Automation in Construction 173 (2025) 106067

Contents lists available at ScienceDirect

Automation in Construction

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

Instance segmentation of reinforced concrete bridge point clouds with transformers trained exclusively on synthetic data

Asad Ur Rahman[a], Vedhus Hoskere[a,b,*]

[a] Department of Civil and Environmental Engineering, University of Houston, 4226 MLK Blvd, Houston, TX 77204, United States
[b] Department of Electrical and Computer Engineering, University of Houston, 4226 MLK Blvd, Houston, TX 77204, United States

A R T I C L E I N F O

Keywords:
Synthetic point clouds
Reinforced concrete bridges
Instance segmentation
Deep learning
Data augmentation

A B S T R A C T

Bridges in the United States require element-level inspections every 24 months, typically relying on laborious manual assessments. Three-dimensional (3D) point clouds from LiDAR or photogrammetry can facilitate these inspections, but are difficult to leverage without automatically identifying individual structural elements. Existing research focuses on semantic segmentation, which classifies points into broader categories rather than identifying each element instance. A major bottleneck is the difficulty of producing instance-level annotations. To address this gap, the paper proposes and evaluates three synthetic data generation approaches to produce automatically labeled point clouds of bridges with element instance-level annotations. An occlusion technique is introduced to increase realism. The synthetic data is then evaluated for training Mask3D transformer model for instance segmentation of field-collected point clouds, achieving mean Average Precision (mAP) scores of 91.7 % on LiDAR data and 63.8 % on photogrammetry. These results demonstrate the potential to enhance element-level bridge inspections and improve overall infrastructure management.

1. Introduction

Bridges are essential transportation infrastructure that connect communities and enable economic growth, making their continued operation, and thus their continued inspection, a matter of paramount importance [1]. Bridges are constantly exposed to varying vehicular, human, and environmental loading, all of which can lead to deterioration over time. Bridge performance can also deteriorate due to design or construction flaws yielding catastrophic results [2]. To facilitate continued operation, the National Bridge Inspection Standards (NBIS) mandate that all bridges must undergo condition assessment every two years [3]. However, NBIS inspections are typically conducted manually and can be inefficient, laborious, expensive, and dangerous and involve assigning condition ratings to each bridge element.

Researchers have sought to digitize the bridge inspection process through the creation of digital twins that can serve as a reliable surrogate of a physical bridge and be used for decision making. In essence, digital twins are digital replicas of the infrastructure asset [4–8] that contain information about the structure and its condition. A first step in developing a DT is the production of a reality model which is 3D representation of the asset produced using photogrammetry or LiDAR using collected raw data. The reality model is then processed to obtain relevant semantic representations and information relating to structural components and damage. Much research has been devoted to the identification of bridge damage using images [9–13] and so we focus here on the less studied problem of identifying bridge components using point clouds of reality models.

To extract information about bridge components from point clouds, 3D semantic segmentation approaches have been researched. Studies related to semantic segmentation of bridge point clouds can be divided into two categories, (i) heuristic methods [14–18], and (ii) learning based methods [19–25]. Xia et al. [20] proposed a multi-scale local descriptor and machine learning-based approach for semantic segmentation of bridge components. Their method outperformed PointNet on the real-world dataset by Lu et al. [14]. Yang et al. [21] adopted the weighted superpoint graph (WSPG) approach for the semantic segmentation of bridge components. The study claimed superior performance of the WSPG model compared to PointNet, DGNN, and superpoint graph (SPG). Lin et al. [23] proposed a framework for the semantic segmentation of road infrastructure bridge components using a mobile LIDAR mapping system. This methodology, evaluated on 27 interstate highway bridges, employed the spearpoint graph (SPG)

* Corresponding author.
E-mail addresses: aurahman@cougarnet.uh.edu (A.U. Rahman), vhoskere@central.uh.edu (V. Hoskere).

https://doi.org/10.1016/j.autcon.2025.106067
Received 22 August 2024; Received in revised form 11 February 2025; Accepted 12 February 2025
Available online 20 February 2025

approach for semantic segmentation, categorizing bridge and road infrastructure into 9 distinct classes. J. S. Lee et al. [24] utilized a hierarchical Dynamic Graph-based Convolutional Neural Network (DGCNN) for semantic segmentation of railway bridges, reporting improved performance, particularly in the vicinity of tall electric poles near the bridge. Yang et al. [25], introduced a framework for semantic segmentation of bridge point clouds. The authors proposed two synthetic data augmentation techniques and employed a graph structured deep metric learning approach to enhance the weighted spearpoint graph (WSPG) model. Most recent study by Shi et al. [26] developed a method for generating large-scale synthetic 3D point cloud datasets. The approach includes generating random bridge types, simulating camera trajectories, using Structure from Motion (SfM) for 3D reconstruction, and training 3D semantic segmentation models for bridge components segmentation.

Instance segmentation provides additional element level information for each point (e.g., girder 1 has a different label from girder 2) beyond just semantic segmentation where all elements of the same type are treated the same. Access to instance segmentation can aid the creation of digital twins and the automation of bridge element ratings. For example, we illustrate a workflow for digital twin generation that utilizes instance segmentation in Fig. 1. The process includes the development of a reality model, semantic and instance segmentation of structural components, and the identification of damages such as cracks and deflections. While reviewing existing research, we discovered only two studies about instance segmentation of bridge point clouds [18,19]. These studies specifically focus on truss bridges. Lamas D et al. [19] uses deep learning, while Lamas D et al. [18] employs a heuristic method involving Principal component analysis (PCA). There has been no research on instance segmentation of reinforced concrete bridges which constitute the bulk of the bridge type in the world.

There has recently been an explosion of research on developing instance segmentation methods for other applications that could potentially be adopted to improve instance segmentation of bridge points clouds. Methods include top-down proposal-based methods [27–31], bottom-up grouping-based methods [32–36], and voting-based approaches [37–41]. The SoftGroup [38] has two-stage pipeline, benefiting from both proposal-based and grouping-based approaches. In the bottom-up stage, high-quality object proposals are generated through soft semantic score grouping, and in the top-down stage, each proposal is processed to refine positive samples and suppress negative ones. Recently, state of the art transformer-based approaches that outperformed the previous studies include SPFormer [42], Mask3D [43], and OneFormer3D [44]. SPFormer [42] is two-stage approach, grouping potential point features from Sparse 3D U-Net into superpoints. Instances are predicted through a novel query-based transformer decoder. Mask3D [43] uses sparse 3D U-Net based backbone for feature extraction, followed by transformer decoder with separate queries for each instance. The instance queries are learned by iteratively attending to the feature maps at multiple levels. OneFormer3D [44] unifies the semantic, instance and panoptic segmentation of 3D point cloud in a single model. Training models for instance structural components requires a huge amount of labeled point cloud data. Acquiring labeled point cloud for training purpose is very expensive [45]. Therefore, researchers have directed their efforts towards the generation of synthetic point clouds [19,46–49] for various tasks.

This study proposes a novel framework for the instance segmentation of reinforced concrete bridge point clouds using synthetic point clouds and state-of-the art transformer model. Specifically, the three main contributions of the framework are as follows. First, we propose a novel approach for generating synthetic point clouds dataset of reinforced concrete bridges with the aim of facilitating instance segmentation of structural components from point clouds of reality models. Second, we propose a novel sparsity-based occlusion algorithm for data augmentation of bridge point clouds that improves generalizability of models trained on synthetic data. Finally, we carefully evaluate the applicability of Mask3D model for the task of instance segmentation of bridge components by conducting experiments for optimal hyperparameter selection. We also studied the effect of various synthetic dataset types and the effect of occlusion on instance segmentation of structural components of bridges.

The structure of this work is organized as follows. Section 3 proposed methodology, Section 4 discusses about the training and evaluation, Section 5 provides a comprehensive analysis of the results from the experimentation. Finally, Section 6 and Section 7 gives the conclusion and limitations.

2. Methodology

This section outlines the methodology crafted to execute the instance segmentation of RC bridges. Our proposed methodology is illustrated in Fig. 2, consisting of five steps namely (i) synthetic RC bridge point clouds generation, (ii) data pre-processing, (iii) data augmentation strategies, and finally (iv) instance segmentation model, and (v) field data collection conducted to validate the proposed approach (Section 3.4). The real-world data (both LiDAR and photogrammetry point

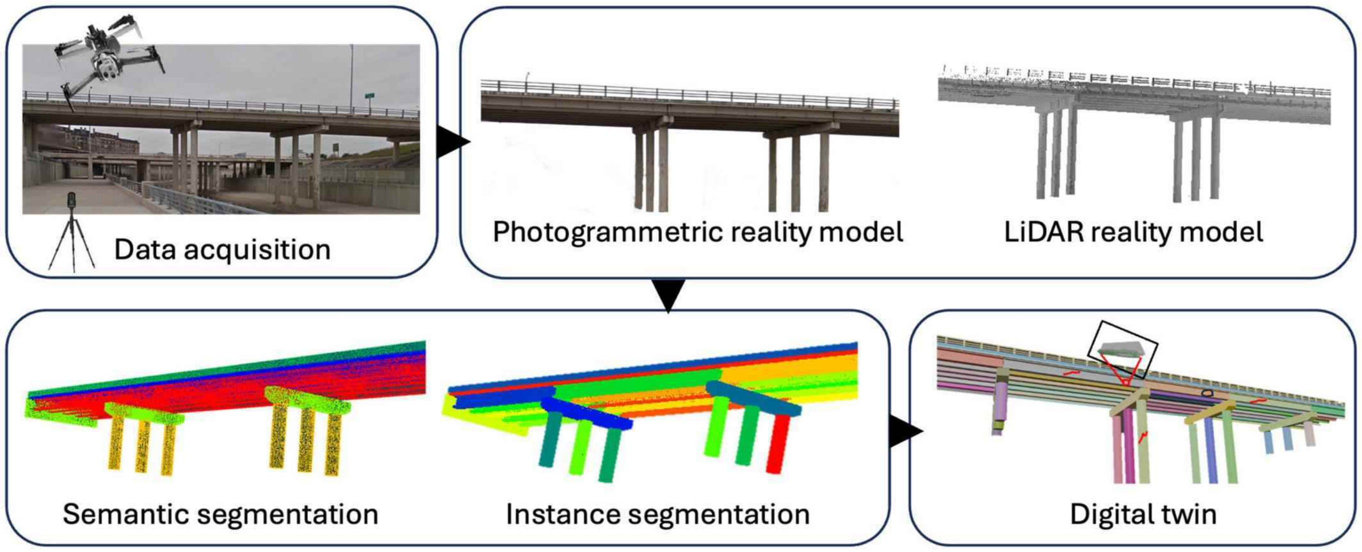

Fig. 1. Workflow for digital twin creation starting with data acquisition through photogrammetry and LiDAR scanning.

