Received: 15 March 2024 | Accepted: 6 August 2024

DOI: 10.1111/2041-210X.14412

Methods in Ecology and Evolution

RESEARCH ARTICLE

Completing 3D point clouds of individual trees using deep learning

Aline Bornand[1,2] | Meinrad Abegg[1] | Felix Morsdorf[2] | Nataliia Rehush[1]

[1]Swiss National Forest Inventory, Swiss Federal Institute for Forest, Snow and Landscape Research WSL, Birmensdorf, Switzerland

[2]Department of Geography, University of Zurich, Zürich, Switzerland

Correspondence
Aline Bornand
Email: aline.bornand@wsl.ch

Funding information
Swiss National Forest Inventory

Handling Editor: Carlos Alberto Silva

Abstract

1. In close-range remote sensing data collected in a forest, occlusion often causes incomplete or sparse point cloud representations of individual trees, impeding accurate 3D reconstruction of tree architecture and estimation of tree height and volume. Recent developments in deep learning (DL) for 3D data have produced approaches for point cloud completion, which could potentially be applied to trees.
2. We explored the potential of a DL approach to fill gaps in dense point clouds representing the main structures of deciduous trees by applying an existing transformer-based completion model (PoinTr). Complete point clouds are required as training data, but even dense terrestrial laser scanning (TLS) data sets contain gaps caused by occlusion, making it nearly impossible to acquire such data. We therefore investigated the ability of point cloud completion models trained on a range of synthetic data sets to handle occlusion patterns in real-world point clouds.
3. Despite the limited data set, we successfully fine-tuned a general pre-trained completion model to fill gaps within 1 m^3 segments of tree point clouds. Fine-tuning on synthetic tree data improved the model's ability to complete tree objects compared with training on diverse artificial objects. However, the quality of the predictions was influenced by the level of sophistication of the synthetic data. Our results demonstrate that incorporating even limited real-world TLS data during training can considerably improve completion results but may introduce additional noise in the predictions.
4. 3D point cloud completion with DL has the potential to improve and fill gaps in point clouds of individual trees, facilitating further steps in the processing and analysis of 3D forest data.

KEYWORDS
artificial intelligence (AI), deep learning (DL), forest, LiDAR, point cloud completion, synthetic data, terrestrial laser scanning (TLS), trees

1 | INTRODUCTION

In forestry and ecology, LiDAR and other point cloud data sources are used extensively to map and monitor state and changes in forest ecosystems. In particular, close-range LiDAR technologies, such as terrestrial laser scanning (TLS), mobile laser scanning (MLS) and unoccupied aerial vehicle laser scanning (UAVLS), have gained prominence in recent years, with a surge of new applications for forests (Calders et al., 2020). These high-density point clouds enable precise measurements of tree structures in a non-destructive manner, facilitating various analyses, such as the estimation of tree structural parameters (Terryn et al., 2023), 3D vegetation density assessments (Grau et al., 2017), and above-ground volume and biomass calculations (Bornand et al., 2023; Demol et al., 2022). Many of these measurements and estimates have the potential to be used in forest inventories (Liang et al., 2016). However, employing close-range LiDAR in forest environments poses major challenges, primarily due to occlusion caused by the density of structures such as trees, branches and leaves (Kükenbrink et al., 2017). Occlusion, as well as increasing distance from the scanner, may result in sparsity and gaps in point cloud representations of individual trees, impeding accurate 3D reconstructions of tree architecture, detailed volume estimations (Morhart et al., 2024), and even tree height estimations (Wang et al., 2019). Careful planning of scanner placement when using TLS (Abegg et al., 2017) and selection of an optimal flight set-up when using UAVLS (Brede et al., 2022) can help reduce occlusions in the resulting point clouds. Data fusion—when TLS or photogrammetric point clouds are combined with UAVLS data—is also a commonly used strategy to deal with occlusion (Terryn et al., 2022). However, a dense and structurally complex forest environment, complex terrain and high costs of additional data acquisition (e.g. UAVLS) can make the acquisition of complete point clouds challenging.

The problem of handling incomplete point clouds of trees has previously been addressed by extracting curve skeletons (Tagliasacchi et al., 2009), using tubular shapes (Ravaglia et al., 2017), covering the occluded regions of tree stems with an a priori model (Morel et al., 2018), or using the tree skeleton approach and local features (Cao et al., 2022; Wang et al., 2023). However, little attention has been paid to addressing occlusion in 3D point clouds of natural objects, such as trees, using deep learning (DL) approaches. Meanwhile, in the forest domain, DL is currently becoming the standard to overcome the two bottlenecks of semantic segmentation tasks (Krisanski et al., 2021) and tree species classification (Allen et al., 2023; Xi et al., 2020).

In other domains, the development of DL-based techniques for reconstruction or completion tasks of 3D point clouds is gaining momentum (Fei et al., 2022). PointNet and its variations (Qi et al., 2017) were the pioneers in directly processing 3D coordinates, and many of the available point cloud completion approaches are based on them. These approaches typically adopt an encoder–decoder framework aimed at generating complete point clouds (Tchapmi et al., 2019; Yuan et al., 2018). In recent years, many other methodologies have been developed, including point-based, convolution-based, folding-based, graph-based, generative-model-based and transformer-based techniques (Fei et al., 2022). These approaches are increasingly being applied to complete and reconstruct point clouds of various objects across a wide range of fields, including applications in vehicles (Ibrahim et al., 2022), dentistry (Toscano et al., 2023) and cultural heritage (Sipiran et al., 2022). However, most of the initial models are trained using point clouds of artificial objects possessing clear continuity and symmetry characteristics (Singer & Asari, 2022). In the only study that involved a tree-specific approach, the model was developed for orchard trees using data collected from agricultural robots (Xu et al., 2023). Currently, the potential of DL for completing morphologically complex natural objects, such as trees, remains uncertain.

A crucial factor influencing the success of a DL model is the quality, quantity and diversity, and especially the representativeness, of the available training data. Supervised methods rely on the availability of paired training data. For classification tasks, this entails generating correct labels—usually through manual annotation. Conversely, for point cloud completion tasks, extensive data sets of complete point clouds are essential as training data. These cannot be manually generated and must therefore be readily available.

Ideally, for completion of point clouds of natural trees, one would thus use a database comprising dense point clouds, such as those acquired through TLS technology, of a representative range of trees. However, even dense TLS scans obtained within forest environments contain gaps, especially in the tree crown, which is not an ideal ground-truth for training completion models. Moreover, in practice, clear criteria for the quality of point cloud data of trees are not yet widely established (Demol et al., 2022). Consequently, LiDAR data on trees collated from various sources are likely to exhibit varying levels of quality and require meticulous manual verification.

The generation of synthetic point clouds enables the creation of almost unlimited data with known parameters, which can be used for model training. For trees, simulated point clouds have already been used to train models for semantic segmentation (Bryson et al., 2023; Vicari et al., 2019) and biomass estimation (Schäfer et al., 2023). Generally, for point cloud completion approaches in other domains, training and testing on synthetic data is the norm. However, Tesema et al. (2023) note that algorithms trained on synthetic data still struggle with performance when applied to real-world data sets. Synthetic data, while useful, may fall short in fully representing the patterns of incompleteness found in real-world scenarios, which include variations in point density and sparsity due to occlusion. These factors contribute to a significant domain gap between synthetic and real-world data, a challenge often overlooked by current methods aiming to replicate real-world probability distributions within training data sets. We assume that these differences between synthetic and real-world data are especially important to consider when applying these

models to 3D representations of complex natural objects, including forest trees.

In this study, the overarching goal was to explore the potential of a DL approach to close gaps in point clouds representing the main structures of deciduous trees, such as the stem and large branches. Further, we investigated the ability of a 3D point-cloud completion model trained on synthetic data to deal with the occlusion pattern in real-world point clouds. Specifically, we explored two key aspects of synthetic tree point cloud data: (1) the degree of naturalism in the shape of synthetic tree objects and (2) the realism of the point distribution in synthetic partial point clouds. For this, we compiled a variety of real-world and synthetic point clouds that represent deciduous trees under leaf-off conditions with differing levels of complexity. We used these point clouds to fine-tune an existing state-of-the-art 3D point-cloud completion model, and we subsequently evaluated the performance of the resulting models when applied to independent real-world TLS data.

2 | MATERIALS AND METHODS

2.1 | Data generation

We created six different data sets to fine-tune an existing DL model and two test data sets to evaluate the fine-tuned models and compare their completion results (Table 1). While the two test data sets only comprised real-world point clouds, some of the training data sets were built from synthetic data. In the following sections, we describe how these data were generated, processed and compiled.

