Automation in Construction 172 (2025) 106021

Contents lists available at ScienceDirect

Automation in Construction

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

3D bridge segmentation using semi-supervised domain adaptation

Maximilian Kellner [a,b,*], Timothy König [c], Jan-Iwo Jäkel [c], Katharina Klemt-Albert [c], Alexander Reiterer [a,b]

[a] Fraunhofer Institute for Physical Measurement Techniques IPM, Freiburg, 79110, Germany
[b] University of Freiburg, Department of Sustainable Systems Engineering INATECH, Freiburg, 79110, Germany
[c] RWTH Aachen University, Institute for Construction Management, Digital Engineering and Robotics in Construction, Aachen, 52070, Germany

ARTICLE INFO

Keywords:
3D deep learning
Bridge segmentation
Point cloud generation
Point cloud processing
Domain adaptation

ABSTRACT

Understanding the scene is crucial for automated bridge inspection. Traditionally, bridges are measured using 3D sensors that produce large point clouds. Manually interpreting the captured data is time-consuming and error-prone. This paper proposes an unsupervised and semi-supervised domain adaptation approach for 3D bridge segmentation using labeled synthetically generated data and no or limited real-world data. To achieve this, a pipeline was developed for automatically generating artificial scenes of bridges and virtually scanning them with an artificial sensor. This data, along with real-world data, is utilized for the proposed methods. In the unsupervised approach, a deep feature alignment method integrates real-world data into the training procedure. Instead of feature alignment, a semi-supervised method is proposed to guide the training using only a small amount of annotated real-world data. The findings demonstrate that performance can be enhanced in an unsupervised manner. However, performance gains are significantly amplified when 10 % of real-world data is integrated with synthetic data and used in the proposed guided training. The approaches are validated using two distinct deep learning architectures.

1. Introduction

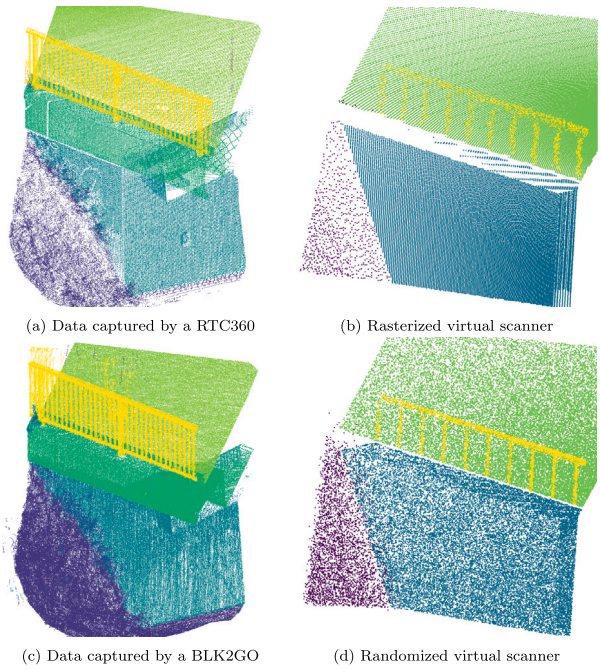
Bridges are essential components of infrastructure that contribute to connectivity, trade, accessibility, resilience, tourism, economic growth, and urban development. Their importance goes beyond the simple transportation infrastructure, affecting various aspects of society, economy, and culture [1]. Therefore, it is essential to maintain and invest in bridges to ensure the continued functionality, safety, and sustainability of infrastructure networks worldwide. In Germany, for instance, the federal highway system includes over 40,000 bridges. According to the March 2023 report of the Federal Highway Research Institute (BASt) [2], 4.6% of these bridges are classified as being in insufficient or worse condition. Furthermore, while a considerable number of bridges are currently in satisfactory or acceptable condition, it is imperative to maintain consistent monitoring and inspection of these structures to identify potential issues at an early stage. This process can be time-consuming and error-prone, but it is necessary to ensure the safety and reliability of the structures [3].

One possible approach is to automate and speed up the process. In this scenario, the Scan-to-BIM or Scan2BIM method [4] can help to produce high quality data in a timely manner to facilitate maintenance decision making. Scan2BIM involves creating detailed three-dimensional (3D) digital models from data captured in the real world and enriching it with semantic information. The captured 3D data consist mainly of point clouds, and the first step for Scan2BIM is to understand the given scene [5]. One of the most common ways to interpret a scene is to segment it semantically. Deep learning is the de facto state of the art for solving such perceptual tasks [6].

A disadvantage of deep learning is that it requires a large amount of training data, which can be costly to produce. The data must first be collected, which can be difficult in some cases. It is also necessary to ensure that the data has sufficient variance to cover as many different scenarios as possible. Once the data has been pre-processed, it needs to be annotated in the case of supervised learning. The complexity of the annotation depends largely on the data and the given list of classes. For example, annotating 3D point clouds is much more time-consuming and error-prone than annotating 2D images [7]. To avoid the mentioned disadvantages, one option is to generate data synthetically. However, applying this approach to real-world problems may not be straightforward due to potential domain gaps. These gaps can be caused by various factors, such as a lack of realism, a distribution shift between the synthetic data and the real-world ones, or not being accurately compared to the complexity and variability of real-world scenarios. It is worth noting that domain gaps may arise not only when

* Corresponding author at: Fraunhofer Institute for Physical Measurement Techniques IPM, Freiburg, 79110, Germany.
E-mail address: maximilian.kellner@ipm.fraunhofer.de (M. Kellner).

https://doi.org/10.1016/j.autcon.2025.106021
Received 18 September 2024; Received in revised form 24 January 2025; Accepted 27 January 2025
Available online 5 February 2025



(a) Data captured by a RTC360 (b) Rasterized virtual scanner

(c) Data captured by a BLK2GO (d) Randomized virtual scanner

Fig. 1. Point clouds acquired by different real sensors (Leica Geosystems) and different virtual sensors.

transferring synthetic data to real-world data, but also within real-world data itself. These variations can be caused by different factors, such as changing sensors, differences in sensor setup, or biases within the data collection process. All the reasons mentioned lead to variations in data distribution. Fig. 1 illustrates the differences in distributions between similar scenes captured by various real and synthetic sensors.

The main idea of learning from different domains is to learn domain invariant representations by aligning the source and target domains [8]. In [9] transfer learning is divided into three categories: inductive transfer learning, transductive transfer learning, and unsupervised transfer learning. The categories are mainly distinguished by whether the task to be solved is different or the domain between the target and the source. Domain Adaptation (DA), which is a special case of inductive transfer learning, refers to the training of machine learning models on data from a different distribution (domain) than the one on which the model is used. The aim is to improve the performance of the model on the target domain. According to [10], DA can be further categorized into four types: closed set, open set, partial, and universal DA. In the closed set, the source and target domains share the same classes, whereas in the open set, the source classes are considered to be a subset of the target classes. The partial DA is the opposite of the open set, which means the target classes are only a subset of the source classes. The universal DA assumes no prior knowledge of the source and target domains.

Most of the work in DA is done with images and mainly with classification problems. However, dealing with point clouds has some disadvantages, as mentioned in [11], which further increase the possible domain gap. A further issue, particularly in the context of bridge applications, is that the scenes to be processed can be of significant size. Indeed, some bridges may extend to hundreds of metres in width, height, or even both dimensions. In Germany, for instance, more than 50% of the federal highway bridges are longer than 30 m, with 7% exceeding 100 m in length [12]. As a result, the scene must be subdivided into smaller sub-clouds, which further increases the difficulty of segmenting the bridge and applying DA techniques. This is because it cannot be guaranteed that the sub-clouds will be of the same sub-part of the bridge or even of the same semantic class. Moreover, the sub-clouds exhibit considerable variation in size and point density.

The main contributions of this paper are the automatic generation of synthetic training data and an easy-to-use joint optimization procedure for the semi-supervised reduction of the domain gap for 3D bridge segmentation. First, a generic and fully automated approach for automatic bridge generation using a component catalog is presented. A 3D scene can be generated by combining the approach with open-source 3D objects. The resulting scene can then be measured with a virtual sensor using virtual viewpoints and ray tracing, which in turn allows for the automatic generation of synthetic annotated point clouds. In total, 24 artificially measured bridges are generated. Secondly, to illustrate the extent to which synthetic data (source domain) can enhance semantic segmentation performance, two real bridges (target domain) were used for aligning purposes, in addition to eight real bridges for validation. This allows us to demonstrate the transferability of the acquired knowledge. Four distinct approaches are employed, comprising two for Unsupervised DA (UDA) and two for Semi-Supervised DA (SSDA). In our first baseline, only the source data is used for training and the trained network is applied to the target domain. For the second experiment, we follow the idea of feature alignment to allow the use of target data without supervision. The second baseline experiment simply mixes the source and target domain randomly using a small amount of real data within optimization. In contrast to the utilization of feature alignment techniques, we propose the concept of employing similar sub-clouds during the optimization process. This involves the selection of a random synthetic sub-cloud, which is then paired with a comparable real-world sub-cloud for joint optimization. Our main contributions can be summarized as follows:

- A automatic pipeline to create bridge environments that take whole scenes into account, making the approach more realistic, without the need of existing data.
- Automatic generation of annotated point cloud data of the created scenes without the necessity for human oversight or supervision.
- Investigation of the application of closed set DA for the purpose of 3D point cloud segmentation of large scenes.
- Approach for SSDA with only a small amount of annotated target data.
- Extensive experiments in four cross-domain scenarios showing that our approach achieves state-of-the-art performance.

2. Related work

2.1. 3D semantic segmentation

3D semantic segmentation is a computer vision technique that classifies objects within a point cloud on a point-by-point basis, providing a detailed understanding of the point cloud content. Today, 3D segmentation is approached in a variety of ways. A general overview is given in [13]. In general, these methods can be divided into four different approaches: projection-based methods, discretization-based methods, point-based methods, and hybrid methods. In [14] the point cloud is projected into a spherical image and processed by 2D convolutions. The point cloud can be discretized into voxels and processed by 3D convolution [15]. Sparse 3D tensors are introduced in [16] to allow processing of large discretized scenes with more efficient sparse 3D convolutions. KPConv of [17] defines kernel points that can be used to process point clouds directly. Another way to work directly on the point cloud is introduced in PointTransformer [18]. Here attention mechanisms are used instead of convolutions. The idea has been extended by using grouped vector attention and grid-based pooling in [19] and further optimized by serializing points and an increased receptive field in [20]. In Swin3D [21], the idea of using hierarchical vision transformers with shifted windows [22], is adapted to 3D. A hybrid method using different representations is shown in [23], where the spherical and bird's eye projections are fused using a KPConv layer.