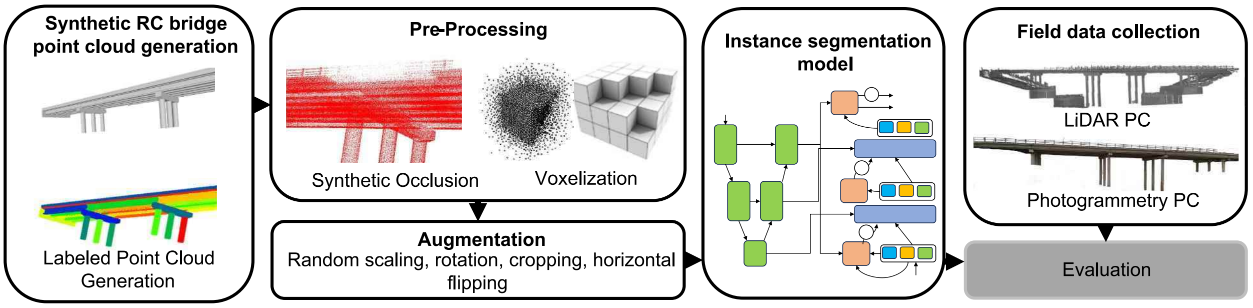
Synthetic RC bridge point cloud generation
Labeled Point Cloud Generation
Pre-Processing
Synthetic Occlusion
Voxelization
Augmentation
Random scaling, rotation, cropping, horizontal flipping
Instance segmentation model
Field data collection
LiDAR PC
Photogrammetry PC
Evaluation

Fig. 2. Diagram illustrates the training and testing pipeline for instance segmentation on bridge point clouds.

clouds) are employed for testing the model's performance.

2.1. Synthetic RC bridge point clouds generation

The lack of labeled RC bridge point cloud dataset necessitates the development of synthetic bridge dataset for training the instance segmentation model. The existing dataset by Lu et al. [14], which only includes point clouds for 10 bridges without labels, is insufficient for deep learning models. A synthetic labeled data featuring diverse bridge designs, is critical for developing models capable of accurately segmenting bridge point clouds.

2.1.1. 3D bridge geometry generation

The initial steps common to creating all three synthetic datasets involve 3D modeling and processing. Using Open bridge designer (OBD), 60 bridges were modeled with randomly defined cross-sections (Fig. 3) for their structural components, employing RC bridge modeling templates for rapid development. These bridge models, focusing on above-ground structural components like slabs, barriers, girders, pier caps, and piers, for instance segmentation purposes, were exported in .obj format. We didn't consider all those hidden parts of the RC bridge which can't be captured with the real Terrestrial Laser Scanner (TLS), such as footings, and piles. Some of the bridge models' examples are illustrated in the Fig. 4. Notably, a few synthetic bridges that were roughly similar to the real test bridge (though not identical in member and cross-sectional dimensions) were included to ensure comprehensive coverage of the real bridge type in the training data distribution. After exporting the bridge model from Open Bridge Modeler (OBM), it undergoes further processing in Blender, where unnecessary or invisible parts, such as wing walls, footings, and piles, are removed to focus on relevant segment classes under study. The model is then centered to the origin, aligned, and its layers are split into separate components, preparing it for the specific requirements of simulation within the IsaacSim Omniverse environment. These processing steps are crucial for ensuring the model's compatibility in subsequent simulation environment.

2.1.2. Point cloud generation approaches

We generate synthetic RC bridge point clouds using two different approaches as shown in the Fig. 5: (i) Mesh Sampled Point cloud (MSP), (ii) Simulated LiDAR Point Cloud (RSLP).

2.1.2.1. Mesh sampled point cloud (MSP). Mesh Sampled Point Cloud (MSP) dataset was generated by sampling points on the 3D meshes of each bridge component using CloudCompare. During preprocessing, the mesh was segmented into distinct structural components based on semantic and instance classifications utilizing the Blender Python library.

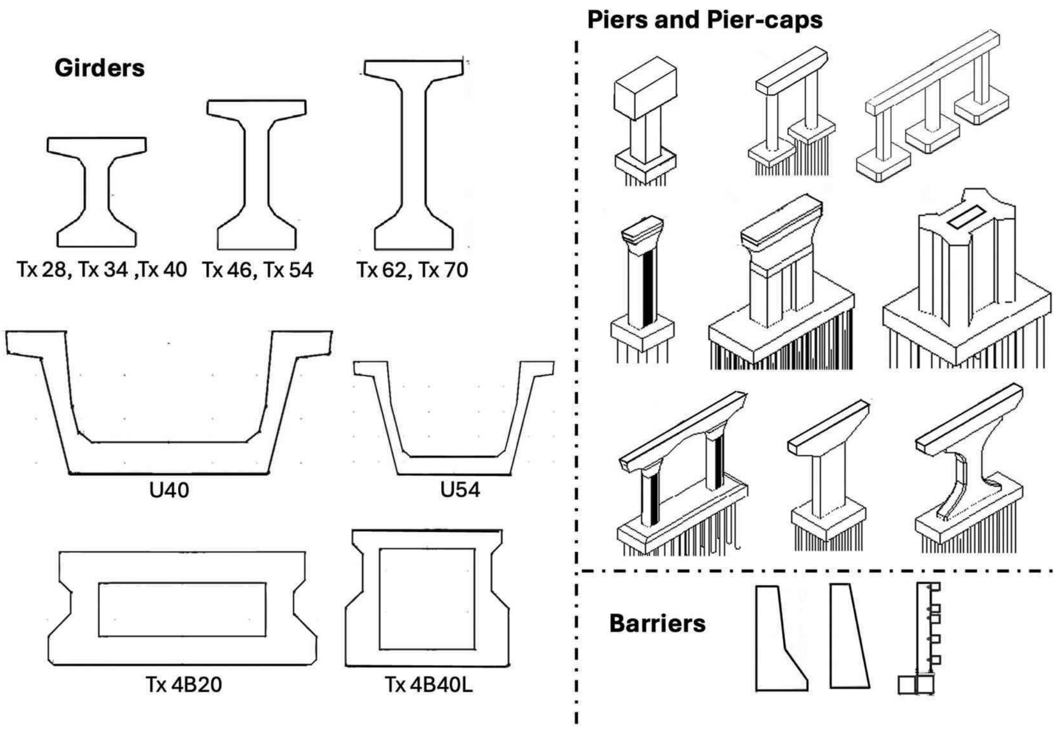

Fig. 3. Cross-sections of bridge elements used to create bridge models.

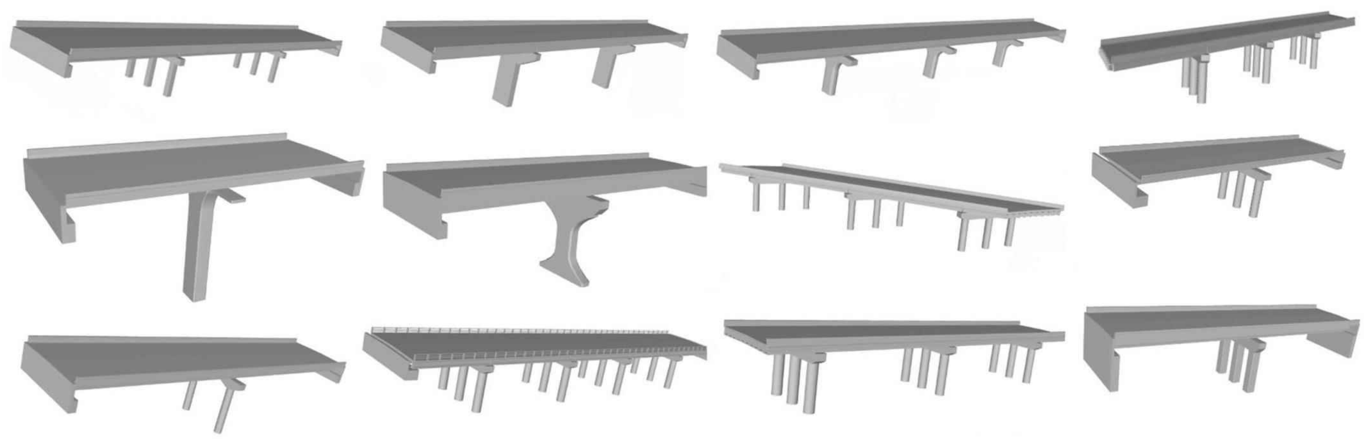

Fig. 4. Structural configurations of synthetic bridge geometries modeled in OpenBridge designer.

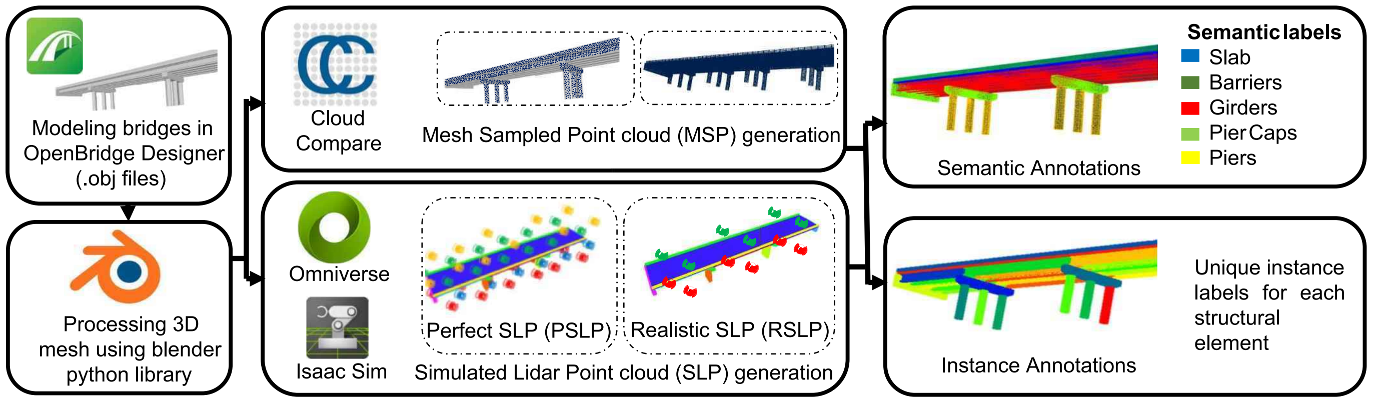
Modeling bridges in OpenBridge Designer (.obj files)
Processing 3D mesh using blender python library
Cloud Compare
Mesh Sampled Point cloud (MSP) generation
Omniverse
Isaac Sim
Perfect SLP (PSLP)
Realistic SLP (RSLP)
Simulated Lidar Point cloud (SLP) generation
Semantic labels
Slab
Barriers
Girders
Pier Caps
Piers
Semantic Annotations
Unique instance labels for each structural element
Instance Annotations

Fig. 5. Illustration of different synthetic bridge point clouds generation processes: Mesh Sampled Point Cloud (MSP), Realistic Simulated LiDAR Point Cloud (RSLP), and Perfectly Simulated LiDAR Point Cloud (PSLP).