2.1.1 | Synthetic trees

We used the Blender software (Blender Online Community, 2015) with two different add-ons to generate the synthetic trees used in this study.

TABLE 1 Description of data sets used to fine-tune and test the PoinTr model.

Dataset name	Point cloud source	Method for partial point cloud	No. samples
SapTreeGen	Random sampling on meshes of 12 simplified synthetic trees generated using the 'Sapling Tree Gen' add-on in Blender	Viewpoint	10,000
TheGrove	Random sampling on meshes of 20 naturalistic synthetic trees generated using the 'The Grove' add-on in Blender	Viewpoint	10,000
HeliosSim	HELIOS++ simulation on meshes of 20 naturalistic synthetic trees generated using the 'The Grove' add-on in Blender	Scan-based	10,000
realTLS	12 TLS point clouds of real forest trees acquired with a BLK360 scanning device, where individual trees were manually segmented; Tree point clouds are not fully complete and contain some smaller occlusions	Scan-based	10,000
TheGrove80+realTLS20	The same data as in the TheGrove dataset, but 2000 samples were replaced with point cloud segments from the realTLS dataset	Viewpoint	10,000
HeliosSim80+realTLS20	The same data as in the HeliosSim dataset, but 2000 samples were replaced with point cloud segments from the realTLS dataset	Scan-based	10,000
Test-viewpoint	Complete segments of real TLS data acquired with a Riegl VZ-400 scanning device	Viewpoint	500
Test-scanbased	Complete segments of real TLS data acquired with a Riegl VZ-400 scanning device	Scan-based	500

Sapling Tree Gen

The 'Sapling Tree Gen' add-on (Hale & Butcher, 2015) is a free software based on the work of Weber and Penn (1995). It can generate 3D tree objects that have a wide range of shapes but are still relatively simple, with conical stems and branches. In this study, we used 12 single tree objects (without leaves) originally created by Abegg et al. (2023) in Sapling Tree Gen. As illustrated in Figure 1 (left), they are of varying size but all follow the same generic geometry, with a vertical stem, helical branch distribution and variable branch curvature.

The Grove

The 'The Grove' add-on (van Keulen, 2023) is a commercial 3D tree growing software, originally intended for use by 3D artists and illustrators. It enables the creation of tree objects for naturalistic illustrations and animations, with a high degree of customisation. We used it to grow 20 tree objects (without leaves) of different shapes and sizes. From the available species presets, we chose five deciduous species common to temperate European forests: ash, beech, linden, maple and oak. These presets provide predefined rules for, for example, branch geometry and growth patterns based on the characteristics of different species, and thus allowed us to create a diverse range of virtual tree structures. Figure 1 (right) illustrates a range of tree objects we created with the 'The Grove' add-on.

2.1.2 | Synthetic point clouds

We then used a series of steps to transform the tree mesh objects generated in Blender into point clouds (Figure 2).

Sampling on mesh

First, we removed subsurface faces using the Meshlab software (v2022.02), leaving hollow mesh objects. This allowed us to perform random point sampling (density = 10,000 pts/m^2) on the mesh in the CloudCompare software v2.11 (Girardeau-Montaut, 2019) to obtain a preliminary point cloud. We then removed small branches up to a diameter of approximately 5 cm from the point cloud using the 'surface density' geometric feature in the CloudCompare software.

HELIOS++ simulations

We also simulated point clouds, which should more realistically reproduce the occlusion patterns found in real TLS data of trees. To achieve this, we employed the open-source LiDAR simulator HELIOS++ (Winiwarter et al., 2022), which enables users to simulate diverse laser scanning platforms on mesh models of individual objects or virtual scenes. Using specifications that are realistic for a Riegl VZ-400 terrestrial laser scanner (0.3 mrad beam divergence, 0.05° angular resolution), we simulated 12 scan positions at ground level distributed around each virtual tree mesh previously created in Blender using the 'The Grove' add-on.

2.1.3 | Real-world point clouds

In addition to the synthetic data sets, we created two data sets from real-world TLS data: one for model training and another for model testing. The primary criteria for selecting these data sets were as follows: (1). completeness and (2). the inclusion of the attributes PointSourceID or GpsTime, which are essential for creating samples with realistic occlusion patterns.

For training, we used 11 individual tree point clouds that had been manually segmented from data acquired using a BLK360 TLS instrument (Leica Geosystems, Heerbrugg, Switzerland). The TLS campaign and the subsequent data processing and tree segmentation are described in Bornand et al. (2023). The co-registered point clouds were sub-sampled to a minimum distance between points of 2 cm. Individual tree segmentation was performed on the CompuTree platform (Othmani et al., 2011) following a semi-automatic approach, which includes manual cleaning as a last step. Each selected individual tree contained points from 3 to 5 scan positions. They consisted of six European beech trees (*Fagus sylvatica* L.) and five oak trees

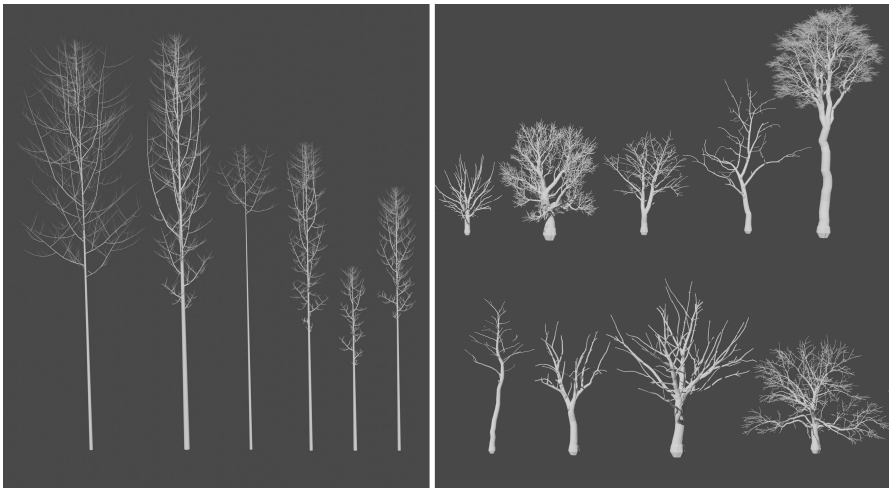

FIGURE 1 Examples of 3D mesh objects generated using the 'Sapling Tree Gen' add-on (left) and the 'The Grove' add-on (right) in the Blender software.

