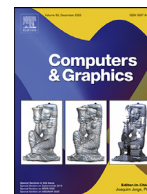




Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on 3DOR 2021

SHREC 2021: 3D point cloud change detection for street scenes

Tao Ku^{a,*}, Sam Galanakis^a, Bas Boom^b, Remco C. Veltkamp^c, Darshan Bangera^d, Shankar Gangisetty^d, Nikolaos Stagakis^e, Gerasimos Arvanitis^e, Konstantinos Moustakas^e^a Department of Information and Computing Sciences, Utrecht University, the Netherlands^b Imec One Planet Research Center, Wageningen, The Netherlands^c Department of Information and Computing Sciences, Utrecht University, The Netherlands^d KLE Technological University, Hubballi 580031, India^e Department of Electrical and Computer Engineering, University of Patras, Rio, Greece

ARTICLE INFO

Article history:

Received 1 April 2021

Revised 30 June 2021

Accepted 2 July 2021

Available online 10 July 2021

Keywords:

SHREC 2021

3D Point cloud change detection

Graph convolutional networks

Siamese networks

ABSTRACT

The rapid development of 3D acquisition devices enables us to collect billions of points in a few hours. However, the analysis of the output data is a challenging task, especially in the field of 3D point cloud change detection. In this Shape Retrieval Challenge (SHREC) track, we provide a street-scene dataset for 3D point cloud change detection. The dataset consists of 866 3D object pairs in year 2016 and 2020 from 78 large-scale street scene 3D point clouds. Our goal is to detect the changes from multi-temporal point clouds in a complex street environment.

We compare three methods on this benchmark, with one handcrafted (PoChaDeHH) and the other two learning-based (HGI-CD and SiamGCN). The results show that the handcrafted algorithm has balanced performance over all classes, while learning-based methods achieve overwhelming performance but suffer from the class-imbalanced problem and may fail on minority classes. The randomized over-sampling metric applied in SiamGCN can alleviate this problem. Also, different siamese network architecture in HGI-CD and SiamGCN contribute to the designing of a network for the 3D change detection task.

© 2021 The Authors. Published by Elsevier Ltd.

This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>)

1. Introduction

Change detection (CD) has been one of the important topics in remote sensing, and has been applied in many practical areas such as forest monitoring, urban sprawl and earthquake assessment [1–3]. 3D change detection, as a subset of the general CD problem, is drawing more and more attention in the area of smart cities [4,5] with the advantages of free from illumination variations and perspective distortions due to the rich 3D geometric information. However, the expensive data acquisition equipment and limited data sources are the main barriers for the 3D CD applications [1,6]. Thanks to the rapid development of 3D acquisition devices, we are able to collect billions of points in a few hours. However, the analysis of the output data is still a challenging problem as there aren't

generic, automated and accurate methodologies appropriate for all 3D change detection applications [7,8].

Change detection approaches can be generalized as direct comparison and classification-based comparison in [9]. Direct comparison is to detect changes directly from raw multi-temporal data. However, classification-based comparison is to classify objects, e.g. building, vegetation, and then detect their changes. Compared with direct comparison approaches, classification-based methods are easier to implement and reduce the difficulty of labelling changes in large-scale 3D point clouds.

Recent studies have focused on 3D data of the ground surface obtained from terrestrial laser scanners (TLS) and aerial laser scanners (ALS). Land cover and building related change detection [10–14] are the mainstream. High resolution 3D point clouds together with color images are combined as hybrid data sets to investigate both spatial and temporal changes. For unsupervised methods, most recent researches focus on calculating the historical differences directly. For example, digital elevation model (DEMs) difference based approaches [15,16] are proposed to identify locations and quantify spatial patterns of geomorphic changes. However, these approaches are usually based on the assumption that

* Corresponding author.

E-mail addresses: t.ku@uu.nl (T. Ku), samme.galanakis6@gmail.com (S. Galanakis), bas.boom@imec.nl (B. Boom), R.C.Veltkamp@uu.nl (R.C. Veltkamp), darshanbangera101@gmail.com (D. Bangera), shankar@kletech.ac.in (S. Gangisetty), nick.stag@ece.upatras.gr (N. Stagakis), arvanitis@ece.upatras.gr (G. Arvanitis), moustakas@ece.upatras.gr (K. Moustakas).

Table 1
Number of annotated objects in our Change3D benchmark.

Labels	nochange	removed	added	change	color_change	Total
Train set	421 (59.13%)	125 (17.56%)	68 (9.55%)	72 (10.11%)	26 (3.65%)	712
Test set	90 (58.44%)	25 (16.23%)	15 (9.74%)	17 (11.04%)	7 (4.55%)	154

the radiometric property of scanned objects are similar in different times which is actually not satisfied in real scenes. The direct and unsupervised difference calculation would introduce errors in change detection, especially for high resolution data. For supervised methods [17–19], there is usually a post-classification process to find valuable objects via an image analysis framework and then detect the changes using multi-modal and multi-temporal data. However, these approaches rely mainly on the imagery information. Furthermore, it is challenging to design a supervised model taking multi-modal and multi-temporal data as input.

In this SHREC track on 3D point cloud change detection for street scenes, we provide a cleaned and annotated 3D point cloud dataset obtained from mobile laser scanners (MLS). Objects in the dataset are initially roughly selected. Then, they are manually annotated with change labels. With the proposed dataset, we aim to compare and develop reliable and accurate change detection techniques for multi-temporal 3D street scenes.

We compare different methods on the proposed Street3D benchmark. The contributions are summarized as:

- We provide a unique classification-based 3D change detection dataset from a complex street environment. There are no other open 3D point cloud datasets released for our purpose.
- We evaluate different algorithms on the dataset and help finding solutions for 3D point cloud change detection tasks.
- The results show that the proposed siamese graph convolutional networks (SiamGCN) are good at extracting representative geometric features and can hereby outperform compared algorithms on the released Change3D dataset.