Table 1
Overview of point cloud datasets with semantic annotations. Number of classes used to evaluate and total number of classes annotated in brackets.

Dataset	Sensor	Scene	Points	Classes
ScanNet [34]	RGB-D	Indoor	242 M	20
S3DIS [35]	TLS	Indoor	215 M	13
Oakland3d [36]	MLS	Outdoor	1.6 M	5 (44)
Semantic3d [37]	TLS	Outdoor	4 000 M	8
Paris-Lille-3D [38]	MLS	Outdoor	143 M	9 (50)
SemanticPOSS [39]	MLS	Outdoor	216 M	14
SemanticKITTI [40]	MLS	Outdoor	4 549 M	25 (28)
SynLiDAR [41]	Synthetic	Outdoor	19 482 M	32
DALES [42]	MLS	Aerial	505 M	8 (9)
ISPRS [43]	MLS	Aerial	1.2 M	9
SensatUrban [44]	UAV	Aerial	2 847 M	13 (31)
STPLS3D [45]	Synthetic	Aerial	150.4 M	18
CLOI [46]	TLS	MEP	140 M	10
Unnamed [47]	TLS	MEP	80 M	5 (6)

A general overview of deep learning for point clouds in the construction industry can be found in the following article [24]. It is evident that the majority of current research is concentrated on indoor building structures and outdoor roads, which frequently emerge from autonomous driving objectives, mechanical, electrical, and plumbing (MEP) systems, or urban scenes. Table 1 presents a selection of publicly accessible state-of-the-art datasets with a primary focus on 3D semantic segmentation. The sensing devices and technologies utilized for each dataset include Terrestrial Laser Scanning (TLS), Mobile Laser Scanning (MLS), Unmanned Aerial Vehicle (UAV) photogrammetry, RGB-D cameras, and synthetic environments.

Given the unique structure of bridges, it is not feasible to make use of the presented datasets with their predefined classes to the problem of semantically segmenting bridges. However, some research has been conducted on the aforementioned task. Instead of using deep learning approaches, [25] uses slicing algorithms and detects and segments pier caps using their surface normal, and beams using oriented bounding boxes and density histograms. To improve efficiency, the authors propose BridgeNet [26], a 3D deep learning neural network that can automate segmentation. To address the scarcity of labeled point cloud data, they create a synthetic dataset to train BridgeNet. However, the authors focus solely on masonry arch bridges and do not consider the environment. A weighted superpoint graph method is used in [27]. Again, synthetic data is used and the real bridges are separated from the environment for validation. The method is further improved in [28] using deep metric learning methods, but still following the idea of separating the whole object from the given environment. In [29] a pipeline for truss bridges using synthetic data is proposed. An adapted JSNet [30] is trained for semantic and instance segmentation. Like the others, it uses only the object itself. The [25] data combined with 10 more realistic synthetic bridges is used to train a RandLA-Net [31] in [32]. The synthetic bridges are virtually scanned using [33].

However, all studies demonstrate that semantic segmentation for bridges is achievable, even when combining synthetic data. Nevertheless, they solely focus on the object and indicate an improvement in performance when using synthetic data, but do not address the use of real data or the required quantity.

2.2. Domain adaptation

DA attempts to address the distribution gap by focusing on general machine learning. While UDA only allows the source domain to be labeled, SSDA involves learning to classify unseen target data with a few labeled and many unlabeled target data along with many labeled source data from a related domain [48]. Semi-Supervised Learning (SSL) aims to make use of the large amount of unlabeled data with limited labeled data to improve classifier performance. The primary distinction between SSL and SSDA lies in the nature of the data utilized for analysis. SSL employs data derived from a single distribution, whereas SSDA utilizes data from two distinct domains, each exhibiting a domain discrepancy as a fundamental characteristic [49].

Approaches to tackle DA can be broadly divided into shallow and deep architectures [50]. In shallow DA, the discrepancy is reduced either by reweighting the source samples and training on the weighted source samples [51], or by generally learning a common shared space in which the distributions of the two datasets are matched [52]. Deep DA typically uses neural networks to bridge the domain gap. There are discrepancy-based, reconstruction-based, and adversarial-based approaches. Several methods in the Deep DA category use distance metrics borrowed from shallow approaches to measure discrimination between feature representations at different layers of the network.

One way to ensure that the representations learned by a model are consistent across different data distributions or modalities is to use a commonly shared feature space. This can be obtained through feature alignment methods. Many approaches use a distance metric, such as Maximum Mean Discrepancy (MMD) [53], KL divergence or Earth Mover's Distance (EMD) (also known as Wasserstein distance) [54], to quantify the disparity between the source and target domains. The authors of CORAL [55] introduced an unsupervised domain adaptation technique that utilizes a linear transformation to align the second-order statistics of the source and target distributions. They also extend this method to Deep CORAL [56], which learns a nonlinear transformation to align correlations of layer activations in deep neural networks. The correlation loss looks as follows:

$$\mathcal{L}_C = \frac{1}{4d^2}\|C_S - C_T\|_F^2 \tag{1}$$

With $\|\cdot\|_F^2$ being the squared matrix Frobenius norm and the feature covariance matrices C_S and C_T for the source and target data:

$$C_S = \mathcal{F}_S A \mathcal{F}_S^T \text{ and } C_T = \mathcal{F}_T A \mathcal{F}_T^T \tag{2}$$

with A being the centering matrix. In [57] a Progressive Feature Alignment Network (PFAN) is proposed to align the discriminative features across domains progressively and effectively, via exploiting the intra-class variation in the target domain.

Both [58,59] are using higher-order correlations since using second-order or lower order statistics might not be enough for complex and non-Gaussian distributions. The complexity of the introduced tricovariance grows exponentially with the dimension of the latent feature representation $\mathcal{O}(z^3)$. Regarding to [60], measuring correlation alignments with the Euclidean metric (Eq. (1)) is inadequate since it fails to capture the internal geometry of the data. For this reason, the Minimal-Entropy Correlation Alignment (MECA) is introduced using the log-Euclidean distance, which serves as a geodesic distance. This changes the calculation of the covariance matrix as follows:

$$\begin{aligned} C_S^{log} &= U\,\mathrm{diag}(\log(\lambda_1^{C_S}), \dots, \log(\lambda_d^{C_S}))U^T \\ C_T^{log} &= V\,\mathrm{diag}(\log(\lambda_1^{C_T}), \dots, \log(\lambda_d^{C_T}))V^T \end{aligned} \tag{3}$$

with U and V are the matrices which diagonalize C_S and C_T, respectively, and $\lambda_i^{C_S}$, $\lambda_i^{C_T}$ for $i = 1, \dots, d$ are the corresponding eigenvalues.

Instead of correlation alignment, maximizing Mutual Information (MI) refers to the process of increasing the amount of information shared between two random variables. It aims to find the optimal configuration of the variables that maximizes the MI between them. MI measures the dependence between the variables and can be used in various fields such as information theory, machine learning, and data analysis. MI is closely related to the entropy H and is obtained by subtracting the marginal entropy of both sets from the joint entropy:

$$MI(X;Y) = H(A,B) - H(A) - H(B) \tag{4}$$

$$= H(Y|X) - H(Y) \tag{5}$$

$$= \int_y \int_x p(x,y) log\left(\frac{p(x,y)}{p(x)p(y)}\right) dxdy \tag{6}$$

In [61], a random transformation is used to obtain a pair of images, and the goal is to maximize the MI between the class assignments

of each pair. This way the authors can classify and segment images unsupervised. [62] follows the same idea but introduces an adversarial training scheme and operates on superpixels instead of single pixels. A module has been developed with the aim of maximizing MI, with the intention of learning features that are invariant across domains and sharing segmentation knowledge between them in [63]. The authors use this approach for the segmentation of brain structures from magnetic resonance images.

For 3D DA, an overview of its application to LiDAR perception for autonomous driving is given in [64]. In order to transfer the knowledge already gained from the 2D field, a common approach is to transfer the point cloud to a 2D representation. In [65] the point cloud is transformed into the bird's eye view and a deep cycle-consistent generative adversarial network (CycleGAN) [66] is used to solve the domain adaption task between real and synthetic generated point clouds, to detect vehicles. The point cloud is transformed into the spherical view in [67] and a three staged pipeline containing learned intensity rendering, geodesic correlation alignment, and progressive domain calibration is proposed for reducing the domain shift. For geodesic correlation alignment, they feed a batch of synthetic data and a batch of real data into the network at each training step. The synthetic one is labeled, while the real one is not. The focal loss [68] is computed on the labeled batch and the geodesic distance [60] between the synthetic and the real one is computed. The total loss is the combination of both losses. Both use the real-world KITTI dataset [69] to validate their results.

There are already some approaches that remain within the 3D space. In [70] a concept is proposed that involves distorting certain areas of the input shape and then using a neural network to reconstruct it, demonstrating a large improvement over previous work on both synthetic and real furniture data. To reduce the domain gap in both the input space and the feature space, [71] developed a point cloud style transfer network and a feature discrimination network based on the CycleGAN architecture. The authors [72] suggested an approach to unsupervised domain adaptation on point clouds, which involves utilizing a self-supervised task that focuses on acquiring knowledge about geometry-aware implicits. A method is proposed to synergize contrastive learning and optimal transport for effective UDA in [73]. They focus on the reduction of domain shift and the learning of transferable point cloud embeddings. For object detection, a mixing method between domains is proposed in [74] that generates an intermediate domain to learn invariant features.

3. Data generation

A pipeline was developed to automatically generate 3D artificial scenes of bridges. The entire data generation pipeline is shown 2. This pipeline employs a bridge component catalog for the individual subparts of bridges, as well as open-source data models for common environmental objects, in order to generate complete scenes. The full scene is generated using a simple rule-based approach that uses all 3D objects. The bridge is scanned using virtual sensor models and scan positions to generate a fully annotated synthetic point cloud. The following subchapters describe the subparts in more detail.

3.1. Bridge model generation

Bridges consist of several main components. The superstructure spans the obstacle, while the abutments support the superstructure at the ends of the bridge. For longer bridges, piers are added to support the superstructure between the abutments. The deck is the surface of the bridge on which people, vehicles, or other loads travel. Railings are added to the sides for safety reasons to prevent people or vehicles from falling off the bridge. The most prominent and important classes for our work have been covered, and we will not discuss any other bridge classes here. The relevant parts generated can be seen in Fig. 3. All components are modeled as parametric families in Autodesk Revit [75] and consolidated into a component catalog.