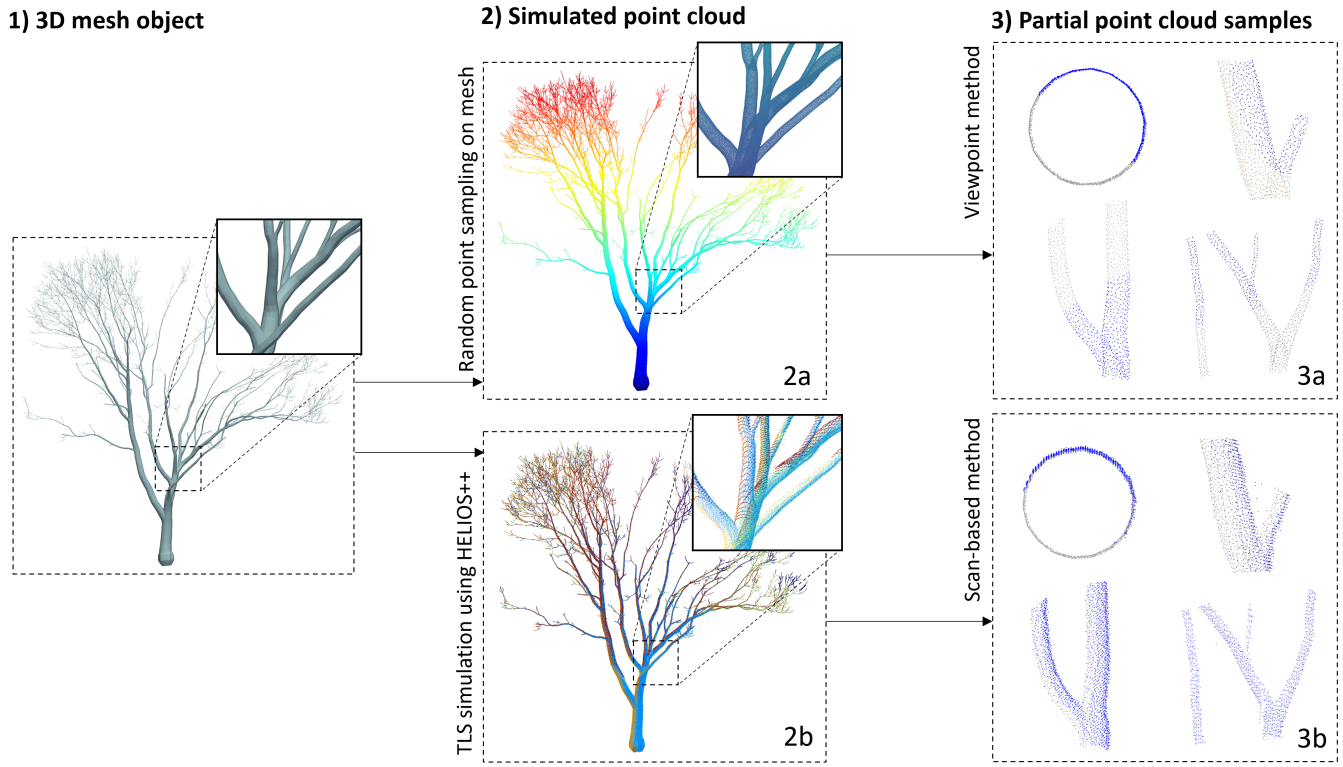

FIGURE 2 Procedure used to generate partial and complete training point cloud samples. Starting with a 3D mesh object (1), point clouds were simulated by sampling on the mesh surface (2a) or by using the HELIOS++ LiDAR simulator (2b). Partial point cloud samples (depicted in blue) were then generated using either the viewpoint method (3a) or the individual scan positions (3b).

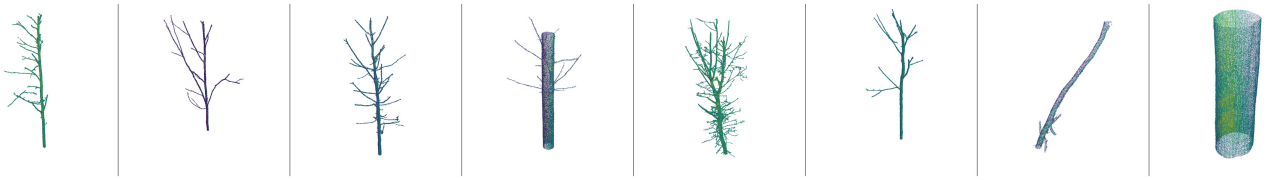

FIGURE 3 Examples of complete real-world point cloud segments used for model testing. The data were collected in a forest stand using a Riegl VZ-400i terrestrial laser scanner.

(*Quercus robur* L.). We chose these 11 trees for their completeness, but they still contained some small gaps caused by occlusions, as they originated from TLS acquisitions in a forest stand.

Since our completion model is intended to be sensor agnostic and capable of handling real-world data, our independent test data set consisted of entirely different TLS data than those used during the training process. We used data of a forest stand acquired with a Riegl VZ-400i TLS instrument (Riegl, Horn, Austria) with scan positions arranged on a 5 m grid pattern. The data were automatically co-registered within the software RISCAN PRO (Riegl, Horn, Austria). Within CloudCompare, point clouds were sub-sampled to a minimum distance between points of 1 cm, and further filtered by removing the 20% of points with lowest reflectance values. We then manually selected and segmented areas of trees with no occlusions (Figure 3).

2.1.4 | Creating point cloud samples

We divided the individual tree point clouds into smaller cube samples (voxels) of 1 m^3 size, by randomly selecting locations for sampling. We down-sampled each cube to contain 8192 points and then centred and normalised the coordinates of the point cloud sample. We generated 10,000 samples for each training data set used in this study. For each test data set, we created 500 samples. We adopted this approach primarily due to the requirements of the PoinTr algorithm, which only accepts point clouds of a specific size as input. Additionally, considering computational time constraints, the division of the point cloud into smaller samples allowed the generation of numerous training samples from a single tree model.

2.1.5 | Generating partial point clouds

We applied and compared two methods for generating partial point clouds that serve as input for the completion models. The first is simpler and can be done on-the-fly, while the second produces more realistic gaps in the point cloud.

Viewpoint method

We generated partial point clouds using a straightforward method that is already implemented in PoinTr (Yu et al., 2021) and has been used previously (e.g. Huang et al., 2020). This method (henceforth called the 'viewpoint method') involves randomly selecting a viewpoint, and removing a random number between 2048 and 6144 (25%–75% of the complete point cloud) of the farthest points from that viewpoint. This is a flexible and efficient strategy to generate a partial point cloud for model training; however, the generated partial point clouds may not be representative of real scanning and occlusion patterns (Tesema et al., 2023).

Scan-based method

This method can be applied to real or simulated data that contain information on which scan position a point originates from (usually an attribute called 'PointSourceID'). We split each complete sample point cloud into its scan positions and used a single scan position or a combination of several scan positions as partial point clouds, which were required to contain at least 2730 points. Through this process, we created between one and nine partial samples for each complete sample.

2.2 | Choice of deep learning network

Our objective was to explore 3D point cloud completion on terrestrial LiDAR data of trees using an existing DL network architecture. A suitable approach should be sensor agnostic and thus require only unstructured XYZ data. Networks with a decoder–encoder structure are particularly suitable for handling unstructured XYZ data, as they are able to generate an output that is the same size as the input. Also, the recently trending transformer-based approaches make it possible to better capture local and global features, thanks to their attention mechanism (Tesema et al., 2023). Although completion may benefit from multimodal inputs (such as the addition of RGB images (Zhang et al., 2021)), this would limit the versatility of such an approach. After these considerations, we opted to apply an approach called PoinTr developed by Yu et al. (2021), who aimed to solve general point cloud completion tasks by leveraging the sequence-to-sequence generation ability of transformer architectures. These authors converted the point cloud to a sequence of point proxies and then employed a geometry-aware transformer encoder–decoder architecture for point generation. With this approach, they were successful in completing a wide range of input point clouds with irregular features and up to 70% missing points. PoinTr is implemented in Python using the PyTorch library (Paszke et al., 2019).

Like in many other 3D completion approaches, the PoinTr network is trained and tested on the ShapeNet data set (Chang et al., 2015). This is an open large-scale repository for 3D CAD models, the ShapeNetCore collection containing over 51,000 shapes of 55 common object categories. However, these shapes mainly represent artificial objects. Yu et al. (2021) provide multiple PoinTr models that have already been pre-trained on this data set.

2.3 | Loss function

For completion tasks of unordered point clouds, the Chamfer distance (CD) is the metric most commonly used, both as a measure for evaluating and as a loss function for optimising learning-based algorithms (Camuffo et al., 2022; Tesema et al., 2023). It is used to assess the dissimilarity between two point clouds, typically a predicted point cloud P and a ground truth point cloud G. CD is calculated by first computing the squared distance between each point in P and its nearest neighbour in G and vice versa, and then combining the average distances in both directions. For each prediction, the CD between the prediction point set P and the ground-truth point set G is calculated as:

$$d_{CD}(P,G) = \frac{1}{|P|}\sum_{p\in P}\min_{g\in G}\|p-g\| + \frac{1}{|G|}\sum_{g\in G}\min_{p\in P}\|g-p\|.$$

An advantage of this metric is that it does not require $|P| = |G|$ and is invariant to the permutation of points. For further details on how CD is implemented as a loss function in the PoinTr model, see Yu et al. (2021).

2.4 | Model training

All model training in this study relied on fine-tuning PoinTr model weights provided by Yu et al. (2021), which they pre-trained on the ShapeNet data set (Chang et al., 2015). Fine-tuning a pre-trained base-model reduces training time and the amount of required data compared with training a DL model from scratch. We used each data set presented in Table 1 to fine-tune the general ShapeNet-based model weights.

While training, we applied the same basic pre-processing methods on-the-fly to each point cloud sample: normalisation and centring of coordinates, and random mirroring transformations. We used a randomly selected training/validation split of 80%/20% throughout. As this study focuses on training data set comparison rather than model optimisation, we kept the same model parameters as proven successful by Yu et al. (2021). These were as follows: an AdamW optimiser, a starting learning rate of 0.0005, 200 epochs, and a batch size of 96. As the scheduler, we used the Cosine Annealing Warm Restarts approach (Loshchilov & Hutter, 2017) instead of LambdaLR.

We carried out all experiments on a Linux machine running Ubuntu 22.04.3 LTS. A typical training time of a model when using

two NVIDIA GeForce RTX 3090 GPUs with 24 GB of memory each was on the order of 6–9 h.