Point clouds for each component were created through density-based sampling on the meshes. Subsequently, these point clouds were labeled according to their respective components and merged into a comprehensive single bridge point cloud. This process was fully automated across all bridges. Unlike the field-collected LiDAR point cloud, which can only capture external surfaces visible to scanner and leave occluded areas like the interior of hollow sections and contact points undetected, mesh sampled point cloud from mesh models using cloud compare include detailed internal geometries (Fig. 6).

2.1.2.2. Simulated LiDAR point cloud (SLP). To address the differences between mesh sampled point clouds and those obtained from real-world LiDAR, a simulated environment was established using IsaacSim Omniverse. The generated bridge models, after modeling in the Bentley OpenBridge modeler and processing in the Blender software, were imported into the simulation environment in .obj format. The LiDAR sensors were strategically placed around the bridge to enhance the point cloud coverage (Fig. 5). These sensors were configured with a minimum and maximum range of 0 m to 600 m, and both the horizontal and vertical fields of view were set at 360° and 180°, respectively, with resolutions of 0.4°. Using the LiDAR Python API, semantic labels were applied to the point clouds, correlating with object layers for structural components such as slabs, barriers, girders, pier caps, and piers. Each structural component of the bridge was also given a unique instance label. Due to the absence of texture or color information in the bridge models, all points were assigned RGB values of 255 (white), representing a uniform coloration. The comprehensive point cloud data was ultimately saved in a .txt file format, containing fields for coordinates (X, Y,

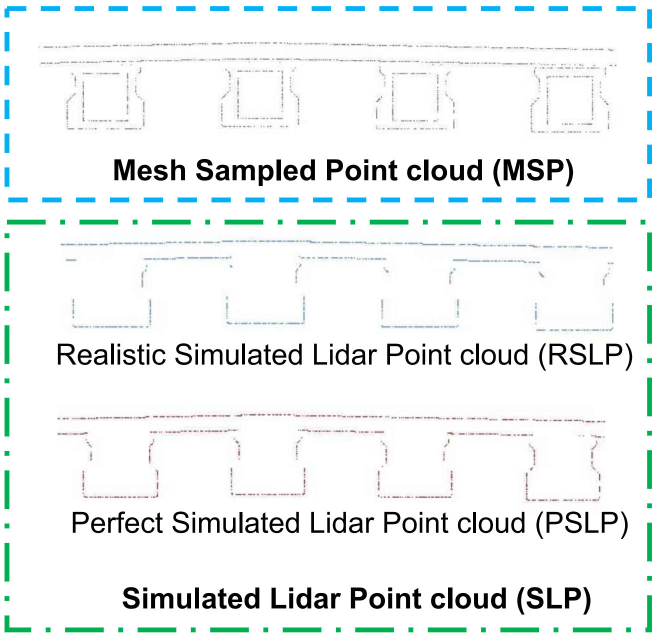
Mesh Sampled Point cloud (MSP)
Realistic Simulated Lidar Point cloud (RSLP)
Perfect Simulated Lidar Point cloud (PSLP)
Simulated Lidar Point cloud (SLP)

Fig. 6. Cross-sections of synthetic point clouds illustrating Mesh Sampled (MSP) with interior detail, Perfectly Simulated LiDAR (PSLP) capturing detailed point cloud, and Realistically Simulated LiDAR (RSLP) mirroring real-world LiDAR capture.

Z), color values (R, G, B), and both semantic (sem) and instance (inst) labels. The synthetic bridges were simulated in the environment with different sensor configurations, resulting in two different datasets: the Realistic Simulated LiDAR Point Cloud (RSLP) and the Perfect Simulated LiDAR Point Cloud (PSLP).

2.1.2.3. Realistic simulated LiDAR point cloud (RSLP). The LiDAR sensors were initially placed in a realistic and practical manner, considering operator accessibility in the field (Fig. 5). Consequently, 12 LiDAR sensors, six above the bridge deck and six at ground levelwere strategically placed to enhance point cloud coverage. It was observed that the LiDAR-generated point cloud omitted some occluded areas, such as those between closely placed girders, highlighting the inherent limitations of LiDAR simulations in capturing complex structural details (Fig. 6).

2.1.2.4. Perfect simulated LiDAR point cloud (PSLP). To address the limitation of missing critical parts with conventional LiDAR setups, we designed a more comprehensive sensor arrangement that, while impractical for real-world applications, ensures no part of the bridge, including occluded areas, is overlooked. This time the LiDAR sensors were deployed in a dense 3D grid with four levels, two above and two below the bridge—tailored to the bridge's size. This configuration varied from 4 to 6 rows and 8 to 12 columns per level, achieving thorough coverage across different bridge dimensions as shown in Fig. 5.

2.2. Data pre-processing

The generated synthetic point cloud data undergoes pre-processing steps that include introducing synthetic occlusion and voxelization of the bridge point cloud. Occlusion is essential for enhancing LiDAR simulation datasets, mimicking real-world scanning challenges like obstructions from bridge components or vegetation. This is achieved by introducing geometric shapes such as cubes, spheres, rectangular prisms, and triangular prisms inside the bridge point cloud. These shapes are positioned randomly throughout the bridge point cloud, with variable sizes to simulate different types of occlusions, as illustrated in the Fig. 7.

Instead of completely removing the occluded points inside these geometric shapes, the points are made sparser using a predetermined sparsity factor, which more accurately reflects realistic LiDAR data capture from multiple angles where some occluded points are still partially visible. This results in a more representative simulation of occlusions that occur due to obstructing objects like bridge supports or surrounding vegetation. Occlusion in the real and synthetic bridge are shown in Fig. 8.

The real bridge point cloud exhibits non-uniform density, introducing unnecessary patterns, and increasing computational costs for training the segmentation model. To address this issue, voxelization is employed, which represents objects within a regular grid of voxels, effectively normalizing the data density [50]. By adopting a voxel size of 2 cm, we were able to down-sample the point clouds, achieving a uniform density across the datasets.

2.3. Instance segmentation model

We employed the Mask3D instance segmentation model proposed by Jonas Schult et al. [43], currently recognized as the leading architecture across multiple benchmark datasets, including STPLS3D [51], ScanNet (v2) [52], and S3DIS. The preprocessing involved cropping the point cloud into equal-sized parallelepiped blocks to manage data input size. The colored point cloud data, initially PxRnx6, was then down sampled into voxels (VxR$_m$x6). The main components of the Mask3D as shown in Fig. 9 are now described.

2.3.1. Sparse feature backbone

Mask3D uses a Sparse Feature Backbone built on the MinkowskiEngine-based symmetric U-Net architecture [53]. This backbone efficiently extracts features from the input point cloud by leveraging sparse convolutional networks. The backbone outputs multi-scale point features that are essential for the subsequent query refinement.

2.3.2. Transformer decoder

Mask3D incorporates a Transformer decoder to refine instance queries through multiple layers of cross-attention with point features extracted from the backbone. Each instance query represents a potential object instance, and the transformer decoder iteratively refines these queries by attending to features at multiple scales. This allows Mask3D to predict instance masks in parallel, capturing both local and global contexts necessary for precise segmentation. The multi-scale property is particularly amenable for bridge component instance segmentation because component identification requires global context but segmentation requires precise action at the interfaces between components that can only be observed at finer scales. Additionally, unlike voting-based methods that rely on geometric priors, the transformer approach in Mask3D is particularly convenient as it is domain-agnostic and can be applied without manually tuning parameters such as object centers or radii. Two important parts of the transformer decoder are the mask modeul (MM) and query refinement.

2.3.2.1. Mask Module (MM). The mask module is responsible for generating binary masks for each instance query. Instance features are combined with point feature maps through a dot product, yielding a similarity score for each point, which is then converted to a binary mask using sigmoid activation and a threshold. This effectively segments individual instances and assigns them semantic labels. The mask generation process also incorporates a hierarchical refinement, where features are progressively refined using attention mechanisms until a final instance mask is produced.

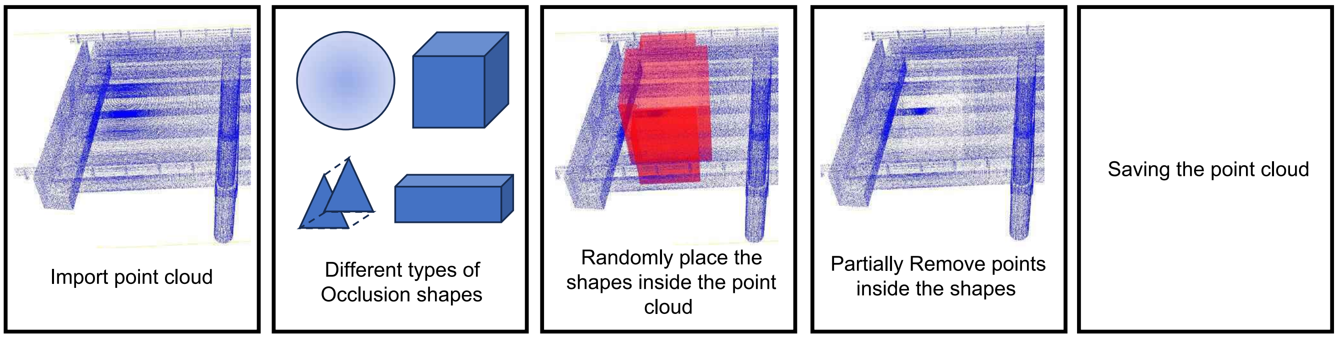
Import point cloud
Different types of Occlusion shapes
Randomly place the shapes inside the point cloud
Partially Remove points inside the shapes
Saving the point cloud

Fig. 7. Illustration of the step-by-step process of the occlusion algorithm.