2. Change3D benchmark

In this comparative evaluation, we provide a change detection dataset named Change3D. The dataset is made publicly available at <https://kutao207.github.io>.

2.1. Dataset description

The dataset is provided by CycloMedia Technology and consists of annotated points of interest in street level colored point clouds gathered in 2016 and 2020 in the city of Schiedam, the Netherlands using vehicle mounted LiDAR sensors. The dataset focuses on street furniture, with the majority of labels corresponding to road-signs although other objects such as advertisements, statues and garbage bins are also included. Labeling was done through manual inspection. The 3D data from CycloMedia are generated from depth maps instead of original LiDAR scans, and they are already registered quite well [20].

We have selected over 78 annotated street-scene 3D point cloud pairs in the year of 2016 and 2020. Each point cloud pair represents a street scene in two different years and contains 866 object pairs of different change type in total. The statistics of the Change3D benchmark are summarized in Tab. 1. Each object pair is assigned one of the following labels:

(1) **nochange** (2) **removed** (3) **added** (4) **change** (5) **color_change**

- **nochange** refers to the case where there is no significant change between the two scans.
- **removed** refers to objects that exists in the first scan but are removed from the second scan.

- **added** refers to objects that do not exist in the first scan but are added during the second scan.
- **change** refers to the case where there is at least significant geometric change but also includes cases where there is also significant change in the RGB space. This includes being replaced by other objects. For example in Fig. 1 (c) a small blue sign is added whilst the rest of the sign stays the same.
- **color_change** refers to the case where there is not significant geometric case but significant change in the RGB space. For example, in Fig. 1 (b), content of an advertisement changed but the rest of the cloud is the same.

2.2. Labeling format

Each data point consists of the coordinates of a point of interest and the corresponding label. The points have been placed on or at the base of the object. A first step for preparing the points for input to a model may be taking all points within a certain x-y radius of the point of interest (resulting cylinders as seen in Fig. 1) from both point clouds. In most cases, apart from the ground this will give a fairly clean representation of the object. There are though cases where this will include other objects (for example signs that are close together) or parts of trees that are above the object.

Corresponding point clouds are saved with file names starting with the same integer (the scene number). The classifications are saved in csv files which also start with the same scene number. The coordinates contained in the csv file correspond to the points of interest.

2.3. Task and evaluation

Our task is to classify the changes of meaningful objects from two different years' 3D point clouds in a complex street environment. We provide scene-level 3D point clouds of year 2016 and 2020 and the corresponding center of meaningful objects. Participants are encouraged to try out different methods for our task.

We adopt the Overall Accuracy (OA) and mean Intersection over Union (mIoU) metrics in our 3D change detection task.

Generally, OA reports the percent of points in the data set which are correctly classified:

$$OA = \frac{N_{\text{correct}}}{N_{\text{total}}} \quad (1)$$

where N_{correct} is the number of correctly classified points and N_{total} is the total number of points. mIoU is the average of per-class IoU. The IoU of class i is defined as:

$$IoU_i = \frac{TP_i}{GT_i + Pred_i - TP_i} \quad (2)$$

where TP_i , GT_i , $Pred_i$ denote the correctly classified number of points, the ground truth point number, and the predicted point number for class i , respectively. The classes are *nochange*, *removed*, *added*, *change* and *color_change*.

3. Methods

3.1. PoChaDeHH: Point cloud change detection with hierarchical histograms

This method is contributed by authors [Nikolaos Stagakis, Gerasimos Arvanitis and Konstantinos Moustakas]. In the follow-

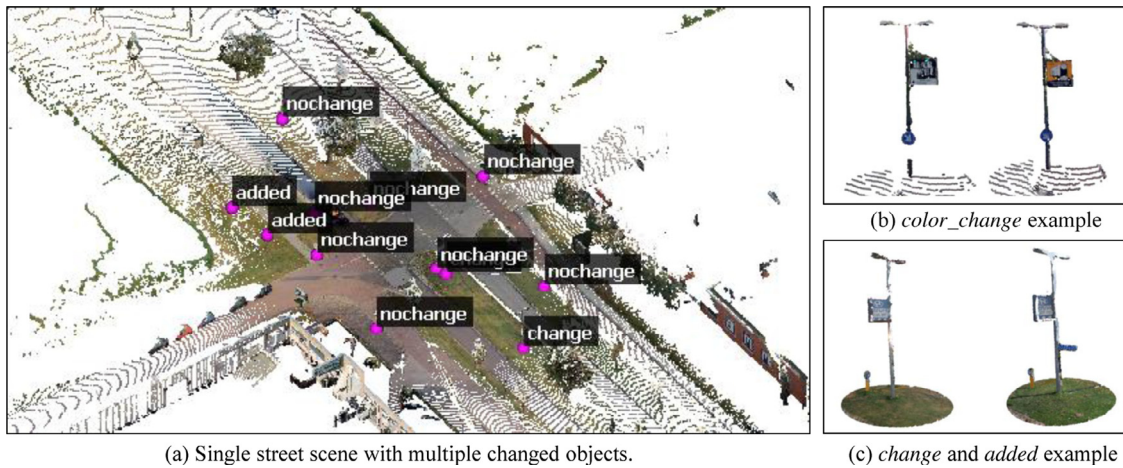


Fig. 1. An example of our Change3D dataset.

ing paragraphs, we will briefly describe all the steps that we follow for the change detection approach, named Point Cloud Change Detection with Hierarchical Histograms (PoChaDeHH) that we developed. The source code of our work is freely available at https://github.com/Stagakis/shrec21_changedetection.

3.1.1. Notation and basic definitions

- as “point-of-interest” we refer to the given point that specifies the area (subscene) in which the object-of-interest exists. The points are found in the csv file of the respective scene.
- as “subscene” we refer to the segment of the original point cloud scene in which the object-of-interest lies. The subscene is extracted by centering a cylinder on the point of interest. In particular, we use a modified version of the python script provided for visualizing the subscenes to save the subscene in a separate “.las” file.
- as “object-of-interest” we refer to the specific object of the subscene that we need to check about possible changes and it is defined based on the given point-of-interest. The rest of the objects extracted by the cylinder (if any) are considered and handled as outliers and noise.

