelled data to be trained. There are some publicly available online datasets such as the S3DIS (Armeni et al., 2016), ScanNet (Dai et al., 2017) and SUN3D (Xiao et al., 2013) which can be used for training and testing DL networks on 3D data. Most other available datasets are focussed on outdoor environments (Fritsch et al., 2013; Geiger et al., 2013; Hackel et al., 2017; Tan et al., 2020). The available indoor datasets containing point clouds are scarce but meshes and RGB-D datasets can be converted to point clouds and therefore can be used for this purpose as was well. Some additional examples of indoor datasets are the NYDV2 (Silberman et al., 2012) and Matterport3D (Chang et al., 2018) datasets consisting of RGB-D images and SceneNN (Hua et al., 2016) captured using RGB-D sensors but already processed to a triangle mesh.

Besides these publicly available datasets, the amount of training data available for 3D DL techniques can still not be compared to the number of available data in 2D DL (Griffiths and Boehm, 2019). To match the successes of 2D DL networks, more high quality training data is needed. The creation of these datasets is a time-consuming and labour-intensive task.

To extend the available training data sets, there are some approaches in literature targeting the creation of synthetic training data or the augmentation of existing data. As many research efforts in this field, the creation of synthetic training data is driven by the field of autonomous driving. Here synthetic data is created by creating point cloud data from a virtual environment or object containing semantic information. This can be done using existing scenes such as from games e.g. GTA-V game scenes are used to create training data point clouds (Hurl et al., 2019; Wu et al., 2018). This use of already existing data from games however raises the problem of insufficient semantic information or detailing to create the required training data (Xiao et al., 2021). To solve this, the use of manually created scenes in 3D modelling software are advised (Xiao et al., 2021). This possibility can be used to create labelled point clouds for training purposes. Similar to above approaches, a scanning simulation (Wang et al., 2019) can be done in other specially created virtual environments such as CARLA (Dosovitskiy et al., 2017).

BIM models can also be used to create synthetic training data. Recent work proposes a workflow spanning 3 commercial software packages to create synthetic training data from BIM models (Ma et al., 2020). Their three steps include an export from Autodesk Revit to an Autodesk AutoCAD file (.dwg) that is than imported and segmented in Trimble SkechUp. Finally, the SkechUp (.skp) file is converted to a labelled point cloud using FME Workbench. They report an increasing IoU when using their synthetic training data combined with real world data. This combination is of specific interest since they also report that models that were only trained on synthetic data tend to underperform on real world data and so do models trained only on synthetic data. Alternatively, the synthetic data can also be augmented e.g. by using PCT (Xiao et al., 2021). PCT is a translator mitigating the difference between real point cloud data and synthetic data (Xiao et al., 2021). The translator alters both the appearance and the sparsity characteristics of the synthetic point cloud data to be more similar to real world data. When using BIM models to generate synthetic training data some other challenges are reported. The absence of clutter and colour

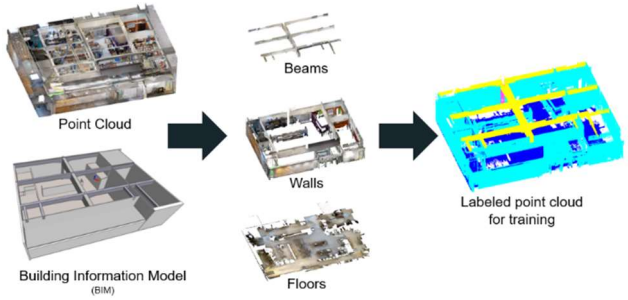

Figure 1. Schematic overview of the Training data creation pipeline.

information forms a major drawback as well as the still needed manual segmentation of the BIM elements (Ma et al., 2020).

Another approach to generate more training data is to make use of data augmentation techniques. In this approach, existing training data is altered with synthetic elements to create new scenes. This is done mostly in the field of point cloud classification. Example methods are PointMixup where an interpolation of different input point clouds leads to new scenes (Chen et al., 2020). Another method is PointAugment which is a framework designed to increase the variety of data when training a network. This is done by optimizing and augmenting the input point clouds (Li et al., 2020). Other approaches create new point cloud scenes by including small point clouds of elements not actually present in the scene. For this purpose, point clouds cut from other real world point clouds can be used or they can be generated from 3D objects (Uggla and Horemuz, 2021). Some other approaches try to minimize the need for training data for example by the use of Transfer learning (Imad et al., 2021) and methods of active learning (Kölle et al., 2020; Lin et al., 2020). Instead of relying on large amounts of training data, these methods increase the efficiency of the already existing training data. As such, they form an important corner stone in the generalisation and practical application of DL-based semantic segmentation on point cloud data.

3. METHODOLOGY

The pipeline presented in this work starts from two common Scan-to-BIM project deliverables, i.e. the point cloud and the As-Built BIM model. Both are aligned in a common coordinate system since the BIM model is derived from the point cloud data. This semantic information or the class to which a point should be assigned, can be derived from the BIM model. An overview of the proposed workflow can be found in Figure 1.

First, a reference point cloud is extracted from the BIM model (Fig. 2). To this end, the BIM elements IFC classes are used to assign the correct labels. Therefore, the labels must be linked to one or more IFC classes with what they correspond. It is important to notice that while for most classes this is a one-on-one mapping i.e Beams solely contain members of the IfcBeam class. Some classes require a more complex mapping. In the presented cases in section 4.1 the Walls class contains the IfcWall, IfcDoor and IfcWindow classes as these classes contain vital wall information but would otherwise be labelled as clutter since they are not part of any structure class. Similarly, Floors and Ceilings only require a subset of the IfcSlab class.