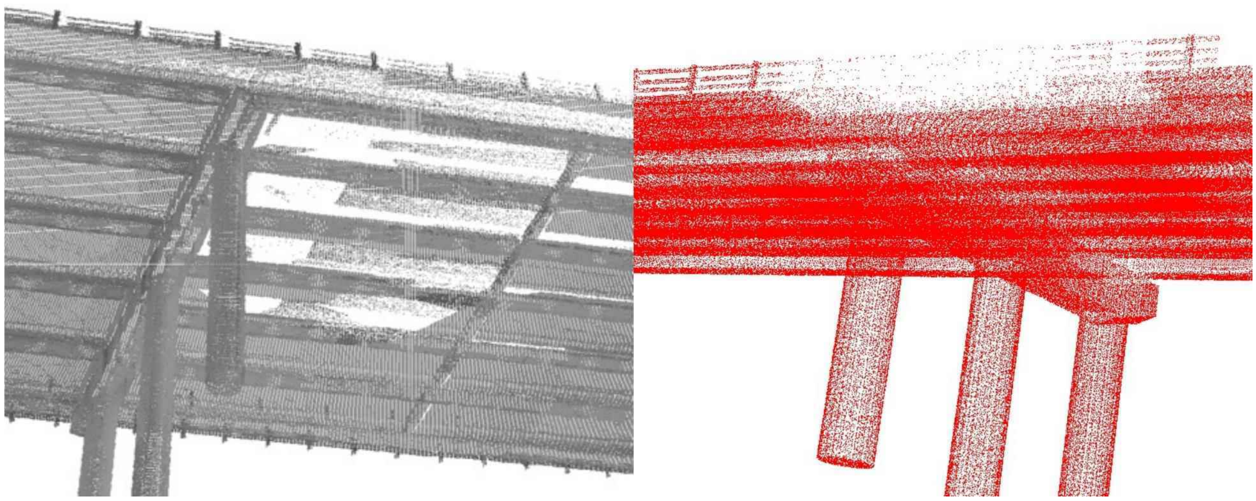

Fig. 8. Observed occlusion in field collected bridge point cloud (gray) and result from applying proposed occlusion approach to synthetic point cloud (red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

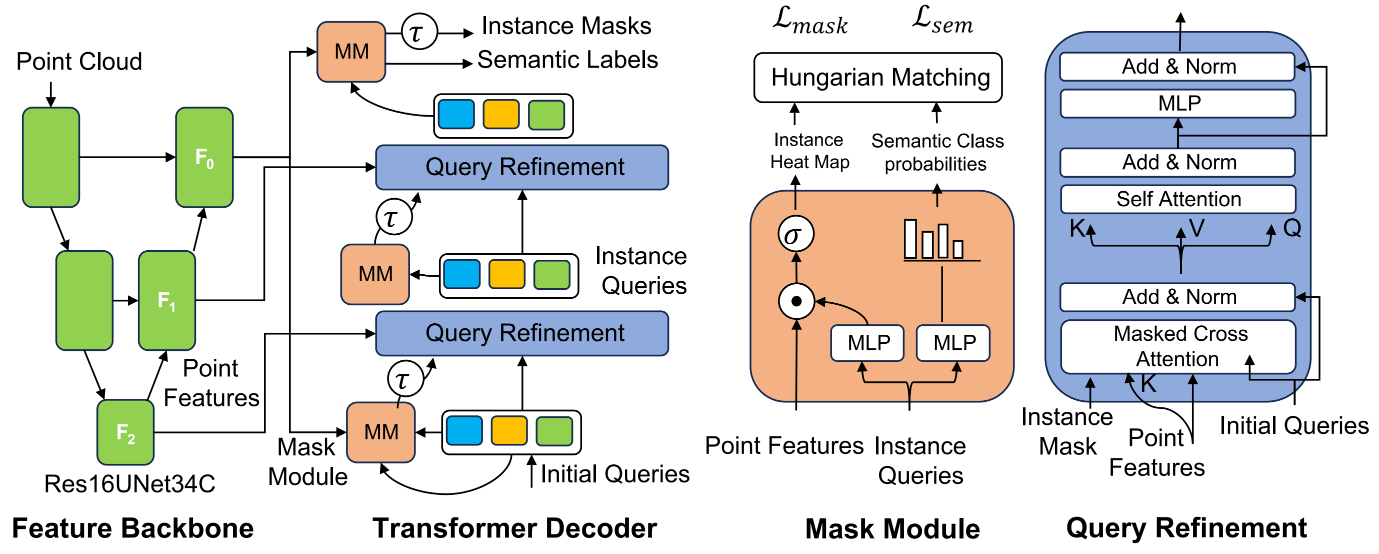

Fig. 9. Mask3D architecture of point cloud segmentation network with a Res16UNet34C backbone, Transformer decoder for instance and semantic segmentation, and query refinement stages for enhanced feature extraction and classification.

2.3.2.2. Query refinement. The instance queries in Mask3D are iteratively refined through a multi-step process to improve their representation and ensure accurate segmentation. Initially, each query is represented by a set of features, and the refinement process is carried out through a stack of Transformer decoder layers. At each layer, the queries cross-attend to voxel features from different levels of the sparse feature backbone. This attention mechanism allows the queries to gather relevant spatial and contextual information from different scales of the point cloud. Additionally, self-attention is employed between the instance queries to prevent multiple queries from latching onto the same object instance, thus avoiding duplicate segmentation. This process is repeated over several layers, progressively refining each query until it accurately represents an object instance in the point cloud.

2.3.2.3. Bipartite graph matching. To establish correspondences between predicted and ground truth instances, bipartite graph matching is employed. The cost matrix C is constructed as given by Eq. (1).

$$C(k,\hat{k}) = \lambda_{dice}\mathscr{L}_{dice}(k,\hat{k}) + \lambda_{BCE}\mathscr{L}_{BCEmask}(k,\hat{k}) + \lambda_{cl}\mathscr{L}^{l}_{CE}(k,\hat{k}) \quad (1)$$

Here, $\mathscr{L}_{dice}$is the Dice loss, $\mathscr{L}_{BCEmask}$ is the binary cross-entropy loss over the foreground and background of the mask, and $\mathscr{L}^{l}_{CE}$ is the multi-class cross-entropy loss for classification. Dice loss was specifically used to address the data imbalance issue in the dataset, where the number of foreground points is significantly lower compared to the background. The Dice loss helps optimize the overlap between predicted and true regions, ensuring better performance in segmenting smaller components. The weights are set to $\lambda_{dice} = \lambda_{cl} = 2.0$ and $\lambda_{BCE} = 5.0$.

After establishing the correspondence using the Hungarian matching the model optimizes the predicted mask with the following Eq. (2).

$$\mathscr{L}_{mask} = \lambda_{BCE}\mathscr{L}_{BCE} + \lambda_{dice}\mathscr{L}_{dice} \quad (2)$$

The overall loss for all auxiliary instance predictions is defined in Eq. (3)

$$\mathscr{L} = \sum_{l}^{L}\mathscr{L}^{l}_{mask} + \lambda_{cl}\mathscr{L}^{l}_{CE} \quad (3)$$

2.3.3. Use of dice loss

The Dice loss was used to address the class imbalance present in the dataset, where smaller components of the bridge, such as piers or cables, are underrepresented compared to larger structures like the deck. The Dice loss is effective in handling this imbalance by optimizing for overlap between predicted and true regions, ensuring accurate segmentation of smaller, yet crucial, bridge components. This enhances the quality of segmentation, particularly for components with fewer points.

2.3.4. Configuration for instance segmentation of RC bridges

We evaluated different configurations of the Mask3D model to identify the most suitable approach for bridge component instance segmentation. The Mask3D model presents different encoder backbones. We found the sparse Res16UNet34C to perform the best for our model and have used that for all our experiments. Additionally, we found the model's performance sensitive to hyper-parameter selection, namely the voxel size and DBSCAN parameters. Understanding the impact of voxel size on model performance is critical due to its influence on resolution and computational demands [54]. In our study, point cloud data was down sampled into voxels, where smaller sizes increase resolution but also computational costs. Additionally, studying the sensitivity of DBSCAN parameters, epsilon (ε) and minimum number of points (MinPts), to refine the segmentation by splitting merged instances into distinct masks. We conduct extensive hyperparameter tuning and show the results in the Results and Discussion section. The Mask3D code can be accessed using the GitHub link https://github.com/JonasSchult/Mask3D.

2.4. Field data collection

We evaluated the deep instance segmentation model using real bridge point cloud data obtained from Terrestrial Laser Scanning (TLS) and photogrammetry. The data were collected from two bridges: one in Houston and one in Austin. For the Houston bridge, a service road bridge over Brays Bayou located at coordinates 29°42′44.4“ N, 95°22'39.8” W, we acquired both LiDAR and photogrammetry point clouds. For the Austin bridge, a highway bridge at coordinates 30°9′31.6686“ N, 97°46'33.978” W, we captured only the LiDAR point cloud.

2.4.1. Point cloud acquisition using terrestrial laser scanner (TLS)

For both the real bridge point cloud data acquisition using Terrestrial Laser Scanner (TLS) we used the RIEGL VZ-400 laser scanning system. The scanner settings included a horizontal field-of-view of 360° and a vertical field-of-view of 100° (ranging from −40° to 60°), with an angular scanning resolution of 0.015°. It operated at a scanning frequency of 1010 kHz, capable of measuring distances up to 580 m with a measurement precision of 2 mm at 100 m. For accurate georeferencing, GNSS technology, supplemented by three reflectors, facilitated the automated alignment and registration of point clouds from various scanner positions. The scans were conducted from six terrestrial laser scanning (TLS) stations beneath the bridge and four above to capture the entire structure comprehensively. The final aligned and georeferenced point cloud was recorded in the Universal Transverse Mercator (UTM) global coordinate system, totaling 2,475,529 points for the Houston Bridge and 9,591,133 for the Austin bridge.

2.4.2. Point cloud acquisition using photogrammetry

For the photogrammetric point cloud, the images data was captured by a Skydio 2+ UAV. This process involved taking 7068 images of the bridge, maintaining a 76 % side lap and overlap, and a consistent 5 ft. distance from the surface. Various scanning techniques were utilized: the upper part of the bridge was documented with a 2D downward scan, the lower deck with a 2D upward scan at gimbal angles of 80 and 60 degrees for complete girder visibility, and the piers and pier caps with 3D capture, while keyframe manual mode was used for the front and back. The collected data were then reconstructed into a detailed 3D model using the WebODM software.

The annotation of the bridge point cloud, captured with Terrestrial Laser Scanning (TLS), was carefully performed using the CloudCompare. Each structural component within the point cloud was labeled with corresponding semantic and instance classes. We defined five semantic classes encompassing slabs, barriers, pier caps, piers, and girders. Additionally, unique instance labels were assigned to every individual component for instance annotation as shown in the Fig. 10. These thoroughly detailed annotations served as the ground truth for evaluating our model, providing a reliable benchmark for assessing segmentation accuracy.

2.5. Synthetic dataset feature similarity

To evaluate the similarity between synthetic and the Houston real bridge point cloud and also the diversity of features across synthetic bridges in the dataset, we implemented a point cloud comparison pipeline described in Fig. 11. The process starts with loading the point cloud data for both real and synthetic bridge. To standardize the data, each point cloud was voxelized with a voxel size of 0.5 units, followed by normalization to ensure consistent spatial positioning and scale. Utilizing a PointNet model pretrained on S3DIS [55] for semantic segmentation, we extracted feature vectors from the last layer of the encoder of size 1088 × 1 for each point cloud. Finally, we compute the cosine similarity between the synthetic and real bridge feature vectors. The feature similarity scores reflect the global and local geometric and structural resemblance.