2.5 | Model testing

We tested the models on the independent real-world data (Section 2.1.3). We evaluated both the original pre-trained model and the models fine-tuned on our six datasets (described in Table 1) twice—once on the test data set created with the viewpoint method and once on the test dataset created with the scan-based method. Additionally, to assess how a model fine-tuned on one data set performs on a different kind of data set, we conducted cross-data set testing. We tested the models trained on the SapTreeGen, TheGrove, HeliosSim and realTLS data sets on each of the other data sets. As evaluation metrics we applied F-Score and CF, which are already implemented in the PoinTr approach, following (Yu et al., 2021) and previous works (Tchapmi et al., 2019; Yuan et al., 2018). Following the previously applied approaches, we used two versions of the CD as evaluation metrics to compare the performance amongst the models fine-tuned on different data sets. CDL1 uses the L1-norm (Manhattan distance) to compute the distance between two points, while CDL2 uses the L2-norm (Euclidean distance) instead.

2.6 | Prediction of entire trees

The ultimate objective of point cloud completion methodologies in forest environments is not only to address areas of 1 m^3 in size, but also to extend their application to entire trees or even larger plots. To apply our completion approach to larger point clouds, we first downsampled and denoised the data and then cut the point cloud into smaller cubes (voxels). We used four voxel grids of sizes between 1 and 1.8 m^3, with the origin of each grid shifted by 0.5 m in the x, y or z direction. This ensured an overlap between the point cloud samples, preventing border effects in the model predictions. We then applied the completion model to each sample. We merged the resulting point clouds and removed noise by filtering with the surface density geometric feature in the CloudCompare software.

3 | RESULTS

3.1 | Model evaluation on an independent data set

The results of the model evaluation on an independent data set, composed of real TLS data with partial point clouds generated using the viewpoint method, are summarised on the left-hand side of Table 2. In particular, the TheGrove data set demonstrates superior performance, achieving the highest F-score and the lowest values for both CDL1 and CDL2. The SapTreeGen data set, on the other hand, which was also generated using the viewpoint method, leads to higher CD values—in a range similar to those from the HeliosSim and realTLS data sets, whose partial samples were generated using the scan-based method.

The results of the model evaluation on an independent data set, composed of real TLS data with partial point clouds generated using the scan-based method, are summarised on the right-hand side of Table 2. For this test data set, the model trained on HeliosSim80+realTLS20 achieves the best results. Notably, there is a considerable improvement compared with HeliosSim when 2000 simulated samples are replaced with real-world TLS data. Overall, this method leads to higher CD values for all data sets compared with testing on partial samples generated using the simpler viewpoint method. We also observe clear differences in the CD metrics between models trained on partial data created using the viewpoint method and models using individual scan positions as partial point clouds.

TABLE 2 Results of model evaluation on an independent data set consisting of real terrestrial laser scanning (TLS) data, where partial point clouds were generated using the viewpoint method, by removing 2048 points from the sample (left) and by using scan positions as partial point clouds (right).

	Partial point cloud generation					
	Viewpoint method			Scan-based method		
Model name	*F*-score	CDL1	CDL2	*F*-score	CDL1	CDL2
ShapeNet	0.55	14.71	2.25	0.13	50.64	13.88
SapTreeGen	0.44	24.85	5.45	0.14	51.75	14.44
TheGrove	0.59	**11.79**	**1.31**	0.17	48.26	14.44
HeliosSim	0.40	28.17	5.74	0.13	42.61	8.83
realTLS	0.49	21.26	5.39	0.21	32.77	7.08
TheGrove80+realTLS20	**0.62**	11.86	1.45	0.14	50.90	15.13
HeliosSim80+realTLS20	0.48	19.08	3.31	**0.21**	**31.31**	**6.18**

Note: We provide the *F*-score metric, the Chamfer distance using L1-norm to calculate the distance between two points (CDL1), and the Chamfer distance using L2-norm (CDL2). The best results are shown in bold.

TABLE 3 Results of cross-data set testing.

Test dataset	Metric	Training data set				
		ShapeNet	SapTreeGen	TheGrove	HeliosSim	realTLS
SapTreeGen	*F*-score	0.62		0.64	0.32	0.53
	CDL1	9.59	–	8.13	41.54	11.39
	CDL2	0.53		0.27	8.70	0.61
TheGrove	*F*-score	0.43	0.35		0.34	0.37
	CDL1	18.83	36.41	–	38.39	26.53
	CDL2	2.80	9.56		9.73	5.18
HeliosSim	*F*-score	0.02	0.03	0.03		0.04
	CDL1	110.85	106.96	98.77	–	79.38
	CDL2	42.64	39.79	34.87		25.05
realTLS	*F*-score	0.11	0.10	0.11	0.09	
	CDL1	64.39	61.68	60.08	48.52	–
	CDL2	16.81	15.24	15.34	9.50	

Note: The models were trained on all datasets (columns) and tested on all other datasets (rows). We provide the *F*-score metric, the Chamfer distance using L1-norm (CDL1) and the Chamfer distance using L2-norm (CDL2).

3.2 | Cross-data set testing

Table 3 shows the results from cross-data set testing. The original model trained on the ShapeNet dataset performs best on the SapTreeGen and TheGrove datasets, where partial point clouds were generated using the viewpoint method, while CD values are much higher for the simulated (HeliosSim) and realTLS scans. We observe similar trends for models trained on the SapTreeGen and TheGrove data sets. Interestingly, the model trained on data simulated with HELIOS++ shows consistency in the CDL2 metric across the test data sets SapTreeGen (8.7), TheGrove (9.7), and realTLS (9.5). By contrast, the model trained on realTLS perform better for the SapTreeGen (CDL2 = 0.61) and TheGrove (CDL2 = 5.2) test data sets compared with the data simulated with Helios++ (CDL2 = 25.05), where the aim of the partial point cloud samples was to mimic real occlusion patterns.

3.3 | Qualitative evaluation of completion results

The qualitative examples in Figure 4 show how the different models filled gaps in point cloud segments of trees. The original PoinTr model trained on the ShapeNet dataset is able to complete simple cylinders but has trouble with tree branches, leading to clumping of predicted points. When fine-tuned on the SapTreeGen data set, the predictions still show similar problems. On the other hand, fine-tuning on the TheGrove data set results in cleaner outputs with minimal noise, maintaining organic shapes. The model trained on HELIOS++ simulations leads to fuzzier point clouds, while the realTLS model can handle various occlusions, albeit with occasional predictions of multiple layers. The model trained on the TheGrove80 + realTLS20 data set produces output with reduced noise and a more even and complete representation compared with that from TheGrove alone. HELIOS++ simulations combined with realTLS data lead to less noise than the simulations alone but have issues with multiple layers, similar to the realTLS dataset.

Figure 5 illustrates an example of point cloud completion of an entire tree (*Fagus sylvatica*) using the model trained on the TheGrove80+realTLS20 data set. The zoomed-in image in Figure 5a shows that this model is capable of successfully closing small gaps in branches and generally increases the point density. In Figure 5b, we see that the model sometimes incorrectly predicts connections between smaller branches. Figure 5c shows that the gaps in the stump area are filled, while the non-cylindrical shape is still preserved.

4 | DISCUSSION

In this study, we evaluated the potential of a DL approach for 3D point cloud completion in the main structures of deciduous trees, such as the stem and large branches. Even with a very limited number of real and synthetic trees, we successfully fine-tuned a general pre-trained completion model to fill some of the gaps within 1 m^3 segments of tree point clouds. As illustrated in Figures 4 and 5, it is possible to close holes in the surface of branches and to close gaps between unconnected branch parts, while retaining their organic shape and not simplifying them as cylinders. However, larger gaps may persist, which can be due to insufficient points within the respective input segment or to a lack of similar shapes in the training data. In such cases, the completion models tend to fill in points in existing structures and only rarely 'invent' incorrect structures, such as the branch connection in Figure 5b.