For this last two classes, an extra processing step is required where the top and bottom of each IfcSlab BIM element is assigned to respectively the Floor and Ceiling class. Also, to have a complete ceiling representation the BIM elements of the IFC class IfcCovering and the bottom of IfcRoof elements are mapped to the Ceiling class. To generate a point cloud from the processed BIM elements, a uniform sampling method is used to generate points on the surfaces of the mesh geometries of the BIM elements. As normal, each point is given the normal of the mesh object from which it was sampled. At the same time, the label corresponding to the BIM element is stored. The result is a labelled reference point cloud Q.

In the presented workflow, “non-visible parts” are left within the reference point cloud because in most cases these points will not find a match during the filtering steps presented later on. In some cases, it is possible this causes mislabelled points, especially when an object is not accurately modelled, for example outside as can be seen in Figure 3. This causes only problems in cases where different surfaces of a BIM object correspond to other classes (Ceilings and Floors).

Second, the point cloud P generated by the laser scanner or any other remote sensing technique and the BIM reference point cloud Q are downsampled using a voxel downsampling with the same voxel size to respectively P’ and Q’. This is done to generate representative point clouds and speed up the computation of the filtering steps.

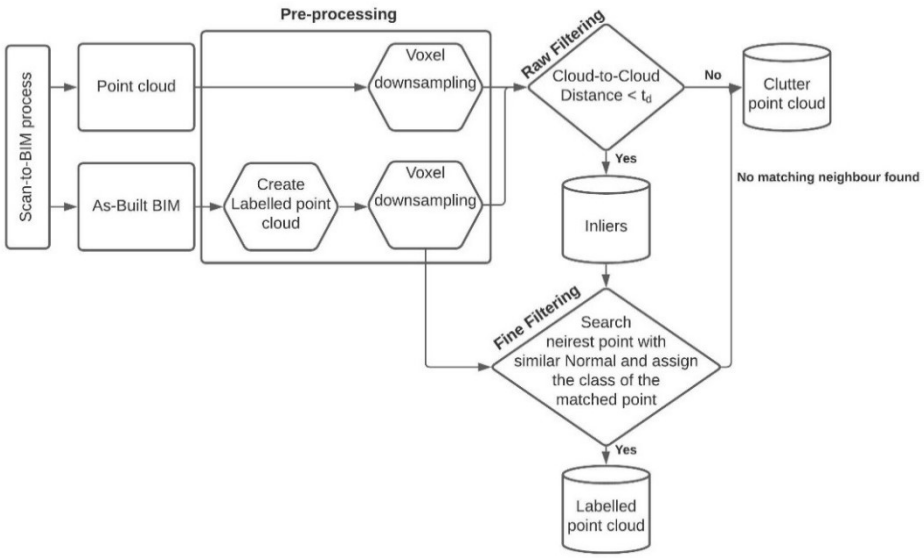

Figure 2. Process map of the proposed training data creation pipeline.

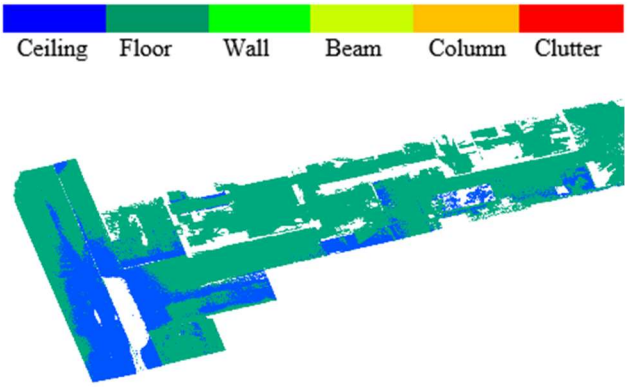

Figure 3. Terrain modelled as IfcSlab can cause misclassifications when the terrain is inaccurately modelled.

Third, a raw filtering is executed by computing the cloud-to-cloud distance between both point clouds. For every point $p_i \in P'$, the closest point in the reference cloud Q' is found based on the Euclidean distance between both points (Eq. 1).

$$I = \{p_i \in P' \mid q_j \in Q' : argmin_{\{q_j\}} (|| p_i - q_j || < t_d)\} \quad \text{(Eq. 1)}$$

Where points that exceed threshold t_d are labelled as Clutter.. This third step makes the process faster because otherwise the fine filtering, which is the most time-consuming step, would result in unnecessary long processing times. For instance, On the dataset of Case 2 the processing times were 1034 seconds for the pre-processing and creation of the reference data, 222 seconds for the raw filtering and 3806 seconds for the fine filtering.