3.1.2. Pre-processing of the scene

The subscenes of the given dataset usually consist of outliers, incomplete and/or noisy objects while other unrelated objects are spatially close to our object-of-interest. To simplify the subscene, we remove these objects, by applying a series of processing steps. These steps described below are repeated for each subscene of all scenes for both of the chronological areas.

3.1.3. Plane area removal

Firstly, we want to remove the floor of the extracted subscene. To do that, we detect the plane with the most inliers of each subscene and discard them. This step helps us later to separate the different objects that may appear in the subscene. In Fig. 2, we present an example of two subscenes and in Fig. 3 the results after the floor removal.

Note here that after this step it is possible that all the information of the subscene to have been removed (i.e., it there was only a flat area). In that case, we either save the original subscene, for later deciding whether the subscene is colorchanged or not, or we replace the subscene with a dummy point at $[0\ 0\ 0]$. The latter is done in the case of geometric classification.

3.1.4. Clustering

We estimate density clusters of each subscene in order to reject these objects that are far away from the object-of-interest and they



Fig. 2. Example of original subscenes for the dataset of 2016 and 2020, correspondingly.



Fig. 3. Original subscenes without the large plane area.

are not semantically related with it. In this step, we use an adaptive approach of the DBSCAN [21] algorithm that separates the objects of the subscene into clusters based on their density. To mention here that we select and merge these cluster(s) which are close



Fig. 4. Results after the process of clustering.

to the point-of-interest (i.e., regarding a predefined threshold). In Fig. 4, we present an example where more than one objects are presented in the subscene. However, only one of them is related with the real object-of-interest that we want to compare, so we have to remove all the other unrelated objects.

3.1.5. Registration

Once, we have found the corresponding clusters, which are related to the given point-of-interest, we apply a registration process to the cluster(s) of the newest date with the corresponding cluster(s) of the oldest date.

3.1.6. Comparison between registered point clouds

To decide the class of each situation, we firstly estimate the mean euclidean distance of each point of the one cluster with the 15 closest point of the other cluster, creating in this way a list of pairs and their corresponding distance. An example that visualizes the heatmap of this distance is presented in Fig. 5. Then, we create two histograms, using these distances, with an equal number of bins (i.e., 50) in order to be easily compared since the values are not normalized, as presented in Fig. 6. These histograms represent each one for the corresponding two clusters of each subscene. According to these histograms, we take into account only the high values of distances assuming that they lie in the histograms between 40–50 bins.

Additionally, to identify small differences between the two compared object-of-interest (i.e., change case) we search for vertices that do not appear in the list of pairs. The number of these unrelated vertices can be used to show a possible change between the two objects.

3.1.7. Color comparison between clustered point clouds

After the initial comparison for determining the geometric class of the object-of-interest, we take another step to find color changes between the point clouds and refine our classification. Given the nature of the colorchanged class we can assume that, regarding the geometry, color changed point clouds are a subset of the nochange class. Thus, we implement a histogram comparison between point clouds that receives as input only the nochange classified objects in the previous step and classifies them as either nochange or color-change. The color comparison is done in the HSV space to reduce

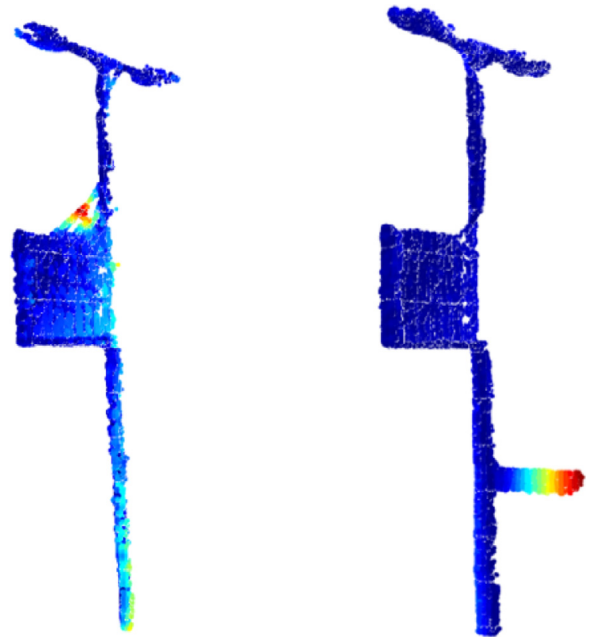


Fig. 5. Heatmap of the mean distance between each point of the one object-of-interest and the closest points of the other.

the effect of luminosity changes and the histograms are aligned. Looking at the statistical distribution of the histogram distance for each class, we chose a threshold to decide whether the input object is truly in the nochange class or should be classified as color-changed.

The entire pipeline of our work is shown in Fig. 7.

3.2. HGI-CD: 3D point cloud change detection for street scenes

This method is contributed by authors [Darshan Bangera and Shankar Gangisetty]. In this work, we propose a hybrid learning-based 3D change detection of bi-temporal point clouds as shown in Fig. 8. Initially, we calculate the change between 2016 and 2020 point clouds by applying a point-to-point Hausdorff distance [22]. Hausdorff distance is the greatest of all the distances from a point in one set to the closest point in the other set. We then designed a Siamese classification network [23] to detect the changes between point cloud street scenes.

3.2.1. Data pre-processing

Given the 3D point cloud scenes of 2016 and 2020 as inputs to the proposed hybrid learning-based framework as shown in Fig. 8. Initially the 3D objects present in 2016 and 2020 scenes are extracted by considering the points that lie within a cylindrical region around the point of interest. Let $(n \times 3)$ and $(m \times 3)$ represent the extracted 2016 and 2020 3D objects, where n and m are the number of points in 2016 and 2020 with (x, y, z) as point coordinates.