To ensure accurate and complete modeling and correct component assignment, and to define and process both geometric and alphanumeric information of the bridge elements, Autodesk Dynamo [76] and the data management package Clockwork [77] are used for generation. Geometric relationships between different components are defined within a cartesian coordinate system, and the terrain model is aligned with the geometric features of the bridge model. To emulate natural terrain characteristics, the z coordinates of selected points within the terrain are randomized uniformly. The parameters height (h), length (l), width (w), and thickness (t) of the supports are pivotal input variables for bridge generation. These values are systematically chosen from standard cross sections, thus ensuring conformity with industry norms [78]. The adjustment process begins with both abutments, followed by the superstructure, columns, and finally the topology of the terrain.

3.2. 3D scene generation

In order to enhance the realism of the scene and to introduce additional elements that may occur in real-world scenarios, we have added additional 3D objects to our generated bridge. We use pedestrians, vehicles, bicycles, road signs, trees, and other vegetation from various 3D model sources [79–82] to create a variation of different objects. To sample the different objects, we use only three predefined rules. Road users such as pedestrians, vehicles, and bicycles can only be generated on the road. Trees and other vegetation must be placed on the ground and must not intersect any of the generated bridge parts. Finally, traffic signs are only allowed near the edge of the road. The number of allowed objects is randomly determined by the size of the given bridge model. To further increase the variation of vegetation objects such as trees, bushes or shrubs, the randomly selected objects are scaled by a random factor before being integrated into the scene.

3.3. Point cloud generation

A terrestrial laser scanner is selected as the reference model for the virtual sensor. It rotates about two of its axes and emits laser pulses to collect distance measurements, generating a point cloud that represents the 3D coordinates of surfaces in the scanned environment. The captured environment is centered in the local coordinate frame, making it suitable for emulation using ray tracing methods. Given that the majority of the information is assumed to be contained within the 3D geometry, it is not necessary to render material properties for the sake of simplicity.

The origin of the virtual sensor is given by O_i. We use $r = 1$, $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi]$ to generate the rays. To follow the idea of terrestrial scanners, the values for $\Delta\theta$ and $\Delta\phi$ are configurable and could be set to the vertical and horizontal angle resolution of a real sensor. The visualized sensor model can be seen in Fig. 4. In theory, the angle resolution could be set to a very high resolution. However, in practice, the points have to be downsampled in order to achieve a more general distribution and a reduced number of points. Consequently, the resolution is reduced to $0.1°$. For the randomized sensor model, we simply define the maximum number of points and sample uniformly angles θ and ϕ between the above defined intervals. Subsequently, the calculated spherical coordinates are mapped to Cartesian coordinates, enabling the seamless integration of multiple scan positions.

The calculated vectors are regarded as rays starting at the origin O. We define the distance (in Meter) to be in range $1.5 < d < 120$, dropping all hits missing the defined region. To make the scan even more realistic, we add different noise levels for different objects following the uniform distribution $\mathcal{U}(a, b)$:

$$f(x) = \begin{cases} \frac{1}{b-a} & \text{for } x \in [a_c, b_c] \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

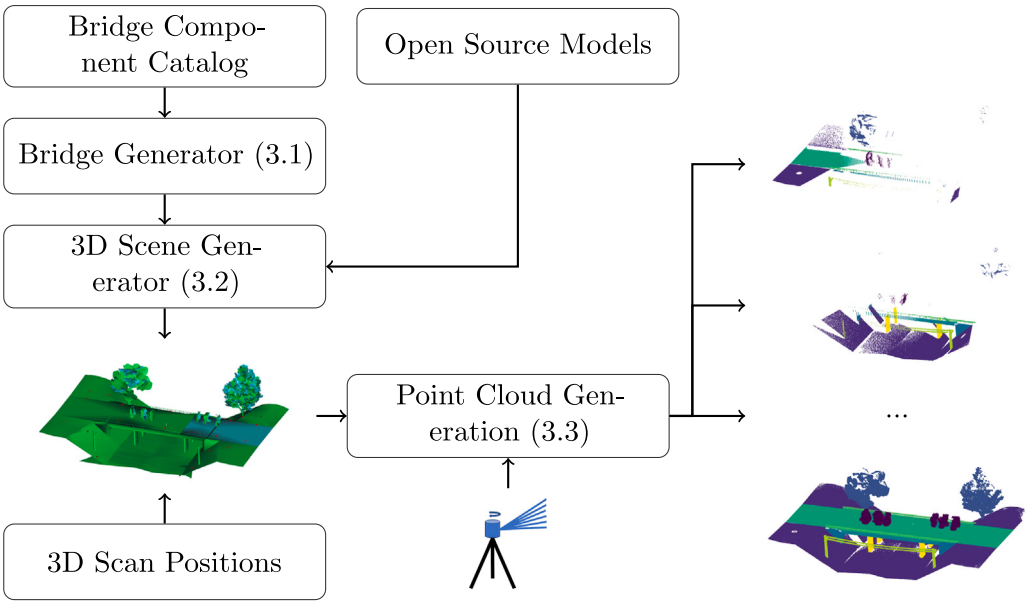


Fig. 2. Data generation pipeline.

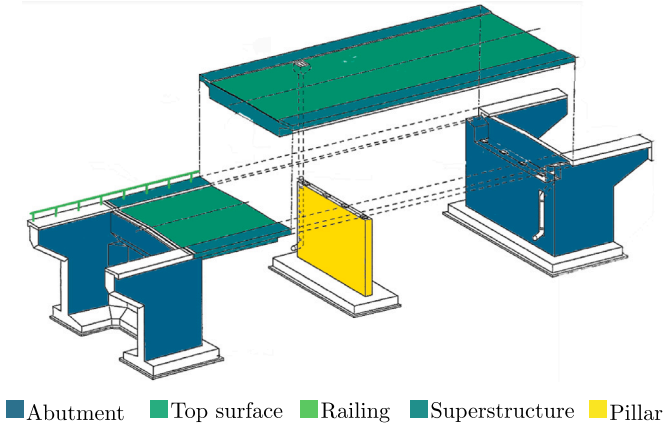


Fig. 3. Colored relevant components for bridges. (Zoom in for best view).

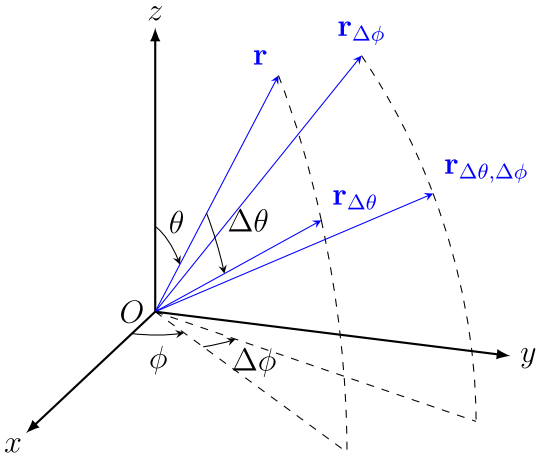


Fig. 4. Spherical sensor model. The virtual scanner is located at the origin O. The horizontal and vertical angle resolution of the sensor are based on the configurable parameters $\Delta\phi$ and $\Delta\theta$.

with $a_0 = 0.5$, $a_1 = 0.1$, $a_2 = 0.2$, $a_3 = 0.001$, $a_4 = 0.001$, $a_5 = 0.001$, $a_6 = 0.1$, $a_7 = 0.1$, $a_8 = 0.01$ and $b_c = -a_c$ for the defined classes. For example, the bridge deck, which is man-made and likely flat, experiences less distortion than the vegetation.

Once all viewpoints have been measured using the virtual sensor, they are merged into a global coordinate system to obtain the desired point cloud of the entire scene.

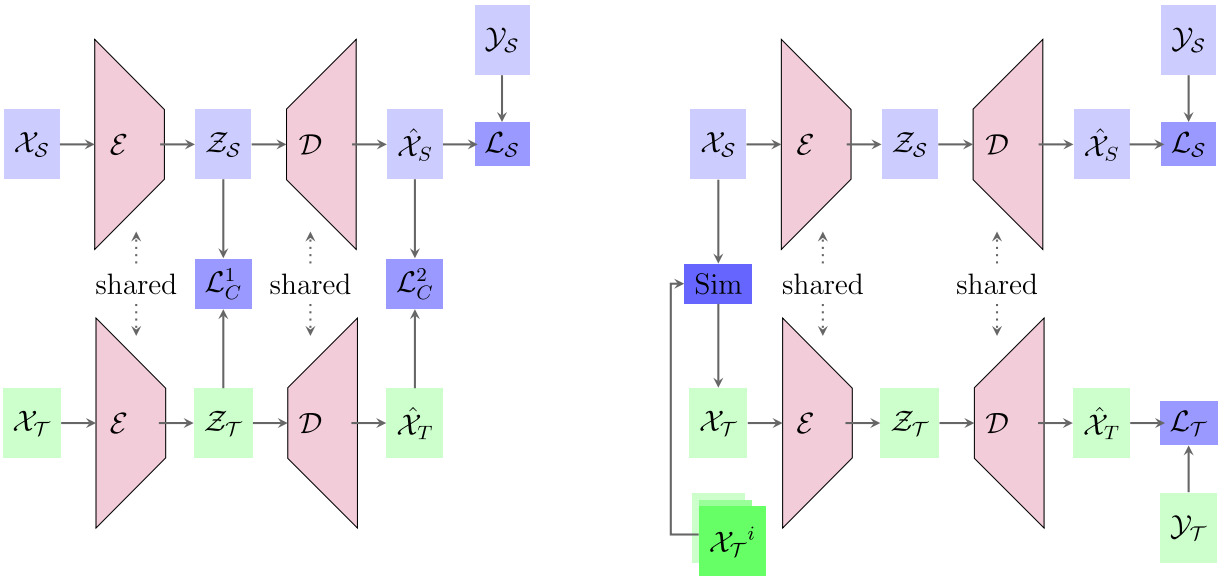
4. Method

4.1. Problem statement and notation

A domain set is defined as $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}, p(x, y)\}$ with the input feature space $\mathcal{X} \in \mathbb{R}^{N\times 3}$ represented as point cloud with N points and its coordinates $(x_i, y_i, z_i)_{i=1}^{N}$, the label space $\mathcal{Y} \in \mathbb{R}^{N}$ and the joint probability distribution $p(x, y)$ over the input and label space. For closed set domain adaptation we have a source domain $\mathcal{S}$ and a target domain $\mathcal{T}$ where $\mathcal{Y}_\mathcal{S} = \mathcal{Y}_\mathcal{T}$. The objective is to develop a universal classifier that accounts for differences in the distributions of source and target domains. The point cloud, irrespective of its origin, serves as the input, while a semantic feature vector $\hat{y}$ represents the required output. The network parameters will be denoted as θ in the following.