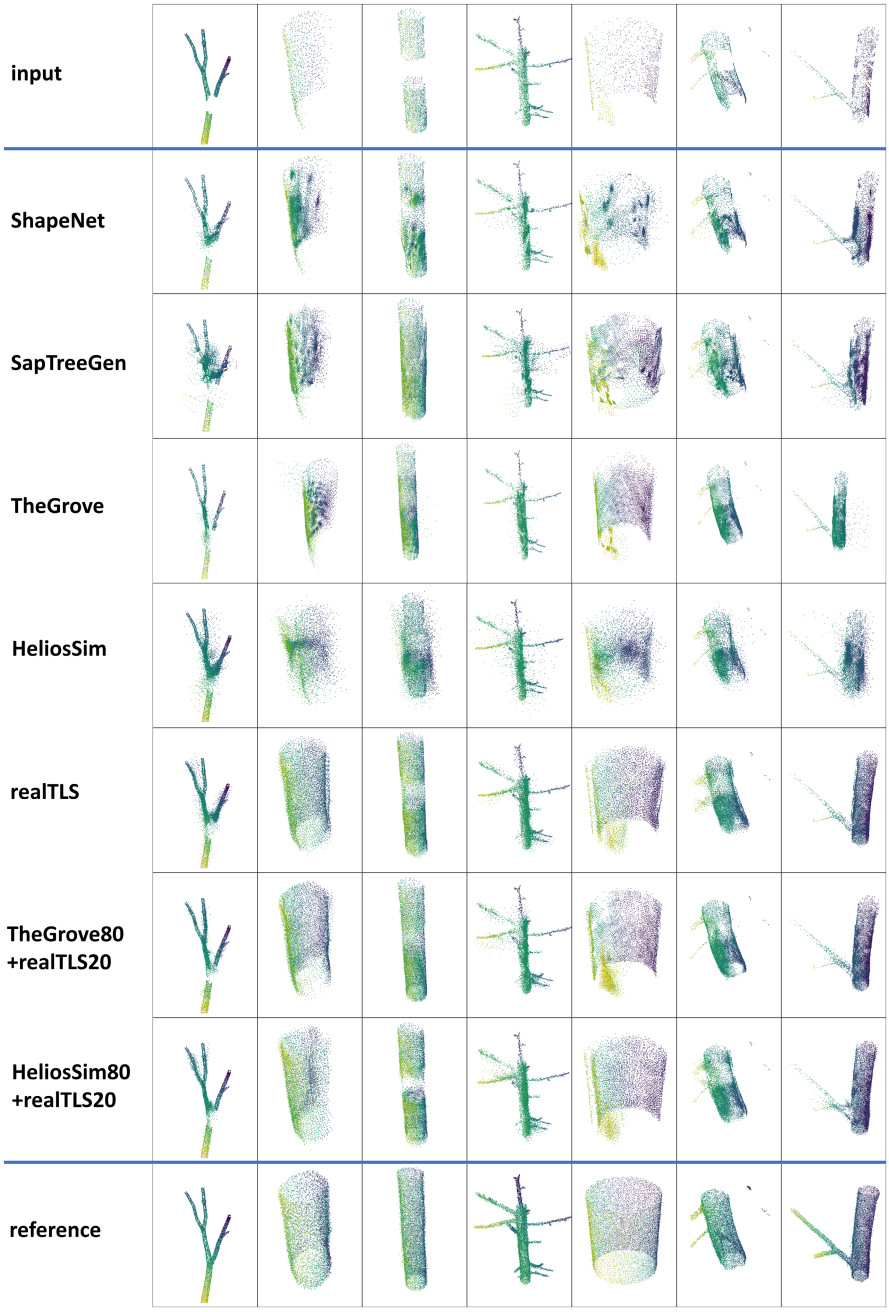

FIGURE 4 Examples of point cloud completion results. The first row depicts the input with gaps, the second the predictions by the original model trained only on ShapeNet (Yu et al., 2021), followed by the fine-tuned models (Table 1). The last row shows the full reference point cloud.

When we used synthetic data for training, both (1) the degree of naturalism of synthetic tree objects and (2) the realism of the point distribution in simulated partial point clouds affected the completion results.

4.1 | Naturalism of synthetic trees

In order for models to effectively complete the shapes of real trees, they need to be trained up to a certain level to deal with the complex morphology inherent in trees. Our tests on real data (Table 2) show that fine-tuning a general completion model using simple synthetic trees (SapTreeGen) may actually diminish its ability to complete sections of real trees. Visual inspection, as illustrated in Figure 4, reveals that the simplified shapes generated by SapTreeGen result in predictions containing clumping and noise. Conversely, fine-tuning on more naturalistic synthetic trees yields the lowest (and hence the best) CD values, indicating that the degree of naturalism in the shapes created by TheGrove is essential and potentially sufficient. At least upon visual comparison of the predicted shapes, there appears to be little difference between models trained on synthetic TheGrove objects and real trees. However, it is important to note that this assessment was based on a limited test data set; a more diverse range of tree shapes would be necessary to determine the circumstances under which the naturalism of TheGrove models is insufficient.

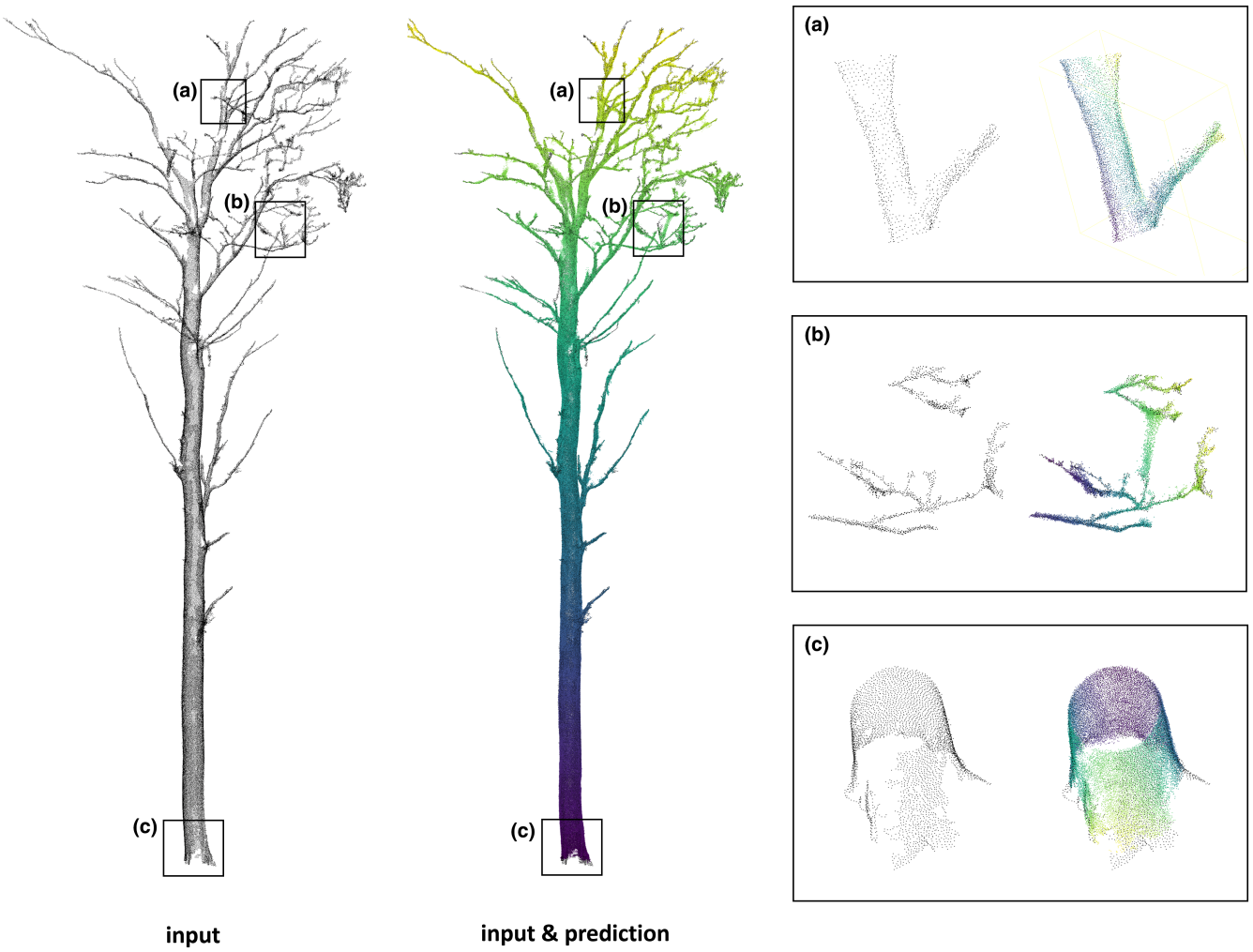

FIGURE 5 Example of point cloud completion of an entire tree using the model fine-tuned on the TheGrov80e+realTLS20 dataset. Input points are shown in black, while predicted and cleaned points are shown in viridis colours.