We then isolate the 3D objects of 2016 $(n' \times 3)$ and 2020 $(m' \times 3)$ from the ground surface by eliminating the large planes that are parallel to the ground surface and pass through the point of interest as shown in Fig. 8. Any of the outliers present are removed using statistical outlier removal technique. If there are no points within the extracted region, we add the point of interest as the sole point in the 3D object.

3.2.2. Change computing

To calculate the geometrical changes between the 2016 and 2020 point clouds, we perform a Hausdorff distance computing between each point in the 2016 object to its nearest neighbour in the

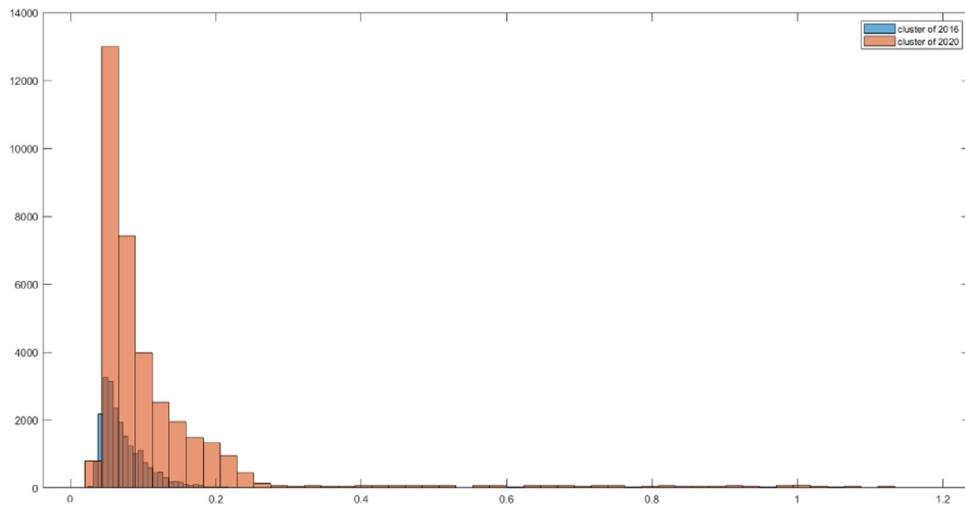


Fig. 6. Histograms of the mean distance between each point of a cluster and the closest points of the other.

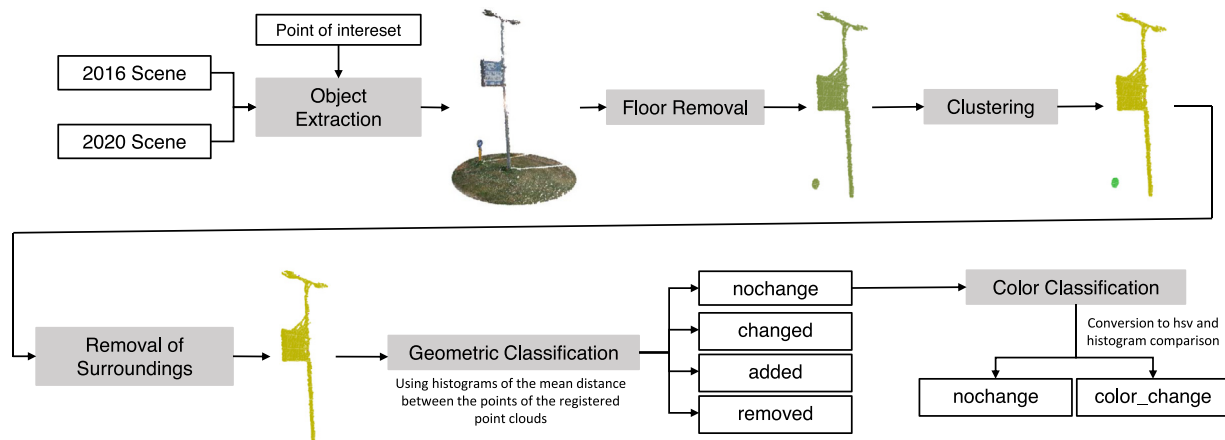


Fig. 7. Histograms of the mean distance between each point of a cluster and the closest points of the other.

2020 point cloud and vice-versa. We then perform thresholding to select the points with significant change in position.

After the change computation, we build color change graphs and geometric change graphs. In the color change graphs, calculation of the changes in colour space is performed by first averaging the RGB values into respective red, green and blue bins. A colour space point cloud is generated by taking the averaged RGB values. The change in color space is calculated by applying a point-to-point Hausdorff distance. For each RGB point with significant change, its corresponding (x, y, z) coordinate from the point cloud is taken. The k-nearest neighbours (KNN) algorithm is used to generate a graph for color and geometrical change point clouds of 2016 and 2020 3D objects, respectively. The fast point feature histograms (FPFH) [24] are calculated for each point as their node features. The results from the three streams (i.e., color change, 2016 points geometry, 2020 points geometry) are then concatenated and passed to multi layer perceptrons to generate the output. In the LiDAR scene scenarios where there are no points of significant change in both 2016 and 2020 point clouds, the object is directly labelled as *nochange*.

3.2.3. Training

We train the hybrid learning-based model using the graph node features and the edge indices as shown in Fig. 9. The loaded data is of dimensions (N, G) for geometrical change detection and (N, C)

for colour change detection, where N is the number of nodes of the 3D object, G is the total features selected for FPFH along the RGB values of the node and the RGB values of its nearest neighbour. The color change graphs and geometric change graphs data is fed to a Siamese graph convolutional networks (GCNs) [25] followed by inception networks [26]. Each of the GCN accepts the node features and edge indices as its input as shown in Fig. 9. The GCN network performs convolutions on the node features but maintains the structure of the graph. The obtained intermediate results have 80 dimensions which is fed to an Inception V1 network as specified in [26]. Each parallel stream (i.e., the color and geometric change) learn to detect if a set of points represent a changed part or not. Global mean pooling is performed on the 180 dimensions output of each stream to ensure that classification is done on the graph as a whole. The outputs are concatenated and fed to a multi-layered perceptron to provide the final labels. The point cloud change detection class labels are *added*, *removed*, *nochange*, *change*, and *color_change*.