All approaches will be performed using an adapted version of 3DUNet [15] and the PointTransformer [18] architectures. We aim to ensure that any architecture can be used within this procedure, regardless of its theoretical background. We have chosen them because they differ greatly in the representation used. In contrast to PointTransformer, which operates directly on the point cloud and employs nearest neighbor to delineate the region of focus, 3DUNet employs a discretization of the given scene, extending the concept of two-dimensional convolution to the third dimension. PointTransformer incorporates geometry by using multilayer perceptrons for encoding the point positions. Structured convolutions, such as 3DUNet, are by nature geometric encodings.

The networks comprise an initial encoding layer with no downsampling, followed by four downsampling layers and four upsampling layers, and a final class prediction layer. We will only use the x, y, z coordinates for training and not the color information, even if it should



(a) The unsupervised correlation alignment approach.

(b) Semi-supervised alignment approach.

Fig. 5. Networks processing data in an unsupervised manner (a) and a semi-supervised manner (b).

be given. The selection of the parameters of the architectures was conducted in accordance with the specifications outlined in the respective publications. The following downsampling rates are employed in the PointTransformer: $N \rightarrow \frac{N}{4} \rightarrow \frac{N}{16} \rightarrow \frac{N}{64} \rightarrow \frac{N}{256}$. Conversely, the feature dimension per point was expanded in each layer: $32 \rightarrow 64 \rightarrow 128 \rightarrow 256 \rightarrow 512$. The number of neighboring points used was limited to $k = 16$. In the 3DUNet model, the grid size and feature dimensions are doubled after each layer. This started with 32 and ended with the same number of feature channels as the point-based method. For our baseline, we only train the models on our synthetic data and simply validate them on the real data. For this reason, the optimization procedure simply looks like the following:

$$\min_\theta \left[\mathcal{L}(\mathcal{X}_S, \mathcal{Y}_S)\right] \tag{8}$$

4.2. Unsupervised domain adaptation

Due to the fact that the correlation is computed for the second order statistics of the feature space (Eq. (1)), the resulting covariance matrices are shaped like the feature space. This allows this approach to be used even when the amount of points between the source and target clouds are different. The networks are split into encoder $\mathcal{E}$ and decoder D. For every iteration the network sees a sub-cloud from the source domain $\mathcal{X}_S$ and the target domain $\mathcal{X}_T$. For the feature alignment we make use of the following:

$$\mathcal{L}_C = \|C_S^{log} - C_T^{log}\|_F^2 \tag{9}$$

We get rid of the factor $\frac{1}{4d^2}$ because we will scale the correlation loss within our target later. As mentioned in [55], the alignment can be done in several steps within the network. For this reason, we align the feature matrix within the latent space $\mathcal{L}_C^1$ and the logit output of D, which is noted as $\mathcal{L}_C^2$. The available labels $\mathcal{Y}_T$ are also used within the loss. The resulting minimization problem with $\lambda_1, \lambda_2 > 0$ looks like the following:

$$\min_\theta \left[\mathcal{L}(\mathcal{X}_S, \mathcal{Y}_S) + \lambda_1 \mathcal{L}_C^1 + \lambda_2 \mathcal{L}_C^2\right] \tag{10}$$

For each point cloud $\mathcal{X}_S$, a point cloud $\mathcal{X}_T$ is also inserted into the network. The selection of the point clouds is random. The whole procedure is visualized in 5(a). The values for λ_1 and λ_2 are critical and will be discussed in Section 5.3.

4.3. Semi supervised domain adaptation

The objective is to facilitate learning by utilizing comparable sub-clouds instead of aligning features. This implies that the network should only be exposed to a limited amount of real-world data that displays identical object classes but with varying appearances. To measure the similarity of point clouds, labels are used. The principle is that point clouds with the same classes are considered similar, while those with different classes are not. For each sub-cloud, we create the label vectors $\mathbf{l}_S \in \mathbb{R}^c$ and $\mathbf{l}_T \in \mathbb{R}^c$ holding the amount of points belonging to each class. To measure the similarity, we simply use the cosine similarity between the two vectors:

$$\text{sim}(\mathbf{l}_S, \mathbf{l}_T) = \frac{\mathbf{l}_S \cdot \mathbf{l}_T}{\|\mathbf{l}_S\| \|\mathbf{l}_T\|} \tag{11}$$

During the training, we will present a synthetic sub-cloud $\mathcal{X}_S$ and the real sub-cloud $\mathcal{X}_T$ that is as similar to it as possible before the optimization. The labels of both sub-clouds will be used for calculating the loss:

$$\min_\theta \left[\mathcal{L}(\mathcal{X}_S, \mathcal{Y}_S) + \mathcal{L}(\mathcal{X}_T, \mathcal{Y}_T)\right] \tag{12}$$

Given the limited quantity of available target data, it is possible that the same target data may be seen on more than one occasion within an epoch. A more detailed examination of this topic will be presented in Section 5.4. The full approach is visualized in 5(b).

5. Experiments

5.1. Dataset and baseline

Our data is generated as described in 3. For testing, we use the data provided by [25], where 10 bridges around Cambridgeshire have been recorded and annotated. With minor adjustments, they match our generated data quite well in terms of labels. Preprocessing all the given point clouds and downsampling them to 1 cm voxels results in the number of points given in Table 2. An example for a real world bridge and a synthetic generated point cloud can be seen in Fig. 6. Since we do not render colors within our pipeline we use the labels for easier visualization.

The division of the real-world data is contingent upon the chosen approach. In the case of the SSDA approach, bridges 2 and 10 are

(a) Real world bridge (b) Synthetic bridge with label

Fig. 6. Point clouds showing a bridge from [25] and a bridge which was automatically generated. Labels are used for easier visualization (color code is defined in Table 2).

Table 2
Semantic segmentation class list with amount of points belonging to it.

Class	ID	Synth	Real	
			SSDA	Test
Unlabeled	0	628.371	271.324	1.498.550
Ground	1	66.630.880	7.738.267	32.243.140
High vegetation	2	10.532.641	2.090.353	4.769.581
Abutment	3	10.838.768	579.530	3.581.851
Superstructure	4	42.497.680	4.605.359	16.920.609
Top surface	5	48.302.362	1.919.373	7.921.600
Railing	6	2.169.738	954.108	3.457.193
Traffic signs	7	341.185	2.635	14.372
Pillar	8	3.462.674	944.899	6.429.369
$\sum$		185.404.299	19.105.848	76.836.265

utilized for training purposes, while all other bridges are employed for testing. It is essential to note that the bridges exhibit considerable similarity, with a consistent presence of a road that serves as an obstacle.

Two baseline experiments are conducted. The first employs exclusively synthetic data for training and exclusively real-world data for testing. This experiment is also used to ascertain the actual discrepancy between the different domains. The second experiment serves as the baseline for the SSDA approach, incorporating the two real-world bridges into the training data. This results in a mixture of synthetic and real data, comprising 12 times more synthetic bridges than real ones. In order to determine the impact of the 3D representation, both of the described architectures were employed.

5.2. Implementation details

All experiments will employ voxel downsampling as a preprocessing technique with a voxel size of 10 cm. It was determined that a relatively high voxel size would be advantageous for the purposes of eliminating at least some of the scanning pattern and to allow for increasing the size of the subsequent defined sub-areas. As the point clouds cover large areas, we sample sub-areas $\mathcal{X}$ of the entire point clouds P using spherical neighborhoods with a radius of 5 m around a random chosen x_c resulting in $\mathcal{X}^p = \{x_i \in P \mid \|x_i - x_c\| \leq r\}$. We sample as many centres x_c as necessary to get each point at least once. For the discretization-based approach, we follow the same idea but make use a cube with an edge length of 2 times the radius, resulting in $\mathcal{X}^p = \{x_i \in P \mid |x_i^x - x_c^x| \leq r,\ |x_i^y - x_c^y| \leq r,\ |z_i^z - x_c^z| \leq r\}$. The network is then trained on these sub-clouds. To improve generalization, these sub-clouds are regenerated every 10 epochs by randomly sampling new centering points x_c. To further avoid overfitting the sub sampled data gets augmented. In the initial step, a random number of points, specified by the distribution $\mathcal{U}(0, 0.3)$, is selected for removal. Subsequently, the coordinates of each point gets altered by the value of $\mathcal{U}(-2, 2)$. The point cloud is subject to random rotation around the z-axis, with an angular range between 30° to 330°. The point cloud gets scaled by a value of $\mathcal{U}(0.8, 1.2)$ and Gaussian noise with $\mathcal{N}(\mu = 0, \sigma^2 = 0.02)$ is added. It is noted that each augmentation, with the exception of the initial augmentation, is applied independently, with a probability of 0.5.

For evaluation we use the mean Intersection over Union (mIoU) and the mean accuracy (mAcc):

$$mIoU = \frac{1}{C} \sum_{c \in C} \frac{TP_c}{TP_c + FP_c + FN_c} \tag{13}$$

$$mAcc = \frac{1}{C} \sum_{c \in C} \frac{TP_c}{TP_c + FN_c} \tag{14}$$

where TP_c, FP_c and FN_c refer to the number of points categorized as true positive, false positive and false negative with respect to class c. In addition, the mean class Accuracy (mAcc) is also calculated using all true positives and the total amount of predicted points.