Fourth, the inlier points I are further evaluated based on proximity and normal similarity. In this filtering, every point $p_i \in I$ will search for a point $q_j \in Q'$, in its neighbourhood with a similar normal. Therefore, for every point p_i, a subset $Q_i \subset Q'$ will be created, containing all points of Q' with an Euclidean distance smaller than t_r (Eq. 2)

$$\boldsymbol{Q_i} = \{q_j \in Q' \mid p_i \in I : d = \|p_i - q_j\| < t_r\} \quad \text{(Eq. 2)}$$

Where $\mathbf{Q} = \{Q_1, Q_2, ..Q_i\}$ represents the sets of neighbours of each $p_i \in I$. Each set of points is then iterated starting from the smallest Euclidean distance until the normal similarity condition is met (Eq. 3). With t_n as threshold for the outcome of the dot product. The parameter for t_n can be dependent on the distance between both point, demanding a lower similarity between close points for example. This can be useful in cases where the model is very accurately represents a brick wall where the normal are typically not perfectly aligned with the abstracted BIM geometries.

$$I' = \left\{p_i \in I \mid q_j \in Q' : \overrightarrow{n_{p_i}} \cdot \overrightarrow{n_{q_j}} > t_n\right\} \quad \text{(Eq. 3)}$$

Where $\overrightarrow{n_{p_i}}$ and $\overrightarrow{n_{q_j}}$ respectively are the normals at point $p_i \in I$ and $q_j \in Q$. Each member of the resulting point cloud I' is then labelled with the same label as q_j. Analogue to the raw filtering, if the conditions from Eq. 3 are not met, the point p_i is labelled as clutter. Finally, the point clouds are exported as PCD format per class which is required by the RandLa-Net network data preparation steps.

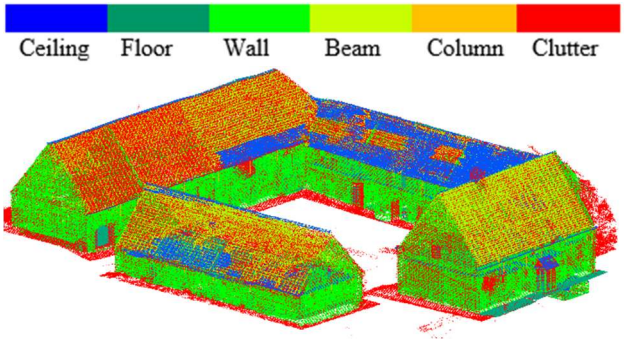

Figure 4. Case 1, a farm containing a farmhouse and barns.

4. EXPERIMENTS

4.1 Training data creation

In this section, several case studies are presented where new training data was created according to the pipeline proposed in section 3. The datasets originate from commercial Scan-to-BIM projects and represent a wide variety of structures including a farm with barns and a retirement home. For all cases the same settings where used, all point clouds and reference clouds are downsampled with a voxel size of 1 cm. Both threshold t_d and t_r are set to 5 cm and t_n was set to 0.9 for points closer than 1 cm and 0.7 for points closer than 0.5 cm.

Case 1 is a complex scene of a farm as shown in Figure 4, containing a farm house and barns. This scene contains several clutter objects and also a complex roof structure that has been modelled in detail using columns and beams. As can be seen in **Figure** 6, the columns and beams are detected and correctly

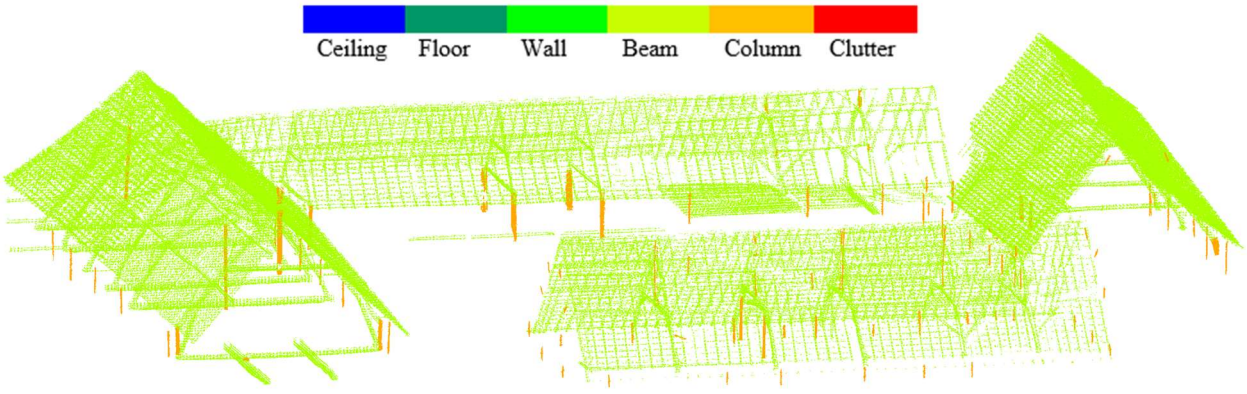

Figure 5. Case 1, overview of the roof structure containing beams and columns.

