Automation in Construction 150 (2023) 104838

Contents lists available at ScienceDirect

Automation in Construction

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

Semantic segmentation of bridge point clouds with a synthetic data augmentation strategy and graph-structured deep metric learning

Xiaofei Yang [*], Enrique del Rey Castillo, Yang Zou, Liam Wotherspoon

Department of Civil and Environmental Engineering, University of Auckland, Auckland 1023, New Zealand

ARTICLE INFO

Keywords:
Deep learning
Bridge component recognition
Weighted superpoint graph
High quality superpoint generation
Boundary segmentation

ABSTRACT

Deep learning techniques are capable of providing versatile solutions to automate classification of bridge point clouds into corresponding constituent components, but training sample scarcity and erroneous boundary segmentation have hindered the extent of their application. To respond to these challenges, this study presents two synthetic data augmentation strategies for alleviating the data scarcity problem and improves the weighted superpoint graph (WSPG) model by using a graph-structured deep metric learning method for generating high quality superpoints to address the erroneous boundary segmentation problem. Evaluation experiments were conducted to validate their effectiveness. Experimental results showed that the synthetic data augmentation strategies can significantly alleviate the training sample scarcity problem. The synthetic superpoint augmentation strategy outperformed the synthetic point cloud augmentation strategy with an increase of around 6% for overall accuracy (OA), 3% for mean class accuracy (mAcc) and 18% for mean Intersection over Union (mIoU). Compared with the original WSPG model, the improved WSPG model with graph-structured deep metric learning method increased the mIoU of the overall dataset and the pier caps by 4% and 21% respectively, while also reducing the boundary segmentation errors.

1. Introduction

Bridges are of critical importance in transportation networks, but the ageing conditions of bridges coupled with increasing traffic load are resulting in excessive deterioration and corrosion. According to a report from the American Society of Civil Engineers (ASCE), the average age of America's bridges is 44 years and 42% of all bridges are over 50 years old. Along with this, 178 million trips are taken across these bridges every day resulting in 7.5% of all bridges are in poor condition [1].

A routine bridge inspection is most commonly performed to ensure the safety of in-service bridges. This inspection is mainly based on visual evaluation by human experts, a process that is recognized as being time-consuming, laborious and costly. Efforts have been made to automatically detect bridge surface damage such as cracks, spalling and exposed rebars in previous research. For example, Zhang et al. leveraged YOLOv3 [2] to automatically detect four types of concrete bridge surface defects [3]. A significant limitation of previous research is that the damage location relative to the specific bridge component is not provided, given that existing inspection guidelines rely on both damage information and corresponding component information to assign coefficient rating values and further assess the bridge condition. Accordingly, bridge condition assessment still needs to manually integrate detected defects within the bridge global structure context [4]. Bridge Information Modelling (BrIM) provides a complete representation of physical and functional characteristics of bridges, which could be a good solution to store, analyze and manage damage data on a component level and provide a global context of the bridge structure [5]. Damage information can be standardized and kept in a usable format, which can be further shared and extracted for use in high level tasks such as automated bridge condition assessment. For example, SeeBridge (Semantic Enrichment Engine for Bridges) aims to develop a comprehensive solution for rapid and smart survey and assessment of bridges [6]. Firstly, advanced remote sensing technologies were leveraged to capture the state of a bridge, followed by a bridge object detection software to reconstruct the geometry of as-is BrIMs. Next, an expert system encoding bridge engineers' knowledge was developed to semantically enrich the BrIMs including material types, internal bridge components and object relationships. Subsequently, a damage measurement tool was devised to associate the damage information with the reconstructed BrIMs at the bridge component level. The output is a BrIM with sufficiently semantic

[*] Corresponding author.
E-mail address: fyan983@aucklanduni.ac.nz (X. Yang).

https://doi.org/10.1016/j.autcon.2023.104838
Received 17 October 2022; Received in revised form 14 February 2023; Accepted 11 March 2023
Available online 16 March 2023

information, which can be used for condition assessment. Notably, most existing bridges do not have BrIMs due to the high cost of the traditional and manual digital twining process [7].

Generation of BrIMs for existing bridges normally consists of four steps: 1) reality reconstruction, 2) semantic segmentation, 3) geometrical modelling, and 4) bridge information modelling [8,9]. In the reality reconstruction process, reality capture techniques such as terrestrial laser scanning, Light Detection and Ranging (LiDAR) and photogrammetry are applied to acquire point clouds for the digital representations of existing bridges [10]. Semantic segmentation is a process that classifies full-scale raw point clouds into corresponding bridge components and then assigns a semantic label to each classified point cluster [11]. Within the geometrical modelling step, the shape of each component and the spatial relationships between these components are represented with geometric parameters [7]. In the final step, the bridge information modelling process leverages the semantic and geometrical parameters of each component to generate bridge information models [6]. Extensive research has focused on the above digital twining processes [12–14], but full automation of the bridge semantic segmentation task remains a global challenge [14].

Previous studies related to the bridge semantic segmentation can be broken down into three categories i.e. bottom-up segmentation approach, top-down partitioning methods and deep learning techniques [11]. Recently, deep learning techniques have been the mainstream as they can provide versatile solutions to various bridge types instead of only adapting to specific types of bridges [15]. Previous studies have investigated the cutting-edge representatives using deep learning techniques such as PointNet [15], PointCNN [16], Dynamic Graph Convolutional Neural Network (DGCNN) [17] and Weighted Superpoint Graph (WSPG) [18]. However, two common challenges remain when using deep learning techniques in this field:

1) The shortage of sufficient real-world training samples significantly hinders their wide application. Realistic bridge point clouds are with the high collection cost, and surveyors suffer from potential traffic risks [11]. The training sample scarcity is further amplified by diverse types of bridges as geometries of distinct bridge types are normally different [14].

2) While erroneous boundary segmentation often occurred, no prior work has paid attention to the segmentation accuracy at the boundary between bridge components. It is worth noting that an accurate boundary segmentation method is of critical importance for the bridge semantic segmentation task. Firstly, accurate boundary segmentation can significantly affect the reconstruction accuracy of digital twining for existing bridges [14] as small segmentation errors on the boundaries can have drastic consequences in the later digital twining process. Secondly, accurate boundary segmentation could also enhance the segmentation performance for minority component categories with much fewer points such as piers and parapets, which would also be beneficial for overall semantic segmentation performance [15].

Synthetic bridge point clouds could be a good substitute for real-world bridge point clouds [33] by providing a nearly unlimited number of diverse training samples with much lower creation costs. The WSPG method is optimal for bridge semantic segmentation tasks within aforementioned deep learning models. Superpoints are sets of points with homogeneous semantics and simple shapes, which can greatly reduce the point cloud redundancy leading to the less computational cost of subsequent semantic segmentation tasks [19]. A graph-structured deep metric learning approach has a higher ability to distinguish similarity and dissimilarity of each point, thus resulting in generated superpoints with higher quality, ensuring each superpoint does not cover different components and carries homogeneous semantic information.

This study aims to meet these challenges by: 1) investigating the feasibility of leveraging synthetic data augmentation strategies to increase the number of training samples, thus addressing the training data scarcity problem; 2) improving the WSPG model by leveraging a graph-structured deep metric learning method to boost the generation of high quality superpoints, allevaiting erroneous boundary segmentation. The contributions of this study involve two aspects: 1) present two synthetic data augmentation strategies for the superpoint-based method and evaluate their performance using synthetic point clouds generated from BrIMs; 2) improve the WSPG model by leveraging a graph-structured deep metric learning method to present a high constrast between component boundaries and thus boost the generation of high quality superpoints, improving the boundary segmentation accuracy.

The start-of-the-art semantic segmentation methods of bridge point clouds and point cloud augmentation methods are introduced in Section 2. Synthetic data augmentation strategies andgraph-structured deep metric learning method and the introduction of the WSPG model are presented in Section 3. Section 4 expounds data preparation, experiment implementation details and evaluation results. The article and potential future research are respectively concluded and discussed in Section 5.

2. Literature review

2.1. Automated bridge point cloud semantic segmentation

Autonomous bridge semantic segmentation task is an essential intermediate step in the digital twining process. Three automated methods exist in this field i.e. bottom-up segmentation algorithms, top-down partitioning methods and deep learning techniques.

2.1.1. Bottom-up segmentation algorithms

Bottom-up approaches commonly implement segmentation and classification with a sequence manner [20], where point clouds' geometric features are extracted from primitive features like points to higher level features such as surface normals, meshes, patches, and nonuniform B-spline surfaces until the segmentation process is finished [11]. Human expertise is then leveraged to classify the sub-point clusters. A large number of studies have focused on generating surface-based primitives, especially planar surfaces that are the most critical geometries of bridge components [14]. These algorithms are represented by Random Sample Consensus (RANSAC) and the region growing (RG)-related algorithms. An early example of the RANSAC method was presented by Schnabel et al. [21], where they grouped point clouds into point clusters with inherent shapes such as planes, spheres, cylinders, cones and tori and a set of remaining points. The RANSAC algorithm was computationally expensive and thus tends to perform well in simple bridges or synthetic point clouds. It was a challenge to use RANSAC-based algorithms for real-world bridges with complicated geometry of bridge components.

A typical example of the region growing algorithm was proposed by Walsh et al. [22]. Their work started by finding a corner or an edge, where the points' normal vectors varied from the neighboring points. Subsequently, smoothly connected point cloud was clustered using a region growing method with smoothness-constraint to avoid over-segmentation. During this process, a specific seed point in planar regions was firstly selected, and then its neighboring points with similar features were clustered into the same segment until reaching an edge. The same process was performed several times until all points were segmented into different regions. These resulting regions were finally classified into different surface types and thus detected into different bridge components.

While bottom-up segmentation algorithms achieved partial success, these methods are still recognized as insufficient. This is due to two aspects: 1) most bridge components present planar patterns that hinder the classification of local surfaces with similar features into different components; 2) bottom-up segmentation algorithms are highly sensitive to incomplete point clouds caused by occlusions, because local point features are inconsistent within incomplete point clouds. This issue limits their application to the real-world bridge scenario as bridge point clouds are normally incomplete due to ambient occlusions such as moving vehicles.

2.1.2. Top-down partitioning methods

Top-down methods aims to simultaneously perform segmentation and classification in a top-down manner based on domain knowledge such as bridge component geometric features and design regulations. The top-down method recognized bridge components heuristically [20]. The first step was to extract different bridge component assemblies from full-scale bridge point clouds to reduce the geometric complexity for subsequent processing. The resulting bridge component assemblies were further classified into individual bridge components according to engineering criteria using bottom-up segmentation algorithms or optimization-based methods [23]. A pioneering example of this work was presented by Lu et al. [11]. Their work started with the alignment of a full-scale bridge point cloud model, followed by vertically segmenting deck assembly and substructures from the whole bridge point clouds leveraging a slicing algorithm. Each bridge component was finally recognized based on its geometric features. While this method showed high segmentation accuracy and time-efficiency, it only recognized limited bridge component categories, and only adapted to simple RC slab bridges and beam-slab bridges with straight decks. To process curved bridges, Yan et al. [24] firstly estimated the traffic direction using a piece-wise linear skeleton model, followed by in combination with an incremental cross-section alignment method to overcome the challenge.

While top-down partitioning methods have often been recognized as superior to the bottom-up segmentation algorithms, these methods are only tailored for a specific bridge type and thus difficult to be generalized across multiple bridge typologies [25]. Additionally, top-down partitioning methods rely on transcendental knowledge related to bridge component geometries or design standards, limiting its application to real-world bridge point clouds as these can differ from the as-designed model.

2.1.3. Deep learning-based techniques

Deep learning techniques have attracted much attention in the realm of automated bridge semantic segmentation due to the increasing use of bridge point cloud models and improvement of computing power. The advantages of the deep learning-based methods mainly lie in two aspects: the end-to-end automated nature and the versatile solutions to various bridge types. Inspired by these benefits, Kim et al. [15] investigated the feasibility of PointNet [26] used for the bridge semantic segmentation task. The PointNet algorithm is not able to directly process full-scale bridge point clouds for the component recognition because this algorithm required a fixed and relatively small input point number (e.g. 2048 points), thus only applicable to small-scale point clouds. To adapt it to bridge point clouds, the authors adopted a concept of subspaces, where the entire bridge point clouds were vertically sliced into small segments that have a fixed length. Each segment is then downsampled to 2048 points and classified into distinct components using the PointNet algorithm. The experiment showed that the PointNet algorithm can adapt to straight, curved and titled bridges with background points. While this work achieved good performance, some drawbacks still remained: 1) bridge global features were generally lost due to the subspace partitioning step; 2) only decks, piers and background information were recognized from bridge point clouds.

PointCNN [16] and DGCNN [17] were also investigated and compared to conduct the bridge semantic segmentation task [27]. The experiment demonstrated DGCNN outperformed the PointNet and PointCNN algorithm. Additionally, to enhance the segmentation accuracy of electric poles, a hierarchical DGCNN (HGCNN) approach was proposed, where the authors leveraged the KNN concept [28] to extend the range of neighbors when obtaining local features and thus improved the performance by 3% of the metric of the Intersection over Union (IoU) [29].