It is evident that the classes contained within the dataset are imbalanced (see Table 2). We calculate a weight for each class $w_c = \frac{1}{\sqrt{f_c}}$ and use it within the cross-entropy loss $\mathcal{L}_{\text{wce}}$:

$$\mathcal{L}_{\text{wce}} = -\frac{1}{\sum_{c \in C} w_c} \sum_{c \in C} w_c y_c \log \hat{y}_c \tag{15}$$

Given our interest in achieving a high mIoU, the Lovász-Softmax loss $\mathcal{L}_{\text{ls}}$ [83] is integrated into our loss calculation. It is defined as:

$$\mathcal{L}_{\text{ls}} = -\frac{1}{|C|} \sum_{c \in C} \Delta_{J_c}(\boldsymbol{e}(c)) \tag{16}$$

where $\boldsymbol{e}(c)$ is the vector of pixel errors for class c and Δ_{J_c} is the Lovász extension of the IoU. The employment of a linear combination of both losses $\mathcal{L} = \mathcal{L}_{\text{wce}} + \mathcal{L}_{\text{ls}}$, allows the optimization of pixel-wise accuracy and the IoU.

All implementations were done with PyTorch [84]. The Stochastic Gradient Descent (SGD) optimizer is employed in conjunction with a momentum term set to 0.9 and a weight decay parameter set to 0.0001. The initial learning rate is set to 0.01, and the exponential decay factor is given by 0.95.

5.3. Unsupervised domain adaptation

By combining the minimization problem defined in Eq. (10) with the loss function defined in Section 5.2 the overall loss function looks like the following:

$$\begin{aligned} \mathcal{L}(\mathcal{X}_S, \mathcal{Y}_S, \mathcal{X}_T) = {} & 0.5\mathcal{L}_{\text{wce}}(\mathcal{X}_S, \mathcal{Y}_S) \\ & + 0.5\mathcal{L}_{\text{ls}}(\mathcal{X}_S, \mathcal{Y}_S) \\ & + \lambda_1 \mathcal{L}_C^1(\mathcal{Z}_S, \mathcal{Z}_T) \\ & + \lambda_2 \mathcal{L}_C^2(\hat{\mathcal{X}}_S, \hat{\mathcal{X}}_T) \end{aligned} \tag{17}$$

The values for λ_1 and λ_2 are considered hyperparameters and are varied with the current epoch t. It is ensured that the sum equals one. The values for λ_1 and λ_2 are varied with each epoch t, using the idea of [85]. To ensure that the values run in opposite directions, we adjust the equation as follows:

$$\begin{aligned} \lambda_1^t &= \lambda_{min} + \frac{1}{2}(\lambda_{max} - \lambda_{min}) \left(1 + \cos\left(\frac{t}{T_{max}}\pi\right)\right) \\ \lambda_2^t &= \lambda_{min} + \frac{1}{2}(\lambda_{max} - \lambda_{min}) \left(1 - \cos\left(\frac{t}{T_{max}}\pi\right)\right) \end{aligned} \tag{18}$$

with $\lambda_{min} = 0.2$, $\lambda_{max} = 0.8$ and T_{max} being the final epoch.

To enhance the evaluation of behavior, the error functions are tracked independently. Fig. 7 displays the results. You can clearly see that the loss with the labels $\mathcal{Y}_S$ decreases continuously, as usual. The

Table 3
Quantitative comparisons of approaches using IoU. Results for the baseline, UDA, random combination, and SSDA, respectively. (In this table, PointTransformer is abbreviated as PT and UNet3D as U3D.)

$\mathcal{X}_S$, $\mathcal{Y}_S$	$\mathcal{X}_T$	$\mathcal{Y}_T$	Model	Unlabeled	Ground	High vegetation	Abutment	Superstructure	Top surface	Railing	Traffic signs	Pillar	mIoU	mAcc
✓			U3D	**0.336**	0.475	0.324	0.079	0.704	0.463	**0.543**	0.0	0.549	0.386	0.662
			PT	0.004	0.573	0.412	0.213	0.484	0.551	0.495	0.0	0.255	0.332	0.623
✓	✓		U3D	0.196	**0.595**	**0.64**	**0.256**	**0.732**	0.371	0.343	0.0	**0.605**	**0.415**	**0.701**
			PT	0.11	0.477	0.555	0.121	0.355	**0.604**	0.388	0.0	0.327	0.326	0.532
✓	✓	✓	U3D	0.775	0.867	0.72	0.49	0.822	0.776	0.824	**0.014**	**0.772**	0.673	0.885
			PT	0.907	**0.905**	**0.926**	0.276	0.749	0.724	0.838	0.0	0.0	0.592	0.881
✓	✓	✓	U3D	**0.932**	0.894	0.796	**0.544**	**0.852**	**0.81**	**0.855**	0.0	0.744	**0.714**	**0.903**
			PT	0.554	0.831	0.786	0.383	0.706	0.767	0.718	0.0	0.606	0.594	0.783

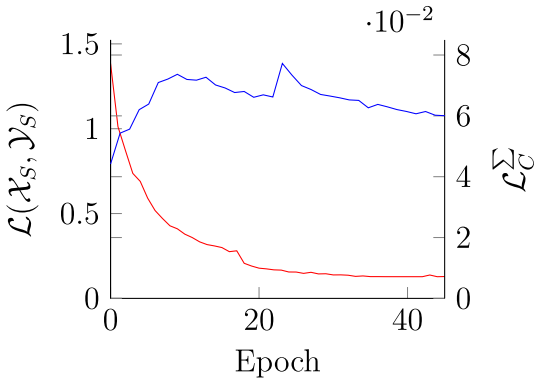


Fig. 7. Loss over epochs. The segmentation loss $\mathcal{L}(\mathcal{X}_S, \mathcal{Y}_S)$ (red) decreases while the correlation loss $\mathcal{L}_C^{\Sigma}$ initially increases but stabilizes over the epochs.

correlation alignment loss, on the other hand, first increases and then stabilizes over time. Interestingly, the recalculation of the data after 10 epochs is not visible within the loss.

5.4. Semi supervised domain adaptation

The loss function changes as follows because both the source and target domain labels are used:

$$\begin{aligned}\mathcal{L}(\mathcal{X}_S, \mathcal{Y}_S, \mathcal{X}_T, \mathcal{Y}_T) = \mathcal{L}_{\text{wce}}(\mathcal{X}_S, \mathcal{Y}_S) \\ + \mathcal{L}_{\text{ls}}(\mathcal{X}_S, \mathcal{Y}_S) \\ + \mathcal{L}_{\text{wce}}(\mathcal{X}_T, \mathcal{Y}_T) \\ + \mathcal{L}_{\text{ls}}(\mathcal{X}_T, \mathcal{Y}_T)\end{aligned} \tag{19}$$

The Eq. (11) is used to calculate similar sub-point clouds and guide the network accordingly. An illustration of which point clouds are similar and which are not can be found in Fig. 8. It is evident that the similarity measure operates as expected. Given that the vectors can be calculated within the preprocessing and that there are only c elements, this can be implemented in a timely and straightforward manner, readily integrated into the training routine.

As previously stated in Section 5.1, we employ a greater number of synthetic bridges during training than real ones. Consequently, it is possible for the same real sub-cloud $\mathcal{X}_T$ to be displayed multiple times within an epoch. Fig. 9 depicts the tracked center points, x_c, over a selected frequency. The training has run for 10 epochs, and it is evident that the partial point clouds in the areas surrounding the abutment are selected at a significantly higher rate than those that only show the ground or the vegetation zones. The most frequently selected point was selected 182 times during the 10 epochs, indicating that this sub-cloud was used 18 times more frequently for optimization than one of the synthetic sub-clouds.

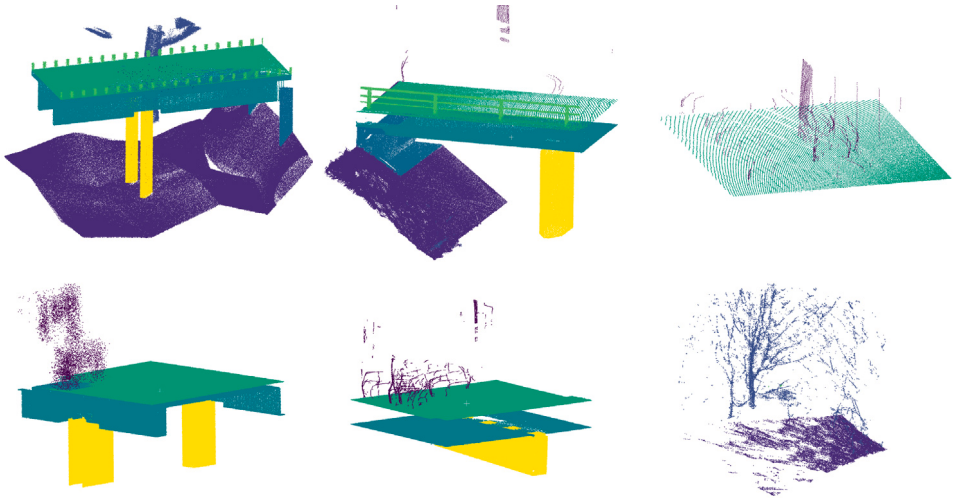
5.5. Comparison and results

The results of all experiments are summarized in Table 3. The initial section presents the baseline results for the exclusive utilization of synthetic data during training. The subsequent section illustrates the UDA outcomes. The Section 3 represents our baseline for the random combination of real and synthetic data. The final section shows our proposed SSDA approach using similar sub-clouds for guidance. The UDA approach does achieve a slight improvement over the baseline experiment using the discretization-based approach. However, the continuous-based approach results in a slight decrease in performance when aligning the feature unsupervised. Regardless of the approach chosen, the use of a limited set of labels is associated with a significant improvement in performance. The guided training approach yields further improvements in results. Notably, the mIoU remains unchanged for the continuous approach, but it does increase when only relevant bridge parts are considered. It should be noted that no synthetic training data was employed in the testing process. This was done intentionally, as it was believed that overfitting was occurring with the training data in use. Nevertheless, this is not a substantial problem, as the objective is to generalize as much as possible to the real data, which was validated using one of the SSDA bridges on a regular basis during training.

The results of the SSDA approaches are additionally shown in Fig. 10. The first column displays the colored point cloud, the second column presents predictions employing the discretized method, and the third column showcases the model utilizing the continuous approach. The initial five bridges originate from the dataset referenced by [25], whereas the last four bridges were recorded in Freiburg, Germany. For the bridges in Freiburg, we have chosen bridges that are as different as possible to the ones given. We also used a different sensor (Leica Geosystems RTC360). The outcomes for the bridges from [25] are promising, with the majority of classes predicted accurately. As expected, by modifying the bridge type and sensor, the outcomes have deteriorated. However, they remain reasonable for both approaches. The outcomes predicted by the discretization-based approach are slightly better than those achieved by the continuous approach.