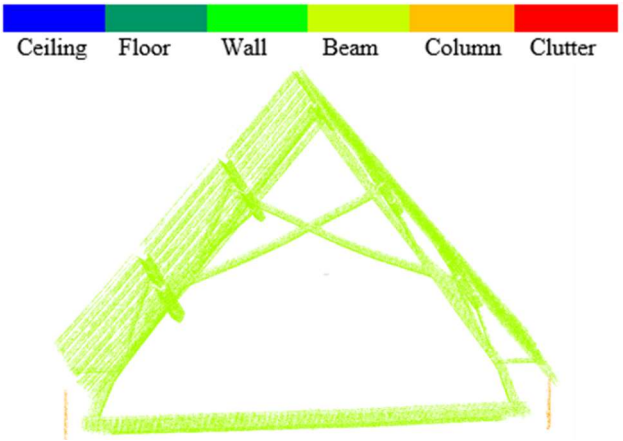

Figure 6. Case 1, detail of the beams correctly labelled in the roof structure.

labelled. Figure 5 shows that some parts of the roof structure are missing, this is because modelling inaccuracies. In this case, it is not possible to increase the threshold parameters because then the roof surface will be considered as the upper side of the beams due to its alignment with the "non-visible" surfaces of the beam geometry which are now neglected due to their bigger difference between the point cloud and the reference cloud. This problem already caused labelling errors on the roof itself, where in some areas both the top and the bottom of the roof layer itself is labelled as ceiling (Fig. 4). This is caused by a roof structure only existing out of roof tiles without any insulation what results in a roof layer smaller than the thresholds resulting in a mislabelling.

Figure 7 shows the cellar of the farm house which has an arched ceiling, this ceiling is made out of bricks and so has varying normals. Despite this diverging normal the ceiling is mostly correctly labelled. The image also shows some clutter that is correctly filtered out of the point cloud such as wires and water hoses.

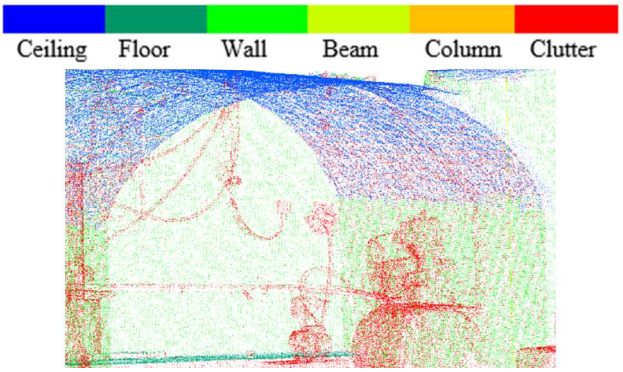

Figure 7. Case 1, detail of the cellar of the farmhouse, containing an arched ceiling.

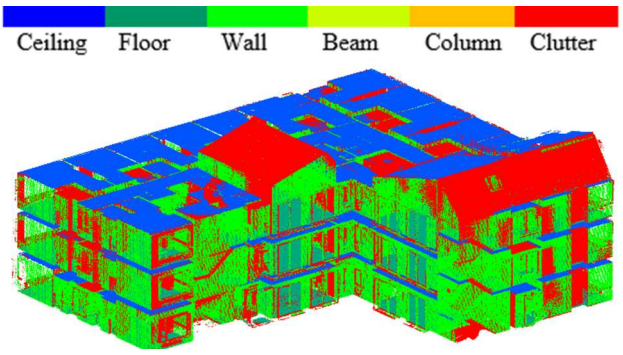

Figure 8. Case 2, a retirement home. Labelled with the thresholds of 5 cm.

Case 2 is a recently completed retirement home. In this case, the BIM model is not derived from the point cloud but is in fact the As-Design BIM that was used during construction. This results in larger inaccuracies between the BIM and the point cloud, where t_d is exceeded (Fig. 8). This results in an excessive labelling of clutter as can be seen in Figure 8. When t_d and t_r are increased to 10 cm the pipeline creates decent training data. Other areas labelled as clutter are: build-in closets, stairs, the ceiling of the top floor which remains a problem due to the large deviations between the As-Design BIM and the acquired point cloud. (Fig. 9).

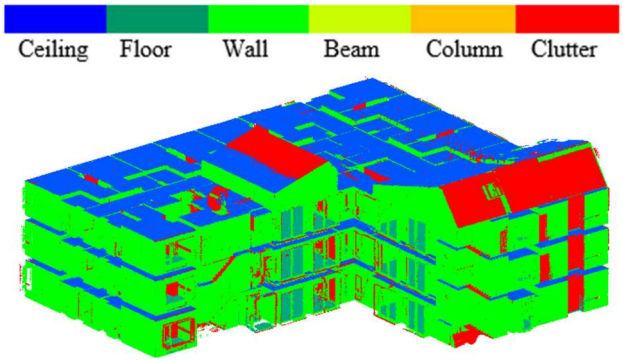

Figure 9. Case 2, with thresholds increased to 10 cm.

This inaccuracies are a result of the mobile mapping method which is used and the accumulated SLAM errors. Other areas labelled as clutter are: build-in closets, stairs, reflections in the windows and open doors. It is important to notice the difference between open and closed doors, where closed doors are part of the wall structure as discussed in section 3 open doors do not represent wall geometry en thus should be labelled as clutter. The dataset contained a minimum of clutter objects because the building was scanned before the building was put into use. A section of the first floor (Fig. 10) shows clearly the proper labelling. This large dataset contains 3 floors and approximate one hundred rooms. The dataset unfortunately does not contain any beams and columns.