3. Training and evaluation

The model underwent training using synthetic data and was subsequently evaluated on real data to assess its performance. For hyperparameter tuning, out of the total 60 synthetic bridges generated, 52 were allocated for the training set while the remaining 8 were designated for validation. These validation bridges were carefully selected to represent the entire spectrum of the data distribution. To maintain consistency in our results, the same set of validation bridges was used across all hyperparameter tunning experiments. This approach allowed for consistent comparative analysis of the model's performance under various conditions.

In subsequent experiments to evaluate the “effect of synthetic dataset type on segmentation” and the “effect of occlusion,” the point clouds generated from the same set of 52 synthetic bridges (using different techniques) were consistently used for training. For validation, the field-collected LiDAR and photogrammetry point clouds were split into two halves; one half was used for validation, and the other half was reserved for testing.

The training process begins with data pre-processing and augmentation. Initially, the dataset underwent preprocessing using the occlusion method proposed. Subsequently, we employ standard augmentation techniques such as random scaling, random rotation, cropping, and horizontal flipping to address discrepancies between synthetic and real-world data. Our augmentation pipeline includes scaling within ±10 % and rotation around the three principal axes, with limits set to full 360 degrees for the z-axis and smaller ranges for the x and y axes to mimic realistic tilting. The bridge point clouds are cropped into cuboids, further diversifying the data by providing variations in scale and altering spatial contexts.

Training was conducted for 70 epochs with an initial learning rate of 10^{-4}. The total number of epochs was determined based on the stabilization of the mean loss curve, as shown in Fig. 12. Extending the training duration beyond this point did not yield better results. We employed a voxel size of 0.2 m and utilized an NVIDIA GeForce RTX 3090, which resulted in training times ranging from approximately 24 to 36 h. The AdamW optimizer [56], along with a one-cycle learning rate schedule [57], was implemented to enhance optimization. To foster model robustness and generalization, we incorporated random occlusion in both training and validation datasets, in addition to our standard augmentation techniques.

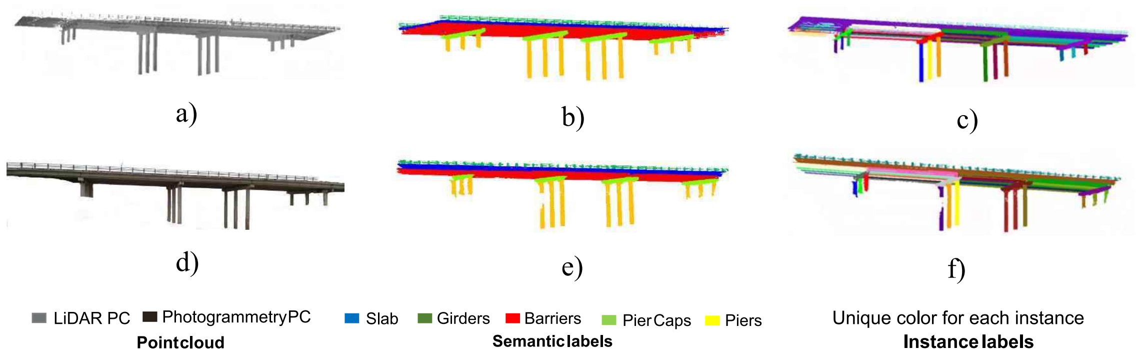

Fig. 10. Field-collected point cloud and ground truth instance labels of Houston bridge.

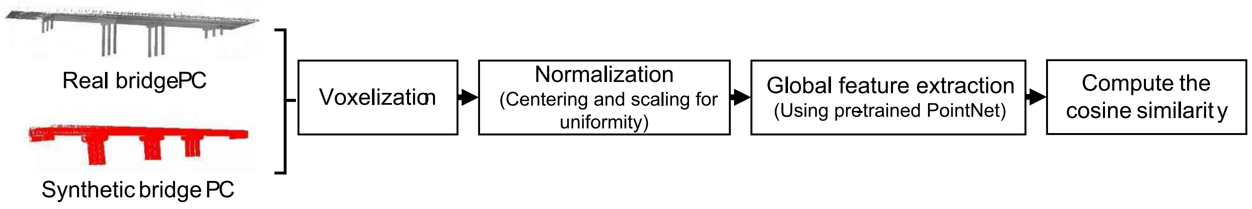
Real bridge PC
Synthetic bridge PC
Voxelization
Normalization (Centering and scaling for uniformity)
Global feature extraction (Using pretrained PointNet)
Compute the cosine similarity

Fig. 11. Steps for implementing the point cloud comparison pipeline.

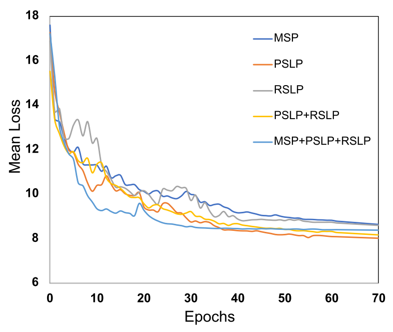

Fig. 12. Showing the training mean loss on the y-axis and epochs on the x-axis. All models converge prior to reaching 70 epochs and we choose that as our epoch number to terminate training uniformly for all models.

The performance of the model trained on synthetic data was evaluated on field-collected LiDAR and photogrammetry point cloud using Average Precision (AP) metrics, specifically mAP, mAP$_{25}$, and mAP$_{50}$. For this evaluation, the model weights corresponding to the peak validation mAP were used to ensure the best possible performance assessment.

$$Precision = \frac{tp}{tp + fp}$$

$tp = true\ positive$

$fp = false\ positive$

AP: Average precision (AP) calculates the mean precision across all the IoU thresholds except 0.25. this average includes IoU thresholds like 0.5, 0.55, 0.6, etc.

AP25: Calculates the mean AP specifically at 25 % IoU threshold.

AP50: Calculates the mean AP specifically at 50 % IoU threshold.

4. Results and discussion

This section presents the results of hyperparameter tuning and examines the impact of different synthetic dataset types on the instance segmentation of bridge point clouds. Unlike synthetic bridge point clouds, which accurately represent bridge geometry, real bridges exhibit occlusions that can affect segmentation performance. To evaluate the impact of occlusion, we conducted experiments introducing occlusion to all synthetic data types. These experiments aimed to assess how occlusion influences the model's segmentation performance.

4.1. Hyperparameter tuning

Hyperparameter tuning played an important role in optimizing our model's performance. We carefully selected a validation set from our synthetic dataset, comprising a fixed size of 8 bridges out of 60 PSLP synthetic dataset to ensure coverage across the entire data distribution. No cross-validation was used; instead, we consistently applied the same validation set to evaluate various hyperparameter adjustments. Key hyperparameters that underwent tuning included voxel size, the epsilon parameter for DBSCAN clustering (dbscan eps), the minimum number of points required to form a cluster in DBSCAN (dbscan min no of points), and variations in the coloration of the point cloud, ranging from uniform white to random and varying colors along the height. Notably, during the tuning phase, occlusion was deliberately not applied to isolate and better understand the effects of other hyperparameter adjustments on model performance.

4.1.1. Voxel size

We tested voxel sizes from 0.1 m to 0.3 m and found that while a 0.1 m size did not improve model precision and caused memory issues, a 0.3 m size led to significant loss in semantic detail. As illustrated in Table 1 the optimal balance was achieved with a 0.2 m voxel size, providing the highest mean AP at 74.2 %, efficiently balancing computational efficiency with sufficient resolution to capture bridge structures accurately.

Table 1
Impact of voxel size on instance segmentation precision.

Voxel Size	*mAP*	*mAP$_{50}$*	*mAP$_{25}$*
0.1	0.713	0.768	0.782
0.2	**0.74**	**0.802**	**0.811**
0.3	0.574	0.748	0.759

This makes a 0.2 m voxel size ideal for our application, as demonstrated by the AP variations across sizes.

4.1.2. DBSCAN parameters

Our sensitivity analysis involved adjusting ε values from 0.5 to 10 and MinPts from 1 to 4, revealing optimal performance at an epsilon (ε) of 0.92 and MinPts of 4. These settings yielded the highest precision metrics (mAP 0.742, AP_{50} 0.8, AP_{25} 0.811) as shown in the Table 2 and Table 3, indicating that a higher MinPts threshold helps form more defined and conservative clusters, thus improving accuracy. During experiments for epsilon (ε), the default MinPts value of 4 was used. Once the optimal epsilon was determined, $\varepsilon = 0.92$ was used for subsequent experiments to find the optimal MinPts.

4.1.3. Color of the point cloud

We evaluated the effect of different color schemes on the performance of the Mask3D model, which segments point clouds based on color and geometric features. The experiment compared three schemes: uniform white, random RGB, and varying colors along the height (*Z*-axis). Initially, point clouds had no color, making this exploration crucial for understanding how color impacts segmentation. Uniform white was used as the baseline, where performance was standard. Introducing random RGB colors slightly reduced the model's accuracy (mAP), likely due to the challenges in segment differentiation, results presented in Table 4. Conversely, applying varying colors along the height slightly improved mAP, providing additional contextual clues that enhanced segmentation.

These findings shown in the Table 4, underscore the importance of color in training synthetic point cloud datasets, with strategic coloring along the Z-axis proving beneficial for improved segmentation accuracy.

The models inference on validation data after optimizing the hyperparameters of the 3D point cloud instance segmentation model are presented in the Fig. 13.

4.2. Effect of synthetic dataset type on segmentation

The instance segmentation performance of the deep learning model trained on various synthetic datasets and tested on real-world bridge LiDAR point cloud shows significant variance depending on the type of synthetic data used, as detailed in Table 5. As outlined in the training and evaluation section, half of the field-collected LiDAR point cloud data was used for validation and the other half for testing. When trained solely on Mesh Sampled Point Cloud (MSP), the model displayed notably low performance when tested on LiDAR point cloud (mAP of 0.010, mAP50 of 0.016, and mAP25 of 0.067). The poor results can be attributed to MSP's significant deviations from real-world data, including differences in the shape of structural components and the inclusion of points sampled from the interiors of structural members, features not typically captured by real-world LiDAR.

In contrast, training with Perfect Simulated LiDAR Point Cloud (PSLP) and Realistic Simulated LiDAR Point Cloud (RSLP) individually showed considerable improvements. PSLP, which simulates a dense grid of LiDAR sensors to avoid occlusions and captures nearly every part of the bridge, achieved a mAP of 0.854. RSLP, being the closest to real-world point clouds, further improved model performance, achieving a mAP of 0.873. The combination of PSLP and RSLP yielded the highest performance (mAP of 0.910, mAP50 of 0.996, mAP25 of 0.996) by providing comprehensive learning on both exact component geometry and realistic LiDAR sensor placement.