5.6. Limitations

A potential limitation of this study is the similarity in the typology of bridges utilized, as all bridges are of the same type, namely a beam bridge. Although the bridge parts generated are of varying sizes, situated in different environments and employing disparate shapes, this homogeneity in typology may nevertheless introduce bias in the data. This gives rise to two principal questions, which remain unanswered in the present study due to the unavailability of the requisite data. Firstly, is the variance of generated beam bridges sufficient to allow for the segmentation of even the most dissimilar bridges of this type? Secondly, how do the models perform in the event of encountering a different bridge type, such as an arch bridge? We suggest that these questions could be investigated in future work.



(a) Selected synthetic sub-cloud (b) Similiar real sub-cloud (c) Not similar real sub-cloud

Fig. 8. Similar (middle) and different (right) calculated sub-clouds. The first column shows the selected synthetic cloud (Color code is defined in Table 2.).

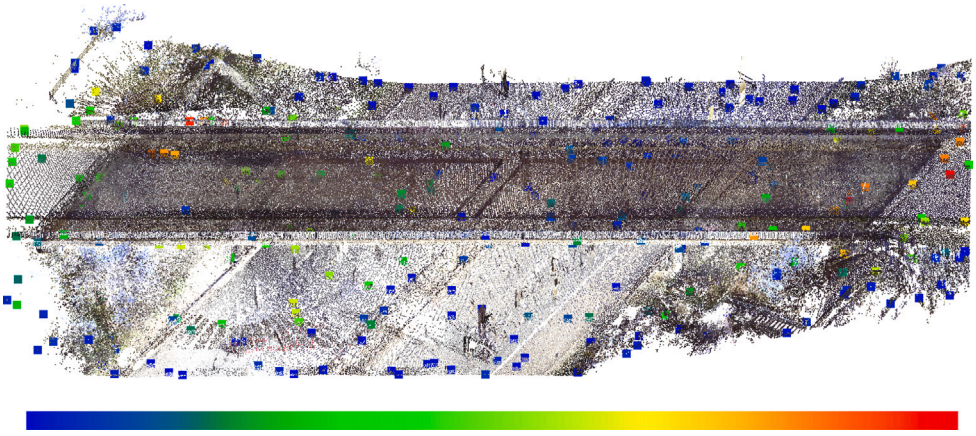


Fig. 9. Selected positions for similar sub-cloud for 10 epochs. Color is scaled linear between 1 (blue) to 182 (red).

6. Discussion

It is crucial to highlight that, in the UDA approach, a random selection between sub-clouds from the source and target domains is not necessary to achieve a match. This is because the similarity match was not employed in this experiment. Nevertheless, we conducted an experiment where the similarity measure was utilized to identify the optimal matching cloud. Our objective was to enhance the metric in this manner, with the eventual goal of modifying the similarity measure to obviate the necessity for labels. Unfortunately, the performance only marginally improved, yet it remains considerably inferior to that achieved by employing a minimal number of labels within the optimization process. The results indicate that the use of the MECA method to align networks at multiple locations has the potential to enhance performance, although the extent of this improvement is limited.

Another noteworthy observation is that, across all experiments conducted across diverse domains, the discretization approach demonstrated superior performance compared to the continuous approach. This was evidenced in all four experiments, where only synthetic data was used for training, the UDA approach for feature alignment, the SSDA approach using guiding as well as the simple approach of shuffling the data and training on all data led to this realization. It has been demonstrated that the continuous approach, trained on the real-world domain, yields superior outcomes compared to the discretization approach. It can be postulated that the discretization step may eliminate certain specific scanning patterns, rendering this approach more suitable for generalization across different domains. Conversely, the continuous approach may be more conducive to the learning and utilization of patterns within a specific domain.

It is evident that utilizing a small proportion of the target data set in conjunction with the annotation during the training phase can result in a considerably greater performance increase than simply using them in an unsupervised manner. Furthermore, by employing the approach presented here to show similar point clouds from different domains in parallel during optimization, the result can be improved even further. This raises the question of whether the generalization for real objects can be further improved by selecting the most diverse real objects possible and incorporating additional synthetic data.

Another crucial question is how the performance metrics change when the sensor is switched. Our experiments demonstrated that performance does degrade when the sensor is changed, but we cannot determine if the change in sensor or the change in bridge environment is the cause. For this reason, further investigation will be conducted in a subsequent study, where the integration of multiple real sensors into the guiding process will be explored.

7. Conclusion

This paper addressed the problem of 3D semantic segmentation of bridges. A data generation pipeline that is fully automated was

(a) Colored bridges (b) 3DUNet (c) PointTransformer

Fig. 10. Prediction results using the SSDA approaches. The Section 1 presents the outcomes of the data provided by [25]. The Section 2 presents the outcomes of the data recorded on bridges in Freiburg, Germany using a different sensor. Color code is defined in Table 2.

developed because there was not enough annotated data. Using this pipeline, we have generated 24 annotated point clouds of bridges and their surroundings. We propose an unsupervised method that employs feature alignment and a semi-supervised approach to guide training using a limited number of annotated real-world bridges. To validate our proposed methods, we scanned and manually annotated real-world bridges. To validate the results against different 3D representations, two different deep learning architectures were used for both approaches. By incorporating the real-world data in an unsupervised manner, we observed a slight improvement. However, this is still significantly less than what can be achieved with the semi-supervised approach. The incorporation of approximately 10% of real-world data into the training data set resulted in a notable enhancement in the evaluation metric, nearly doubling the baseline. Using the similarity between the training data and the real-world data to guide the joint optimization resulted in an even more significant improvement, particularly with respect to the relevant classes of bridges. Despite the success of this work in automatic semantic segmentation for bridges, further investigation is necessary on several topics. For instance, it is necessary to examine the extent to which performance is affected by the use of a different sensor and the degree to which generalization can be achieved using the approach. Additionally, the transferability of the approach to different bridge types, such as arch bridges, should be investigated. To this end, a greater number of different bridge types should be measured.

CRediT authorship contribution statement

Maximilian Kellner: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Timothy König:** Writing – original draft, Data curation. **Jan-Iwo Jäkel:** Writing – review & editing, Project administration. **Katharina Klemt-Albert:** Resources, Funding acquisition. **Alexander Reiterer:** Writing – review & editing, Supervision, Resources, Funding acquisition.

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.

Acknowledgments

The work has been funded by the BMDV as part of the mFUND project "Partially automated creation of object-based inventory models using multi-data fusion of multimodal data streams and existing inventory data - mdfBIM+" (FKZ: 19FS2021B). The authors would like to thank Ruodan Lu and Ioannis Brilakis for sharing the bridge data with us. We would also like to thank all the creative designers who have made their objects publicly available for free.

Data availability

Data will be made available on request.

References

[1] K. Klemt-Albert, R. Hartung, S. Bahlau, Enhancing resilience of traffic networks with a focus on impacts of neuralgic points like urban tunnels, in: Resilience Engineering for Urban Tunnels, 2018, pp. 55–70, http://dx.doi.org/10.1061/9780784415139.ch05, (Ch. Chapter 5). arXiv:https://ascelibrary.org/doi/pdf/10.1061/9780784415139.ch05 URL https://ascelibrary.org/doi/abs/10.1061/9780784415139.ch05.

[2] B.f.S. BASt, Zustandsnoten der Brücken, 2023, https://www.govdata.de/web/guest/daten/-/details/zustandsnoten-der-bruckenfe8db. (Accessed 29 September 2023), Data licence Germany – attribution – Version 2.0.

[3] J.-I. Jäkel, R. Hartung, K. Klemt-Albert, A concept of an automated damage management for the maintenance of bridge structures in the context of a life cycle oriented approach, in: Proceedings of the 2022 European Conference on Computing in Construction, in: Computing in Construction, Vol. 3, European Council on Computing in Construction, Rhodes, Greece, ISBN: 978-8-875902-26-1, 2022, http://dx.doi.org/10.35490/EC3.2022.211, URL https://ec-3.org/publications/conference/paper/?id=EC32022_211.

[4] F. Bosché, M. Ahmed, Y. Turkan, C.T. Haas, R. Haas, The value of integrating scan-to-BIM and Scan-vs-BIM techniques for construction monitoring using laser scanning and BIM: The case of cylindrical MEP components, Autom. Constr. (ISSN: 0926-5805) 49 (2015) 201–213, http://dx.doi.org/10.1016/j.autcon.2014.05.014, URL https://www.sciencedirect.com/science/article/pii/S0926580514001319, 30th ISARC Special Issue.

[5] M. Kellner, H. Vassilev, A. Busch, R. Blaskow, M. Ferrandon Cervantes, K.N. PokuAgyemang, A. Schmitt, S. Weisbrich, H. Maas, F. Neitzel, A. Reiterer, J. Blankenbach, Scan2BIM - A review on the automated creation of semantic aware geometric as-is models of bridges, Z. Bereiche Geodäsie Geoinf. 03/2024 (2024) 159–181, http://dx.doi.org/10.14627/avn.2024.3.4.

[6] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature (ISSN: 1476-4687) 521 (7553) (2015) 436–444, http://dx.doi.org/10.1038/nature14539.

[7] D. Merkle, A. Reiterer, Overview of 3D point cloud annotation and segmentation techniques for smart city applications, 2022, http://dx.doi.org/10.1117/12.2635845.

[8] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, J. Vaughan, A theory of learning from different domains, Mach. Learn. 79 (2010) 151–175, URL http://www.springerlink.com/content/q6qk230685577n52/.

[9] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359, http://dx.doi.org/10.1109/TKDE.2009.191.

[10] A. Farahani, S. Voghoei, K. Rasheed, H.R. Arabnia, A brief review of domain adaptation, in: R. Stahlbock, G.M. Weiss, M. Abou-Nasr, C.-Y. Yang, H.R. Arabnia, L. Deligiannidis (Eds.), Advances in Data Science and Information Engineering, Springer International Publishing, Cham, 2021, pp. 877–894, http://dx.doi.org/10.1007/978-3-030-71704-9_65.

[11] M. Kellner, B. Stahl, A. Reiterer, Reconstructing geometrical models of indoor environments based on point clouds, Remote Sens. (ISSN: 2072-4292) 15 (18) (2023) http://dx.doi.org/10.3390/rs15184421, URL https://www.mdpi.com/2072-4292/15/18/4421.

[12] B.f.D.u.V. BMDV, Brücken an bundesfernstraß en bilanz und ausblick, 2024, https://bmdv.bund.de/SharedDocs/DE/Anlage/K/presse/bruecken-an-bundesfernstrassen-bilanz-und-ausblick.pdf?_blob=publicationFile. (Accessed 10 September 2024), Data licence Germany – attribution – Version 2.0.