4.2 | Realism of the point distribution in simulated partial point clouds

Most completion algorithms that rely on synthetic datasets are trained to complete partial point clouds generated by, for instance, back-projecting 2.5D depth images into 3D (Tchapmi et al., 2019; Yuan et al., 2018) or simply removing points based on their distance from a selected viewpoint (Yu et al., 2021). While such partial point clouds may yield satisfactory results for artificial objects or urban scenes characterised by mostly planar surfaces and solid objects, the intricate and dense structure of forests, containing semi-transparent spaces caused by small branches and foliage, can result in occlusion patterns unique to these environments. Additionally, real-world data typically have non-uniform point densities because points measured from very different distances are combined in the resulting point cloud (Tesema et al., 2023). For instance for TLS data on trees, the points contained in the tree top are measured from farther away and are also subject to distance effects.

Our comparison of data sets derived from partial point clouds generated using the viewpoint method versus the more realistic scan-based method revealed a notable discrepancy in their impact on tree completion. When we evaluated the models fine-tuned on test data generated via the viewpoint method, the computed CD values fell within a similar range to those obtained by Yu et al. (2021) when they evaluated the PoinTr model using benchmark datasets (CDL1 between 4.75 and 10.9) and those achieved with other approaches, such as TopNet (Tchapmi et al., 2019), which achieved CDL1 values between 7.6 and 15.7 for the same benchmark.

In contrast, test data generated through the scan-based method consistently yielded higher (i.e. worse) CD values. Tesema et al. (2023) note that, generally, the performance of learning-based completion algorithms is still quite poor for real-world data sets. In the case of trees, this might be at least partly due to the fact that the incompleteness in real-world point clouds is caused by a complex mixture of occlusion, distance and noise effects. To explore this further, we fine-tuned the completion model on partial data generated from virtual LiDAR scans simulated using HELIOS++ (Winiwarter et al., 2022), as well as partial data obtained from single real-world scans. Our quantitative evaluation indicates that combining simulated scan data (HeliosSim) with 2000 samples of

real data yields the best performance for the scan-based test dataset, followed by the model trained on real TLS data only (Table 2). In general, mixing synthetic LiDAR data with real data when only a limited number of real training samples is available is a strategy that has been recommended for other tasks, such as tree segmentation (Bryson et al., 2023) and forest biomass modelling (Schäfer et al., 2023).

With our current approach, visual inspection of the predictions reveals that data sets derived from real-world scan-based methods often produce fuzzy point clouds or exhibit double layers of points, resembling misalignments of scan positions, despite the absence of such patterns in the real TLS data used for training. Conversely, datasets derived from the viewpoint method sometimes leave larger gaps but yield smoother, less noisy surfaces with more uniform point densities. Depending on the application (e.g. for 3D reconstruction), one might even prefer the resulting prediction to be smoother than real TLS or MLS data and therefore choose a model that was mainly trained on synthetic data.

4.3 | Considerations for model applicability

To apply the completion approach outlined in this study to entire trees or field plots, the input data must first be partitioned into smaller cubes. The method selected for this segmentation affects the prediction results considerably, making it crucial to determine the optimal size and potential overlaps of these segments. Additionally, we recommend the strategy of performing predictions on multiple rotations of the same sample and then merging the resulting point clouds, for example, using voting techniques. As part of data preparation, we excluded very fine branches (diameter less than 5 cm) from our training data sets; a similar prefiltering step would be necessary before applying the completion model to other data.

Before completion approaches can be integrated into processing chains for 3D point clouds of trees, it is crucial to investigate how completion affects the estimation of parameters such as stem diameter, tree height or volume, and if quantitative structure models (QSMs) could benefit from it.

Our study focussed on leaf-off broadleaf trees, aiming to fill gaps in their complex branch architectures. We see potential for applying this completion approach after using a leaf-wood separation algorithm (Krisanski et al., 2021; Vicari et al., 2019; Wang et al., 2020), where the resulting point cloud of the branch structure often contains small occlusions caused by foliage.

Conversely, training the model to complete foliage on trees poses greater challenges. Producing point cloud data of leaf-on trees without occlusions for training is difficult, and synthetic 3D objects of foliage are challenging to simulate realistically. For these reasons, we believe it less practical to train this model to complete branches of evergreen conifers, as needles produce significant scattering and are generally not well-resolved in TLS data. We expect that our approach can be used to complete the stems of conifers; however, most conifer stems can also be reconstructed sufficiently accurately by fitting cylindrical or conical shapes, or using taper curves assuming a conical stem shape. In dense conifer stands, tree tops are often fully occluded (e.g. Torresan et al., 2018). For such cases, future studies could focus on developing tree completion models on a larger scale, working with coarser point clouds and focusing more on the overall crown shape of the conifer.

4.4 | Relation to tree reconstruction approaches

3D reconstruction is crucial for extracting information from dense point cloud data of individual trees. Common methods include TreeQSM (Raumonen et al., 2013), SimpleTree (Hackenberg et al., 2015) and AdTree (Du et al., 2019), which rely on high-quality data and can struggle with occlusion, low point density or high noise. A newer approach by Wang et al. (2023) improves reconstruction from incomplete point clouds using a tree skeleton node generation technique, a spline curve tree cross-section method for the stem, and an alpha shape morphology-guided tree adaptive growth algorithm to complete branches in the crown. While effective, this complex combination of rule-based algorithms may limit flexibility across different datasets.

Adapting a general DL-based point cloud completion model to fill gaps in tree structures is a novel approach not yet explored on the scale of dense TLS data. Currently, the only similar DL-based approach focused on completing coarser point clouds of entire trees (Xu et al., 2023).

Our approach, when related to tree reconstruction, has potential as a pre-processing step to fill gaps before applying a cylinder fitting method such as TreeQSM. Alternatively, increasing the point density of tree surfaces could help fit mesh surfaces that better represent organic shapes than simple cylinders. However, both geometric reconstruction methods and our completion approach still face challenges with the complex structures of small branches.

4.5 | Outlook

The exploration of the topic discussed in this study is still in its early stages, which provides numerous opportunities for further investigation. Based on our experiences, we outline four potential avenues for future research:

This study focussed on small gaps in dense point clouds. Currently, the model we used can handle only a limited number of points per input sample, due to the size of the model and the associated computational cost. To complete larger structures at once, we would need to work with sparser point clouds. This would also require retraining the model with data from a broader range of trees and suitable point densities. Another option would be to use a different model architecture which can handle larger input data, such as the approach by Singer and Asari (2022), which was able to complete real-world aerial LiDAR scans.

To date, we have only applied the data augmentations originally done in PoinTr (Yu et al., 2021). With this set-up, we observe that the models are generally better at completing vertical stems than horizontal branches. Therefore, a suitable data augmentation strategy would be to rotate input samples in random directions and also to randomly scale the size of the input point clouds; this would increase the diversity in shapes and point densities.

We focussed on real-world and synthetic TLS data. However, this completion approach should be sensor-agnostic and applicable to many kinds of point cloud data. Thus, an important next step is to test this method on MLS or UAVLS data or on point clouds derived from photogrammetry. We assume that differences in noise level and point density would need to be taken into account when training on data from other sources.

Our current methodology involves cutting the entire point cloud into smaller segments for completion, trusting the model's ability to learn which areas need to be completed. A more efficient approach could involve first identifying the areas that require completion. Techniques such as occlusion mapping (Kükenbrink et al., 2017) could help pinpoint regions requiring completion, providing insight into which voxels were potentially occluded from the sensor and to what extent. Introducing this information during model training could also positively impact completion results.

5 | CONCLUSIONS

The rapid advancements in DL approaches for 3D data provide many possibilities for applications in ecology and forestry, although much exploration is still needed to adapt these technologies to real-world environments. For tasks such as point cloud completion, extensive datasets of complete point clouds are needed for training, but manual generation of such data is impractical. Using synthetic data has the potential to generate the required volume of controlled training data. Here, we investigated the potential of DL-based point cloud completion for trees, using dense point clouds of deciduous trees in a leaf-off state as an example. Our findings demonstrate that fine-tuning a network on synthetic tree data can enhance the model's ability to complete tree objects compared with a general model trained on diverse artificial objects. However, the quality of predictions is influenced by the level of sophistication of the synthetic data. Moreover, our assessment was based on a limited test data set, meaning that a more diverse range of tree shapes would be needed to assess the general applicability of our models. Before integration into processing workflows, it is also crucial to investigate how completion impacts the estimation of tree parameters. Importantly, incorporating even limited real-world TLS data during training can significantly improve completion results but may introduce additional noise in predictions. Based on our findings, we recommend the use of models trained on partial point clouds with realistic occlusion patterns to fill gaps in real-world point clouds.