After evaluating the model's performance with the field-collected LiDAR bridge point cloud, we tested the same models with a field-collected photogrammetry bridge point cloud. The average precision values were lower compared to those for the LiDAR point cloud, as shown in Table 6. This discrepancy is attributed to the fundamentally different nature of the photogrammetry point cloud, which is less accurate and exhibits more irregularities than the LiDAR point cloud [58]. These irregularities make inference challenging for the model, which was trained on synthetic bridge point clouds that are more precise and better represent the bridge geometry. A similar trend was observed for the photogrammetry bridge point cloud, with the best performance achieved using the PSLP+RSLP dataset (mAP of 0.638, mAP50 of 0.903, mAP25 of 0.903).

Table 2
Impact of varying DBSCAN epsilon values on model performance across three precision metrics.

DBSCAN eps (ε)	*mAP*	*mAP_{50}*	*mAP_{25}*
0.5	0.711	0.76	0.781
0.92	**0.742**	**0.802**	**0.811**
2	0.738	0.78	0.8
4	0.724	0.771	0.79
10	0.736	0.78	0.802

Table 3
Influence of minimum number of points (MinPts) in DBSCAN clustering on the segmentation.

DBSCAN MinPts	*mAP*	*mAP_{50}*	*mAP_{25}*
1	0.64	0.732	0.778
2	0.729	0.751	0.797
3	0.731	0.786	0.795
4	**0.742**	**0.802**	**0.811**

Table 4
Impact of color variation on the performance of the model, with varying colors along the height showing slightly favorable results.

Point cloud color	*mAP*	*mAP_{50}*	*mAP_{25}*
White	0.732	0.798	0.811
Random RGB	0.71	0.782	0.801
Varying color along the height	**0.741**	**0.807**	**0.815**

4.3. Effect of occlusion

To test the effect of occlusion, a variation of the original training dataset was created with occluded samples at different sparsity factors. For instance, in the PSLP+occN% scenarios, the training data included 52 original bridges without occlusion and an additional 52 bridges with occlusion, resulting in a total of 104 synthetic bridge point clouds.

The model is first trained on the PSLP datasets with occlusion at various sparsity factors and tested against the LiDAR point cloud, as shown in the Table 7. The results indicate that optimal performance is not achieved by removing all points within the occlusion geometry, as different parts of the object may be visible from various stations. Instead, the optimum occlusion sparsity rate should be identified, or the training data should be processed with various sparsity factors. It was observed that for the PSLP a 60 % sparsity factor yielded the highest performance (mAP 0.920, mAP50 0.989, and mAP25 0.991) followed by decline at 80 % sparsity, indicating the importance of optimum sparsity rate of occlusion in training data.

After optimizing the sparsity factor, the optimum rate of 60 % was used for all the preceding training datasets, and the model's performance was evaluated using field-collected LiDAR (Table 8) and photogrammetry (Table 9) point clouds. The results, as shown in Table 8, indicate that the PSLP+RSLP+Occ60 % dataset yielded the highest (mAP 0.917, mAP50 0.998, and mAP25 0.998) values for the LiDAR point cloud, due to the introduction of occlusion.

In contrast, the model's performance declined when occlusion was added and evaluated with the photogrammetry point cloud. This decrease can be attributed to the fundamentally different nature of photogrammetry point clouds. Unlike LiDAR point clouds,

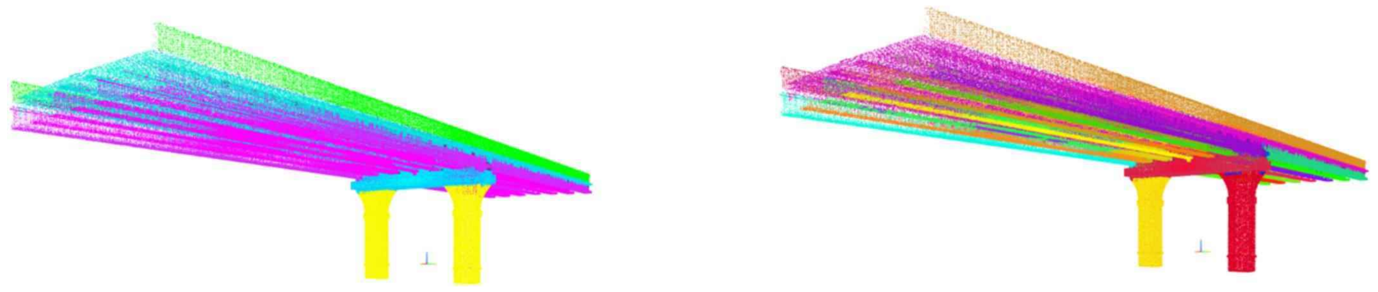

Fig. 13. Results of semantic (left) and instance (right) segmentation on a synthetic bridge point cloud from validation data, following hyper-parameter tuning.

Table 5
Comparative performance of deep learning model trained on different synthetic datasets and tested on real-world bridge point clouds.

Exp #	*Training data type*	*Test data*	*mAP*	*mAP$_{50}$*	*mAP$_{25}$*
1	MSP	LiDAR PC	0.010	0.016	0.067
2	PSLP	LiDAR PC	0.854	0.963	0.977
3	RSLP	LiDAR PC	0.873	0.974	0.981
4	PSLP+RSLP	LiDAR PC	**0.910**	**0.996**	**0.996**

Table 6
Comparative performance of deep learning model trained on different synthetic datasets and tested with photogrammetric bridge point clouds.

Exp #	*Trained with*	*Test*	*mAP*	*mAP$_{50}$*	*mAP$_{25}$*
1	MSP	Photogrammetry	0.001	0.005	0.10
2	PSLP	Photogrammetry	0.431	0.615	0.746
3	RSLP	Photogrammetry	0.595	0.890	**0.904**
4	PSLP+RSLP	Photogrammetry	**0.638**	**0.903**	0.903

Table 7
Comparative performance of deep learning model trained with PSLP data with various occlusion sparsity factors and tested with the field-collected bridge point clouds from LiDAR.

Sparsity factor	*Test*	*mAP*	*mAP$_{50}$*	*mAP$_{25}$*
20 %	LiDAR PC	0.804	0.956	0.979
40 %	LiDAR PC	0.664	0.889	0.924
60 %	LiDAR PC	**0.920**	**0.989**	**0.991**
80 %	LiDAR PC	0.763	0.822	0.955

Table 8
Comparative performance of deep learning model trained on different synthetic datasets with 60 % occlusion sparsity rate and tested on bridge point clouds from LiDAR.

Exp #	Trained with	Test	*mAP*	*mAP$_{50}$*	*mAP$_{25}$*
1	MSP+Occ60 %	LiDAR PC	0.01	0.017	0.07
2	PSLP+Occ60 %	LiDAR PC	0.916	0.994	0.996
3	RSLP+Occ60 %	LiDAR PC	0.824	0.911	0.919
4	PSLP+RSLP+Occ60 %	LiDAR PC	**0.917**	**0.998**	**0.998**

Table 9
Performance of deep learning model trained on different synthetic datasets with 60 % occlusion sparsity rate and tested on bridge point clouds from photogrammetry.

Exp #	Trained with	Test data	*mAP*	*mAP$_{50}$*	*mAP$_{25}$*
1	MSP+Occ60 %	Photogrammetry	0.004	0.07	0.13
2	PSLP+Occ60 %	Photogrammetry	**0.496**	**0.826**	0.843
3	RSLP+Occ60 %	Photogrammetry	0.458	0.603	0.697
4	PSLP+RSLP+Occ60 %	Photogrammetry	0.475	0.763	**0.928**

photogrammetry point clouds have very little occlusion because the image data was collected extensively, allowing for a complete reconstruction of the bridge without missing any occluded areas. Therefore, adding occlusion to the training data did not positively impact the model's inference for photogrammetry point cloud.

4.4. *Feature similarity*

The similarity analysis shows the individual and mean cosine similarity scores between the synthetic bridge datasets and the Houston bridge. We analyze the top two generation approaches, namely the PSLP and the RSLP methods with 60 % occlusion and find that the PSLP point clouds are consistently more similar to the Houston point cloud. This observation correlates directly with the performance of the model on the field-collected data shown in Table *7* demonstrating the PSLP+Occ60 % achieving mAP 0.916, compared to RSLP+Occ60 % with mAP 0.824.

Additionally, we observed synthetic bridges exhibiting structural features similar to the Houston bridge, such as length, cross-sectional shape, and number of spans, achieved higher similarity scores, as illustrated in Fig. *14* below. Specifically, two-span short bridges showed significant differences from the real bridge, whereas four-span bridges in the synthetic dataset exhibited the highest similarity.

4.5. *Qualitative comparison*

Based on the above experiments, this study proposes two important considerations to maximize the model performance when using synthetic point cloud data for instance segmentation of bridge point clouds, including i) using the combination of PSLP and RSLP datasets for photogrammetry point clouds, and ii) addition of optimal occlusion to the training data improves the model performance significantly in case of LiDAR point cloud. The inference of LiDAR and photogrammetry point clouds with the proposed techniques are illustrated in Fig. 15.

5. Limitations

We successfully demonstrated the potential of using synthetic data to train deep learning models for instance segmentation of RC bridge point clouds. However, the following limitations are acknowledged.

Our synthetic dataset is developed based on the standard state of Texas, USA section types used in bridges in the state of Texas. Because the section shapes of many Texas bridges are constrained by such standardization, the instance segmentation of such components is particularly amenable to synthetically trained deep learning models, which tend to perform well when training and test data are independently and identically distributed. While we expect that our synthetic data generation method could be applied to other bridges, one requirement is that there is prior knowledge about the types of potential component sections that could be encountered in bridges and that this information is encoded properly in synthetic data generated. In the absence of this information, producing data that can train a highly generalizable model will likely be more challenging.

