ISPRS Journal of Photogrammetry and Remote Sensing 215 (2024) 192–213

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

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

Deep learning with simulated laser scanning data for 3D point cloud classification

Alberto M. Esmorís [a,e], Hannah Weiser [a], Lukas Winiwarter [a,c,d], Jose C. Cabaleiro [e,f], Bernhard Höfle [a,b,*]

[a] 3DGeo Research Group, Institute of Geography, Heidelberg University, Im Neuenheimer Feld 368, Heidelberg, 69120, Baden-Württemberg, Germany
[b] Interdisciplinary Center for Scientific Computing (IWR), Heidelberg University, Im Neuenheimer Feld 205, Heidelberg, 69120, Baden-Württemberg, Germany
[c] Research Unit Photogrammetry, Department of Geodesy and Geoinformation, TU Wien, Wiedner Hauptstraße 8-10, 1040, Vienna, Austria
[d] Unit of Geometry and Surveying, Faculty of Engineering Sciences, University of Innsbruck, Technikerstraße 13, Innsbruck, 6020, Tyrol, Austria
[e] Centro Singular de Investigación en Tecnoloxías Intelixentes, Universidade de Santiago de Compostela, Rúa de Jenaro de la Fuente Domínguez, Santiago de Compostela, 15782, A Coruña, Spain
[f] Departamento de Electrónica e Computación, Universidade de Santiago de Compostela, Rúa Lope Gómez de Marzoa, Santiago de Compostela, 15782, A Coruña, Spain

ARTICLE INFO

Keywords:
Virtual laser scanning
LiDAR simulation
Point clouds
Machine learning
Point-wise classification
Leaf-wood segmentation

ABSTRACT

Laser scanning is an active remote sensing technique applied in many disciplines to acquire state-of-the-art spatial measurements. Semantic labeling is often necessary to extract information from the raw point cloud. Deep learning methods constitute a data-hungry solution for the semantic segmentation of point clouds. In this work, we investigate the use of simulated laser scanning for training deep learning models, which are applied to real data subsequently. We show that training a deep learning model purely on virtual laser scanning data can produce results comparable to models trained on real data when evaluated on real data. For leaf-wood segmentation of trees, using the KPConv model trained with virtual data achieves 93.7% overall accuracy, while the model trained on real data reaches 94.7% overall accuracy. In urban contexts, a KPConv model trained on virtual data achieves 74.1% overall accuracy on real validation data, while the model trained on real data achieves 82.4%. Our models outperform the state-of-the-art model FSCT in terms of generalization to unseen real data as well as a baseline model trained on points randomly sampled from the tree mesh surface. From our results, we conclude that the combination of laser scanning simulation and deep learning is a cost-effective alternative to real data acquisition and manual labeling in the domain of geospatial point cloud analysis. The strengths of this approach are that (a) a large amount of diverse laser scanning training data can be generated quickly and without the need for expensive equipment, (b) the simulation configurations can be adapted so that the virtual training data have similar characteristics to the targeted real data, and (c) the whole workflow can be automated through procedural scene generation.

1. Introduction

Light Detection and Ranging (LiDAR) can be used for three-dimensional (3D) observation of various environments, making it one of the most important geospatial data acquisition technologies (Shan and Toth, 2018). Point clouds are the main representation of LiDAR measurements. They have unstructured Euclidean data (geometric data) defining the spatial coordinates of the points (Otepka et al., 2013) and may include other data such as radiometric or backscatter features. Typically, the raw point clouds need to be segmented or classified in order to extract meaningful geoinformation or perform further analysis.

Virtual Laser Scanning (VLS) simulates laser scanning to generate virtual point clouds (Winiwarter et al., 2022; Dosovitskiy et al., 2017; Gastellu-Etchegorry et al., 2016), also called simulated or synthetic point clouds in other studies. In this work, we compare the performance of deep learning (DL) models trained with real-world and virtually scanned and thereby labeled point clouds. While there is a growing interest in deep learning on 3D point clouds, manual labeling is an extremely time-consuming task and is the main bottleneck in this area (Griffiths and Boehm, 2019).