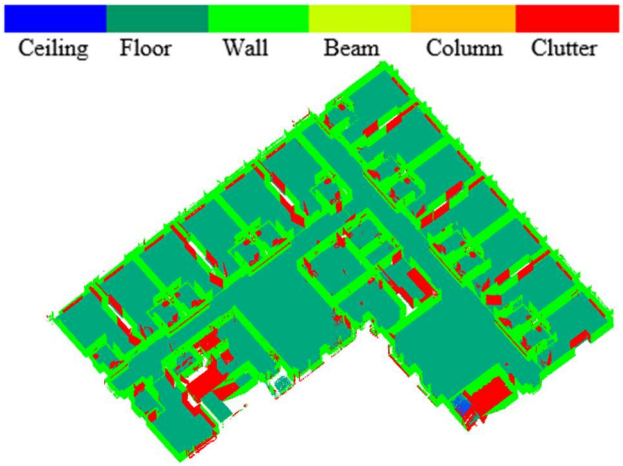

Figure 10. Case 2, section of the first floor showing the room configuration which is similar to the other floors and the labelling of the points.

4.2 Training and testing

The labelled point cloud of case 2 from section 4.1 is used to train and test RandLA-Net. This tests show whether the generated training data can be used to train a DL network. Besides the data generated in section 4.1 the S3DIS dataset is also used and was therefore processed to only 6 classes instead of 13.

To make the coming sections more understandable, the test and train areas will be named, Area 1-6 are the corresponding areas of the S3DIS dataset. Area 7 is the first additional dataset i.e. the entire labelled point cloud of Case 2 containing three storeys. The second dataset that was added is Area 8. This dataset contains the same point clouds as Area 7 but here they are split per storey to better resemble the S3DIS datasets.

For the experiments, the RandLA-Net (Hu et al., 2020) network was used to train and test a model on the data. All models are trained with the same settings. Each model is trained for a 100 epochs consisting of 500 training steps and 150 validation steps. The initial learning rate of the model is set to 0.01 and decreases by 5% every epoch. The training and testing was performed using a NVIDIA GeForce RTX 2080 Ti Turbo GPU server. The separate training processes took approximate 10-12 hours after which the models were saved and tested.

The first experiment tests how models trained on online available training data behave on real-world data. To this end, a set of models was trained on the areas of the S3DIS Dataset. For this first experiment Area 1 and 3 of the S3DIS Dataset were used. After training according to the previously mentioned settings, the model was tested four times. Each model was tested on the same data it was trained on and on an independent Area of the S3DIS dataset. Additionally, every model is tested on both Area 7 and Area 8 of the generated data. Area 4 was used as the independent area because this Area was not used for training the model in this work. The results of all tests are presented in Table 1.

Looking at the results, the areas that are a part of the S3DIS dataset provide results in line with the literature. When providing the models with real-world data, the model clearly underperforms. Especially Area 7 underperforms with a mIoU of only around 50%. Upon inspection, large parts of the floors are mislabelled as ceiling (Fig.11). On the ground floor, the labelling is correct but there are significant errors with the upper floors. In contrast, mislabelled ceilings as floors occur to a much lesser extent. As shown in Figure 11, this mislabelling is mostly found in the centre of rooms which is also the case for the ceilings labelled as floors. When the same model is tested on Area 8, the floors and ceilings are labelled correctly (Fig.12). However, there are more mislabelled points on the walls then in Area 7.

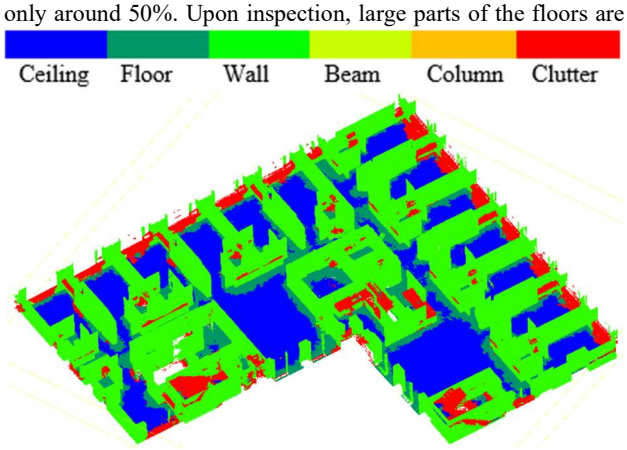

Figure 11. Semantic segmentation results of Area 7 with a model trained on Area 1. This section shows large parts of the floor our mislabelled as ceiling.

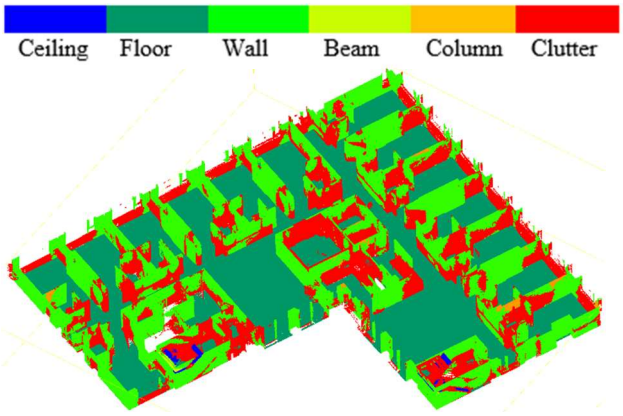

Figure 12. Semantic segmentation results on Area 8 with a model trained on Area 1. Showing the correct labelling of floors and ceilings.