There are common drawbacks for all the aforementioned methods: 1) reliance on the subspace partition procedure for processing the entire bridge point clouds; 2) only validated on a limited number of component categories because of the shortage of training data; 3) the classification accuracy for minor bridge component categories is relatively low. The WSPG method presented by Yang et al. [18] was able to overcome these drawbacks. The first step was to cluster full-scale bridge point clouds into hundreds of superpoints using the geometric partitioned method. These superpoints were semantically homogeneous and geometrically simple. The resulting superpoints were then fed into PointNet and Graph Neural Networks, where superpoints with the same semantics were clustered together and classified into distinct component categories. Their proposed WSPG method was able to directly perform component recognition from full-scale bridge point clouds and was validated on 8 bridge component categories based on a synthetic dataset generated from 22 BrIMs. This approach also improved the segmentation accuracy of minority bridge component categories compared to other existing methods by alleviating the data imbalance problem. In addition, a recent study focused on using a small number of training samples to conduct bridge semantic segmentation tasks due to the training data scarcity [14]. The researchers firstly leveraged manually designed local geometric descriptors to extract geometric features and then trained a deep classification neural network to conduct semantic segmentation. Finally, a refinement algorithm was applied to optimize above segmentation results. While this method required less training data, it heavily depends on the quality of descriptors and performance may significantly drop when the point density changes.

Deep learning-based methods have achieved great success and become the mainstream for bridge semantic segmentation tasks because they can generally adapt to all types of bridges with complex geometries. However, some common challenges still remain when using deep learning techniques: 1) the scarcity of realistic bridge point clouds hinders their wide application; 2) little research has focused on the segmentation accuracy for boundaries between adjacent bridge components, which significantly affects the reconstruction accuracy of digital twining for existing bridges. Thus, this paper employs synthetic data augmentation strategies to alleviate the training data scarcity problem and a graph-structured deep metric learning method to improve the segmentation accuracy for bridge component boundaries.

2.2. Point cloud data augmentation

The data scarcity of bridge point clouds compromized the application of deep learning techniques in the realm of bridge point cloud semantic segmentation. Data augmentation strategies are desirable for deep learning techniques as they increase the number and diversity of the training samples, avoiding overfitting issues and enhancing the network generalization ability. Conventional data augmentation methods such as rotation, flipping, scaling, jittering and stochastic noise enhanced the training sample diversity and demonstrated their effectiveness. Nevertheless, these conventional data augmentation methods did not increase the diversity of bridge component geometric structures. To address this challenge, previous research leveraged 3D CAD models to generate synthetic point clouds such as ShapeNet [30] and ModelNet [31] for augmenting the training dataset. Recently, several studies have focused on using BIMs to generate well-annotated synthetic point clouds without tedious manual annotation as BIMs are able to carry structural geometry information and rich semantic information relevant to building components. For example, Ma et al. [32] presented a user-friendly synthetic point cloud generation method from BIMs. They demonstrated that augmenting a small amount of real-world point clouds with synthetic point clouds was effective, achieving a 7.1% IoU improvement when training the DGCNN algorithm with the augmented dataset for semantic segmentation tasks. Similarly, IFCNet [33], which is a synthetic point cloud dataset for industrial entities, was established to offer sufficient point cloud data for achieving better classification performance. A limitation of these studies was that the generated synthetic

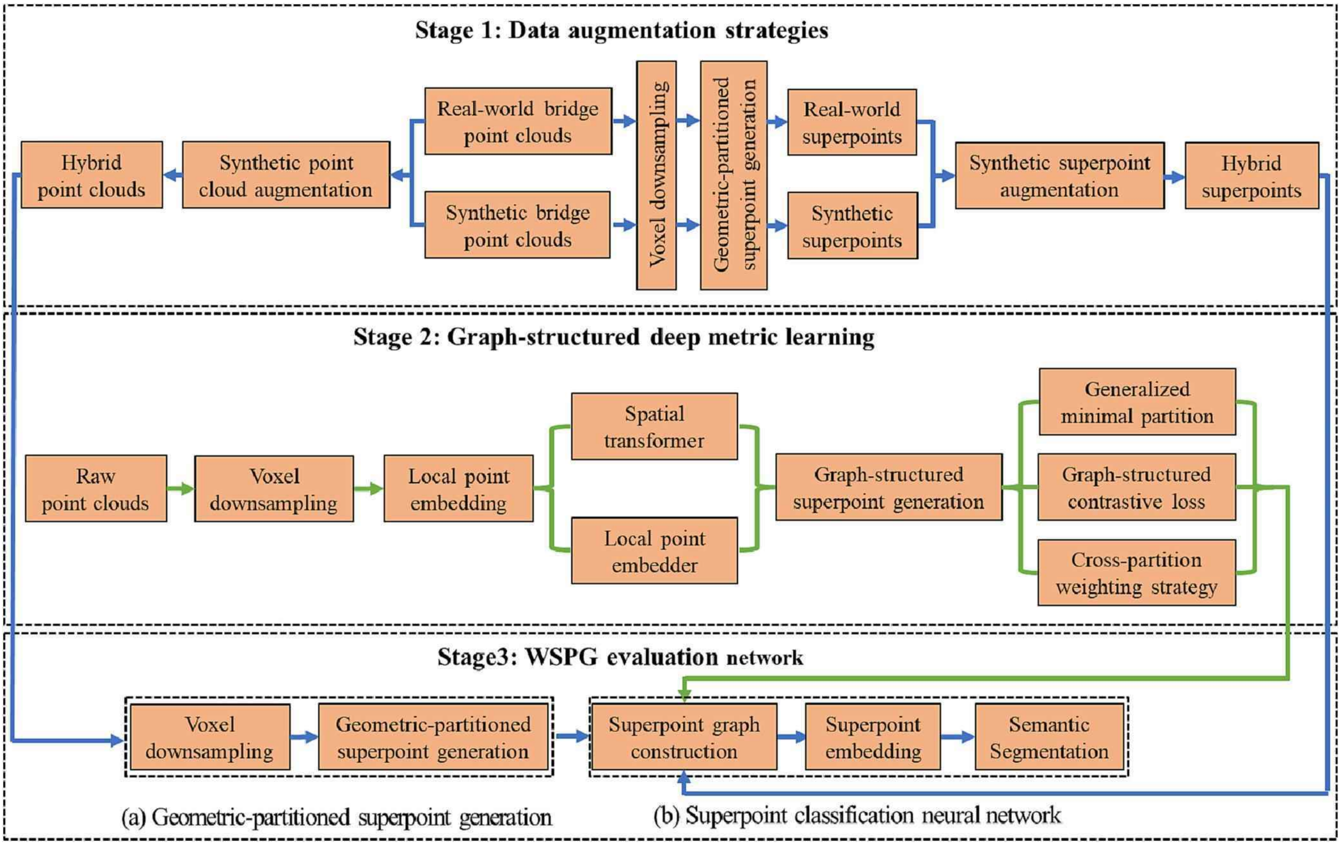

Stage 1: Data augmentation strategies
Hybrid point clouds
Synthetic point cloud augmentation
Real-world bridge point clouds
Synthetic bridge point clouds
Voxel downsampling
Geometric-partitioned superpoint generation
Real-world superpoints
Synthetic superpoints
Synthetic superpoint augmentation
Hybrid superpoints
Stage 2: Graph-structured deep metric learning
Raw point clouds
Voxel downsampling
Local point embedding
Spatial transformer
Local point embedder
Graph-structured superpoint generation
Generalized minimal partition
Graph-structured contrastive loss
Cross-partition weighting strategy
Stage3: WSPG evaluation network
Voxel downsampling
Geometric-partitioned superpoint generation
Superpoint graph construction
Superpoint embedding
Semantic Segmentation
(a) Geometric-partitioned superpoint generation
(b) Superpoint classification neural network

Fig. 1. Summary of the workflow for each stage in the methodology.

dataset did not include point cloud color information. A recent example presented by Zhai et al. [34] established a color-based synthetic building point cloud dataset and combined synthetic point clouds and real-world point clouds together to train a superpoint-based network. Their results demonstrated the effectiveness of using synthetic point clouds to augment limited and low-quality real-world point cloud dataset.

To summarize, current studies mainly focused on using synthetic data augmentation methods in the field of building point clouds and no existing work has investigated synthetic data augmentation methods in the bridge point cloud scene. This then posed the question whether synthetic data augmentation strategies would be effective in the realm of bridge point clouds due to the larger difference between the synthetic and real-world bridge point clouds compared to building point clouds. Firstly, real-world bridge point clouds are normally incomplete due to the ambient occlusion such as vegetation and moving vehicles. Secondly, real-world bridge point clouds have larger point cloud registration errors than building point clouds due to the lack of salient geometric features such as corners. Lastly, bridge point clouds are captured outdoors with a higher level of noise compared to indoor environments, compromizing the data quality. Another knowledge gap is that previous work including those proposed in Ma et al. [32] and Zhai et al. [34] only mixed the synthetic and real-world point clouds together, and then leveraged the mixed dataset to train the deep learning model. No prior study has investigated the influence of different synthetic data augmentation strategies for the superpoint-based method.

3. Methodology

3.1. Overview

The summary of the workflow for each stage in the methodology is illustrated in Fig. 1. The first stage elaborates on the implementation process of two synthetic data augmentation strategies to alleviate the data scarcity challenge. In the second stage, the graph-structured deep metric learning algorithm is introduced in detail, which can generate high quality superpoints. The WSPG model enhanced by the graph-structured deep metric learning algorithm is able to address boundary segmentation errors and further improve the segmentation accuracy of superpoint-based methods. In the third stage, the principle and computational workflow of the original WSPG model are elaborated. The original WSPG model is leveraged to evaluate the effectiveness of synthetic data augmentation strategies and the improved WSPG model enhanced by graph-structured deep metric learning method. These stages are described in more detail in the following sections.

3.2. Synthetic data augmentation strategies

This section investigates synthetic data augmentation strategies for superpoint-based algorithms. According to the computational workflow of superpoint-based algorithms, input point clouds are firstly processed to generate superpoints using a geometric partitioned method that is dependant on the geometric features of each bridge component type, which can be seen as a generalized minimal partition problem [35]. Different input point clouds are partitioned into distinct superpoints due to the distinct geometric features (e.g. linearity, planarity, scattering, and normalized elevation) contained in the point clouds. Next, these superpoints are fed into the superpoint classification neural network within the WSPG model to conduct the semantic segmentation task. Based on the computational workflow of the superpoint-based algorithms, two data augmentation methods are proposed, which are illustrated in detail as below. In addition, the performance of two data augmentation strategies are validated through comparison of superpoint classification accuracy using the WSPG model.

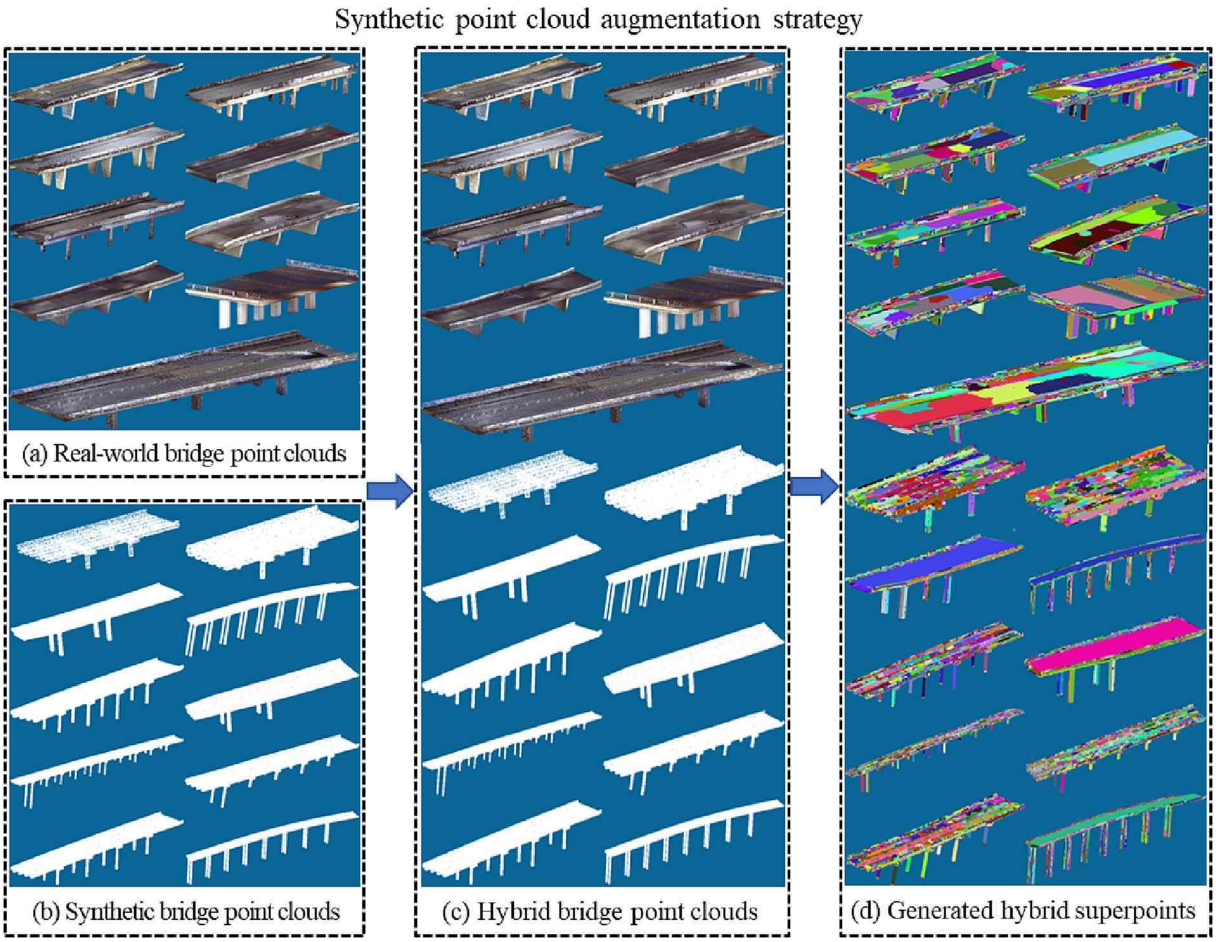

Synthetic point cloud augmentation strategy
(a) Real-world bridge point clouds
(b) Synthetic bridge point clouds
(c) Hybrid bridge point clouds
(d) Generated hybrid superpoints

Fig. 2. Visualization of each step of the synthetic point cloud augmentation strategy.