We use VLS to generate perfectly annotated virtual point clouds from 3D scenes in a time and cost-efficient way to train deep learning

* Correspondence to: Heidelberg University, Institute of Geography, Im Neuenheimer Feld 368, 69120 Heidelberg, Germany
E-mail address: hoefle@uni-heidelberg.de (B. Höfle).

https://doi.org/10.1016/j.isprsjprs.2024.06.018
Received 19 April 2023; Received in revised form 14 June 2024; Accepted 20 June 2024
Available online 13 July 2024

models that generalize to real data. The 3D scenes for VLS can be manually designed, often made available in public 3D model repositories, derived from real data acquired with different sensors (e.g., LiDAR or photogrammetry), or computed through procedural scene generation algorithms (Zahs et al., 2023). Thus, VLS can be used to (a) create huge and diverse synthetic datasets, (b) generate targeted training data for critical classes or mimic the characteristics of a real dataset by modifying the scene, platform, and scanner configuration with little effort, and (c) provide perfectly annotated datasets by transferring labels from the 3D scene with no need for point cloud annotation or real data acquisition.

Our experiments focus on two common point-wise point cloud classification problems: (1) semantic segmentation of urban scenes (Kölle et al., 2021) and (2) leaf-wood segmentation of trees (Ferrara et al., 2018; Krisanski et al., 2021b; Han and Sánchez-Azofeifa, 2022). For (1), we selected the mesh model of the Hessigheim3D urban benchmark dataset (Kölle et al., 2021) to create the virtual scene for VLS. For (2), we use computer-generated tree models for a fully virtual scene and the Wytham Woods 3D forest scene, which was reconstructed from real data (Calders et al., 2018; Liu et al., 2022).

Our objectives are to:

1. Develop a theoretical definition of the Virtual Laser Scanning meets Deep Learning (VLS-DL) model.
2. Empirically validate the VLS-DL model using state-of-the-art data and algorithms to train deep learning models on virtual point clouds that generalize to real point clouds.
3. Obtain proof-of-concept by applying the VLS-DL model to prominent and representative classification tasks covering natural and urban environments.

In doing so, the following research questions will be answered:

1. To what degree do deep learning models trained with virtual laser scanning data generalize to real point clouds?
2. What are the quantitative differences between fitting a DL model to virtual or real-world point clouds?
3. What is the quantitative difference between using VLS-DL with meshes derived from real point clouds compared to fully computer-generated scenes?

2. Related work

2.1. Labeled datasets

Several open-access labeled point cloud datasets in the literature have been used for benchmarking (Guo et al., 2021). Most of them are from indoor or urban scenes. For instance, the widely used Stanford 3D Indoor Scene Dataset (S3DIS) (Armeni et al., 2016) and the outdoor Semantic3D dataset (Hackel et al., 2017) were acquired using static terrestrial laser scanning (TLS). Point clouds are also acquired with airborne platforms such as the DALES objects dataset covering ground, vegetation, vehicles, buildings, fences, and powerlines (Singer and Asari, 2021). Some state-of-the-art datasets, such as the KITTI 3D Object Detection benchmark (Geiger et al., 2012), are specifically oriented to robotics, which combines high-resolution video cameras and laser scanning.

While the availability of high-quality labeled datasets is increasing, they are still limited, e.g., to certain geographic regions, specific sensor systems, or specific objects (e.g., tree species). Annotating laser scanning point clouds manually is a cumbersome, time-consuming, and labor-intensive task. On top of that, interpreter bias and imperfect human annotation often cause label noise (Kölle et al., 2021; Hackel et al., 2016), especially for fine-scale structures and in partly occluded areas. This can influence the success rate of DL-based semantic segmentation. For example, González-Collazo et al. (2023) found a correlation ($R^2 = 0.765$) between the F1-score of a PointNet++ model and the percentage of concordance points (same class assigned by all annotators) in the training data.