AUTHOR CONTRIBUTIONS

Aline Bornand was involved in conceptualisation, methodology, software, validation, formal analysis, investigation, data curation, writing—original draft, writing—review and editing, and visualisation. Meinrad Abegg was involved in conceptualisation, data curation, resources, writing—review and editing, supervision and funding acquisition. Felix Morsdorf was involved in conceptualisation, writing—review and editing, supervision and funding acquisition. Nataliia Rehush was involved in conceptualisation, methodology, software, investigation, resources, writing—review and editing, and supervision.

ACKNOWLEDGEMENTS

The project was funded by the Swiss National Forest Inventory. We thank Birgit Eben for helping with the manual tree segmentation and Daniel Kükenbrink and Mauro Marty for providing additional data and helping with data acquisition and curation. We also thank Melissa Dawes for professional language editing.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

PEER REVIEW

The peer review history for this article is available at https://www.webofscience.com/api/gateway/wos/peer-review/10.1111/2041-210X.14412.

DATA AVAILABILITY STATEMENT

Data available via https://doi.org/10.5281/zenodo.13303159 (Bornand, 2024a) and code available via https://github.com/al-Brnd/treePoinTr or https://doi.org/10.5281/zenodo.13303681 (Bornand, 2024b).

ORCID

Aline Bornand https://orcid.org/0000-0001-7712-5571
Meinrad Abegg https://orcid.org/0000-0001-5151-5889
Felix Morsdorf https://orcid.org/0000-0001-6713-1599
Nataliia Rehush https://orcid.org/0000-0003-4966-1945

REFERENCES

Abegg, M., Bösch, R., Kükenbrink, D., & Morsdorf, F. (2023). Tree volume estimation with terrestrial laser scanning—Testing for bias in a 3D virtual environment. *Agricultural and Forest Meteorology*, *331*, 109348. https://doi.org/10.1016/j.agrformet.2023.109348

Abegg, M., Kükenbrink, D., Zell, J., Schaepman, M. E., & Morsdorf, F. (2017). Terrestrial laser scanning for forest inventories-tree diameter distribution and scanner location impact on occlusion. *Forests*, *8*, 184. https://doi.org/10.3390/f8060184

Allen, M. J., Grieve, S. W., Owen, H. J., & Lines, E. R. (2023). Tree species classification from complex laser scanning data in Mediterranean forests using deep learning. *Methods in Ecology and Evolution*, *14*, 1657–1667. https://doi.org/10.1111/2041-210X.13981

Blender Online Community. (2015). *Blender—A 3D modelling and rendering package*. https://www.blender.org

Bornand, A. (2024a). Data and model weights for completing 3D point clouds of individual trees using deep learning. *Zenodo*, https://doi.org/10.5281/zenodo.13303159

Bornand, A. (2024b). *treePoinTr: Code for the publication "Completing 3D point clouds of individual trees using deep learning"*. https://doi.org/10.5281/zenodo.13303681

Bornand, A., Rehush, N., Morsdorf, F., Thürig, E., & Abegg, M. (2023). Individual tree volume estimation with terrestrial laser scanning: Evaluating reconstructive and allometric approaches. *Agricultural and Forest Meteorology*, *341*, 109654. https://doi.org/10.1016/j.agrformet.2023.109654

Brede, B., Bartholomeus, H. M., Barbier, N., Pimont, F., Vincent, G., & Herold, M. (2022). Peering through the thicket: Effects of UAV LiDAR scanner settings and flight planning on canopy volume discovery. *International Journal of Applied Earth Observation and Geoinformation*, *114*, 103056. https://doi.org/10.1016/j.jag.2022.103056

Bryson, M., Wang, F., & Allworth, J. (2023). Using synthetic tree data in deep learning-based tree segmentation using LiDAR point clouds. *Remote Sensing*, *15*, 2380. https://doi.org/10.3390/rs15092380

Calders, K., Adams, J., Armston, J., Bartholomeus, H., Bauwens, S., Bentley, L. P., Chave, J., Danson, F. M., Demol, M., Disney, M., Gaulton, R., Krishna Moorthy, S. M., Levick, S. R., Saarinen, N., Schaaf, C., Stovall, A., Terryn, L., Wilkes, P., & Verbeeck, H. (2020). Terrestrial laser scanning in forest ecology: Expanding the horizon. *Remote Sensing of Environment*, *251*, 112102. https://doi.org/10.1016/j.rse.2020.112102

Camuffo, E., Mari, D., & Milani, S. (2022). Recent advancements in learning algorithms for point clouds: An updated overview. *Sensors*, *22*, 1357. https://doi.org/10.3390/s22041357

Cao, W., Wu, J., Shi, Y., & Chen, D. (2022). Restoration of individual tree missing point cloud based on local features of point cloud. *Remote Sensing*, *14*, 1346. https://doi.org/10.3390/rs14061346

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., & Yu, F. (2015). *ShapeNet: An information-rich 3D model repository*.

Demol, M., Verbeeck, H., Gielen, B., Armston, J., Burt, A., Disney, M., Duncanson, L., Hackenberg, J., Kükenbrink, D., Lau, A., Ploton, P., Sewdien, A., Stovall, A., Takoudjou, S. M., Volkova, L., Weston, C., Wortel, V., & Calders, K. (2022). Estimating forest above-ground biomass with terrestrial laser scanning: Current status and future directions. *Methods in Ecology and Evolution*, *13*, 1628–1639. https://doi.org/10.1111/2041-210X.13906

Du, S., Lindenbergh, R., Ledoux, H., Stoter, J., & Nan, L. (2019). AdTree: Accurate, detailed, and automatic modelling of laser-scanned trees. *Remote Sensing*, *11*, 2074. https://doi.org/10.3390/rs11182074

Fei, B., Yang, W., Chen, W. M., Li, Z., Li, Y., Ma, T., Hu, X., & Ma, L. (2022). Comprehensive review of deep learning-based 3D point cloud completion processing and analysis. *IEEE Transactions on Intelligent Transportation Systems*, *23*, 22862–22883. https://doi.org/10.1109/TITS.2022.3195555

Girardeau-Montaut, D. (2019). *CloudCompare*. https://cloudcompare.org

Grau, E., Durrieu, S., Fournier, R., Gastellu-Etchegorry, J. P., & Yin, T. (2017). Estimation of 3D vegetation density with terrestrial laser scanning data using voxels. A sensitivity analysis of influencing parameters. *Remote Sensing of Environment*, *191*, 373–388. https://doi.org/10.1016/j.rse.2017.01.032

Hackenberg, J., Spiecker, H., Calders, K., Disney, M., & Raumonen, P. (2015). SimpleTree—An efficient open source tool to build tree models from TLS clouds. *Forests*, *6*, 4245–4294. https://doi.org/10.3390/f6114245

Hale, A., & Butcher, A. (2015). *Sapling tree gen blender add-on*. https://docs.blender.org/manual/en/latest/addons/add_curve/sapling.html

Huang, Z., Yu, Y., Xu, J., Ni, F., & Le, X. (2020). PF-Net: Point fractal network for 3D point cloud completion. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 7659–7667. https://doi.org/10.1109/CVPR42600.2020.00768

Ibrahim, Y., Nagy, B., & Benedek, C. (2022). Multi-view based 3D point cloud completion algorithm for vehicles. In *Proceedings—international conference on pattern recognition* (pp. 2121–2127). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ICPR56361.2022.9956459

Krisanski, S., Taskhiri, M. S., Aracil, S. G., Herries, D., & Turner, P. (2021). Sensor agnostic semantic segmentation of structurally diverse and complex forest point clouds using deep learning. *Remote Sensing*, *13*, 1413. https://doi.org/10.3390/rs13081413

Kükenbrink, D., Schneider, F. D., Leiterer, R., Schaepman, M. E., & Morsdorf, F. (2017). Quantification of hidden canopy volume of airborne laser scanning data using a voxel traversal algorithm. *Remote Sensing of Environment*, *194*, 424–436. https://doi.org/10.1016/j.rse.2016.10.023

Liang, X., Kankare, V., Hyyppä, J., Wang, Y., Kukko, A., Haggrén, H., Yu, X., Kaartinen, H., Jaakkola, A., Guan, F., Holopainen, M., & Vastaranta, M. (2016). Terrestrial laser scanning in forest inventories. *ISPRS Journal of Photogrammetry and Remote Sensing*, *115*, 63–77. https://doi.org/10.1016/j.isprsjprs.2016.01.006

Loshchilov, I., & Hutter, F. (2017). SGDR: Stochastic gradient descent with warm restarts. *5th International Conference on Learning Representations, ICLR 2017—Conference Track Proceedings*, *International Conference on Learning Representations, ICLR*. https://arxiv.org/abs/1608.03983v5