3.2.4. Implementation details

We train the model using the Adam optimizer with a learning rate of 0:001 and a decay rate of 0:001. Empirical we set a threshold of 0:2 to select the points of significant geometric change and generate a k-NN graph with $k = 10$. Since the data are imbalanced,

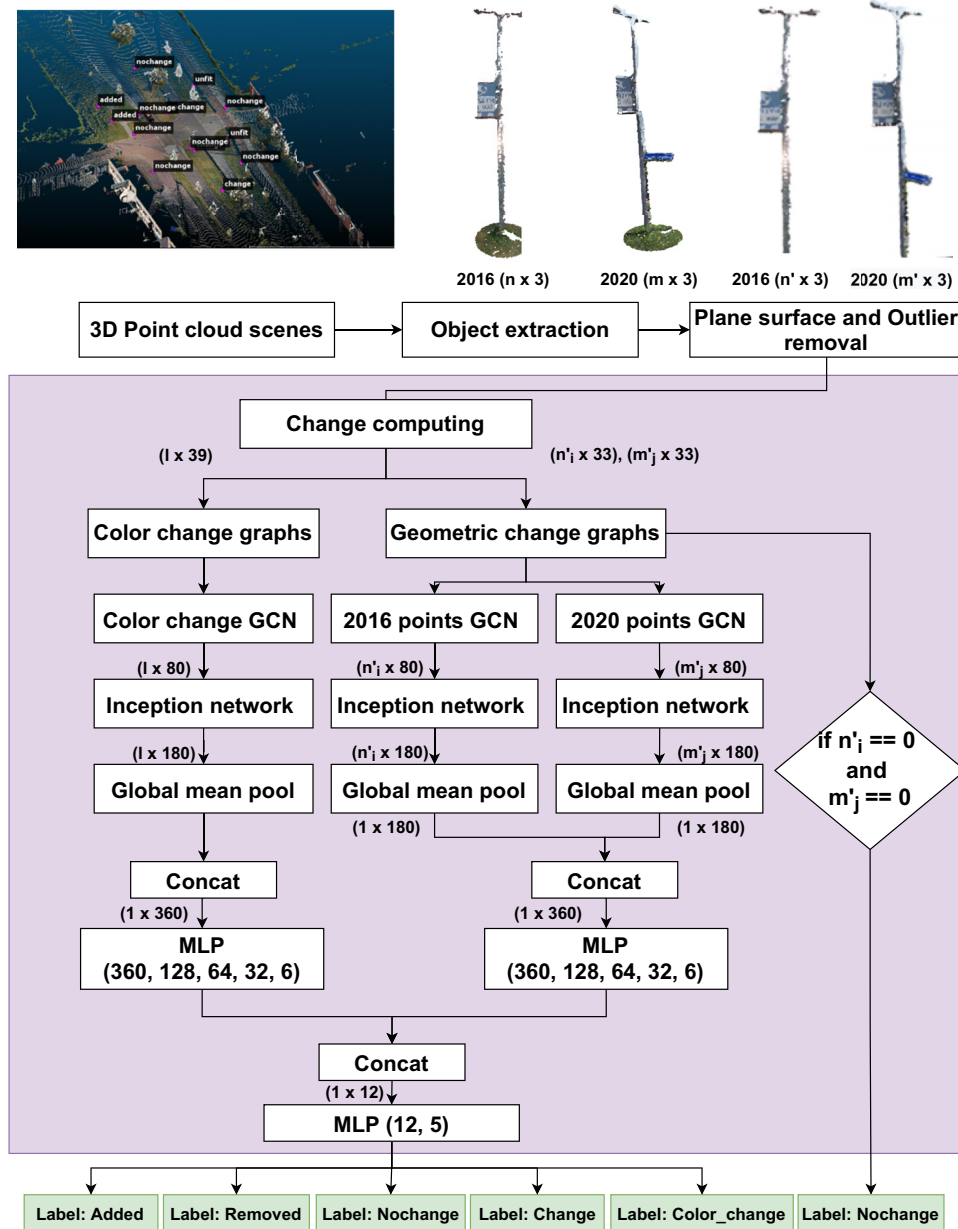


Fig. 8. Proposed hybrid learning-based point cloud change detection.

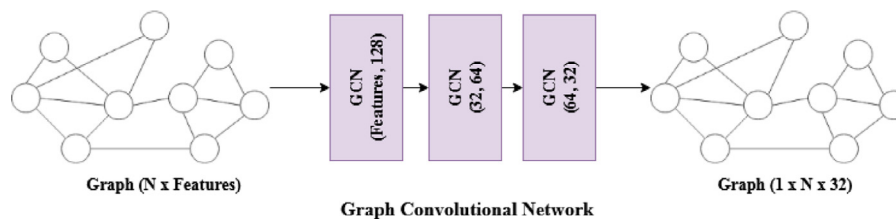


Fig. 9. Graph convolution network for point cloud change detection.

we perform data augmentation on each of the different classes by considering jitter and rotation to balance the dataset. We used a train : validation split of 80 : 20 for experimental analysis. The model is trained on hardware comprising of 16 GB CPU and a single NVIDIA Quadro P600 GPU.

After acceptance the clean code will be released. For evaluation the code at following GitHub link is submitted: <https://github.com/darshanbanger/3D-Change-Detection>.