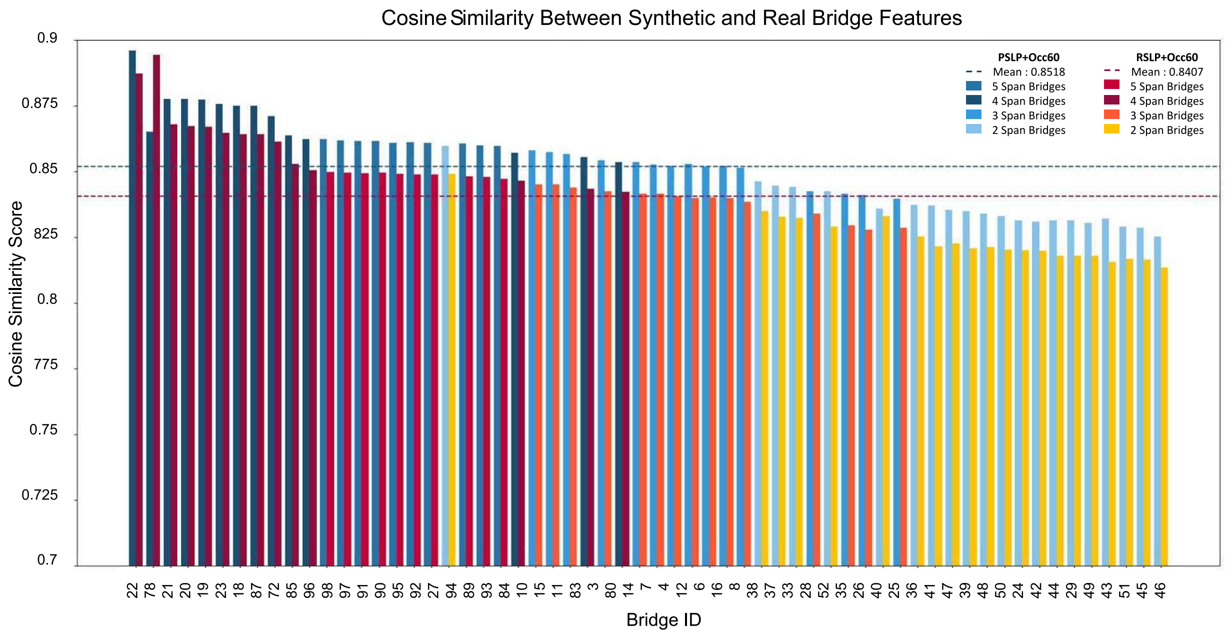

Fig. 14. Similarity between bridges generated by the PSLP+Occ60 % and the RSLP+Occ60 % approaches and the Houston bridge point cloud.

We were only able to validate our method with two real bridges given the challenges of collecting data and the lack of available datasets of RC beam slab bridges. Evaluating the approach on additional bridges will help shed more light on further improvements that could be incorporated into the synthetic data generation process.

6. Conclusion

This paper proposed a methodology for producing synthetic RC bridge point clouds to effectively train generalizable deep learning models for instance segmentation of structural elements in field-collected point clouds. To demonstrate this proposed methodology, three datasets of the same 60 bridges were developed, each with distinct methods of sampling points from 3D bridge models: Mesh Sampled Point Clouds (MSP), Perfect Simulated LiDAR Point Clouds (PSLP), and Realistic Simulated LiDAR Point Clouds (RSLP). The latter two were developed by densely (PSLP) or practically (RSLP) placing virtual LiDAR sensors around bridge models, respectively, with RSLP more closely mirroring real-world accessible locations.

The proposed synthetic data approach was found to be highly suitable for the training of generalizable point-cloud instance segmentation models. Specifically, the Mask3D model performed best on field-collected LiDAR point cloud when training with both the PSLP and RSLP data and pre-processed with optimum occlusion (60 % sparsity), achieving a mAP of 0.917, mAP_{50} of 0.998 and mAP_{25} of 0.998. This performance is attributed to the fact the PSLP provides information on how a perfect point cloud might look, and the RSLP provides additional information on what may occur if the bridge coverage is limited, and the 60 % sparsity occlusions resemble real-world occlusion patterns observed in the dataset. The combination of these three processes resulted in improved generalizability to field-collected point clouds compared to applying any process in isolation.

Notably, the pretrained PointNet feature similarity between synthetic and field-collected bridge point clouds was found to be a good indicator of the suitability of the data for training generalizable models. In particular, synthetic data produced with PSLP+Occ60 % had highest similarity to the field-collected LiDAR point clouds, translating into superior performance on field-collected bridge LiDAR point clouds, with PSLP+Occ60 % achieving mAP 0.916 (mAP_{50} 0.994, mAP_{25} 0.996), compared to RSLP+Occ60 % with mAP 0.824 (mAP_{50} 0.911, mAP_{25} 0.919). Further, the synthetic bridge point clouds that structurally and geometrically resembled real-world bridges showed higher similarity scores.

For the field-collected photogrammetry point cloud, the highest performance (mAP 0.638, mAP_{50} 0.903, and mAP_{25} 0.903) was achieved when the model was trained with the combination of PSLP and RSLP data without occlusion pre-processing. Likely due to the minimal occlusion in photogrammetry data. However, the model performance on the field-collected photogrammetry point clouds was lower than on the field-collected LiDAR point clouds, likely due to the inherent irregularities and less precise geometry in photogrammetry data. As expected, MSP data proved inadequate for training models because of its discrepancy from real-world scenarios.

Additionally, the study found that varying or randomizing point cloud colors did not significantly impact the model's performance, and that finer voxel resolutions did not necessarily lead to better results. Future research could study the applicability of the proposed approaches to other bridge types such as arch, truss, cable stayed, and suspension bridges. Additionally, research may study utilizing the results of instance segmentation in downstream tasks such as in automating the bridge inspection process by aiding element level bridge inspection and in creating digital twins of bridges.

Funding

This research is supported by the Texas Department of Transportation (TxDOT) under project number TxDOT 0–7181.

CRediT authorship contribution statement

Asad Ur Rahman: Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Vedhus Hoskere:** Writing – review & editing, Supervision, Project administration, Funding acquisition, Conceptualization.

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.

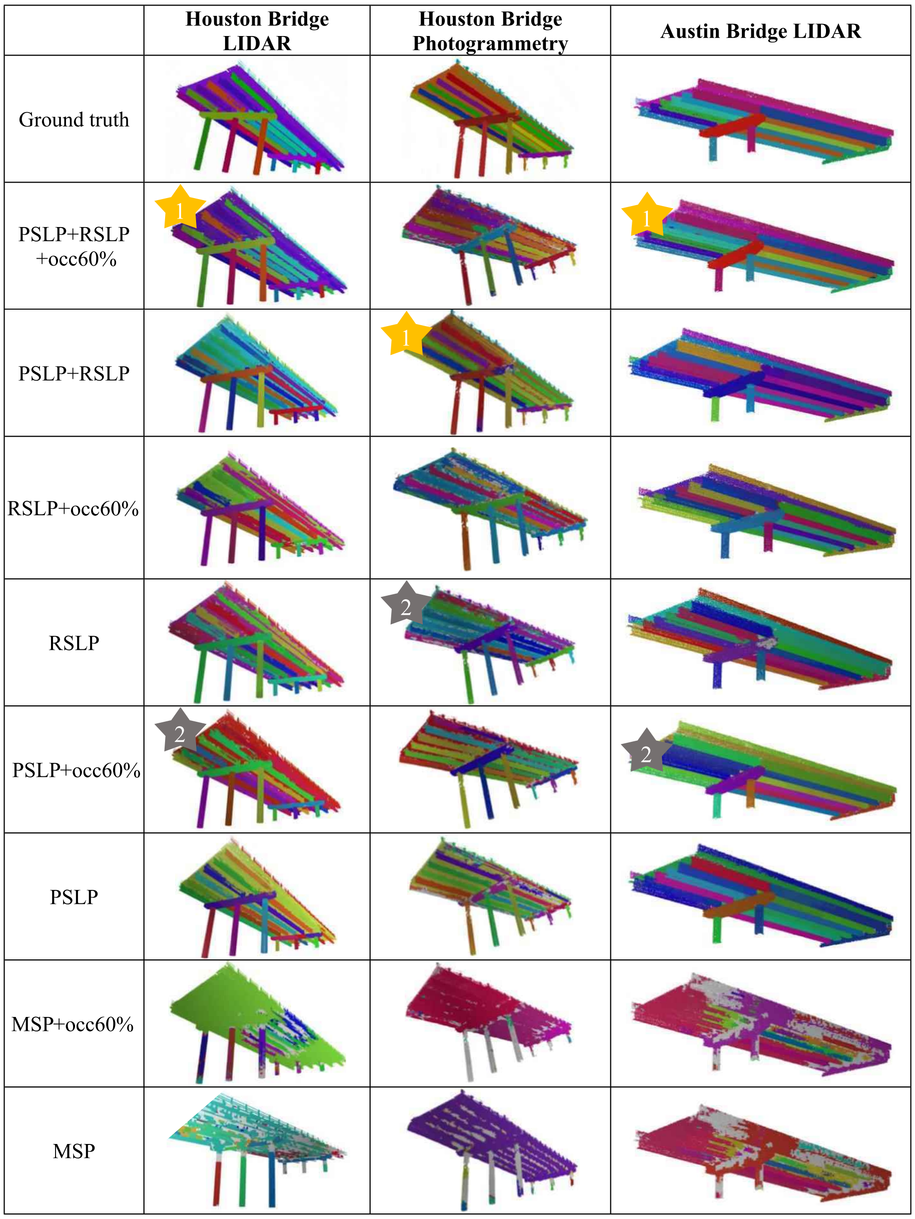

Fig. 15. Visualization of the test inference from the field-collected bridge point clouds.

Data availability

Data will be made available on request.

References

[1] B.F. Spencer Jr., V. Hoskere, Y. Narazaki, Advances in computer vision-based civil infrastructure inspection and monitoring, Engineering 5 (2019) 199–222, https://doi.org/10.1016/j.eng.2018.11.030.

[2] K.C. Crawford, Bridge Deterioration and Failures, In Failure Analysis, IntechOpen, 2023, https://doi.org/10.5772/intechopen.109927.

[3] M. Nasrollahi, G. Washer, Estimating inspection intervals for bridges based on statistical analysis of National Bridge Inventory Data, J. Bridg. Eng. 20 (2015), https://doi.org/10.1061/(asce)be.1943-5592.0000710.

[4] E. Febrianto, L. Butler, M. Girolami, F. Cirak, Digital twinning of self-sensing structures using the statistical finite element method, Data-Centric Eng. 3 (2022) e31, https://doi.org/10.48550/arXiv.2103.13729.

[5] S. Kaewunruen, J. Sresakoolchai, W. Ma, O. Phil-Ebosie, Digital twin aided vulnerability assessment and risk-based maintenance planning of bridge infrastructures exposed to extreme conditions, Sustainability 13 (2021) 2051, https://doi.org/10.3390/su13042051.

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

[7] C.-S. Shim, N.-S. Dang, S. Lon, C.-H. Jeon, Development of a bridge maintenance system for prestressed concrete bridges using 3D digital twin model, Struct. Infrastruct. Eng. 15 (2019) 1319–1332, https://doi.org/10.1080/15732479.2019.1620789.