Morel, J., Bac, A., & Véga, C. (2018). Surface reconstruction of incomplete datasets: A novel Poisson surface approach based on CSRBF. *Computers and Graphics (Pergamon)*, *74*, 44–55. https://doi.org/10.1016/j.cag.2018.05.004

Morhart, C., Schindler, Z., Frey, J., Sheppard, J. P., Calders, K., Disney, M., Morsdorf, F., Raumonen, P., & Seifert, T. (2024). Limitations of estimating branch volume from terrestrial laser scanning. *European Journal of Forest Research*, *143*, 687–702. https://doi.org/10.1007/s10342-023-01651-z

Othmani, A., Piboule, A., Krebs, M., Stolz, C., & Lew Yan Voon, L. F. C. (2011). Towards automated and operational forest inventories with T-Lidar. *Silvilaser 2011*, Hobart, Australia.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*. Neural Information Processing Systems Foundation. https://arxiv.org/abs/1912.01703v1

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. *Proceedings—30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 77–85. https://doi.org/10.1109/CVPR.2017.16

Raumonen, P., Kaasalainen, M., Åkerblom, M., Kaasalainen, S., Kaartinen, H., Vastaranta, M., Holopainen, M., Disney, M., & Lewis, P. (2013). Fast automatic precision tree models from terrestrial laser scanner data. *Remote Sensing*, *5*, 491–520. https://doi.org/10.3390/rs5020491

Ravaglia, J., Bac, A., & Fournier, R. A. (2017). Extraction of tubular shapes from dense point clouds and application to tree reconstruction from laser scanned data. *Computers and Graphics (Pergamon)*, *66*, 23–33. https://doi.org/10.1016/j.cag.2017.05.016

Schäfer, J., Winiwarter, L., Weiser, H., Novotný, J., Höfle, B., Schmidtlein, S., Henniger, H., Krok, G., Stereńczak, K., & Fassnacht, F. E. (2023). Assessing the potential of synthetic and ex situ airborne laser scanning and ground plot data to train forest biomass models. *Forestry:*

An International Journal of Forest Research, *97*, 512–530. https://doi.org/10.1093/forestry/cpad061

Singer, N., & Asari, V. K. (2022). View-agnostic point cloud generation for occlusion reduction in aerial Lidar. *Remote Sensing*, *14*, 2955. https://doi.org/10.3390/rs14132955

Sipiran, I., Mendoza, A., Apaza, A., & Lopez, C. (2022). Data-driven restoration of digital archaeological pottery with point cloud analysis. *International Journal of Computer Vision*, *130*, 2149–2165. https://doi.org/10.1007/s11263-022-01637-1

Tagliasacchi, A., Zhang, H., & Cohen-Or, D. (2009). Curve skeleton extraction from incomplete point cloud. *ACM Transactions on Graphics, ACM PUB27*, New York, NY, USA. https://doi.org/10.1145/1531326.1531377

Tchapmi, L. P., Kosaraju, V., Rezatofighi, H., Reid, I., & Savarese, S. (2019). Topnet: Structural point cloud decoder. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society*, pp. 383–392. https://doi.org/10.1109/CVPR.2019.00047

Terryn, L., Calders, K., Åkerblom, M., Bartholomeus, H., Disney, M., Levick, S., Origo, N., Raumonen, P., & Verbeeck, H. (2023). Analysing individual 3D tree structure using the R package ITSMe. *Methods in Ecology and Evolution*, *14*, 231–241. https://doi.org/10.1111/2041-210X.14026

Terryn, L., Calders, K., Bartholomeus, H., Bartolo, R. E., Brede, B., D'hont, B., Disney, M., Herold, M., Lau, A., Shenkin, A., Whiteside, T. G., Wilkes, P., & Verbeeck, H. (2022). Quantifying tropical forest structure through terrestrial and UAV laser scanning fusion in Australian rainforests. *Remote Sensing of Environment*, *271*, 112912. https://doi.org/10.1016/j.rse.2022.112912

Tesema, K. W., Hill, L., Jones, M. W., Ahmad, M. I., & Tam, G. K. (2023). Point cloud completion: A survey. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–20. https://doi.org/10.1109/tvcg.2023.3344935

Torresan, C., Chiavetta, U., & Hackenberg, J. (2018). Applying quantitative structure models to plot-based terrestrial laser data to assess dendrometric parameters in dense mixed forests. *Forest Systems*, *27*, e004. https://doi.org/10.5424/fs/2018271-12658

Toscano, J. D., Zuniga-Navarrete, C., Siu, W. D. J., Segura, L. J., & Sun, H. (2023). Teeth mold point cloud completion via data augmentation and hybrid RL-GAN. *Journal of Computing and Information Science in Engineering*, *23*, 041008. https://doi.org/10.1115/1.4056566

van Keulen, W. (2023). *The Grove*. https://www.thegrove3d.com

Vicari, M. B., Disney, M., Wilkes, P., Burt, A., Calders, K., & Woodgate, W. (2019). Leaf and wood classification framework for terrestrial LiDAR point clouds. *Methods in Ecology and Evolution*, *10*, 680–694. https://doi.org/10.1111/2041-210X.13144

Wang, D., Momo Takoudjou, S., & Casella, E. (2020). LeWoS: A universal leaf-wood classification method to facilitate the 3D modelling of large tropical trees using terrestrial LiDAR. *Methods in Ecology and Evolution*, *11*, 376–389. https://doi.org/10.1111/2041-210X.13342

Wang, W., Li, Y., Huang, H., Hong, L., Du, S., Xie, L., Li, X., Guo, R., & Tang, S. (2023). Branching the limits: Robust 3D tree reconstruction from incomplete laser point clouds. *International Journal of Applied Earth Observation and Geoinformation*, *125*, 103557. https://doi.org/10.1016/J.JAG.2023.103557

Wang, Y., Lehtomäki, M., Liang, X., Pyörälä, J., Kukko, A., Jaakkola, A., Liu, J., Feng, Z., Chen, R., & Hyyppä, J. (2019). Is field-measured tree height as reliable as believed—A comparison study of tree height estimates from field measurement, airborne laser scanning and terrestrial laser scanning in a boreal forest. *ISPRS Journal of Photogrammetry and Remote Sensing*, *147*, 132–145. https://doi.org/10.1016/j.isprsjprs.2018.11.008

Weber, J., & Penn, J. (1995). Creation and rendering of realistic trees. *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pp. 119–126. https://doi.org/10.1145/218380.218427

Winiwarter, L., Esmorís Pena, A. M., Weiser, H., Anders, K., Martínez Sánchez, J., Searle, M., & Höfle, B. (2022). Virtual laser scanning with HELIOS++: A novel take on ray tracing-based simulation of topographic full-waveform 3D laser scanning. *Remote Sensing of Environment*, *269*, 112772. https://doi.org/10.1016/j.rse.2021.112772

Xi, Z., Hopkinson, C., Rood, S. B., & Peddle, D. R. (2020). See the forest and the trees: Effective machine and deep learning algorithms for wood filtering and tree species classification from terrestrial laser scanning. *ISPRS Journal of Photogrammetry and Remote Sensing*, *168*, 1–16. https://doi.org/10.1016/j.isprsjprs.2020.08.001

Xu, D., Chen, G., & Jing, W. (2023). A single-tree point cloud completion approach of feature fusion for agricultural robots. *Electronics*, *12*, 1296. https://doi.org/10.3390/electronics12061296

Yu, X., Rao, Y., Wang, Z., Liu, Z., Lu, J., & Zhou, J. (2021). PoinTr: Diverse point cloud completion with geometry-aware transformers. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 12478–12487. https://doi.org/10.1109/ICCV48922.2021.01227

Yuan, W., Khot, T., Held, D., Mertz, C., & Hebert, M. (2018). PCN: Point completion network. *Proceedings—2018 International Conference on 3D Vision, 3DV 2018*, pp. 728–737. https://wentaoyuan.github.io/pcn, https://doi.org/10.1109/3DV.2018.00088

Zhang, X., Feng, Y., Li, S., Zou, C., Wan, H., Zhao, X., Guo, Y., & Gao, Y. (2021). View-guided point cloud completion. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 15885–15894. https://doi.org/10.1109/CVPR46437.2021.01563

SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

Appendix S1: Additional illustration.

How to cite this article: Bornand, A., Abegg, M., Morsdorf, F., & Rehush, N. (2024). Completing 3D point clouds of individual trees using deep learning. *Methods in Ecology and Evolution*, *15*, 2010–2023. https://doi.org/10.1111/2041-210X.14412