3.3. SiamGCN: Siamese graph convolutional network for 3D point cloud change detection

This method is contributed by authors [Tao Ku, Sam Galanakis, Bas Boom and Remco C. Veltkamp]. In this section, we introduce a Siamese Graph Convolutional Network (SiamGCN) for 3D point cloud change detection. The edge convolution operator is adopted to extract representative features from point clouds. Then,

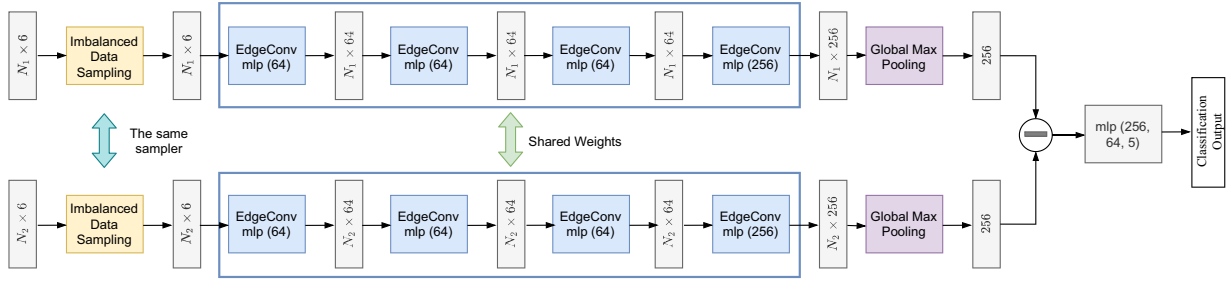


Fig. 10. The network architecture of the proposed SiamGCN.

a siamese architecture [27] based on the graph convolutional networks is proposed to identify the change type of any two input point sets from two different years. The code of our approach is publicly available at <https://github.com/kutao207/SiamGCN>.

3.3.1. Preprocessing

We design a siamese network for the task. However, the provided data are in the form of large point clouds and only object center coordinates are given. The size and shape of the objects are uncertain. To correctly identify the change type, we need to properly extract object pairs around given center coordinates and input these point cloud pairs to our proposed network for training and evaluation. Since the task is to classify change types instead of object classes, we don't need to extract the objects correctly. Considering that the point clouds are well registered, we extract the cylinders around the given centers with an experimental distance of 3 m for all samples. We don't expect the extracted point clouds have the same input point number as our network is well-designed to deal with this issue.

3.3.2. Network architecture

As shown in Fig. 10, the proposed SiamGCN consists of two branches of graph convolutional networks. These two branches share the same weights. The point clouds from two different times are fed into corresponding branch and output two one-dimension vectors through the global max pooling layer. By subtracting these two vectors, we use an MLP to get the final classification output.

Imbalanced Data Sampler: As shown in Tab. 1, the data are class-imbalanced. Around 60% samples are labelled as nochange and only around 3% are labelled as color_change, which will lead to a biased training process and result in failure cases in predicting minority classes. In order to address this problem, we adopt the randomized over-sampling metric in [28]. The general idea is to randomly duplicate samples in the minority class to make sure that samples of each class has the same probability being adopted in the training process.

Graph Construction: In order to apply graph convolution on point clouds, we need to construct graphs for input points. Considering the irregular data format of point clouds, we use graphs to encoding the geometric relations among points. For a point cloud with N points, suppose $X \in \mathcal{R}^{N \times 3}$ denotes the XYZ coordinates of N points. A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represents the local structure, where $\mathcal{V} = 1, 2, \dots, n$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ are vertices and edges. In our approach, we use the k -nearest neighbor (k NN) algorithm to build the graph of X and obtain corresponding adjacency matrix \mathbf{A} . We set the k NN query number as 16 throughout our experiments.

Graph Convolution: Inspired by [27], we adopt the edge-conditioned graph convolution operator in our proposed SiamGCN architecture. Edge convolution [27] is adopted to dynamically update the edge features. Generally, the edge convolution is to apply a channel-wise symmetric aggregation on the edge fea-

tures. Mathematically,

$$\mathbf{x}_i^{(k)} = \text{ReLU}(\theta_m \cdot (\mathbf{x}_j^{(k-1)} - \mathbf{x}_i^{(k-1)}) + \phi_m \cdot \mathbf{x}_i^{(k-1)}) \quad (3)$$

where θ_m and ϕ_m are the weights of filters and can be implemented as a shared MLP, \cdot denotes the Euclidean inner product. \mathbf{x}_i^k denotes the i -th edge feature output of the k -th layer output.

Then, we adopt the graph convolution concept in [29] to encode geometric information as

$$\mathbf{Y} = \mathbf{A}\mathbf{X}\mathbf{W} \quad (4)$$

where $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\} \in \mathbb{R}^{N \times C}$ is the input features, $\mathbf{Y} \in \mathbb{R}^{N \times D}$ is the filtered output, $\mathbf{W} \in \mathbb{R}^{C \times D}$ denotes the weight matrix, and \mathbf{A} is the adjacency matrix.

4. Results and discussion

4.1. Experimental results

Quantitative evaluation results on the Change3D benchmark are summarized in Tab. 2. Overall Accuracy (OA) and mean Intersection over Union (mIoU) are evaluated for all classes. We have also calculated the classification accuracy and IoU for each class respectively. In Fig. 11, confusion matrices are plotted to intuitively show the strength and weakness of each method.

We have evaluated the three submitted methods: PoChaDeHH, HGI-CD and SiamGCN. PoChaDeHH is based on hand-crafted detectors, while HGI-CD and SiamGCN are learning-based and both adopt graph convolution networks and Siamese network architecture.

From Tab. 2, it is straightforward that SiamGCN outperforms the other two methods on accuracy and IoU. PoChaDeHH, as a non-learning based method can achieve 61.04% overall accuracy and relatively balanced performance on classes. HGI-CD, as one of the learning-based methods, achieves good result on classifying majority class *nochange*, but fails on minority classes *change* and *color_change*. The SiamGCN has very good performance on the dataset, especially on minority classes which achieves 95.24% and 98.57% accuracy on minority classes *change* and *color_change*.

4.2. Discussion

Handcrafted v.s. Learning-based: Three algorithms including one handcrafted and two learning-based are evaluated on the 3D change detection dataset. Although the handcrafted algorithm can achieve relatively balanced results on the overall and per-class accuracy and mIoU, it's still obvious that learning-based methods can achieve overwhelming performance.