3.2.1. *Synthetic point cloud augmentation strategy*

The synthetic point cloud augmentation strategy is an approach to directly add synthetic bridge point clouds into a real-world bridge point cloud training dataset in order to increase the number of training samples, which is similar to the synthetic data augmentation method used for building structures [32,34]. The newly generated training dataset with the both real-world and synthetic bridge point clouds is called the hybrid training dataset. The hybrid training dataset is then leveraged to generate superpoints using the geometric partitioned method. The workflow of the synthetic point cloud augmentation strategy is shown in Stage 1 of Fig. 1. The visualization of each step of the synthetic point cloud augmentation strategy is presented in Fig. 2, where the generated hybrid superpoints are annotated in different colors. Directly leveraging the hybrid bridge point clouds to generate superpoints may lead to coarse superpoint generation due to the different geometric features between the synthetic and real-world point clouds, affecting superpoints carrying homogeneous semantic information.

3.2.2. *Synthetic superpoint augmentation strategy*

The synthetic superpoint augmentation strategy respectively generates superpoints from the synthetic and real-world bridge point clouds using the geometric partitioned method, followed by merging the synthetic superpoints with the real-world superpoints, as shown in Stage 1 of Fig. 1. The visualization of each step of the synthetic superpoint augmentation strategy is illustrated in Fig. 3. Separately generated synthetic superpoints and real-world superpoints result in higher superpoint quality as they can fully leverage their respective intrinsic geometric features and do not influence each other during the superpoint generation process.

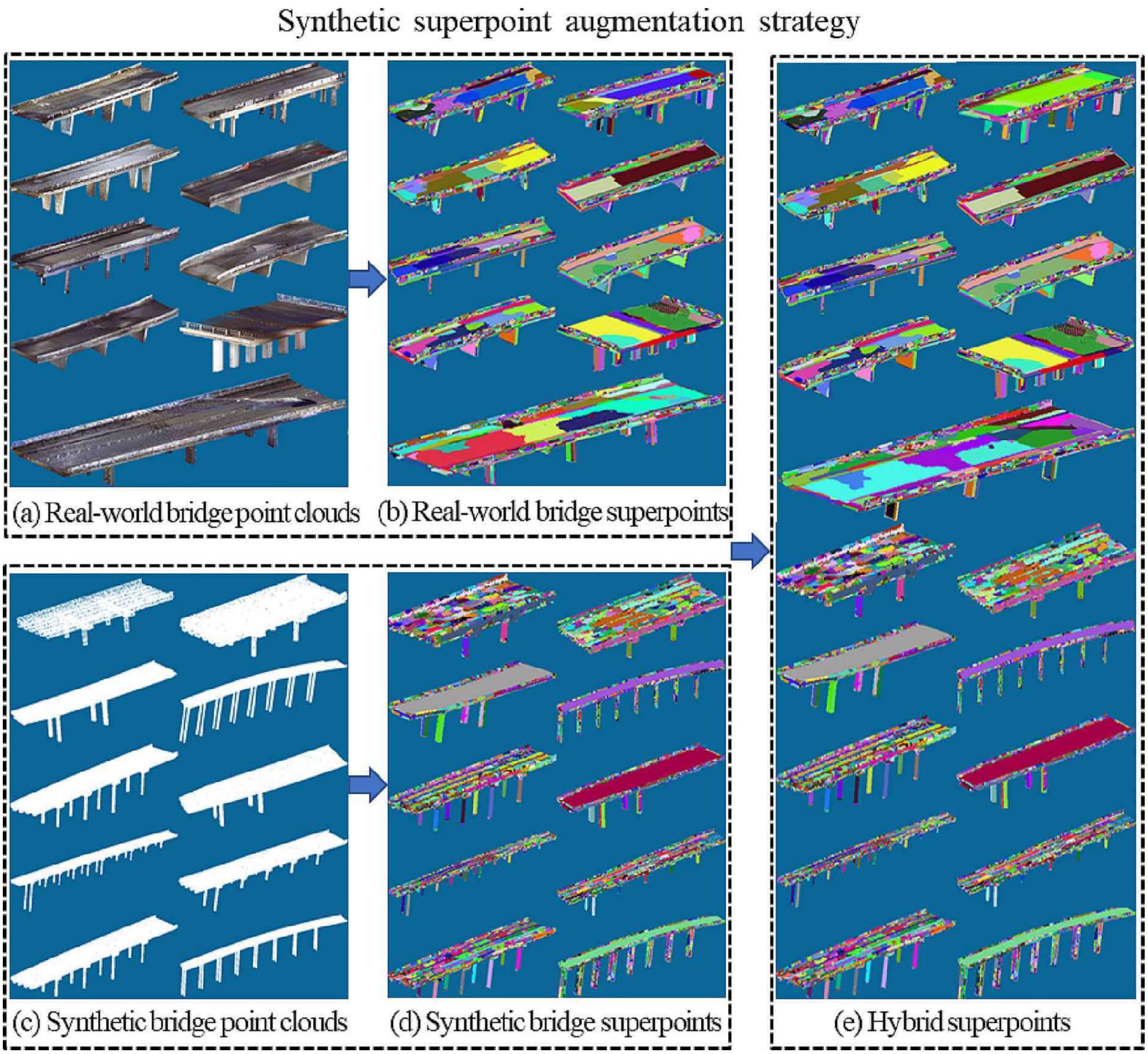
3.3. *Superpoint generation with graph-structured deep metric learning*

Deep metric learning method is able to build similarity or dissimilarity between different points by reducing the distance between points of same class and increasing the distance between points of different class, in such a way that points in the same component are merged together and points in different components are pushed apart. Hence, the deep metric learning method is able to boost generation of high-quality superpoints. High-quality superpoints are defined by three aspects: 1) semantic-purity ($P1$), which means superpoints cannot overlap over different bridge components; 2) border recall ($P2$), meaning the intersection between superpoints must align to the boundaries between different components; and 3) regularity ($P3$), which represents superpoints' contour must be simple [36].

This section summarises a high-quality superpoint generation method using the graph-structured deep metric learning method [36]. This method can be divided into two steps: 1) local point embedding: encoding original 3 dimensional (3D) data points into feature vectors by leveraging a simple yet efficient neural network; 2) graph-structured superpoint generation: generating superpoints by comparing similarity between a pair of feature vectors using graph-structured method.

3.3.1. *Local point embedding*

Local point embedding associates each point with m-dimensional embedding e_i defined by its point features (e.g. position) and set features (e.g. the geometry of each point's local neighborhood). These embeddings are normalized to be within the m-unit sphere (S_m) to avoid collapse during the training process. The local point embedding process consists of two sub-stages: 1) spatial transformer and 2) local point



(a) Real-world bridge point clouds (b) Real-world bridge superpoints

(c) Synthetic bridge point clouds (d) Synthetic bridge superpoints

(e) Hybrid superpoints

Fig. 3. Visualization of each step of the synthetic superpoint augmentation strategy.

embedder, as shown in Fig. 4.

As illustrated in Fig. 4 Stage 1, the spatial transformer takes a point's coordinate c_i and its neighbors' coordinates C_i as inputs, where the local neighborhood size (k) is set to 20. '1' stands for associating each point with its neighborhood. Subsequently, neighbors' coordinates of each point is normalized around point's coordinate c_i, where 'r' is the radius of a point's neighborhood and computed by Eq. (1). This process is notated as '2' and defined by Eq. (2), resulting in the standard deviation of point's coordinate to be 1.

$$rad = std(C_i) \tag{1}$$

$$\hat{C}_i = (C_i - c_i)\Big/rad \tag{2}$$

Then, with the help of a 2 × 2 rotation matrix $\varDelta$, this normalized neighborhood of each point is rotated around the z axis, notated as process '3'. Within this step, each point's vertical coordinate $c_i^{(z)}$ is firstly extracted, notated as 'z', and then a PointNet-like network ('PTN') is employed to output a 2 × 2 rotation matrix $\varDelta$ around z axis. The rotation matrix and the rotated normalized neighborhood can be defined by Eqs. (3) and (4):

$$\varDelta = PTN\left(\hat{C}_i\right) \tag{3}$$

$$\vec{C_i} = \left\{ c \times \Delta \middle| c \in \hat{C}_i \right\} \tag{4}$$

Along with the rotated normalized neighborhood $\vec{C_i}$, the output of the spatial transform also includes local geometric point features $\vec{c_i}$describing the elevation $c_i^{(z)}$, the neighborhood radius *rad* and its original orientation defined by the rotation matrix [$\varDelta_{x,\,x}, \varDelta_{x,\,y}, \varDelta_{y,\,x}, \varDelta_{y,\,y}$].

To summarize, the objective of the the spatial transform step is to standardize the position of each point's neighborhood, which also boosts the following local point embedder step to learn the position distribution [37].

In Fig. 4 Stage 2, the local point embedder is a small PointNet-like neural network [26], which takes the outputs of the spatial transformer $\vec{C_i}$ and $\vec{c_i}$ as inputs. The rotated normalized neighborhood $\vec{C_i}$ is firstly processed by a multi-layer perceptron, notated as MLP$_1$ (widths 32, 128), followed by a maxpooling operation into a point-feature. The resulting point-feature is then concatenated with the point features $\vec{c_i}$ obtained from the spatial transformer step. Subsequently, the concatenated vector is sent to the MLP$_2$ (widths 64,32,32,m) and then normalized on the unit sphere through a normalization block called L$_2$ block, resulting in the e_i vector.

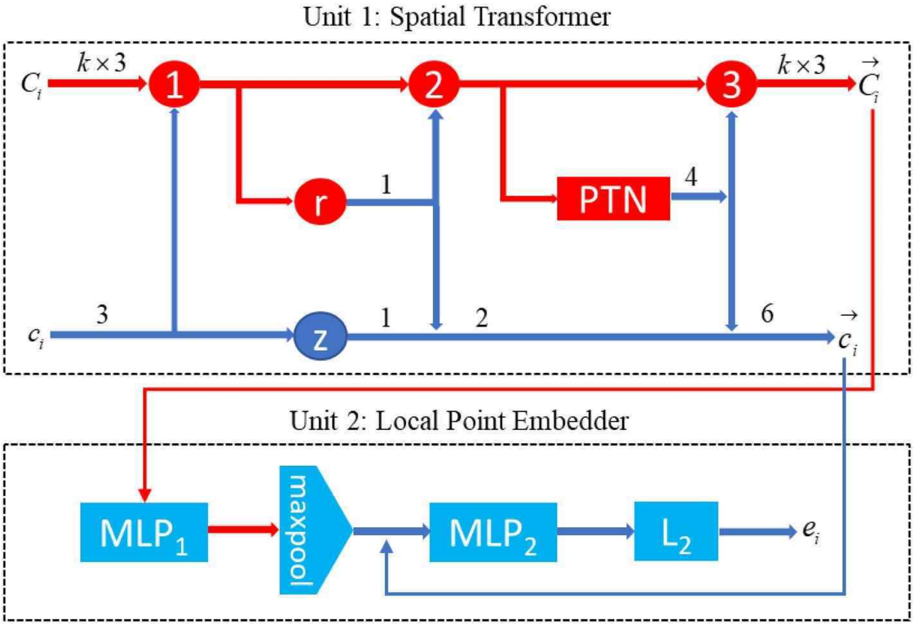


Fig. 4. The workflow of the local point embedding process. The number above each line represents the channel size.