The outdoor Hessigheim benchmark (Kölle et al., 2021), which we use in our study, is a particularly interesting labeled point cloud dataset for three reasons: (1) Acquired using an unoccupied aerial vehicle (UAV), it has a high point density of about $800\,\mathrm{pts/m^2}$ and achieves state-of-the-art representation of fine-grain features and also vertical elements, (2) it covers eleven different classes in the broader categories ground, buildings, vegetation and urban objects, and (3) it contains multiple epochs, captured in different seasons in 2018 and 2019. There is an online benchmark with many results for the March 2018 epoch comparing some well-known models (Gao et al., 2022; Qi et al., 2017b; Thomas et al., 2019).[1] Moreover, the Hessigheim 3D (H3D) dataset also includes annotated 3D meshes derived by combining the point cloud and oblique images. Table 1 represents the point-wise classification distribution of the different epochs, split into training and validation by the benchmark organizers.

For the case of leaf-wood separation, Vicari et al. (2019) have published the real-world and virtual validation data (Boni Vicari et al., 2018a,b) for their Python library TLSeparation (14 trees in total). Wang et al. (2020) provided the 61 labeled tree point clouds from Momo Takoudjou et al. (2018), which they used for validation of the automatic leaf-wood segmentation tool LeWoS (Wang et al., 2021). The labeled Quantitative Structure Models (QSMs) obtained from point clouds of the Wytham Woods research forest (Oxfordshire, UK) used by Calders et al. (2018) and Liu et al. (2022) to model radiative transfer in forest stands have also been made openly available. We use the Wytham Woods dataset for VLS-based model training for leaf-wood classification. The main real-world leaf-wood dataset for our experiments consists of 11 TLS point clouds of different species, which have been manually labeled (Weiser et al., 2023). The tree point clouds have between 500,000 and 10,600,000 points and there are always more leaf points than wood points. Furthermore, we apply our models to two additional datasets from the literature to investigate how the models generalize. Appendix B contains a detailed description on the real-world training and validation point cloud datasets.

We are considering two types of leaf-wood experiments: isolated and near trees (Appendix A.2, Appendix B). For the “leaf-wood isolated” case, the trees are placed isolated from each other such that any input neighborhood for the deep learning model will contain points from a single tree. In the “leaf-wood near“ case, the trees are close to each other, so their crowns overlap. Consequently, input neighborhoods potentially contain points from different trees.

2.2. Deep learning on point clouds

The PointNet model is generally accepted as the first relevant milestone of deep learning applied to point clouds because it achieved permutation invariance concerning input data while capturing the local structure of neighborhoods (Qi et al., 2017a). It was later extended to the PointNet++ model, which achieves hierarchical feature extraction similar to typical convolutional neural networks (CNN) following an incremental multiscale approach (Qi et al., 2017b). There are also extensions of PointNet++, such as alsNet, which uses a batching framework strategy to process vast airborne laser scanning (ALS) point clouds (Winiwarter et al., 2019). Other models, such as Kernel Point Convolution (KPConv), define the kernel as a finite set of points in the Euclidean space and a convolution operator based on a linear correlation where the distance between the kernel’s points and the input neighborhood’s points weights the contribution of each particular point to the extracted feature (Thomas et al., 2019). Other deep

[1] https://ifpwww.ifp.uni-stuttgart.de/benchmark/hessigheim/results.aspx (Accessed on 7 March 2023).

Table 1
Percentage of points per class for the Hessigheim datasets. The classes from left to right are low vegetation (C00), impervious surface (C01), vehicle (C02), urban furniture (C03), roof (C04), facade (C05), shrub (C06), tree (C07), soil/gravel (C08), vertical surface (C09), and chimney (C10).