[13] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, M. Bennamoun, Deep learning for 3D point clouds: A survey, 2019, CoRR abs/1912.12033 arXiv:1912.12033 URL http://arxiv.org/abs/1912.12033.

[14] A. Milioto, I. Vizzo, J. Behley, C. Stachniss, RangeNet++: Fast and Accurate LiDAR Semantic Segmentation, in: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS, 2019.

[15] Ö. Çiçek, A. Abdulkadir, S. Lienkamp, T. Brox, O. Ronneberger, 3D U-Net: Learning dense volumetric segmentation from sparse annotation, in: Medical Image Computing and Computer-Assisted Intervention, MICCAI, in: LNCS, Vol. 9901, Springer, 2016, pp. 424–432, URL http://lmb.informatik.uni-freiburg.de/Publications/2016/CABR16 (available on arXiv:1606.06650 [cs.CV]).

[16] C. Choy, J. Gwak, S. Savarese, 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, Los Alamitos, CA, USA, 2019, pp. 3070–3079, http://dx.doi.org/10.1109/CVPR.2019.00319, URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00319.

[17] H. Thomas, C.R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, L. Guibas, KPConv: Flexible and Deformable Convolution for Point Clouds, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV, IEEE Computer Society, Los Alamitos, CA, USA, 2019, pp. 6410–6419, http://dx.doi.org/10.1109/ICCV.2019.00651, URL https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00651.

[18] H. Zhao, L. Jiang, J. Jia, P. Torr, V. Koltun, Point transformer, in: 2021 IEEE/CVF International Conference on Computer Vision, ICCV, 2021, pp. 16239–16248, http://dx.doi.org/10.1109/ICCV48922.2021.01595.

[19] X. Wu, Y. Lao, L. Jiang, X. Liu, H. Zhao, Point transformer V2: grouped vector attention and partition-based pooling, in: Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS '22, Curran Associates Inc., Red Hook, NY, USA, ISBN: 9781713871088, 2024, http://dx.doi.org/10.48550/arXiv.2210.05666.

[20] X. Wu, L. Jiang, P.-S. Wang, Z. Liu, X. Liu, Y. Qiao, W. Ouyang, T. He, H. Zhao, Point Transformer V3: Simpler, Faster, Stronger, 2024, http://dx.doi.org/10.1109/CVPR52733.2024.00463, URL https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.00463.

[21] Y.-Q. Yang, Y.-X. Guo, J. Xiong, Y. Liu, H. Pan, P.-S. Wang, X. Tong, B. Guo, Swin3D: A pretrained transformer backbone for 3D indoor scene understanding, 2023, URL https://api.semanticscholar.org/CorpusID:258170015.

[22] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, F. Wei, B. Guo, Swin transformer V2: Scaling up capacity and resolution, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2022, pp. 11999–12009, http://dx.doi.org/10.1109/CVPR52688.2022.01170.

[23] M. Kellner, B. Stahl, A. Reiterer, Fused projection-based point cloud segmentation, Sensors (ISSN: 1424-8220) 22 (3) (2022) http://dx.doi.org/10.3390/s22031139, URL https://www.mdpi.com/1424-8220/22/3/1139.

[24] H. Yue, Q. Wang, H. Zhao, N. Zeng, Y. Tan, Deep learning applications for point clouds in the construction industry, Autom. Constr. (ISSN: 0926-5805) 168 (2024) 105769, http://dx.doi.org/10.1016/j.autcon.2024.105769, URL https://www.sciencedirect.com/science/article/pii/S0926580524005053.

[25] R. Lu, I. Brilakis, C.R. Middleton, Detection of structural components in point clouds of existing RC bridges, Comput.-Aided Civ. Infrastruct. Eng. 34 (3) (2019) 191–212, http://dx.doi.org/10.1111/mice.12407, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/mice.12407 URL https://onlinelibrary.wiley.com/doi/abs/10.1111/mice.12407.

[26] 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. (ISSN: 0926-5805) 142 (2022) 104459, http://dx.doi.org/10.1016/j.autcon.2022.104459, URL https://www.sciencedirect.com/science/article/pii/S0926580522003326.

[27] 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. (ISSN: 0926-5805) 142 (2022) 104519, http://dx.doi.org/10.1016/j.autcon.2022.104519, URL https://www.sciencedirect.com/science/article/pii/S0926580522003922.

[28] 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. (ISSN: 0926-5805) 150 (2023) 104838, http://dx.doi.org/10.1016/j.autcon.2023.104838, URL https://www.sciencedirect.com/science/article/pii/S0926580523000985.

[29] 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. (ISSN: 0926-5805) 158 (2024) 105176, http://dx.doi.org/10.1016/j.autcon.2023.105176, URL https://www.sciencedirect.com/science/article/pii/S0926580523004363.

[30] F. Chen, F. Wu, G. Gao, Y. Ji, J. Xu, G.-P. Jiang, X.-Y. Jing, JSPNet: Learning joint semantic & instance segmentation of point clouds via feature self-similarity and cross-task probability, 122, (ISSN: 0031-3203) 2022, 108250, http://dx.doi.org/10.1016/j.patcog.2021.108250, URL https://www.sciencedirect.com/science/article/pii/S0031320321004301,

[31] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, A. Markham, RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, Los Alamitos, CA, USA, 2020, pp. 11105–11114, http://dx.doi.org/10.1109/CVPR42600.2020.01112, URL https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.01112.

[32] M. Mafipour, C. Alici, S. Shakeel, A. Kalkavan, Semantic segmentation of real and synthetic point cloud data for digital twinning of bridges, in: Proceedings of 33. Forum Bauinformatik, 2022, URL https://mediatum.ub.tum.de/node?id=1688410.

[33] L. Winiwarter, A.M. Esmorís Pena, H. Weiser, K. Anders, J. Martínez Sánchez, M. Searle, B. Höfle, Virtual laser scanning with HELIOS++: A novel take on ray tracing-based simulation of topographic full-waveform 3D laser scanning, Remote Sens. Environ. (ISSN: 0034-4257) 269 (2022) http://dx.doi.org/

10.1016/j.rse.2021.112772, URL https://www.sciencedirect.com/science/article/pii/S0034425721004922.

[34] A. Dai, A.X. Chang, M. Savva, M. Halber, T. Funkhouser, M. Niessner, ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, Los Alamitos, CA, USA, (ISSN: 1063-6919) 2017, pp. 2432–2443, http://dx.doi.org/10.1109/CVPR.2017.261, URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.261.

[35] I. Armeni, O. Sener, A.R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3D Semantic Parsing of Large-Scale Indoor Spaces, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, Los Alamitos, CA, USA, (ISSN: 1063-6919) 2016, pp. 1534–1543, http://dx.doi.org/10.1109/CVPR.2016.170, URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.170.

[36] D. Munoz, N. Vandapel, M. Hebert, Onboard contextual classification of 3-D point clouds with learned high-order Markov random fields, in: 2009 IEEE International Conference on Robotics and Automation, 2009, pp. 2009–2016, http://dx.doi.org/10.1109/ROBOT.2009.5152856.

[37] T. Hackel, N. Savinov, L. Ladicky, J.D. Wegner, K. Schindler, M. Pollefeys, SEMANTIC3D.NET: A new large-scale point cloud classification benchmark, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, IV-1-W1, 2017, pp. 91–98, URL https://api.semanticscholar.org/CorpusID:7526065.

[38] X. Roynard, J.-E. Deschaud, F. Goulette, Paris-Lille-3D: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification, Int. J. Robot. Res. 37 (6) (2018) 545–557, http://dx.doi.org/10.1177/0278364918767506, arXiv:https://doi.org/10.1177/0278364918767506.

[39] Y. Pan, B. Gao, J. Mei, S. Geng, C. Li, H. Zhao, SemanticPOSS: A point cloud dataset with large quantity of dynamic instances, in: 2020 IEEE Intelligent Vehicles Symposium, IV, 2020, pp. 687–693, http://dx.doi.org/10.1109/IV47402.2020.9304596.

[40] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, J. Gall, C. Stachniss, Towards 3D LiDAR-based semantic scene understanding of 3D point cloud sequences: The SemanticKITTI Dataset, 40, (8–9) 2021, pp. 959–967, http://dx.doi.org/10.1177/02783649211006735,

[41] A. Xiao, J. Huang, D. Guan, F. Zhan, S. Lu, Transfer learning from synthetic to real LiDAR point cloud for semantic segmentation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 2795–2803, URL https://arxiv.org/abs/2107.05399.

[42] N. Varney, V.K. Asari, Q. Graehling, DALES: A Large-scale Aerial LiDAR Data Set for Semantic Segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, CVPRW, IEEE Computer Society, Los Alamitos, CA, USA, 2020, pp. 717–726, http://dx.doi.org/10.1109/CVPRW50498.2020.00101, URL https://doi.ieeecomputersociety.org/10.1109/CVPRW50498.2020.00101.

[43] F. Rottensteiner, G. Sohn, J. Jung, M. Gerke, C. Baillard, S. Benitez, U. Breitkopf, The ISPRS benchmark on urban object classification and 3D building reconstruction, ISPRS Ann. Photogramm. Remote. Sens. Spat. Inf. Sci. I- 3 Nr. 1 1 (1) (2012) 293–298, http://dx.doi.org/10.5194/isprsannals-I-3-293-2012, (2012).

[44] Q. Hu, B. Yang, S. Khalid, W. Xiao, N. Trigoni, A. Markham, Towards semantic segmentation of urban-scale 3D point clouds: A dataset, benchmarks and challenges, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, http://dx.doi.org/10.48550/arXiv.2009.03137.

[45] M. Chen, Q. Hu, T. Hugues, A. Feng, Y. Hou, K. McCullough, L. Soibelman, STPLS3D: A large-scale synthetic and real aerial photogrammetry 3D point cloud dataset, 2022, arXiv:2203.09065 URL https://arxiv.org/abs/2203.09065.

[46] E. Agapaki, A. Glyn-Davies, S. Mandoki, I. Brilakis, CLOI: A shape classification benchmark dataset for industrial facilities, American Society of Civil Engineers, 2019, http://dx.doi.org/10.17863/CAM.36600, URL https://www.repository.cam.ac.uk/handle/1810/289351.