[8] C.-S. Shim, H. Kang, N.S. Dang, D. Lee, Development of BIM-based bridge maintenance system for cable-stayed bridges, Smart Struct. Syst. 20 (2017) 697–708, https://doi.org/10.12989/sss.2017.20.6.697.

[9] Z.A. Bukhsh, N. Jansen, A. Saeed, Damage detection using in-domain and cross-domain transfer learning, Neural Comput. Applic. 33 (2021) 16921–16936, https://doi.org/10.32657/10356/12234.

[10] M.M. Islam, M.B. Hossain, M.N. Akhtar, M.A. Moni, K.F. Hasan, CNN based on transfer learning models using data augmentation and transformation for detection of concrete crack, Algorithms 15 (2022) 287, https://doi.org/10.3390/a15080287.

[11] Y. Cha, W. Choi, G. Suh, S. Mahmoudkhani, O. Büyüköztürk, Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types, Comput. Aided Civ. Inf. Eng. 33 (2018) 731–747, https://doi.org/10.1111/mice.12334.

[12] P. Prasanna, K.J. Dana, N. Gucunski, B.B. Basily, H.M. La, R.S. Lim, H. Parvardeh, Automated crack detection on concrete bridges, IEEE Trans. Autom. Sci. Eng. 13 (2014) 591–599, https://doi.org/10.1109/TASE.2014.2354314.

[13] H. Zoubir, M. Rguig, M. Elaroussi, Crack recognition automation in concrete bridges using Deep Convolutional Neural Networks, in: MATEC Web of

Conferences, EDP Sci. (2021) 03014, https://doi.org/10.1051/matecconf/202134903014.

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

[15] 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.

[16] B. Riveiro, M.J. DeJong, B. Conde, Automated processing of large point clouds for structural health monitoring of masonry arch bridges, Autom. Constr. 72 (2016) 258–268, https://doi.org/10.1016/j.autcon.2016.02.009.

[17] N. Gyetvai, L. Truong-Hong, D.F. Laefer, Laser scanning-based diagnostics in the structural assessment of historic wrought Iron, Bridges (2018), https://doi.org/10.1680/jenhh.17.00018.

[18] D. Lamas, A. Justo, M. Soilán, M. Cabaleiro, B. Riveiro, Instance and semantic segmentation of point clouds of large metallic truss bridges, Autom. Constr. 151 (2023) 104865, https://doi.org/10.1016/j.autcon.2023.104865.

[19] D. Lamas, A. Justo, M. Soilán, B. Riveiro, Automated production of synthetic point clouds of truss bridges for semantic and instance segmentation using deep learning models, Autom. Constr. 158 (2024) 105176, https://doi.org/10.1016/j.autcon.2023.105176.

[20] 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.

[21] 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.

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

[23] Y.-C. Lin, A. Habib, Semantic segmentation of bridge components and road infrastructure from mobile LiDAR data, ISPRS Open J. Photogrammetry Rem. Sens. 6 (2022) 100023, https://doi.org/10.1016/j.ophoto.2022.100023.

[24] 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.

[25] X. Yang, E. del Rey Castillo, Y. Zou, L. Wotherspoon, Semantic segmentation of bridge point clouds with a synthetic data augmentation strategy and graph-structured deep metric learning, Autom. Constr. 150 (2023) 104838, https://doi.org/10.1016/j.autcon.2023.104838.

[26] M. Shi, H. Kim, Y. Narazaki, Development of large-scale synthetic 3D point cloud datasets for vision-based bridge structural condition assessment, Adv. Struct. Eng. (2024) 13694332241260076, https://doi.org/10.1177/13694332241260077.

[27] B. Yang, J. Wang, R. Clark, Q. Hu, S. Wang, A. Markham, N. Trigoni, Learning object bounding boxes for 3D instance segmentation on point clouds, Adv. Neural. Inf. Process Syst. 32 (2019), https://doi.org/10.48550/arXiv.1906.01140.

[28] J. Hou, A. Dai, M. Nießner, 3D-sis: 3d semantic instance segmentation of rgb-d scans, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4421–4430, https://doi.org/10.48550/arXiv.1812.07003.

[29] W. Sun, D. Rebain, R. Liao, V. Tankovich, S. Yazdani, K.M. Yi, A. Tagliasacchi, NeuralBF: neural bilateral filtering for top-down instance segmentation on point clouds, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 551–560, https://doi.org/10.1109/wacv56688.2023.00062.

[30] M. Kolodiazhnyi, A. Vorontsova, A. Konushin, D. Rukhovich, Top-down beats bottom-up in 3d instance segmentation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024, pp. 3566–3574, https://doi.org/10.48550/arXiv.2302.02871.

[31] L. Yi, W. Zhao, H. Wang, M. Sung, L.J. Guibas, Gspn: Generative shape proposal network for 3d instance segmentation in point cloud, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3947–3956, https://doi.org/10.48550/arXiv.1812.03320.

[32] T. Vu, K. Kim, T.M. Luu, T. Nguyen, C.D. Yoo, Softgroup for 3d instance segmentation on point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 2708–2717, https://doi.org/10.1109/cvpr52688.2022.00273.

[33] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, J. Jia, Pointgroup: Dual-set point grouping for 3d instance segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 4867–4876, https://doi.org/10.48550/arXiv.2004.01658.

[34] Z. Liang, Z. Li, S. Xu, M. Tan, K. Jia, Instance segmentation in 3D scenes using semantic superpoint tree networks, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 2783–2792, https://doi.org/10.48550/arXiv.2108.07478.

[35] S. Chen, J. Fang, Q. Zhang, W. Liu, X. Wang, Hierarchical aggregation for 3d instance segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 15467–15476, https://doi.org/10.48550/arXiv.2108.02350.

[36] T. He, C. Shen, A. Van Den Hengel, Dyco3d: Robust instance segmentation of 3d point clouds through dynamic convolution, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 354–363, https://doi.org/10.48550/arXiv.2011.13328.

[37] S. Chen, J. Fang, Q. Zhang, W. Liu, X. Wang, Hierarchical aggregation for 3d instance segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 15467–15476, https://doi.org/10.48550/arXiv.2108.02350.

[38] T. Vu, K. Kim, T.M. Luu, T. Nguyen, C.D. Yoo, Softgroup for 3d instance segmentation on point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 2708–2717, https://doi.org/10.1109/cvpr52688.2022.00273.

[39] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, J. Jia, Pointgroup: Dual-set point grouping for 3d instance segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 4867–4876, https://doi.org/10.48550/arXiv.2004.01658.

[40] F. Engelmann, M. Bokeloh, A. Fathi, B. Leibe, M. Nießner, 3D-mpa: Multi-proposal aggregation for 3d semantic instance segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9031–9040, https://doi.org/10.48550/arXiv.2003.13867.

[41] L. Han, T. Zheng, L. Xu, L. Fang, Occuseg: Occupancy-aware 3d instance segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 2940–2949, https://doi.org/10.48550/arXiv.2003.06537.

[42] J. Sun, C. Qing, J. Tan, X. Xu, Superpoint transformer for 3d scene instance segmentation, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2023, pp. 2393–2401, https://doi.org/10.1609/aaai.v37i2.25335.

[43] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, B. Leibe, Mask3d for 3d semantic instance segmentation, ArXiv Preprint ArXiv:2210.03105 (2022), https://doi.org/10.1109/ICRA48891.2023.10160590.

[44] M. Kolodiazhnyi, A. Vorontsova, A. Konushin, D. Rukhovich, OneFormer3D: One transformer for unified point cloud segmentation, ArXiv Preprint ArXiv:2311.14405 (2023), https://doi.org/10.48550/arXiv.2311.14405.

[45] J. Balado, R. Sousa, L. Diaz-Vilarino, P. Arias, Transfer learning in urban object classification: online images to recognize point clouds, Autom. Constr. 111 (2020) 103058, https://doi.org/10.1016/j.autcon.2019.103058.

[46] A.J. Rios, V. Plevris, M. Nogal, Synthetic Data Generation For The Creation Of Bridge Digital Twins What-If Scenarios, 2023, https://doi.org/10.7712/120123.10760.21262.

[47] Y. Jing, B. Sheil, S. Acikgoz, Segmentation of large-scale masonry arch bridge point clouds with a synthetic simulator and the BridgeNet neural network, Autom. Constr. 142 (2022) 104459, https://doi.org/10.1016/j.autcon.2022.104459.

[48] 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.

[49] F. Noichl, F.C. Collins, A. Braun, A. Borrmann, Enhancing point cloud semantic segmentation in the data-scarce domain of industrial plants through synthetic data, Comput. Aided Civ. Inf. Eng. (2024), https://doi.org/10.1111/mice.13153.

[50] Y. Xu, X. Tong, U. Stilla, Voxel-based representation of 3D point clouds: methods, applications, and its potential use in the construction industry, Autom. Constr. 126 (2021) 103675, https://doi.org/10.1109/iros45743.2020.9340992.

[51] M. Chen, Q. Hu, Z. Yu, H. Thomas, A. Feng, Y. Hou, K. McCullough, F. Ren, L. Soibelman, Stpls3d: A large-scale synthetic and real aerial photogrammetry 3d point cloud dataset, ArXiv Preprint ArXiv:2203.09065 (2022), https://doi.org/10.48550/arXiv.2203.09065.

[52] A. Dai, A.X. Chang, M. Savva, M. Halber, T. Funkhouser, M. Nießner, Scannet: Richly-annotated 3d reconstructions of indoor scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5828–5839, https://doi.org/10.48550/arXiv.1702.04405.

[53] C. Choy, J. Gwak, S. Savarese, 4d spatio-temporal convnets: Minkowski convolutional neural networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3075–3084, https://doi.org/10.1109/cvpr.2019.00319.

[54] W. Tjong, G.J. Kazakia, A.J. Burghardt, S. Majumdar, The effect of voxel size on high-resolution peripheral computed tomography measurements of trabecular and cortical bone microstructure, Med. Phys. 39 (2012) 1893–1903, https://doi.org/10.1118/1.3689813.

[55] I. Armeni, O. Sener, A.R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3d semantic parsing of large-scale indoor spaces, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1534–1543, https://doi.org/10.1109/cvpr.2016.170.

[56] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, ArXiv Preprint ArXiv:1711.05101 (2017), https://doi.org/10.48550/arXiv.1711.05101.

[57] L.N. Smith, N. Topin, Super-convergence: Very fast training of neural networks using large learning rates, in: Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications, SPIE, 2019, pp. 369–386, https://doi.org/10.1117/12.2520589.

[58] D. Moon, S. Chung, S. Kwon, J. Seo, J. Shin, Comparison and utilization of point cloud generated from photogrammetry and laser scanning: 3D world model for smart heavy equipment planning, Autom. Constr. 98 (2019) 322–331, https://doi.org/10.1016/j.autcon.2018.07.020.