Dataset	Point-wise class distribution percentage (%)										
	C00	C01	C02	C03	C04	C05	C06	C07	C08	C09	C10
March 2018 (training)	35.96	17.53	0.43	1.95	10.56	2.02	1.81	13.60	14.45	1.64	0.04
March 2018 (validation)	25.85	22.21	1.27	3.15	21.10	3.82	2.36	15.34	4.10	0.70	0.11
March 2019 (training)	36.67	18.42	0.71	1.57	19.21	2.63	4.61	14.03	0.85	1.20	0.10
March 2019 (validation)	27.45	18.99	1.18	2.85	26.92	4.27	5.49	10.79	0.99	0.93	0.15

learning proposals aim to solve the problem of point clouds being unstructured data spaces without topological information by estimating the implicit topology to improve the representation capabilities of raw point clouds. For instance, the Dynamic Graph Convolutional Neural Network (DGCNN) model uses a convolutional operator on the graph's edges representing a local neighborhood updated from layer to layer (Wang et al., 2019). Furthermore, there are models based on the sparse 3D convolutional neural networks introduced by Graham (2015). The main idea is to mitigate the dimensionality curse that arises when generalizing discrete 2D convolutions to 3D by using a hash table where the keys correspond to non-empty spatial locations, and the values are the index of the associated row in the input matrix. These sparse convolutional models have been successfully applied to point-wise semantic segmentation of point clouds (Graham et al., 2018; Schmohl and Sörgel, 2019).

The PointNet++ model has been modified to support 20,000 instead of 1024 points per sample to achieve sensor agnostic segmentation in forest point clouds (Krisanski et al., 2021b). This PointNet++-like model is part of the Forest Structural Complexity Tool (FSCT), which we also use in our work to compare virtual-to-real generalization with real-to-real generalization for leaf-wood segmentation (Krisanski et al., 2021a). Other works use a point-wise CNN to segment stems from leaves, e.g., to study maize plants from terrestrial LiDAR point clouds (Ao et al., 2022). Some approaches combine geometric features with corrected optical features and achieve 96.20% and 94.98% overall accuracy (OA) when performing leaf-wood segmentation on broadleaf and coniferous plants, respectively, and with an 84.26% OA on mixed vegetation contexts (Wu et al., 2020). The leaf-wood segmentation problem has also been approached as a time series problem comparing CNN, Long Short-Term Memory convolutional neural networks (LSTMCNN), and Residual Network (ResNet) models (Han and Sánchez-Azofeifa, 2022). Recent works are thoroughly studying the performance of PointNet++ for leaf-wood-flower segmentation depending on the number of scan positions and the amount of noise (Rousseau et al., 2022).

2.3. Virtual laser scanning

The high cost and inherent errors of point cloud annotation suggest exploring VLS as an alternative or complement to real data. There are software solutions to compute laser scanning simulations covering different scanner and scene configurations. For instance, the Blender Sensor Simulation Toolbox (BlenSor) modified Blender (Blender Online Community, 2023) to support casting many simultaneous rays, making it an efficient unified VLS and scene modeling framework (Gschwandtner et al., 2011). Recent studies used laser scanning simulations within Blender to create training data for machine learning on 3D point clouds. For instance, Hildebrand et al. (2022) worked on the classification of indoor scenes (like office and apartment rooms). However, for virtual point cloud data of larger outdoor scenes, simulations have to be physically more sophisticated, i.e., taking into account beam divergence, multiple returns, and complex sensors while handling large scenes and high simulated pulse frequencies. Blender-based solutions inherit the modeling and performance limitations from software oriented to 3D animation, visual effects, video games, and many more. We argue that obtaining the best VLS results demands specific VLS-oriented software. Consequently, we decided to use the general-purpose high-performance Heidelberg LiDAR Operations Simulator HELIOS++ (Winiwarter et al., 2022), to study its potential for successfully training deep learning models solely from virtual data. This software supports many input formats, such as triangle meshes, voxels, geographic tagged images (GeoTIFF) as raster models, and point clouds. Besides, it supports detailed XML specifications for different platforms and scanners. Instead of a ray tracing implementation similar to the native ray tracing of computer games such as Grand Theft Auto V (GTA V) and 3D graphics software such as Blender, HELIOS++ provides an efficient object-aware ray tracing algorithm that supports the high-performance parallel computation of different physical models on the light ray (Esmorís et al., 2022).