[47] C. Yin, B. Wang, V.J. Gan, M. Wang, J.C. Cheng, Automated semantic segmentation of industrial point clouds using ResPointNet++, Autom. Constr. (ISSN: 0926-5805) 130 (2021) 103874, http://dx.doi.org/10.1016/j.autcon.2021.103874, URL https://www.sciencedirect.com/science/article/pii/S0926580521003253.

[48] Y. Zhang, H. Zhang, B. Deng, S. Li, K. Jia, L. Zhang, Semi-supervised models are strong unsupervised domain adaptation learners, 2021, http://dx.doi.org/10.48550/arXiv.2106.00417.

[49] A. Singh, CLDA: Contrastive learning for semi-supervised domain adaptation, in: A. Beygelzimer, Y. Dauphin, P. Liang, J.W. Vaughan (Eds.), Advances in Neural Information Processing Systems, 2021, URL https://openreview.net/forum?id=1ODSsnoMBav.

[50] M. Wang, W. Deng, Deep visual domain adaptation: A survey, Neurocomputing (ISSN: 0925-2312) 312 (2018) 135–153, http://dx.doi.org/10.1016/j.neucom.2018.05.083, URL https://www.sciencedirect.com/science/article/pii/S0925231218306684.

[51] L. Bruzzone, M. Marconcini, Domain adaptation problems: A DASVM classification technique and a circular validation strategy, IEEE Trans. Pattern Anal. Mach. Intell. 32 (5) (2010) 770–787, http://dx.doi.org/10.1109/TPAMI.2009.57.

[52] M. Gheisari, M.S. Baghshah, Unsupervised domain adaptation via representation learning and adaptive classifier learning, Neurocomputing (ISSN: 0925-2312) 165 (2015) 300–311, http://dx.doi.org/10.1016/j.neucom.2015.03.020, URL https://www.sciencedirect.com/science/article/pii/S0925231215002921.

[53] A. Gretton, K.M. Borgwardt, M.J. Rasch, B. Schölkopf, A. Smola, A kernel two-sample test, J. Mach. Learn. Res. 13 (25) (2012) 723–773, URL http://jmlr.org/papers/v13/gretton12a.html.

[54] Y. Rubner, C. Tomasi, L.J. Guibas, L.J. Guibas, The earth mover's distance as a metric for image retrieval, 2000, pp. 99–121, URL https://www.cs.duke.edu/~tomasi/papers/rubner/rubnerIjcv00.pdf,

[55] B. Sun, J. Feng, K. Saenko, Return of frustratingly easy domain adaptation, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI '16, AAAI Press, 2016, pp. 2058–2065, URL https://arxiv.org/abs/1511.05547.

[56] B. Sun, K. Saenko, Deep CORAL: Correlation alignment for deep domain adaptation, in: G. Hua, H. Jégou (Eds.), Computer Vision – ECCV 2016 Workshops, Springer International Publishing, Cham, ISBN: 978-3-319-49409-8, 2016, pp. 443–450, URL https://arxiv.org/abs/1607.01719.

[57] C. Chen, W. Xie, W. Huang, Y. Rong, X. Ding, Y. Huang, T. Xu, J. Huang, Progressive feature alignment for unsupervised domain adaptation, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, Los Alamitos, CA, USA, 2019, pp. 627–636, http://dx.doi.org/10.1109/CVPR.2019.00072, URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00072.

[58] R. Li, X. Jia, J. He, S. Chen, Q. Hu, T-SVDNet: Exploring High-Order Prototypical Correlations for Multi-Source Domain Adaptation, in: 2021 IEEE/CVF International Conference on Computer Vision, ICCV, IEEE Computer Society, Los Alamitos, CA, USA, 2021, pp. 9971–9980, http://dx.doi.org/10.1109/ICCV48922.2021.00984, URL https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.00984.

[59] Z. Cheng, C. Chen, Z. Chen, K. Fang, X. Jin, Robust and high-order correlation alignment for unsupervised domain adaptation, 33, (12) (ISSN: 1433-3058) 2021, pp. 6891–6903, http://dx.doi.org/10.1007/s00521-020-05465-7,

[60] P. Morerio, J. Cavazza, V. Murino, Minimal-entropy correlation alignment for unsupervised deep domain adaptation, Int. Conf. Learn. Represent. (2018) URL https://openreview.net/forum?id=rJWechg0Z.

[61] X. Ji, A. Vedaldi, J. Henriques, Invariant information clustering for unsupervised image classification and segmentation, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV, IEEE Computer Society, Los Alamitos, CA, USA, 2019, pp. 9864–9873, http://dx.doi.org/10.1109/ICCV.2019.00996, URL https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00996.

[62] S.E. Mirsadeghi, A. Royat, H. Rezatofighi, Unsupervised image segmentation by mutual information maximization and adversarial regularization, IEEE Robot. Autom. Lett. 6 (2021) 6931–6938, URL https://api.semanticscholar.org/CorpusID:235727535.

[63] Q. Hu, Y. Wei, J. Pang, M. Liang, Unsupervised domain adaptation for brain structure segmentation via mutual information maximization alignment, Biomed. Signal Process. Control (ISSN: 1746-8094) 90 (2024) 105784, http://dx.doi.org/10.1016/j.bspc.2023.105784, URL https://www.sciencedirect.com/science/article/pii/S174680942301217X.

[64] L.T. Triess, M. Dreissig, C.B. Rist, J.M. Zöllner, A survey on deep domain adaptation for LiDAR perception, in: 2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops), Institute of Electrical and Electronics Engineers (IEEE), ISBN: 978-1-66547-921-9, 2021, pp. 350–357, http://dx.doi.org/10.1109/IVWorkshops54471.2021.9669228.

[65] K. Saleh, A. Abobakr, M. Attia, J. Iskander, D. Nahavandi, M. Hossny, S. Nahvandi, Domain adaptation for vehicle detection from Bird's Eye View LiDAR point cloud data, in: 2019 IEEE/CVF International Conference on Computer Vision Workshop, ICCVW, IEEE Computer Society, Los Alamitos, CA, USA, 2019, pp. 3235–3242, http://dx.doi.org/10.1109/ICCVW.2019.00404, URL https://doi.ieeecomputersociety.org/10.1109/ICCVW.2019.00404.

[66] J.-Y. Zhu, T. Park, P. Isola, A.A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, in: 2017 IEEE International Conference on Computer Vision, ICCV, 2017, pp. 2242–2251, http://dx.doi.org/10.1109/ICCV.2017.244.

[67] B. Wu, X. Zhou, S. Zhao, X. Yue, K. Keutzer, SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud, 2019 Int. Conf. Robot. Autom. (ICRA) (2018) 4376–4382, URL https://api.semanticscholar.org/CorpusID:52815788.

[68] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: 2017 IEEE International Conference on Computer Vision, ICCV, 2017, pp. 2999–3007, http://dx.doi.org/10.1109/ICCV.2017.324.

[69] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The KITTI vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3354–3361, http://dx.doi.org/10.1109/CVPR.2012.6248074.

[70] I. Achituve, H. Maron, G. Chechik, Self-supervised learning for domain adaptation on point clouds, in: 2021 IEEE Winter Conference on Applications of Computer Vision, WACV, IEEE Computer Society, Los Alamitos, CA, USA, 2021, pp. 123–133, http://dx.doi.org/10.1109/WACV48630.2021.00017, URL https://doi.ieeecomputersociety.org/10.1109/WACV48630.2021.00017.

[71] Y. Song, Z. Sun, Y. Wu, Y. Sun, S. Luo, Q. Li, Learning semantic segmentation on unlabeled real-world indoor point clouds via synthetic data, in: 2022 26th International Conference on Pattern Recognition, ICPR, 2022, pp. 3750–3756, http://dx.doi.org/10.1109/ICPR56361.2022.9956612.

[72] Y. Shen, Y. Yang, M. Yan, H. Wang, Y. Zheng, L. Guibas, Domain adaptation on point clouds via geometry-aware implicits, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, Los Alamitos, CA, USA, 2022, pp. 7213–7222, http://dx.doi.org/10.1109/CVPR52688.2022.00708, URL https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.00708.

[73] S. Katageri, A. De, C. Devaguptapu, V.S.S.V. Prasad, C. Sharma, M. Kaul, Synergizing Contrastive Learning and Optimal Transport for 3D Point Cloud Domain Adaptation, in: 2024 IEEE/CVF Winter Conference on Applications of Computer Vision, WACV, IEEE Computer Society, Los Alamitos, CA, USA, 2024, pp. 2930–2939, http://dx.doi.org/10.1109/WACV57701.2024.00292, URL https://doi.ieeecomputersociety.org/10.1109/WACV57701.2024.00292.

[74] Y. Wang, J. Yin, W. Li, P. Frossard, R. Yang, J. Shen, SSDA3D: semi-supervised domain adaptation for 3D object detection from point cloud, in: Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence, in: AAAI'23/IAAI'23/EAAI'23, AAAI Press, ISBN: 978-1-57735-880-0, 2023, http://dx.doi.org/10.1609/aaai.v37i3.25370.

[75] Autodesk, revit, 2023, URL https://www.autodesk.com/products/revit/overview?term=1-YEAR&tab=subscription.

[76] Autodesk dynamo, 2023, URL https://primer2.dynamobim.org/.

[77] A. Dieckmann, Clockwork for dynamo, 2023, URL https://github.com/andydandy74/ClockworkForDynamo.

[78] Guidelines for the Design of Motorways, FGSV-Verlag, 2008, URL https://www.fgsv-verlag.de/pub/media/pdf/202_E_PDF.v.pdf.

[79] A. Denoyel, C. Pinson, P.-A. Passet, Sketchfab, 2012, URL https://sketchfab.com/features/free-3d-models.

[80] M. Kalytis, Cgtrader, 2011, URL https://www.cgtrader.com/de/frei-3d-modelle.

[81] Free3d, 1997, URL https://free3d.com/de/.

[82] M. Wisdom, A. Wisdom, Turbo squid, 2000, URL https://www.turbosquid.com/de/.

[83] M. Berman, A. Rannen Triki, M.B. Blaschko, The Lovasz-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4413–4421, http://dx.doi.org/10.1109/CVPR.2018.00464.

[84] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in PyTorch, 2017, URL https://openreview.net/forum?id=BJJsrmfCZ.

[85] I. Loshchilov, F. Hutter, SGDR: Stochastic gradient descent with warm restarts, in: International Conference on Learning Representations, 2017, URL https://openreview.net/forum?id=Skq89Scxx.