3.3.2. *Graph-structured superpoint generation*

3.3.2.1. The generalized minimal partition problem. The objective of this step is to segment the input point clouds into superpoints that are point clusters with semantic homogeneity and geometrically simple shapes. An undirected adjacency graph $G = (C,E)$ was used to define superpoints, where C stands for the input point cloud and E derives from the e_i vector (the final output of the local point embedding process) and defined by the differences of the two adjacent point feature vectors. Superpoints are defined as the constant connected components in G of the solution of the Equation below [38]:

$$f^* = \underset{f\in R^{C\times m}}{argmin}\sum_{i\in C}\|f_i - e_i\|^2 + \sum_{(i,j)\in E} w_{i,j}[f_i \neq f_j] \tag{5}$$

where f^* denotes the approximate solution of above Eq. (5) and $w_{i,\ j}$ represents the edges' weight, while $[f_i \neq f_j]$ is 1 if true and 0 otherwise.

In this study, we further augmented each point's local embeddings e_i by concatenating them with the point's global spatial embedding. The point's global spatial embedding is denoted by the product of point's coordinate values and dataset-specific spatial embedding coefficient. The augmented point's embeddings denoted as $\overrightarrow{e_i}$ are then fed into the Eq. (5) to replace e_i to generate superpoints, computed as below:

$$\overrightarrow{e_i} = e_i \oplus \gamma^* xyz \tag{6}$$

$$f^* = \underset{f\in R^{C\times m}}{argmin}\sum_{i\in C}\|f_i - \overrightarrow{e_i}\|^2 + \sum_{(i,j)\in E} w_{i,j}[f_i \neq f_j] \tag{7}$$

where $\oplus$ stands for the concatenated operation, while γ represents dataset-specific spatial embedding coefficient, and xyz denotes the point's coordinate values.

The benefits of this improvement are: 1) limiting the maximum size of the generated superpoints, resulting in relatively uniform superpoints; and 2) increasing the number of training superpoints for the superpoint classification neural network, leading to a better semantic segmentation performance.

The edges' weight is defined as $w_{i,j} = \lambda exp\left(\frac{-1}{\gamma}\|\overrightarrow{e_i} - \overrightarrow{e_j}\|^2\right)$ with λ and γ being real numbers in the positive domain (λ, $\gamma \in R^+$). This definition encourages superpoints' boundaries to align to high contrast areas. Parameter λ is the regularization strength and controls the superpoint contour making the superpoint shape simple, which ensures regularity ($P3$).

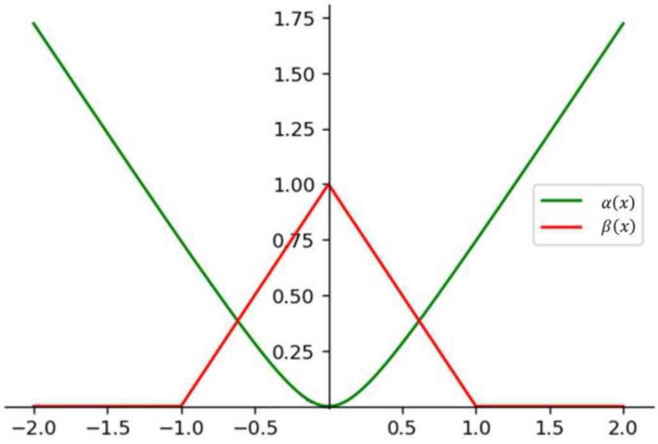


Fig. 5. Visualization of functions α and βin the graph-structured contrastive loss [36].

Eq. (7) is a generalized minimal partition (GMP) problem [35]. Here note that GMP is a optimization problem with the non-continuous non-differentiable non-convex property, and thus the global minimum of Eq. (7) is not able to be obtained. As such, the $\ell_0 - cut$ pursuit algorithm is leveraged to provide an approximate solution for Eq. (7) [38].

3.3.2.2. Graph-structured contrastive loss function. A graph-structured contrastive loss function is proposed herein to ensure correct detection of the bridge component boundaries making superpoints and bridge components share the same boundaries, and thus meet two superpoint properties (i.e. semantic-purity $P1$ and border recall $P2$). This graph-structured contrastive loss [36] is defined by the Equation below:

$$\ell(e) = \frac{1}{|E|}\left(\sum_{(i,j)\in E_{\text{intra}}}\alpha(e_i - e_j) + \sum_{(i,j)\in E_{\text{inter}}}\mu_{i,j}\beta(e_i - e_j)\right) \tag{7}$$

where intra-edges E_{intra} denote the set of edges of G between points within the same component and inter-edges E_{inter} represent the set of

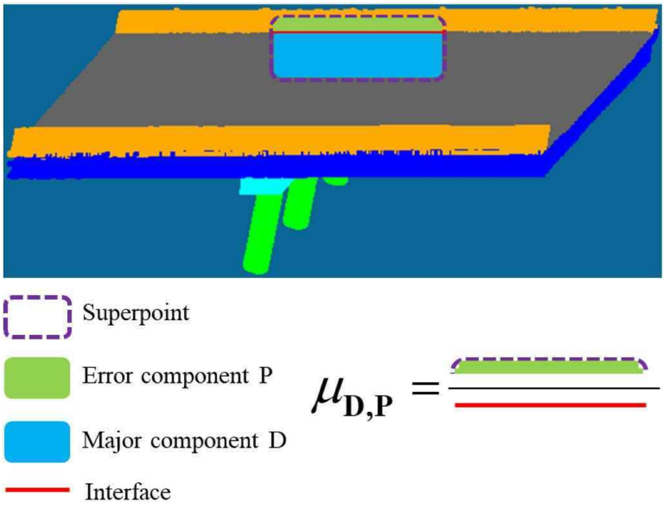


Fig. 6. Visualization of the cross-partition strategy applied to a bridge with a superpoint covering the deck (D) and parapet (P).

edges of Gbetween points from adjacent components. α and β represent a function with minimal and maximal at 0 respectively, as shown in Fig. 5. $\mu_{i,\,j} \in R^{E_{\text{inter}}}$ stands for a weight applying to inter-edges.

This contrastive loss function encourages the similar points' embeddings e_i, when points are linked by intra-edges, while boosting different points' embeddings when points linked by inter-edges.

Fig. 5 presents the functions α (in green) and β (in red) leveraged in the graph-structured contrastive loss.

Function $\alpha(e_i - e_j)$ aims to boost the smooth homogeneity of embeddings within the same component, and is defined as the Equation below:

$$\alpha(e_i - e_j) = \partial\left(\sqrt{\|e_i - e_j\|^2 / \partial^2 + 1} - 1\right) \tag{8}$$

where ∂ value is set as 0.3 so that the distance between embeddings of points linked by intra-edges is reduced.

On the other hand, function $\beta(e_i - e_j)$ punishes similar embeddings at the boundary between adjacent components, and is defined as the following Equation:

$$\beta(e_i - e_j) = max(1 - \|e_i - e_j\|, 0) \tag{9}$$

We leveraged $\beta(e_i - e_j)$ to set a threshold for differences of two points' embeddings larger than 1 because the points' embeddings are constrained to the unit sphere. This function learns the embeddings of vertices on the bounaries linked by inter-edges to be an euclidean distance of 1 rather than pursuiting a larger difference. Note that 0 loss is obtained when any embeddings contained in the same component are constant. Otherwise, the difference of embeddings of points between adjacent components is great than or equal to 1.

3.3.2.3. Cross-partition weighting. The weight on inter-edges is represented as $\mu_{i,\,j} \in R^{E_{\text{inter}}}$, which significantly influences the performance of

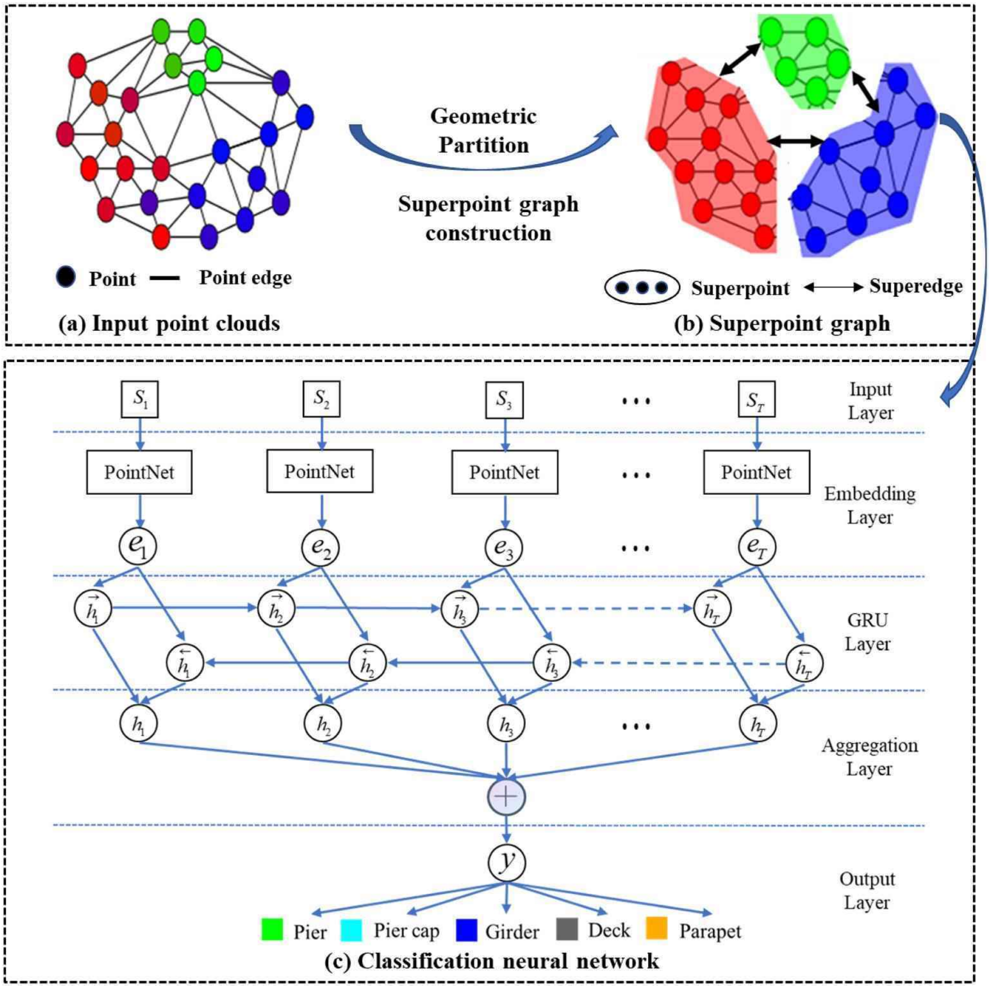
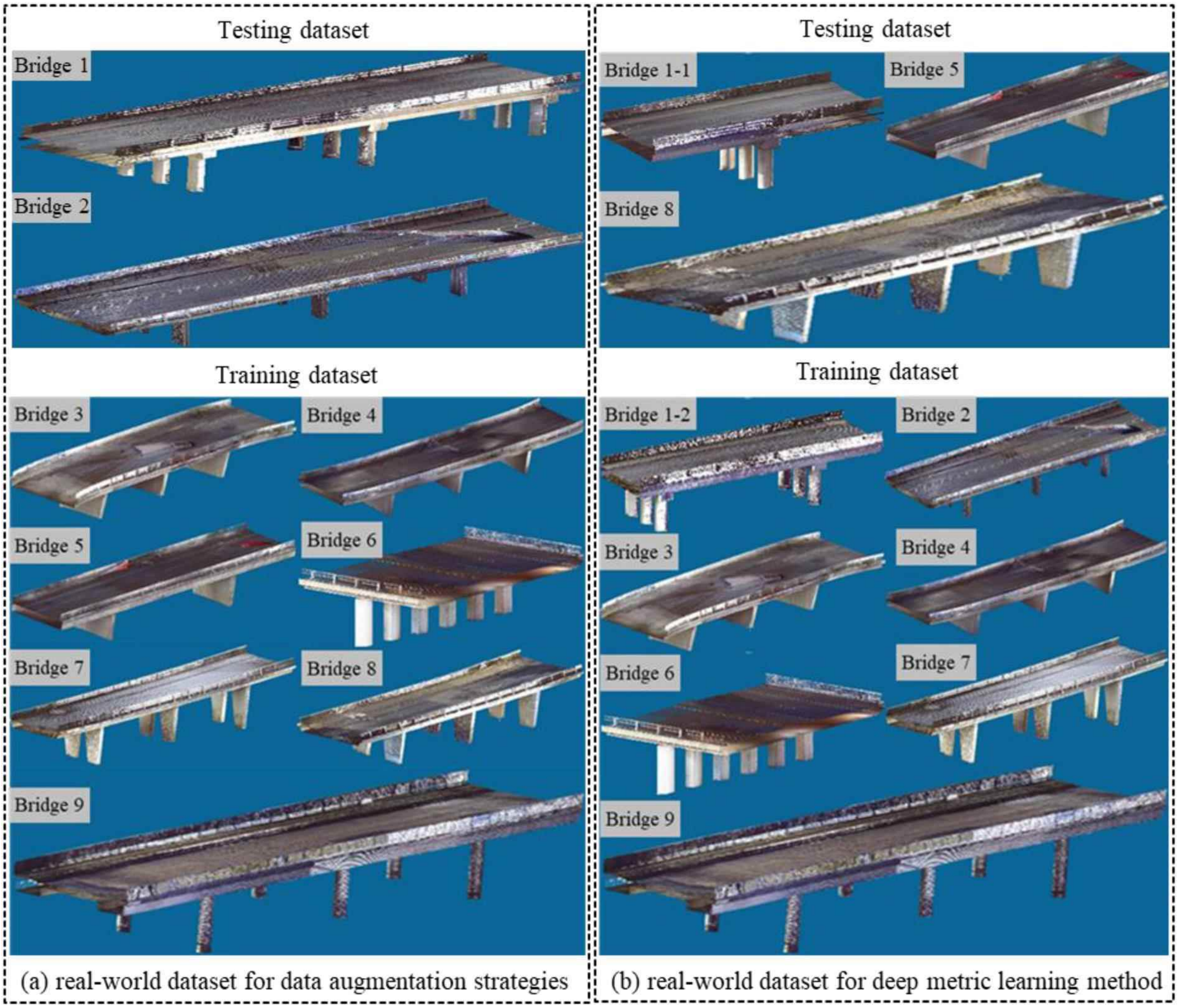


Fig. 7. Workflow of the weighted superpoint graph model.