There has been some previous work using synthetic data to train DL models. Hurl et al. (2019) achieve a 5% improvement in average precision when using the Precise Synthetic Image and LiDAR (PreSIL) dataset to pre-train object detection models validated against the KITTI 3D Object Detection benchmark (Geiger et al., 2012). The PreSIL dataset includes point clouds generated from GTA V virtual scenes. It uses an alternative to classical ray casting based on projecting the rays on an image plane to compute a depth interpolation that leads to more accurate object representations. Others have implemented an in-game LiDAR in GTA V to generate virtual 3D point clouds and improve the performance of CNN-like models at point-wise semantic segmentation. Enriching real data with virtual data boosted classification intersection over union (IoU) on real validation data by around 8.9% (Wu et al., 2018). Furthermore, Bryson et al. (2023) explored a variant of leaf-wood segmentation using virtual data to train a deep learning model based on the PointNet++ architecture. Instead of typical leaf and wood classes, they considered stem (main tree stem and large branches) and foliage (canopy and small branches). They found that models trained on virtual data can outperform models trained on real data when there are not enough labeled real point clouds to achieve convergence. However, with abundant real data, models trained on real data outperform those trained on virtual data.

3. Method

Central to our approach is the combination of virtual laser scanning (VLS) and deep learning (DL) for point cloud semantic segmentation. In this section, we first consider the main theoretical components of VLS and DL applied to point clouds. We then present our experimental framework, where we train our deep learning model with (a) only real data and (b) only VLS data and then evaluate the performance by predicting class labels on a withheld validation set of the real data. The whole workflow is illustrated in Fig. 1. We provide a detailed description of the virtual scene modeling and the simulated platform and scanner in Appendix A and the real training and validation point clouds in Appendix B.

3.1. Theoretical model

The theoretical VLS-DL model illustrates the key components of a virtually trainable model based on connecting simulation and deep