Class imbalance: Class imbalance poses a challenge for learning-based modeling as most machine learning algorithms are designed with the assumption of an equal number of examples for each class. However, due to the limitations of data acquisition, class-imbalanced problem cannot be avoided. Around 60% objects

Table 2
Quantitative results on the Change3D benchmark.

Method	OA (%)	mIoU (%)	nochange		removed		added		change		color_change	
			Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU
PoChaDeHH	61.04	30.22	76.70	61.06	48.00	31.58	66.70	40.00	5.90	4.17	28.60	14.29
HGI-CD	53.90	17.17	81.11	55.30	28.00	16.28	20.00	14.29	0.00	0.00	0.00	0.00
SiamGCN	92.42	76.63	84.25	55.03	89.80	72.07	94.70	89.82	95.24	69.57	98.57	96.67

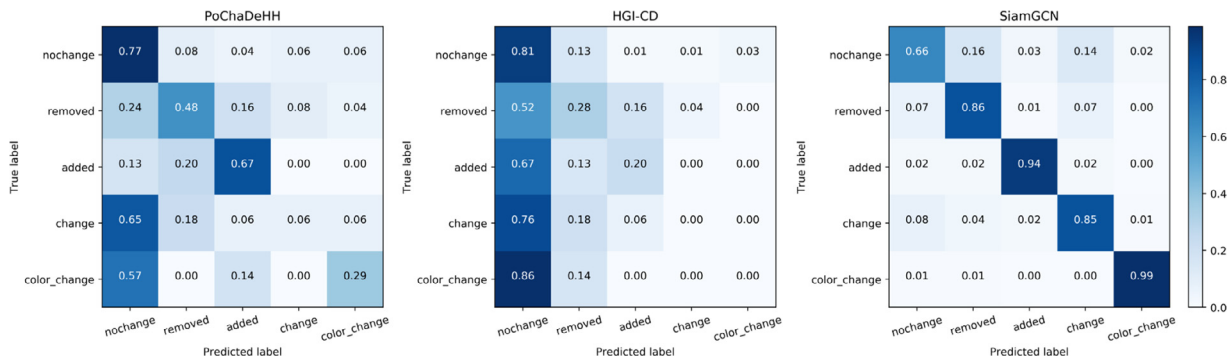


Fig. 11. Confusion matrix for all the three comparison algorithms. The x-axis represents the predicted labels while the y-axis denotes the groundtruth labels. In this figure, each subfigure shows clearly the classification ability of each algorithm. We normalized each row of the confusion matrix so that it is intuitive to show the probability of correct predictions and incorrect predictions.

Table 3
The comparison between concatenation operator and subtraction operator in SiamGCN.

	OA (%)	nochange	removed	added	change	color_change
Concat	76.54	39.60	76.06	88.37	82.64	97.97
Sub	92.42	84.25	89.80	94.70	95.24	98.57

are labelled as *nochange* and only 3.65% labelled as *color_change*. Hence, it is important to deal with the class imbalance for the participated algorithms. SiamGCN adopts randomized oversampling metric to avoid the class-imbalanced problem and achieves good results on minority classes. However, HGI-CD lacks this procedure and fails on the minority classes which indicates the importance of resampling part when designing an algorithm for the class-imbalanced change detection dataset.

Network architecture: When adopting deep learning based methods, the network design is extremely important for robust and effective modeling. For this 3D change detection task, both HGI-CD and SiamGCN adopts the graph convolutional networks and Siamese architecture. But there are still many differences. When dealing with the output features of the siamese network, HGI-CD uses a concatenation operator while SiamGCN adopts a subtraction operator, which makes a difference to the final performance. In order to investigate the importance of these two operators, we compare the performance of these two operators in the SiamGCN method. The results are summarized in [Tab. 3](#). The subtraction operator greatly increases the performance on both overall and per class accuracy. Although the concatenation operator has been widely used in classification and semantic segmentation, subtraction is more natural and reasonable for change detection as it is to classify the differences.

5. Conclusions

In conclusion, this comparative evaluation contributes to 3D point cloud change detection for street scenes with multiple approaches. We provide a street-scene 3D change detection dataset composed of 78 scans with 866 annotated object pairs in year 2016 and 2020. Five class labels are included for the change type.

We introduce three novel and different methodologies including one handcrafted method and two learning-based methods (HGI-CD and SiamGCN). It shows that learning-based can achieve overwhelming performance on the dataset. SiamGCN solves the class-imbalanced problem by adopting randomized oversampling and proposes a well-designed siamese graph convolutional network architecture for the 3D change detection task. Comparison results shows that over-sampling and using subtraction operator are the key for the SiamGCN to achieve the best performance on the released Change3D benchmark. The three compared algorithms contribute on how to design a proper framework for the 3D change detection task.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

No funding was received for this work.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

CRedit authorship contribution statement

Tao Ku: Methodology, Software, Writing – original draft. **Sam Galanakis:** Data curation, Writing – review & editing. **Bas Boom:** Data curation. **Remco C. Veltkamp:** Writing – review & editing. **Darshan Bangera:** Methodology, Software, Writing – original draft, Writing – review & editing. **Shankar Gangisetty:** Methodology, Software, Writing – original draft, Writing – review & editing. **Nikolaos Stagakis:** Methodology, Software, Writing – original draft, Writing – review & editing. **Gerasimos Arvanitis:** Methodology, Software, Writing – original draft, Writing – review & editing. **Konstantinos Moustakas:** Methodology, Software, Writing – original draft, Writing – review & editing.