(a) real-world dataset for data augmentation strategies

(b) real-world dataset for deep metric learning method

Fig. 8. Real-world bridge datasets used for data augmentation strategies and the deep metric learning method.

the graph-structured contrastive loss. The cross-partition weighting strategy described here was leveraged when computing the aforementioned contrastive loss function to ensure the semantic purity of superpoints during the superpoint generation process. In this strategy, the cross-segmentation graph $\delta = (\upsilon, \varepsilon)$ is firstly computed, which is the adjacent relationship of cross-parition υ of the input point cloud C between the superpoints partition $S$and the component partition Ω. Note that the cross-partition υ is a set of superpoints along the component border. Additionally, ε represents sets of super-edges, while υ and ε are defined by the following Equations:

$$\upsilon = \{\Omega \cap S | \Omega \in \Omega, S \in S\} \tag{10}$$

$$\varepsilon = \{\{(i,j) \in (\mathrm{M} \times N) \cap E_{\mathrm{inter}}\} | \mathrm{M}, N \in \upsilon\} \tag{11}$$

where M and N represent two superpoints in υ, while super-edge $(\mathrm{M}, \mathrm{N}) \in \varepsilon$ denotes the set of inter-edges of E_{inter} between M and N.

The weights $\mu_{\mathrm{M},\,\mathrm{N}}$ and $\mu_{i,\,j}$ are associated with each super-edge (M, N) and each edge respectively, defined by the Equations below:

$$\mu_{\mathrm{M},N} = \frac{\mu min(|\mathrm{M}|, |N|)}{|(\mathrm{M}, N)|}, \text{with } (\mathrm{M}, N) \in \varepsilon \tag{12}$$

$$\mu_{i,j} = \mu_{\mathrm{M},N}, \text{with all } (i,j) \in (\mathrm{M}, N) \tag{13}$$

where $|\mathrm{M}|$ and $|\mathrm{N}|$ refer to the number of points within superpoints M and N, while $|(\mathrm{M}, \mathrm{N})|$ points to the number of edges within the interface.

The cross-partition weighting strategy is further illustrated in a bridge scene, as shown in Fig. 6, where a superpoint (in purple) overlaps the deck and is divided into two sub-superpoints (i.e. major component D and error component P) by the interface between D and P. The cross-partition weight $\mu_{D,\,P}$ can be calculated by the number of points in the P dividing the number of edges constituting the interface, which evenly distributes the weight over the edges contained in the interface.

3.4. Weighted superpoint graph evaluation network

The WSPG model was leveraged as the evaluation network. The workflow of the original WSPG model is presented in the Stage 3 of Fig. 1 and illustrated in Fig. 7. Input point clouds are firstly clustered into distinct individual superpoints according to the local geometric feature descriptors (e.g. linearity, planarity, scattering, and normalized elevation) [35], as shown in Fig. 7(a). Subsequently, the superpoint graph $G = (S, \varepsilon, F)$ was constructured based on superpoints S, superedges ε, and superedge features F [39], as illustrated in Fig. 7(b). Finally, superpoints were fed into the superpoint classification neural network and classified into different bridge component categories, as illustrated in Fig. 7(c). Within the superpoint classification neural network, PointNet architecture was firstly leveraged for local embedding learning to compute descriptors for each superpoint, followed by inputing the resulting descriptors of each superpoint S_i into a simplified RNN network called GRU [40] to construct long-range dependencies between superpoints

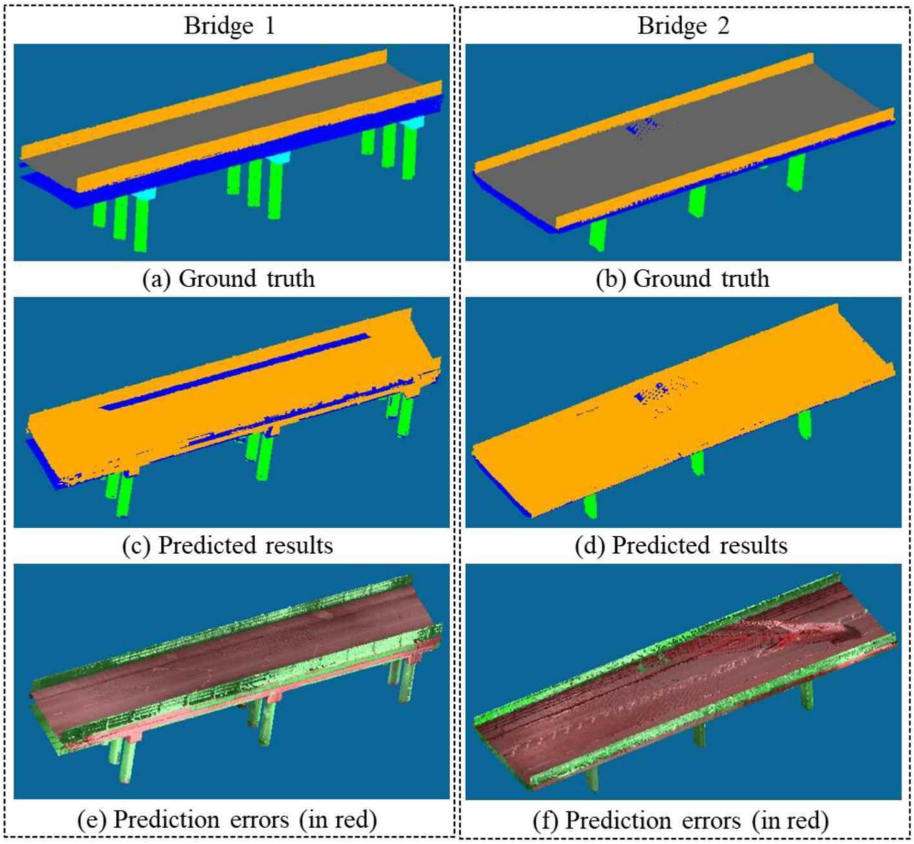


Fig. 9. Visualization of the prediction results of the WSPG model without synthetic data augmentation strategies on Bridge 1 and Bridge 2.

within the superpoint graph.

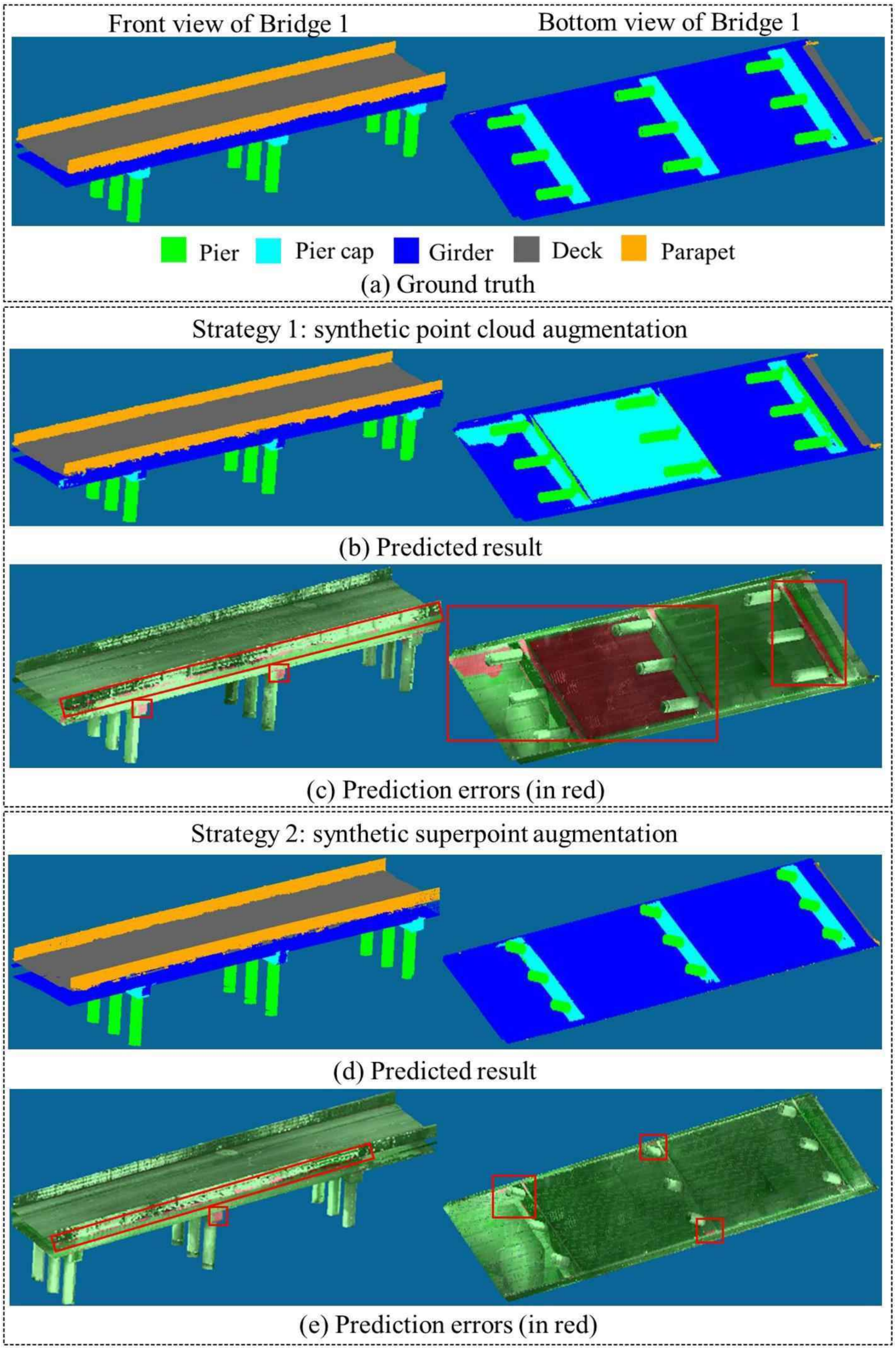
4. Experiment and results

4.1. Data preparation

We firstly established two well-annotated datasets for both real-world and synthetic bridge point clouds containing 5 types of bridge components (i.e. piers, pier caps, girders, decks and parapets), which was leveraged to validate the viability of our proposed synthetic data augmentation strategies. The real-world dataset based on a raw open-source dataset shown in Fig. 8(a) (provided by Lu et al. [11]), where Bridge 1 and Bridge 2 were assigned to the testing dataset and the rest of bridges were assigned to the training dataset. This division was used to simulate training sample scarcity, with the training dataset not including some bridge characteristics of Bridge 1 and Bridge 2 such as different types of piers and pier caps. As summarised in Appendix Table 1, the real-world training dataset only had one pier cap and did not include cuboid piers that were present in the real-world testing dataset. The synthetic bridge point cloud dataset contained 10 bridge point clouds. These synthetic bridge point clouds were generated by randomly sampling points from the surface of mesh models that were converted from as-designed BrIMs, resulting in an evenly distributed point density across the entire bridge, as shown in Fig. 2(b). The as-designed BrIMs had the same bridge types with the real-world dataset, so that the synthetic bridge point clouds carry similar geometric characteristics (i.e. linearity, planarity, scattering, verticality and elevation) to the real-world data. Nevertheless, synthetic bridge point clouds are slightly different from the real-world bridge point clouds. The reasons for these differences can be split into two groups. Firstly, the point density of real-world bridge point clouds is highly uneven across the bridge, where components closer to the laser scanner are more densely captured than the ones further away [41]. By contrast, the synthetic point clouds have an evenly distributed point density across the entire bridge. The Voxel Grid Filter algorithm can be used to downsample point clouds and obtain relatively uniform point clouds, but the differences in the point distributions between the synthetic and real-world point clouds still influence the representation of the point cloud feature. Secondly, real-world data includes noise from sources such as outlier points and incomplete point clouds caused by ambient occlusions, which also affects the geometric feature representation. Additionally, to verify the effectiveness of the proposed graph-structured deep metric learning method, we reorganized the real-world testing dataset, as shown in Fig. 8(b), with Bridge 5, Bridge 8 and a portion of Bridge 1 being assigned to the testing dataset and the remaining bridges to the training dataset. This ensured that the testing and training datasets both had similar features.