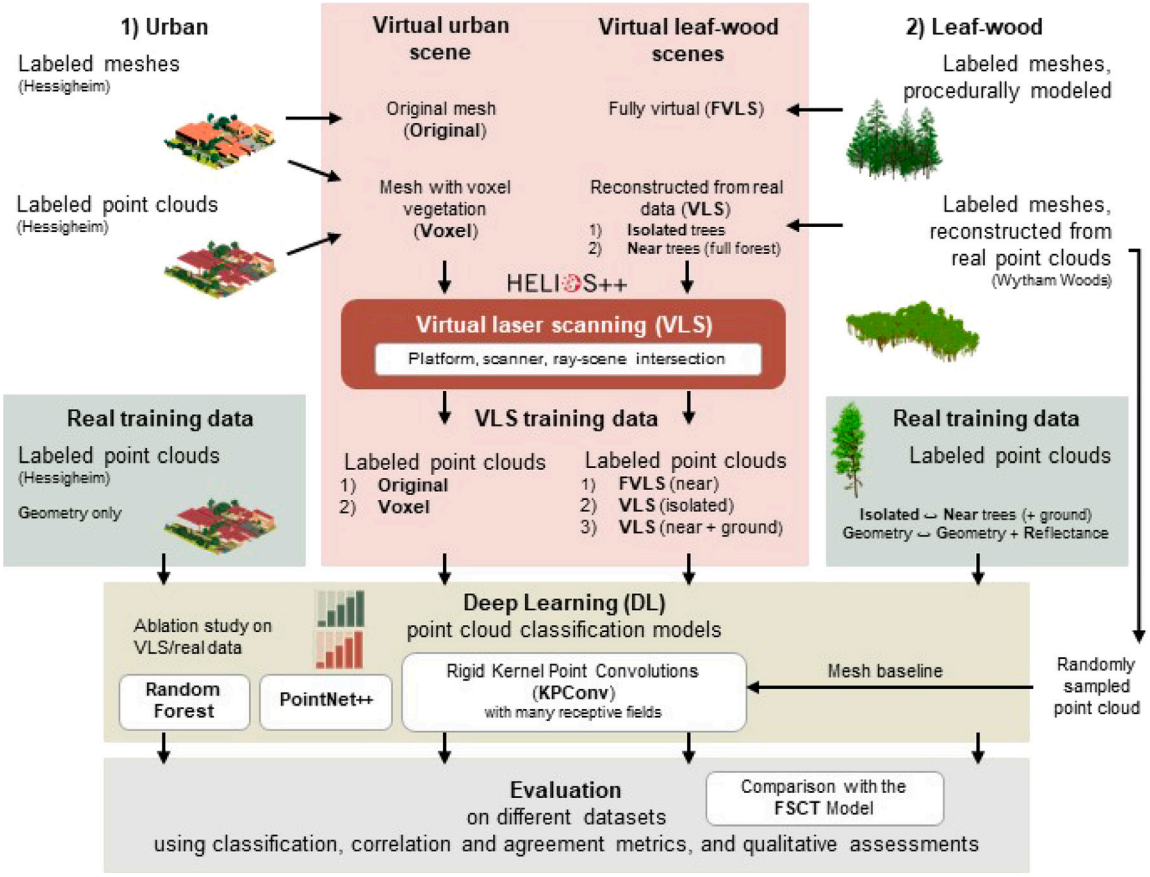

Fig. 1. The workflow summarizes the entire experimental framework. The top shows the generation of 3D scenes, which are virtually scanned with HELIOS++ to create virtual labeled point clouds for training the VLS-DL models for (1) urban classification and (2) leaf-wood classification. For the urban case, meshes and point clouds from the Hessigheim3D dataset (Kölle et al., 2021) are used. For the leaf-wood case, synthetic meshes generated with the algorithm by Weber and Penn (1995), and the Wytham Woods 3D model (Calders et al., 2018; Liu et al., 2022) are used. Real labeled point clouds (Hessigheim for the urban case, point clouds from Weiser et al. (2023) for the leaf-wood case) serve as training data for the deep learning (DL) models. The performance of the models trained on real data and those trained on virtual data are evaluated and compared quantitatively and qualitatively. The leaf-wood models are also compared with the FSCT model (Krisanski et al., 2021a) and a mesh sampling baseline model. The ablation study compares the performance of VLS and real training datasets on many models with different amounts of annotated points.

learning. In this section, we describe the many submodels and how they are connected from the first (parametric ray generation) to the last (neural network).

The VLS-DL model on a 3D Euclidean space starts by simulating a trajectory using a parametric model (Stewart, 2012) with position $(x_1(t), x_2(t), x_3(t))$ and associated direction $(\varphi_1(t), \varphi_2(t), \varphi_3(t))$. A ray is defined from an origin point $\boldsymbol{o}_i$ and an associated director vector $\boldsymbol{v}_i$ such that $\{\boldsymbol{o}_i + t\boldsymbol{v}_i : t \geq 0\}$ (Boyd and Vandenberghe, 2004). However, for VLS, the travel time of the ray is used to estimate the distance. Consequently, a ray in VLS context must be defined as $\{\boldsymbol{o}_i + t\boldsymbol{v}_i : t \geq \epsilon\}$, where $\epsilon \in \mathbb{R}_{>0}$ is the minimum distance threshold defining the scanner. Thus, it is possible to express the finite set of rays in the simulation such that $\mathfrak{r}_i = \{\boldsymbol{o}_i = o\left[t_i, x_1(t_i), \ldots, \varphi_3(t_i)\right], \boldsymbol{v}_i = v\left[t_i, x_1(t_i), \ldots, \varphi_3(t_i)\right]\}$, where o and v map the parametric model to the final ray. These maps can be as simple as the identity function or as complicated as the composition of many reference systems mixed with non-linear deflection models.