References

- [1] Qin R, Tian J, Reinartz P. 3D change detection – approaches and applications. *ISPRS J Photogramm Remote Sens* 2016;122:41–56.
- [2] Tewkesbury AP, Comber AJ, Tate NJ, Lamb A, Fisher PF. A critical synthesis of remotely sensed optical image change detection techniques. *Remote Sens Environ* 2015;160:1–14.
- [3] Song D-X, Huang C, Sexton JO, Channan S, Feng M, Townshend JR. Use of landsat and corona data for mapping forest cover change from the mid-1960s to 2000s: case studies from the eastern united states and central brazil. *ISPRS J Photogramm Remote Sens* 2015;103:81–92. *Global Land Cover Mapping and Monitoring*
- [4] Daniel S, Doran M-A. Geosmartcity: Geomatics contribution to the smart city. In: *Proceedings of the 14th Annual International Conference on Digital Government Research*. New York, NY, USA: Association for Computing Machinery; 2013. p. 65–71. ISBN 9781450320573.
- [5] Zolanvari SMI, Ruano S, Rana A, Cummins A, da Silva RE, Rahbar M, et al. Dublincity: Annotated lidar point cloud and its applications. In: *30th British Machine Vision Conference 2019, BMVC 2019, Cardiff, UK, September 9–12, 2019*. BMVA Press; 2019. p. 44.
- [6] Xiao W, Vallet B, Paparoditis N. Change detection in 3d point clouds acquired by a mobile mapping system. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* 2013;1(2):331–6.
- [7] Karantzalos K. Recent advances on 2d and 3d change detection in urban environments from remote sensing data. *Computational Approaches for Urban Environments* 2015:237–72.
- [8] Mazzanti P, Caporossi P, Brunetti A, Mohammadi FI, Bozzano F. Short-term geomorphological evolution of the poggio baldi landslide upper scarp via 3d change detection. *Landslides* 2021:1–15.
- [9] Singh A. Review article digital change detection techniques using remotely-sensed data. *Int J Remote Sens* 1989;10(6):989–1003.
- [10] Boehm HV, Liesenberg V, Limin SH. Multi-temporal airborne lidar-survey and field measurements of tropical peat swamp forest to monitor changes. *IEEE J Sel Top Appl Earth Obs Remote Sens* 2013;6(3):1524–30.
- [11] Ventura G, Vilardo G, Terranova C, Sessa EB. Tracking and evolution of complex active landslides by multi-temporal airborne lidar data: the montaguto landslide (southern italy). *Remote Sens Environ* 2011;115(12):3237–48.
- [12] Jaboyedoff M, Oppikofer T, Abellán A, Derron M-H, Loye A, Metzger R, et al. Use of lidar in landslide investigations: a review. *Natural hazards* 2012;61(1):5–28.
- [13] Abellán A, Vilaplana JM, Calvet J, García-Sellés D, Asensio E. Rockfall monitoring by terrestrial laser scanning—case study of the basaltic rock face at castellfolit de la roca (catalonia, spain). *Nat Hazards Earth Syst Sci* 2011;11(3):829–41.
- [14] Mineo S, Pappalardo G, Mangiameli M, Campolo S, Mussumeci G. Rockfall analysis for preliminary hazard assessment of the cliff of taormina saracen castle (sicily). *Sustainability* 2018;10(2):417.
- [15] James LA, Hodgson ME, Ghoshal S, Latiolais MM. Geomorphic change detection using historic maps and dem differencing: the temporal dimension of geospatial analysis. *Geomorphology* 2012;137(1):181–98.
- [16] Rumsby B, Brasington J, Langham J, McLelland S, Middleton R, Rollinson G. Monitoring and modelling particle and reach-scale morphological change in gravel-bed rivers: applications and challenges. *Geomorphology* 2008;93(1–2):40–54.
- [17] Tzotsos A, Karantzalos K, Argialas D, Weng Q. Multiscale segmentation and classification of remote sensing imagery with advanced edge and scale-space features. In: *Scale Issues in Remote Sensing*. Wiley Online Library; 2014. p. 170–96.
- [18] Berger C, Voltersen M, Eckardt R, Eberle J, Heyer T, Salepci N, et al. Multi-modal and multi-temporal data fusion: outcome of the 2012 grss data fusion contest. *IEEE J Sel Top Appl Earth Obs Remote Sens* 2013;6(3):1324–40.
- [19] Schneider A. Monitoring land cover change in urban and peri-urban areas using dense time stacks of landsat satellite data and a data mining approach. *Remote Sens Environ* 2012;124:689–704.
- [20] Someren Bv. Neural multi-view segmentation-aggregation for joint lidar and image object detection; 2017.
- [21] Ester M, Kriegel H-P, Sander J, Xu X. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. AAAI Press; 1996. p. 226–31.
- [22] Huttenlocher DP, Klanderman GA, Rucklidge W. Comparing images using the hausdorff distance. *IEEE Trans Pattern Anal Mach Intell* 1993;15(9):850–63.
- [23] Melekhov I, Kannala J, Rahtu E. Siamese network features for image matching. In: *23rd International Conference on Pattern Recognition, ICPR 2016, Cancún, Mexico, December 4–8, 2016*. IEEE; 2016. p. 378–83.
- [24] Rusu RB, Blodow N, Beetz M. Fast point feature histograms (FPFH) for 3d registration. In: *2009 IEEE International Conference on Robotics and Automation, ICRA 2009, Kobe, Japan, May 12–17, 2009*. IEEE; 2009. p. 3212–17.
- [25] Wu F, AHS Jr, Zhang T, Fifty C, Yu T, Weinberger KQ. Simplifying graph convolutional networks. In: Chaudhuri K, Salakhutdinov R, editors. *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9–15 June 2019, Long Beach, California, USA*. *Proceedings of Machine Learning Research*, 97. PMLR; 2019. p. 6861–71.
- [26] Szegedy C, Liu W, Jia Y, Sermanet P, Reed SE, Anguelov D, et al. Going deeper with convolutions. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7–12, 2015*. IEEE Computer Society; 2015. p. 1–9.
- [27] Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM. Dynamic graph CNN for learning on point clouds. *ACM Trans Graph* 2019;38(5):146:1–146:12.
- [28] Batista GEAPA, Prati RC, Monard MC. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor* 2004;6(1):20–9.
- [29] Chen S, Varma R, Sandryhaila A, Kovacevic J. Discrete signal processing on graphs: sampling theory. *IEEE Trans Signal Process* 2015;63(24):6510–23.