The second experiment tests the performance of models that are trained solely on the generated training data. To this end, different configurations of areas 7 and 8 are evaluated. First, a model was trained from each dataset individually. However, the model trained only on area 7 did not generate any results and during training the mIoU topped around 2%. This is likely due to the extreme size of the point cloud or the simultaneous use of different stories which might confuse RandLA-Net. For Area 8, the model does yield results but significantly underperforms compared to the two S3DIS Areas despite Area 8 containing 2.4 times more points than Area 1 and 5.7 times more points than as Area 3. The lack of performance is thus probably caused by the fact that most rooms in Area 8 are very similar to each other where the S3DIS Areas have more variation of scenes. When this model is tested on Area 7 it gives better results than when tested on the same data it was trained on.

The third experiment tests the performance of models that are trained both on available benchmark data and generated data., To this end, models were trained on both a S3DIS area and Area 7 or Area 8 (Table 1). As expected, the mIoU of models trained on a combination of S3DIS data and data created using the pipeline proposed in section 3 decreases. This is probably caused by the large amount of new data in Area 7 and Area 8 that have very similar scenes. The results of configurations containing Area 7 remain poor and when trained on a S3DIS area extended with Area 8 the labelling of Area 7 completely fails. Even when the

	Tested on								
	Same as training		Area 4		Area 8		(Area 7)		
Trained on	mIoU [%]	OA [%]	mIoU [%]	OA [%]	mIoU [%]	OA [%]	mIoU [%]	OA [%]	
Area 1	77.04	90.96	**90.79**	**97.83**	64.06	73.48	54.30	77.93	
Area 3	76.83	**91.57**	89.69	95.32	61.77	69.40	50.60	74.62	
Area 8	68.54	78.58	83.00	91.74	-	-	**80.82**	**88.74**	
Area 1+7	66.63	78.25	75.58	87.85	84.46	92.42	52.70	73.11	
Area 3+7	59.42	75.95		73.98	87.36	**85.30**	**93.05**	53.12	73.74
Area 1+8	**78.67**	88.20		72.32	84.94	79.13	87.89	5.14	20.57
Area 3+8	73.24	86.32		76.74	84.71	77.46	86.24	2.76	11.06

Table 1. results of the conducted tests. For each test, the training areas, testing areas and the achieved mIoU and OA are provided.

model is trained on an S3DIS area combined with Area 7 there is no real improvement in the results. The same model does generate significantly better results when tested on Area 8. In contrast, the model trained on a S3DIS area and Area 8, clearly underperforms when tested on Area 7. The other results of these models trained on both the S3DIS data and Area 8 look promising. For instance, the results on Area 8 increase significantly. When using this model the results of Area 4 and Area 8 are similar. This was not the case at the first experiments where areas similar to the training area preformed clearly better. The results when tested on Area 8 as can be seen in Figure 13 are clearly better than those of the models trained on only one area.

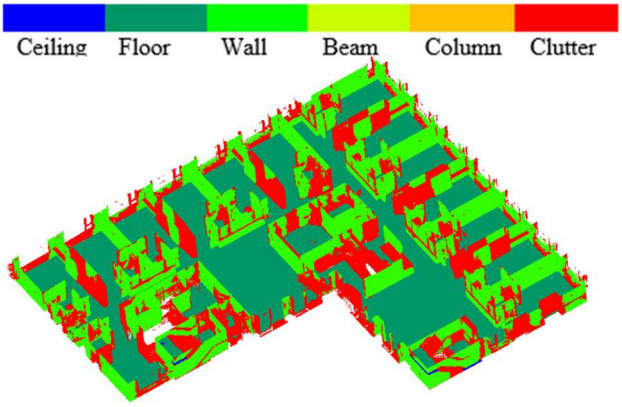

Figure 13. Semantic segmentation results on Area 8 with a model trained on Area 8 an Area 3. Showing a more correct labelling of the wall and clutter regions from for instance the build-in closets.

5. DICUSSION & FUTURE WORK

The training data creation pipeline proposed in section 3 offers the possibility to use existing Scan-to-BIM projects to generate more training data. The inputs of the pipeline are in-line with traditional Scan-to-BIM project deliverables, this allows for a smooth integration with current workflows. The generation of the new training data can be done without user intervention but requires some pre-processing steps and will save large amounts of time compared to manually labelling millions of points. These pre-processing steps such as correctly aligning the point cloud and the BIM model and exporting them to the correct formats (IFC for the BIM model and PCD for the point cloud data) that the pipeline can handle. Support for other formats can be implemented in the future but for BIM the IFC standard is chosen because this is considered the standard and in theory software independent. Nevertheless during testing only IFC files exported from Autodesk Revit where successfully processed.

Also, the reading and writing of the point cloud data can be speed up using other formats. Also, the presence of those "non-visible" elements, discussed in section 4.1, can in some cases result in mislabelling of certain point. This mainly causes problems for small floor and ceiling objects because for those object the top and bottom of the BIM geometry represent different classes. Other problems that occur are problems concerning the accuracy of the registration of the point clouds, the modelling or scanning accuracy. Due to misalignments between the point cloud data and the BIM, parts of structures will be mislabelled. The same problem occurs when the BIM is not updated to the most recent state of the building (as seen in Case 2 in section 4.1), but because mostly the BIM is created from the point cloud this is not a problem in most cases. Another important aspect is the modelling of the BIM, during which the modeller must pay close attention to the elements that are used and the way of modelling. For example, when a flat roof is modelled as a floor, this will be labelled as floor. Also, modelling of unnecessary surfaces close to each other (within 5cm for example) should be avoided, this can occur in scenes were beneath the floor a ceiling is modelled directly against the floor. This will result in mislabelling of the ceiling. To avoid this, one IfcSlabs element should be used and have a sufficient thickness.