Let V be a finite set of points in $\mathbb{R}^3$ that lie on a common plane where they define a convex polygon. This convex polygon can be seen as the feasible region of a linear programming problem (LP) (Solow, 2014). Moreover, as the objective function is null, any solution is optimal, which means the LP can be seen as a feasibility problem. Let $\boldsymbol{V} \in \mathbb{R}^{2\times|V|}$ be the matrix representing the vertices in V on the local coordinates of their plane. Since each convex polygon is a convex set, any convex combination of its vertices $\sum_j^{|V|} \lambda_j \boldsymbol{v}_{*j}$ satisfying $\sum_j^{|V|} \lambda_j = 1$ and $\lambda_j \geq 0$ will be inside the polygon (Boyd and Vandenberghe, 2004), where $\boldsymbol{v}_{*j}$ is the row j of matrix $\boldsymbol{V}$. The convex combination of the vertices of a convex polygon is illustrated on the left side of Fig. 2 , while the finite set of feasible regions is illustrated on the right side. Any scene point $\boldsymbol{p}$ must belong to the plane defined by a set of vertices V_i and lie inside a feasible region when expressed as the point $\boldsymbol{q}$ in the local coordinates of this plane. Then, all scene points can be modeled as belonging to the union of a finite set of planes π_i constrained by linear systems $\boldsymbol{V}_i'\lambda_i = \boldsymbol{q}'$ subject to $\lambda_i \geq 0$. Note for any polygon defined by the vertices V_i the constraint $\sum_j^{|V_i|}(\lambda_i)_j = 1$ is implicit on the definition of $\boldsymbol{V}_i'$ and $\boldsymbol{q}'$ given in Eq. (1), both expanded with ones.

$$\boldsymbol{V}_i' = \begin{bmatrix} \boldsymbol{V}_i \\ \mathbf{1}^\intercal \end{bmatrix}, \boldsymbol{q}' = \begin{bmatrix} \boldsymbol{q} \\ 1 \end{bmatrix} \tag{1}$$

To conclude the VLS model, note that the feasible regions contain infinite points, but the amount of rays is finite. Thus, the intersection between the rays and the feasible regions leads to a finite set of points constituting the baseline solution of the ray tracing VLS model. The final output point set P generated through VLS can be defined as the set of points that satisfy all the constraints given in Eq. (2). In this equation, each $\psi_k \leq \tau_k$ stands for each of the K non-necessarily linear filters. These filters can be physical-based (e.g., $-\psi_k$ is the power of the reflected light at the sensor and $-\tau_k$ is the minimum power the sensor can detect) or simulation conveniences (e.g., ψ_k is the distance between origin and target points and τ_k is an arbitrary maximum distance threshold). The input vector $\boldsymbol{\theta}$ represents other potential simulation

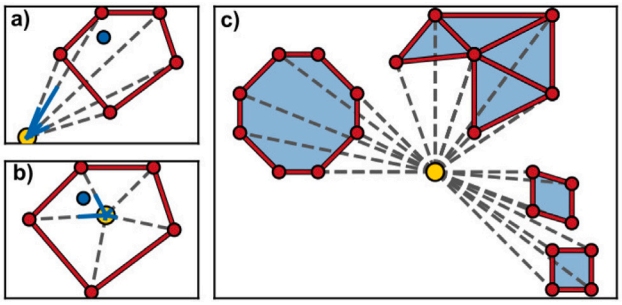

Fig. 2. The left side figures (a) and (b) show how a point (blue) inside a convex set can be expressed as a convex combination of its vertices (red) understood as vectors (gray) from the origin (yellow). Note that the blue highlighted segments represent the magnitude of displacement on the vector for the convex combination. The right figure (c) shows that many intersections of halfspaces can be seen as many different linear feasible regions (blue) representing a linear system each. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

parameters (e.g., the wavelength).

$$P = \left\{ \boldsymbol{p} \in \mathbb{R}^3 : \begin{array}{ll} \boldsymbol{p} = \boldsymbol{o}_j + t\boldsymbol{v}_j \in \pi_i, & t \geq \epsilon, \\ \boldsymbol{V}_i' \lambda_i = \boldsymbol{q}', & \lambda_i \geq 0, \\ \bigwedge_k^K \psi_k(\boldsymbol{p}, \pi_i, \tau_j, t, \boldsymbol{\theta}) \leq \tau_k & \end{array} \right\} \tag{2}$$