4.2. Implementation details

This study leveraged the WSPG model as the evaluation model. Pytorch [42] was selected as the deep learning framework, and evaluation experiments were implemented on an Ubuntu 20.04 system configured with an Intel Xeon CPU and a Tesla T4 GPU. A voxel size of 0.1 m was selected as a trade off between the segmentation accuracy and the training and inference speed. For training the deep metric learning to generate high quality superpoints, the dimension of point embeddings was set to 4, and the Adam optimizer [43] was leveraged for minimizing the overall loss. Additionally, we chose a batch size of 4 and an initial learning rate of 0.01. The deep metric learning network was trained for 50 epochs, where a stepwise learning rate decay was set as 0.7 at epochs 20, 35 and 45. When training the superpoint classification neural network within the WSPG model, the batch size of 5 was set and the Adam optimizer was also adopted. The WSPG network was trained for


Front view of Bridge 1
Bottom view of Bridge 1
🟩 Pier
🟦 Pier cap
🟦 Girder
⬛ Deck
🟧 Parapet
Strategy 1: synthetic point cloud augmentation
Strategy 2: synthetic superpoint augmentation

(a) Ground truth

(b) Predicted result

(c) Prediction errors (in red)

(d) Predicted result

(e) Prediction errors (in red)

Fig. 10. Visualization of the prediction results for the WSPG model using two synthetic data augmentation strategies on the front and bottom views of Bridge 1.

350 epochs with an initial learning rate of 0.01 and a learning rate decay of 0.7 at epochs 280 and 320.

For evaluating experiments, the metrics of overall accuracy (OA), mean class Accuracy (mAcc), and mean Intersection over Union (mIoU) were selected to evaluate the semantic segmentation performance and defined as below:

$$OA = \frac{\sum_{i=1}^{n} p_{ii}}{\sum_{i=1}^{n}\sum_{j=1}^{n} p_{ij}} \tag{14}$$

$$IoU = \frac{p_{ii}}{\sum_{j=1}^{n} p_{ij} + \sum_{j=1}^{n} p_{ji} - p_{ii}} \tag{15}$$

$$mIoU = \frac{1}{n}\sum_{i=1}^{n} \frac{p_{ii}}{\sum_{j=1}^{n} p_{ij} + \sum_{j=1}^{n} p_{ji} - p_{ii}} \tag{16}$$

$$mAcc = \frac{1}{n}\sum_{i=1}^{n} \frac{p_{ii}}{\sum_{j=1}^{n} p_{ij}} \tag{17}$$

where n denotes the number of bridge component categories, p_{ij} refers to the points belonging to the i category that are wrongly predicted to the j category.

4.3. Evaluation

The evaluation experiments can be divided into two groups: (1)

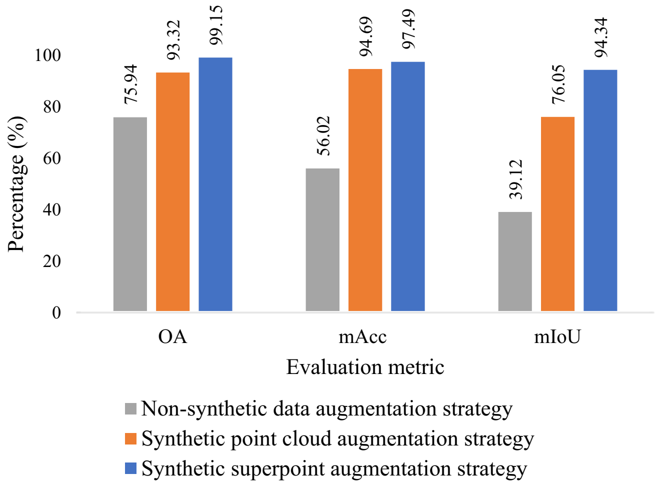


Fig. 11. Overall evaluation results (%) across the different data augmentation strategies.

evaluation of data augmentation strategies (the effectiveness of the proposed synthetic point cloud augmentation and synthetic superpoint augmentation strategies), and (2) evaluation of deep metric learning for high quality superpoint generation (its ability to address the low segmentation accuracy on boundaries). The details are discussed in the following sections.

4.3.1. Evaluation of data augmentation strategies

4.3.1.1. Qualitative evaluation. Visualization of the semantic segmentation results on the real-world testing dataset using the WSPG model without the synthetic data augmentation strategy are presented in Fig. 9. This Figure visually compared the difference between prediction results and ground truth as well as labelled prediction errors in red. Most bridge components such as pier caps and decks (labelled in red in Fig. 9(e) and (f)) were not successfully recognized from the full-scale bridge point clouds, despite data scarcity only existed in piers and pier caps. It can be concluded that data scarcity of specific bridge component categories not only decreased the recognition performance of themselves but also affected other component categories. These results also demonstrated that deep learning techniques are heavily dependent on a sufficient number of training samples.

The predicted results for the WSPG model with the two data augmentation strategies (Strategy1: synthetic point cloud augmentation strategy and Strategy 2: synthetic superpoint augmentation strategy) are presented in Fig. 10. The two synthetic data augmentation strategies both significantly improved the semantic segmentation performance compared to the WSPG model trained without synthetic data augmentation. As can be seen from Fig. 10 Strategy 1, most bridge components were correctly recognized from the entire bridge point clouds using the synthetic point cloud augmentation strategy, but two problems still remain. Firstly, most boundaries between adjacent bridge components were not successfully distinguished such as boundaries between girders and parapets as shown in Fig. 10(c). Secondly, the pier caps and girders were not correctly classified. The second issue was because the training samples comprised of real-world and synthetic point clouds may have difficulty generating semantically homogeneous superpoints. In other words, the different geometric features within real-world and synthetic point clouds may result in coarse superpoints that cover multiple bridge components, which will significantly affect the accuracy of the superpoint classification neural network.

As presented in Fig. 10(e), the performance of the synthetic superpoint augmentation strategy was significantly better than the performance of the synthetic point cloud augmentation strategy, especially for girders and pier caps, despite some small misclassification at the boundaries between components and on pier caps. Possible reasons for the improved performance of this strategy can be broke down into two groups. Firstly, separately generated superpoints from real-world bridge point clouds and synthetic bridge point clouds may result in higher superpoint quality. Secondly, the generated real-world superpoints were augmented by synthetic superpoints to increase the number of training samples, as superpoints were the input of the posterior classifaction neural network. Additionally, the reasons for the small failure in pier caps was the lack of training point clouds of these components, despite 10 synthetic point clouds being leveraged to augment training samples.

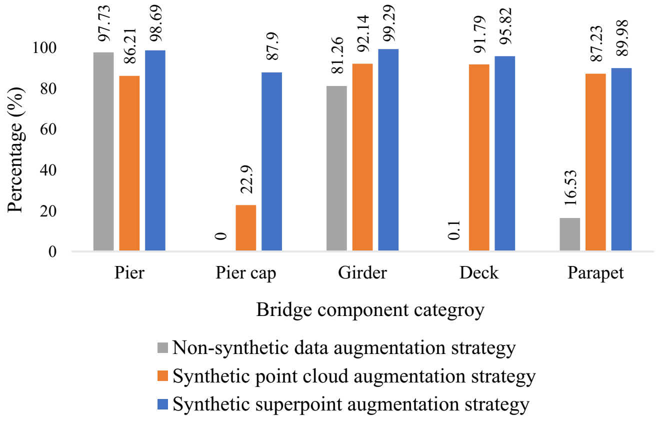


Fig. 12. Metrics of IoU (%) for each component category over different data augmentation strategies.

4.3.1.2. Quantitative evaluation. The quantitative evaluation of the two synthetic data augmentation strategies used in this research focussed on the evaluation of the overall performance and evaluation of the performance at the component level. Firstly, overall performance was comnpared across three augmentation strategies (i.e. non-synthetic data augmentation, synthetic point cloud augmentation strategy and synthetic superpoint augmentation strategy), as presented in Fig. 11. The non-synthetic data augmentation strategy has the worst performance due to a lack of sufficient training samples. Both synthetic point cloud and superpoint augmentation strategies significantly improved the semantic segmentation performance across all the overall evaluation metrics. The improvements of synthetic point cloud and superpoint augmentation strategies respectively exceeded the non-synthetic data augmentation strategy by 17.38% and 23.21% for OA, 38.67% and 41.47% for mAcc as well as 36.93% and 55.22% for mIoU.

It is worth noting that the synthetic superpoint augmentation strategy outperformed the synthetic point cloud augmentation strategy on all overall evaluation metrics, with an increase of 5.83% for OA, 2.8% for mAcc and 18.29% for mIoU.

A comparison for the three augmentation strategies across IoU metrics for each component category is summarised in Fig. 12. The synthetic superpoint augmentation strategy achieved the best results across all bridge component categories (in blue in Fig. 12). Here note that the segmentation accuracy for pier caps and parapets was generally lower than the accuracy for the other bridge components across the strategies. This phenomenon was a result of the geometric features contained in the real-world bridge point clouds being slightly different from those extracted from the synthetic bridge point clouds, resulting in a relative scarcity of training samples for pier caps and parapets. Hence, augmenting real-world bridge point clouds using synthetic data with similar geometries had a slight influence on the final semantic segmentation performance, demonstrating that synthetic data is not able to completely replace real-world data.

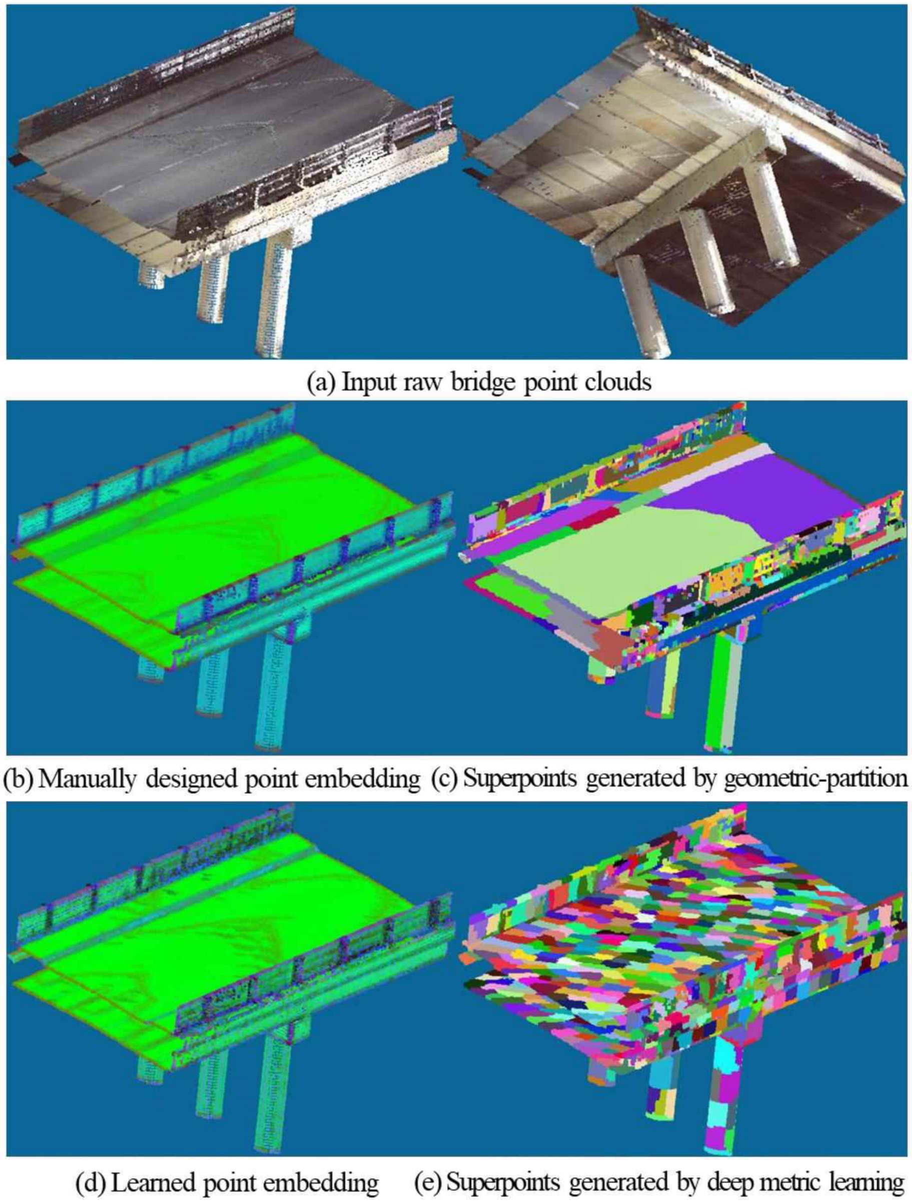

(a) Input raw bridge point clouds
(b) Manually designed point embedding
(c) Superpoints generated by geometric-partition
(d) Learned point embedding
(e) Superpoints generated by deep metric learning

Fig. 13. Visualization examples of the superpoint generation process using the original geometric partitioned method and the graph-structured deep metric learning method.