Another advantage of the proposed workflow is the realistic character of the scenes, by using the point clouds of real-world scenes the training data contains cluttered objects, occluded areas and noise. These elements are difficult to create in synthetic data. Future work could focus on the combination of the proposed pipeline and the use of synthetic data. By augmenting the data with additional, typically scarce, objects such as beams and columns which could increase their detection rates.

The tests from section 4.2 show that a model preforms significantly better on data similar to the data it was trained on. Also, variation in the training data is needed to obtain decent results. The conducted experiments also suggest a possible problem for processing entire projects at once. According to the experiments, point clouds spanning multiple floors should be avoided or should be split in separate point clouds per floor to provide good results. Adding data spanning multiple floors during training did not solve the problems and training a new model on a point cloud spanning multiple floors also did not succeed. Therefore, it is suggested to split the data before processing either manually or automatically.

Through the findings presented in this section, a "one model fits all" approach of the Scan-to-BIM semantic segmentation seems unlikely. A possible approach lies in using multiple models where every model focusses on a different type of building. Afterwards, the results of those different models could be compared and possibly with human intervention, the correct model could be selected. To be able to use such a solution, the processing time needed to semantically segment a point cloud should be reasonable. The network used in section 4.2 can process 103M points under 5 minutes. Such processing times make the suggested approach a viable option to be explored.

6. CONCLUSION

This work proposes a training data creation pipeline where traditional Scan-to-BIM project outputs are used for automated training data generation. Without pre-processing, the process can generate the data without human intervention. By automating this process and minimizing the manual efforts needed to label millions of points, the creation of new training data does not form a hazard for training new Deep Learning models. The experiments show that the created data can be seamlessly combined with already existing online datasets. Nevertheless, it is also indicated that the quality of the results is highly dependent on the type of data the model is trained on. On the one hand the data has to be similar to the training data of the model to achieve decent results. On the other hand, the date used for training needs to contain enough variety since training on monotone or highly similar scenes provides poor results.

ACKNOWLEDGEMENTS

This project has received funding from the VLAIO BAEKELAND programme (grant agreement 552HBC.2020.2819), MEET HET BV for the data and use of equipment, the FWO Postdoc grant (grant agreement: 1251522N) and the Geomatics research group of the Department of Civil Engineering, TC Construction at the KU Leuven in Belgium.

REFERENCES

Ahmed, E., Saint, A., Shabayek, A., Cherenkova, K., Das, R., Gusev, G., Aouada, D., 2019. A survey on Deep Learning Advances on Different 3D DataRepresentations 2657, 1–9. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S., 2016. 3D semantic parsing of large-scale indoor spaces. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 2016-Decem, 1534–1543. https://doi.org/10.1109/CVPR.2016.170

Bello, S.A., Yu, S., Wang, C., Adam, J.M., Li, J., 2020. Review: Deep learning on 3D point clouds. Remote Sens. 12, 1–34. https://doi.org/10.3390/rs12111729

Chang, A., Dai, A., Funkhouser, T., Halber, M., Niebner, M., Savva, M., Song, S., Zeng, A., Zhang, Y., 2018. Matterport3D: Learning from RGB-D Data in Indoor Environments, in: Proceedings - 2017 International Conference on 3D Vision, 3DV 2017. pp. 667–676. https://doi.org/10.1109/3DV.2017.00081

Chen, Y., Hu, V.T., Gavves, E., Mensink, T., Mettes, P., Yang, P., Snoek, C.G.M., 2020. PointMixup: Augmentation for Point Clouds, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer International Publishing. https://doi.org/10.1007/978-3-030-58580-8_20

Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Niebner, M., 2017. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes, in: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V., 2017. CARLA: An Open Urban Driving Simulator 1–16.

Fritsch, J., Kuhnl, T., Geiger, A., 2013. A new performance measure and evaluation benchmark for road detection algorithms, in: IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC. pp. 1693–1700. https://doi.org/10.1109/ITSC.2013.6728473

Geiger, A., Lenz, P., Stiller, C., Urtasun, R., 2013. Vision meets robotics: The KITTI dataset. Int. J. Rob. Res. 32, 1231–1237. https://doi.org/10.1177/0278364913491297

Griffiths, D., Boehm, J., 2019. A Review on deep learning techniques for 3D sensed data classification. Remote Sens. https://doi.org/10.3390/rs11121499

Hackel, T., Savinov, N., Ladicky, L., Wegner, J.D., Schindler, K., Pollefeys, M., 2017. Semantic3D.net: A new Large-scale Point Cloud Classification Benchmark.

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A., 2020. Randla-Net: Efficient semantic segmentation of large-scale point clouds. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 11105–11114. https://doi.org/10.1109/CVPR42600.2020.01112

Hua, B.S., Pham, Q.H., Nguyen, D.T., Tran, M.K., Yu, L.F., Yeung, S.K., 2016. SceneNN: A scene meshes dataset with aNNotations. Proc. - 2016 4th Int. Conf. 3D Vision, 3DV 2016 92–101. https://doi.org/10.1109/3DV.2016.18

Hurl, B., Czarnecki, K., Waslander, S., 2019. Precise Synthetic Image and LiDAR (PreSIL) Dataset for Autonomous Vehicle Perception.