4.3.2. Improved WSPG model with the graph-structured deep metric learning

We conducted comprehensive evaluation experiments (both qualitative and quantitative evaluations) to validate the effectiveness of the improved WSPG model with the graph-structured deep metric learning method for alleviating the erroneous boundary segmentation issue in semantic segmentation tasks. We leveraged the evaluation metrics of OA, mAcc and mIoU to indirectly quantify the effectiveness of the proposed method because no specific metric has been designed for measuring the boundary segmentation performance [44].

4.3.2.1. Qualitative evaluation. Visualization examples of superpoint generation using the original geometric partitioned method and the graph-structured deep metric learning method are presented in Fig. 13. From Fig. 13(b) and (d), the deep metric learning method is able to learn more accurate local point embedding along the component boundaries than the original method. In addition, as shown by Fig. 13(c) and (e), the deep metric learning method can segment distinct bridge components with clearer boundaries, leading to higher quality superpoints.

The predicted results for the front and bottom views of part of Bridge 1 based on the WSPG models with both the original geometric partitioned superpoint generation method and the deep metric learning superpoint generation method are visually presented in Fig. 14. Most bridge components were correctly recognized from the entire bridge point clouds, with the exception of pier caps and boundaries between adjacent bridge components such as the boundary between pier caps and girders. As expected, the performance of the improved WSPG model with the deep metric learning method exceeded the performance of the original WSPG model with the geometric partitioned superpoint generation method, especially the identification of the bridge component boundaries, as shown in Fig. 14(d) and (e).

The original WSPG model with the geometric partitioned superpoint generation method (method 1 in Fig. 14(b)) resulted in segmentation errors for the pier, pier cap and the component boundaries. As labelled in red in Fig. 14(c), the errors were caused by coarse superpoints that overlaps adjacent components. These coarse superpoints generated by the original method was reliant on the predefined geometric feature

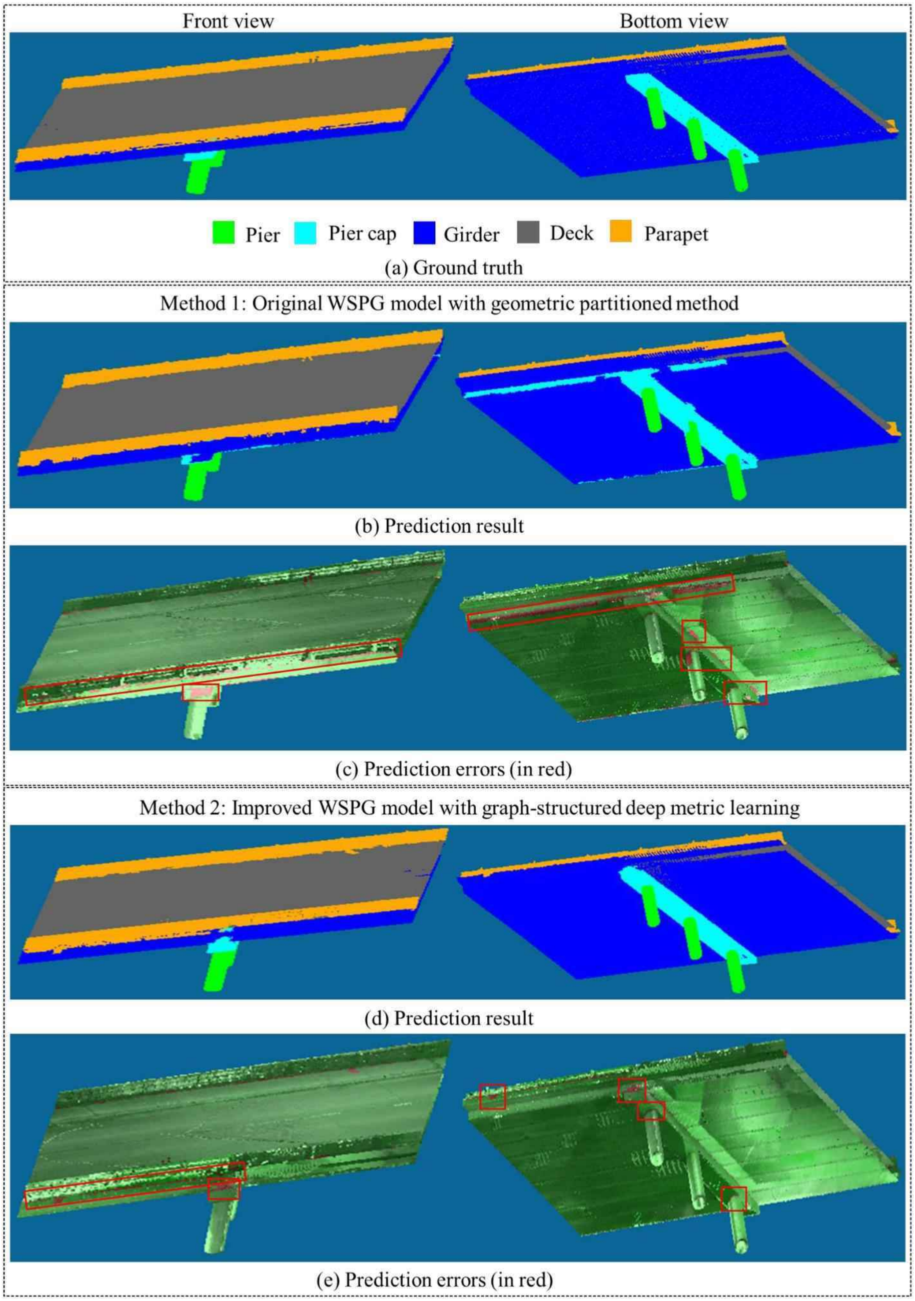

Front view
Bottom view
🟩 Pier 🟦 Pier cap 🟦 Girder ⬛ Deck 🟧 Parapet
(a) Ground truth
Method 1: Original WSPG model with geometric partitioned method
(b) Prediction result
(c) Prediction errors (in red)
Method 2: Improved WSPG model with graph-structured deep metric learning
(d) Prediction result
(e) Prediction errors (in red)

Fig. 14. Visualization of prediction results for the original WSPG model with geometric partitioned method and improved WSPG model with graph-structured deep metric learning method for the front and bottom views of part of Bridge 1.

descriptors (e.g. linearity, planarity, scattering, and normalized elevation), but the predefined geometric descriptors may not fully represent the real geometric features contained in the various bridge components, increasing the coarseness of the generated superpoints. The coarseness compromises the generated superpoints with homogeneous component semantic information.

The improved WSPG model with the graph-structured deep metric learning method (method 2 in Fig. 14) produced small errors at the pier cap and bridge component boundaries. The small errors at the boundaries may depend on the following intrinsic properties of bridge component boundaries:

1) vegetation, such as grass and moss, that often appears on the component boundaries and especially on the boundary between deck and parapet, causing inconsistency in the geometric features along the boundaries;
2) poor definition of the boundaries between adjacent components because of their semantic ambiguity, meaning that a point may belong to multiple classes at the same time;
3) subjective annotation of the ground truth (i.e. the manual annotated results) by human modellers, which may result in mislabeling, particularly in the boundaries.

4.3.2.2. Quantitative evaluation. Fig. 15 displays the comparison results of overall evaluation metrics for the original WSPG model with geometric partitioned method and improved WSPG model with the graph-structured deep metric learning method using the metrics of OA, mAcc and mIoU. The improved WSPG model with the graph-structured deep metric learning method increased the metric of OA slightly and the

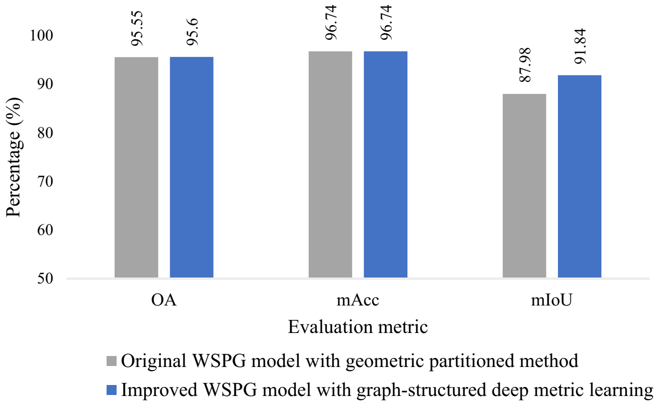


Fig. 15. Comparison of the overall evaluation metrics between the original and improved WSPG model.

metric of mIoU by 4%, but did not change the metric of mAcc compared with the original WSPG method. The improvement of the mIoU indirectly demonstrated the ability of the improved WSPG model with the graph-structured deep metric learning method to reduce the erroneous boundary segmentation. It is worth noting that the number of points on the boundaries is only a small proportion of the total number of points. This phenomenon explains the relatively small improvement on the overall evaluation metrics, despite the significant improvement of the boundary segmentation performance. Future work should focus on designing a series of specific evaluation metrics for evaluating the boundary segmentation performance.

Along with the evaluation on the overall metrics, Fig. 16 presents the IoU comparison results for each bridge component category. The improved WSPG model with the graph-structured deep metric learning method achieved good results for the pier (99.36%), pier cap (94.6%), girder (93.64%) and performed satisfactorily for the deck (83.18%) and parapet (88.43%). The possible reason for the lower performance of the deck and parapet is that vegetation on the boundary between deck and parapet caused inconsistency in the geometric features, resulting misclassification along the boundary, as presented in Fig. 14(e). The improved WSPG model with the graph-structured deep metric learning method achieved the best results across all components apart from the parapets, where there was a slight decrease in performance compared to the original method. The improvement is especially significant for pier caps, where the performance improved by 20.75% IoU. These results further demonstrate the ability of the improved WSPG model with the graph-structured deep metric learning method to boost the performance of minority component categories by improving the boundary segmentation performance.

5. Conclusions

As-built BrIMs present great potential to store, analyze and manage inspection data, enabling full automation of bridge condition assessment. The semantic segmentation of bridge point clouds is an essential intermediate step for the reconstruction of BrIMs for existing bridges. Currently, two main issues have hindered the application of deep learning techniques for the autonomous bridge semantic segmentation task: real-world training sample scarcity and erroneous boundary segmentation. This study firstly presented two synthetic data augmentation strategies (i.e. synthetic point cloud augmentation strategy and synthetic superpoint augmentation strategy) for superpoint-based algorithms, and analyzed their effectiveness based on the WSPG model. The evaluation experiments demonstrated that synthetic data augmentation strategies can greatly enhance the semantic segmentation performance of bridge components. The synthetic superpoint augmentation strategy achieved the best overall evaluation performance, with an increase of around 6% OA, 3% mAcc and 18% mIoU compared with the synthetic point cloud augmentation strategy. In addition, the synthetic superpoint augmentation strategy outperformed the synthetic point cloud augmentation strategy based on IoUs for all component categories.

We then validated the effectiveness of the improved WSPG model with graph-structured deep metric learning method for generating high quality superpoints to boost the boundary segmentation performance. The experiments demonstrate its strength for alleviating the erroneous boundary segmentation issue by improving mIoU by 4%, thus also improving the segmentation performance for minor bridge component categories like pier caps.

While this study has presented a strategy to improve the semantic segmentation of bridge point clouds, some challenges still remain:

1) synthetic bridge point clouds generated from BrIMs are not able to completely replace real-world datasets because they do not include bridge texture information and their geometric features do not include the outlier points and incomplete point clouds contained in real-world data caused by ambient occlusions; 2) no specific evaluation metric is designed for boundary segmentation performance, with this study leveraging generic evaluation metrics such as OA, mAcc and mIoU to indirectly measure the boundary segmentation performance; 3) The

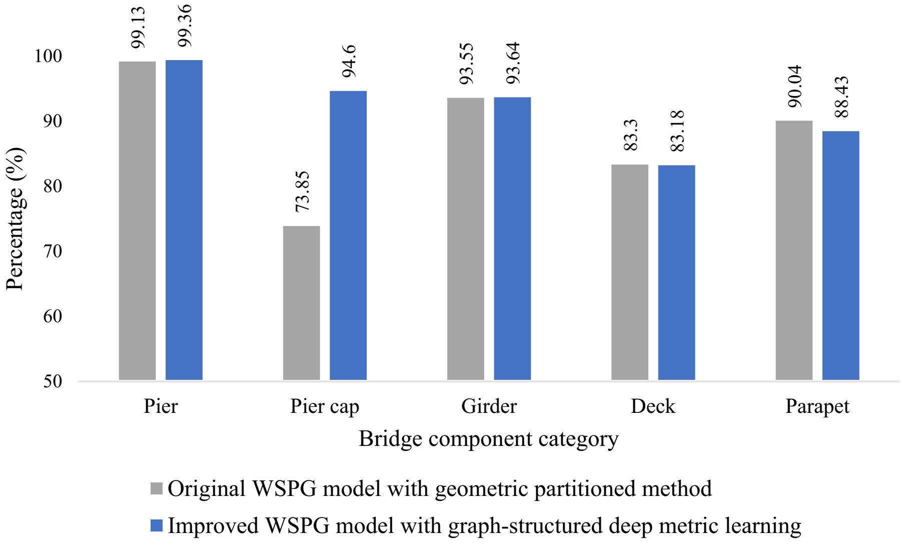


Fig. 16. Comparison of the IoU (%) for each component category based on the original and improved WSPG model.

proposed method is only validated on the concrete bridges and may be not effective on other bridge types such as truss bridges as these bridge point clouds present complicated geometry and have more minor components as well as are normally incomplete missing geometric features of their components.