Imad, M., Doukhi, O., Lee, D.J., 2021. Transfer learning based semantic segmentation for 3d object detection from point cloud. Sensors 21. https://doi.org/10.3390/s21123964

Jiang, M., Wu, Y., Zhao, T., Zhao, Z., Lu, C., 2018. PointSIFT: A SIFT-like Network Module for 3D Point Cloud Semantic Segmentation. CoRR abs/1807.0.

Kölle, M., Walter, V., Schmohl, S., Soergel, U., 2020. Hybrid Acquisition of High Quality Training Data for Semantic Segmentation of 3D Point Clouds Using Crowd-Based Active Learning, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Copernicus GmbH, pp. 501–508. https://doi.org/10.5194/isprs-annals-V-2-2020-501-2020

Li, R., Li, X., Heng, P.A., Fu, C.W., 2020. PointAugment: An Auto-Augmentation Framework for Point Cloud Classification. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 6377–6386. https://doi.org/10.1109/CVPR42600.2020.00641

Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B., 2018. PointCNN: Convolution on X-transformed points. Adv. Neural Inf. Process. Syst. 2018-Decem, 820–830.

Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y., 2020. Efficient Training of Semantic Point Cloud Segmentation Via Active Learning, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Copernicus GmbH, pp. 243–250. https://doi.org/10.5194/isprs-annals-V-2-2020-243-2020

Ma, J.W., Czerniawski, T., Leite, F., 2020. Semantic segmentation of point clouds of building interiors with deep learning: Augmenting training datasets with synthetic BIM-based point clouds. Autom. Constr. 113, 103144. https://doi.org/10.1016/j.autcon.2020.103144

Maturana, D., Scherer, S., 2015. VoxNet: A 3D Convolutional Neural Network for real-time object recognition, in: IEEE International Conference on Intelligent Robots and Systems. Institute of Electrical and Electronics Engineers Inc., pp. 922–928. https://doi.org/10.1109/IROS.2015.7353481

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017a. PointNet: Deep learning on point sets for 3D classification and segmentation. Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017 2017-Janua, 77–85. https://doi.org/10.1109/CVPR.2017.16

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017b. PointNet++: Deep hierarchical feature learning on point sets in a metric space. Adv. Neural Inf. Process. Syst. 2017-Decem, 5100–5109.

Riegler, G., Ulusoy, A.O., Geiger, A., 2017. OctNet: Learning Deep 3D Representations at High Resolutions Deep Learning for 3D Data Shape Classification. Proc. IEEE Conf. Comput. Vis. Pattern Recognit. 3577–3586.

Silberman, N., Hoiem, D., Kohli, P., Fergus, R., 2012. Indoor segmentation and support inference from RGBD images Lecture Notes in Computer Science. ECCV'12 Proc. 12th Eur. Conf. Comput. Vis. - Vol. Part V Part V, 746–760.

Tan, W., Qin, N., Ma, L., Li, Y., Du, J., Cai, G., Yang, K., Li, J.,

2020. Toronto-3D: A Large-scale Mobile LiDAR Dataset for Semantic Segmentation of Urban Roadways. https://doi.org/10.1109/CVPRW50498.2020.00109

Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L., 2019. KPConv: Flexible and deformable convolution for point clouds. Proc. IEEE Int. Conf. Comput. Vis. 2019-Octob, 6410–6419. https://doi.org/10.1109/ICCV.2019.00651

Uggla, G., Horemuz, M., 2021. Towards synthesized training data for semantic segmentation of mobile laser scanning point clouds: Generating level crossings from real and synthetic point cloud samples. Autom. Constr. 130, 103839. https://doi.org/10.1016/J.AUTCON.2021.103839

Wang, F., Zhuang, Y., Gu, H., Hu, H., 2019. Automatic Generation of Synthetic LiDAR Point Clouds for 3-D Data Analysis. IEEE Trans. Instrum. Meas. 68, 2671–2673. https://doi.org/10.1109/TIM.2019.2906416

Wu, B., Wan, A., Yue, X., Keutzer, K., 2018. SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud. Proc. - IEEE Int. Conf. Robot. Autom. 1887–1893. https://doi.org/10.1109/ICRA.2018.8462926

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2015. 3D ShapeNets: A Deep Representation for Volumetric Shapes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Xiao, A., Huang, J., Guan, D., Zhan, F., Lu, S., 2021. Transfer Learning from Synthetic to Real LiDAR Point Cloud for Semantic Segmentation.

Xiao, J., Owens, A., Torralba, A., 2013. SUN3D: A database of big spaces reconstructed using SfM and object labels. Proc. IEEE Int. Conf. Comput. Vis. 1625–1632. https://doi.org/10.1109/ICCV.2013.458

Yang, Y., Feng, C., Shen, Y., Tian, D., 2018. FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation BT - Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). Proc. IEEE Conf. Comput. Vis. Pattern Recognit. 3, 206–215.

Zhang, Z., Hua, B.S., Yeung, S.K., 2019. ShellNet: Efficient point cloud convolutional neural networks using concentric shells statistics. Proc. IEEE Int. Conf. Comput. Vis. 2019-Octob, 1607–1616. https://doi.org/10.1109/ICCV.2019.00169

Zhao, H., Jiang, L., Fu, C.W., Jia, J., 2019. Pointweb: Enhancing local neighborhood features for point cloud processing. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 2019-June, 5560–5568. https://doi.org/10.1109/CVPR.2019.00571