To address aforementioned limitations, future work should: 1) explore a user-friendly generation approach for real-world-approached synthetic data including color information and noise resulted from ambient occlusions; 2) design a series of metrics to evaluate the boundary segmentation quality; 3) improve the proposed method to make it applicable to a wider range of bridge types.

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.

Data availability

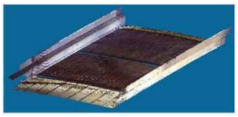
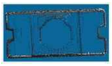
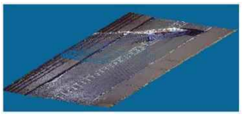
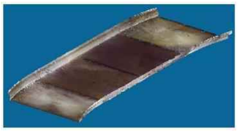
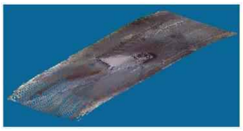
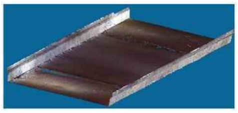
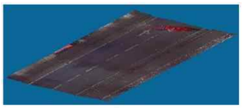
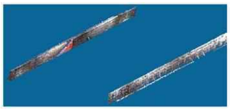
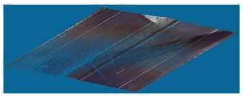
Data will be made available on request.

Acknowledgements

The authors would like to acknowledge the support by University of Auckland FRDF Grant (Project No. 3716476).

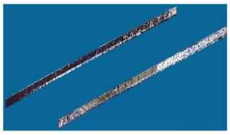
Appendix A. Appendix

Table 1
Visualization examples of component categories for each bridge in the real-world dataset.

Dataset	Bridge No.	Number of component categories	Visualization of component categories for each bridge				
			Piers	Pier caps	Girders	Decks	Parapets
Testing dataset	1	5					
	2	4		/			
Training dataset	3	4		/			
	4	4		/			
	5	4		/			
Training dataset	6	4		/			
	7	4		/			

(continued on next page)

Table 1 (*continued*)

Dataset	Bridge No.	Number of component categories	Visualization of component categories for each bridge				
			Piers	Pier caps	Girders	Decks	Parapets
	8	4		/			
	9	5					

References

[1] ASCE, 2021 Report Card for America's Infrastructure American Society of Civil Engineers, Available at, https://infrastructurereportcard.org/wp-content/uploads/2020/12/National_IRC_2021-report.pdf, 2021. Access date: 14 February, 2021.

[2] A. Farhadi, J. Redmon, Yolov3: An Incremental Improvement, Computer Vision and Pattern Recognition, Springer, Berlin/Heidelberg, Germany, 2018, pp. 1804–2767, https://doi.org/10.48550/arXiv.1804.02767.

[3] C. Zhang, C.C. Chang, M. Jamshidi, Concrete bridge surface damage detection using a single-stage detector, Comp. Aid. Civ. Infrastruct. Eng. 35 (4) (2020) 389–409, https://doi.org/10.1111/mice.12500.

[4] FHWA, National Bridge Inspection Standards, Available at, https://www.fhwa.dot.gov/bridge/nbis2022.cfm, 2022. Access date: 14 February, 2022.

[5] T. McKenna, M. Minehane, B. O'Keeffe, G. O'Sullivan, K. Ruane, Bridge information modelling (BrIM) for a listed viaduct, in: Proceedings of the Institution of Civil Engineers-Bridge Engineering 170, Thomas Telford Ltd, 2017, pp. 192–203, https://doi.org/10.1680/jbren.16.00007.

[6] R. Sacks, A. Kedar, A. Borrmann, L. Ma, I. Brilakis, P. Hüthwohl, S. Daum, U. Kattel, R. Yosef, T. Liebich, SeeBridge as next generation bridge inspection: overview, information delivery manual and model view definition, Autom. Constr. 90 (2018) 134–145, https://doi.org/10.1016/j.autcon.2018.02.033.

[7] R. Lu, I. Brilakis, Digital twinning of existing reinforced concrete bridges from labelled point clusters, Autom. Constr. 105 (2019), 102837, https://doi.org/10.1016/j.autcon.2019.102837.

[8] T. Czerniawski, F. Leite, Automated digital modeling of existing buildings: a review of visual object recognition methods, Autom. Constr. 113 (2020), 103131, https://doi.org/10.1016/j.autcon.2020.103131.

[9] V. Pătrăucean, I. Armeni, M. Nahangi, J. Yeung, I. Brilakis, C. Haas, State of research in automatic as-built modelling, Adv. Eng. Inform. 29 (2) (2015) 162–171, https://doi.org/10.1016/j.aei.2015.01.001.

[10] C. Popescu, B. Täljsten, T. Blanksvärd, L. Elfgren, 3D reconstruction of existing concrete bridges using optical methods, Struct. Infrastruct. Eng. 15 (7) (2019) 912–924, https://doi.org/10.1080/15732479.2019.1594315.

[11] R. Lu, I. Brilakis, C.R. Middleton, Detection of structural components in point clouds of existing RC bridges, Comp. Aid. Civ. Infrastruct. Eng. 34 (3) (2019) 191–212, https://doi.org/10.1111/mice.12407.

[12] D. Isailović, V. Stojanovic, M. Trapp, R. Richter, R. Hajdin, J. Döllner, Bridge damage: detection, IFC-based semantic enrichment and visualization, Autom. Constr. 112 (2020), 103088, https://doi.org/10.1016/j.autcon.2020.103088.

[13] H.V. Dang, M. Tatipamula, H.X. Nguyen, Cloud-based digital twinning for structural health monitoring using deep learning, IEEE Trans. Industrial Inform. 18 (6) (2021) 3820–3830, https://doi.org/10.1109/TII.2021.3115119.

[14] T. Xia, J. Yang, L. Chen, Automated semantic segmentation of bridge point cloud based on local descriptor and machine learning, Autom. Constr. 133 (2022), 103992, https://doi.org/10.1016/j.autcon.2021.103992.

[15] H. Kim, J. Yoon, S.H. Sim, Automated bridge component recognition from point clouds using deep learning, Struct. Control. Health Monit. 27 (9) (2020), e2591, https://doi.org/10.1002/stc.2591.

[16] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, Pointcnn: convolution on x-transformed points, Adv. Neural Inf. Proces. Syst. 31 (2018), https://doi.org/10.48550/arXiv.1801.07791.

[17] Y. Wang, Y. Sun, Z. Liu, S.E. Sarma, M.M. Bronstein, J.M. Solomon, Dynamic graph cnn for learning on point clouds, ACM Trans. Graphics (tog). 38 (5) (2019) 1–12, https://doi.org/10.1145/3326362.

[18] X. Yang, E. del Rey Castillo, Y. Zou, L. Wotherspoon, Y. Tan, Automated semantic segmentation of bridge components from large-scale point clouds using a weighted superpoint graph, Autom. Constr. 142 (2022), 104519, https://doi.org/10.1016/j.autcon.2022.104519.

[19] L. Landrieu, M. Simonovsky, Large-scale point cloud semantic segmentation with superpoint graphs, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2018) 4558–4567, https://doi.org/10.1109/CVPR.2018.00479.

[20] E. Borenstein, S. Ullman, Combined top-down/bottom-up segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 30 (12) (2008) 2109–2125, https://doi.org/10.1109/TPAMI.2007.70840.

[21] R. Schnabel, R. Wahl, R. Klein, Efficient RANSAC for Point-Cloud Shape Detection, Computer Graphics Forum vol. 26, Wiley Online Library, 2007, pp. 214–226, https://doi.org/10.1111/j.1467-8659.2007.01016.x.

[22] S.B. Walsh, D.J. Borello, B. Guldur, J.F. Hajjar, Data processing of point clouds for object detection for structural engineering applications, Comp. Aid. Civ. Infrastruct. Eng. 28 (7) (2013) 495–508, https://doi.org/10.1111/mice.12016.

[23] Z. Dong, B. Yang, P. Hu, S. Scherer, An efficient global energy optimization approach for robust 3D plane segmentation of point clouds, ISPRS J. Photogramm. Remote Sens. 137 (2018) 112–133, https://doi.org/10.1016/j.isprsjprs.2018.01.013.

[24] Y. Yan, J.F. Hajjar, Automated extraction of structural elements in steel girder bridges from laser point clouds, Autom. Constr. 125 (2021), 103582, https://doi.org/10.1016/j.autcon.2021.103582.

[25] F. Hu, J. Zhao, Y. Huang, H. Li, Structure-aware 3D reconstruction for cable-stayed bridges: a learning-based method, Comp. Aid. Civ. Infrastruct. Eng. 36 (1) (2021) 89–108, https://doi.org/10.1111/mice.12568.

[26] C.R. Qi, H. Su, K. Mo, L.J. Guibas, Pointnet: deep learning on point sets for 3d classification and segmentation, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2017) 652–660, https://doi.org/10.1109/CVPR.2017.16.

[27] H. Kim, C. Kim, Deep-learning-based classification of point clouds for bridge inspection, Remote Sens. 12 (22) (2020) 3757, https://doi.org/10.3390/rs12223757.

[28] S. Zhang, X. Li, M. Zong, X. Zhu, R. Wang, Efficient kNN classification with different numbers of nearest neighbors, IEEE Trans. Neural Netw. Learn. Syst. 29 (5) (2017) 1774–1785, https://doi.org/10.1109/TNNLS.2017.2673241.

[29] J.S. Lee, J. Park, Y.-M. Ryu, Semantic segmentation of bridge components based on hierarchical point cloud model, Autom. Constr. 130 (2021), 103847, https://doi.org/10.1016/j.autcon.2021.103847.

[30] A.X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, Shapenet: An information-rich 3d model repository, arXiv (2015), https://doi.org/10.48550/arXiv.1512.03012 preprint arXiv:1512.03012.

[31] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3d shapenets: a deep representation for volumetric shapes, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2015) 1912–1920, https://doi.org/10.48550/arXiv.1406.5670.

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

[33] C. Emunds, N. Pauen, V. Richter, J. Frisch, C. van Treeck, IFCNet: a benchmark dataset for IFC entity classification, in: Proceedings of the EG-ICE 2021 Workshop on Intelligent Computing in Engineering, 2021, pp. 166–175, https://doi.org/10.48550/arXiv.2106.09712.

[34] R. Zhai, J. Zou, Y. He, L. Meng, BIM-driven data augmentation method for semantic segmentation in superpoint-based deep learning network, Autom. Constr. 140 (2022), 104373, https://doi.org/10.1016/j.autcon.2022.104373.

[35] S. Guinard, L. Landrieu, Weakly supervised segmentation-aided classification of urban scenes from 3D LiDAR point clouds, international archives of the photogrammetry, remote sensing and spatial, Inf. Sci. 42 (1/W1) (2017), https://doi.org/10.5194/isprs-archives-XLII-1-W1-151-2017.

[36] L. Landrieu, M. Boussaha, Point cloud oversegmentation with graph-structured deep metric learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 7440–7449, https://doi.org/10.48550/arXiv.1904.02113.

[37] M. Jaderberg, K. Simonyan, A. Zisserman, Spatial transformer networks, Adv. Neural Inf. Proces. Syst. 28 (2015) 2017–2025,, doi: 1506.02025.

[38] L. Landrieu, G. Obozinski, Cut pursuit: fast algorithms to learn piecewise constant functions on general weighted graphs, SIAM J. Imag. Sci. 10 (4) (2017) 1724–1766, https://doi.org/10.1137/17M1113436.

[39] M. Erwig, The graph Voronoi diagram with applications, Netw. Int. J. 36 (3) (2000) 156–163, https://doi.org/10.1002/1097-0037(200010)36:3<156::AID-NET2>3.0.CO;2-L.

[40] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, arXiv (2014) 1724–1734, preprint arXiv: abs/1406.1078, doi: 1406.1078.

[41] L. Du, X. Ye, X. Tan, J. Feng, Z. Xu, E. Ding, S. Wen, Associate-3Ddet: perceptual-to-conceptual association for 3D point cloud object detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 13329–13338, https://doi.org/10.48550/arXiv.2006.04356.

[42] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, Pytorch: an imperative style, high-performance deep learning library, Adv. Neural Inf. Proces. Syst. 32 (2019). https://pytorch.org/.

[43] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv (2014), https://doi.org/10.48550/arXiv.1412.6980 preprint arXiv:1412.6980.

[44] L. Tang, Y. Zhan, Z. Chen, B. Yu, D. Tao, Contrastive boundary learning for point cloud segmentation, arXiv (2022), https://doi.org/10.48550/arXiv.2203.05272 preprint arXiv:2203.05272.