From now on, the matrix $\boldsymbol{P} = [\boldsymbol{X} \mid \boldsymbol{F} \mid \boldsymbol{y}] \in \mathbb{R}^{m\times 5}$ will refer to the output point cloud of m points generated with VLS where $\boldsymbol{X} \in \mathbb{R}^{m\times 3}$ represents the simulated geometric data, $\boldsymbol{F} \in \mathbb{R}^{m\times 1}$ is a column-vector matrix of ones to enable feature extraction with deep learning, and $\boldsymbol{y} \in \mathbb{Z}^{m\times 1}$ is a column-vector such that y_i is an integer representing the class to which the i-th point belongs. This matrix $\boldsymbol{P}$ can also be seen as the input to a fitting algorithm where a loss function $\mathcal{L}$ is defined for an estimator $\hat{y}(\boldsymbol{X}, \boldsymbol{F}; \boldsymbol{\omega})$ that uses the parameters $\boldsymbol{\omega}$ to predict the class. Since deep-learning models are based on gradient descent-like optimization methods (Zhang, 2017; Goodfellow et al., 2016a), the VLS-DL model can be summarized as Eq. (3), where α represents the learning rate. Before adventuring forth, it is worth mentioning that the proposed training data generation where $\boldsymbol{y} \in \mathbb{Z}^m$ can be easily adapted to regression problems where $\boldsymbol{y} \in \mathbb{R}^m$ (e.g., $\boldsymbol{y}$ can be a point-wise surface roughness metric).

$$\boldsymbol{\omega}_{t+1} = \boldsymbol{\omega}_t - \alpha \nabla_{\boldsymbol{\omega}_t} \mathcal{L}\left(\boldsymbol{P}; \boldsymbol{\omega}_t\right) \tag{3}$$

Deep learning applied to point clouds often requires a domain-specific operator for feature extraction. For example, it is possible to address this issue using the Kernel-Point Convolution (KPConv) operator (Thomas et al., 2019) as the feature extraction method for the estimator $\hat{y}$.

First, let $Q = \left\{\boldsymbol{Q} \in \mathbb{R}^{K\times 3}, \boldsymbol{W}_1, \ldots, \boldsymbol{W}_K \in \mathbb{R}^{D_{\text{IN}}\times D_{\text{OUT}}}\right\}$ represent a kernel of K points (regularly distributed by solving an energy minimization problem) at a given layer that maps D_{IN} input features to D_{OUT} output features. For then, any point from the geometric data $\boldsymbol{x}_{i*} \in \mathbb{R}^3$ can be convolved considering its neighborhood $\mathcal{N}_{\boldsymbol{x}_{i*}}$, e.g., $\mathcal{N}_{\boldsymbol{x}_{i*}} = \left\{\boldsymbol{x}_{j*} : \|\boldsymbol{x}_{j*} - \boldsymbol{x}_{i*}\| \leq r\right\}$ (where r is the radius). The rigid KPConv operator based on linear correlation is formally described in Eq. (4). In this equation, $\boldsymbol{q}_{k*}$ are the points defining the kernel in the Euclidean space, i.e., rows from the $\boldsymbol{Q}$ matrix, the $\boldsymbol{W}_k$ matrices are the weights for each kernel point, the $\boldsymbol{f}_{j*}$ vector is the j-th row of the input features matrix $\boldsymbol{F}$, and σ defines the influence distance.

$$(\boldsymbol{P} * Q)(\boldsymbol{x}_{i*}) = \sum_{\boldsymbol{x}_{j*}\in\mathcal{N}_{\boldsymbol{x}_{i*}}} \sum_{k=1}^{K} \max\left\{0, 1 - \frac{\|\boldsymbol{x}_{j*} - \boldsymbol{x}_{i*} - \boldsymbol{q}_{k*}\|}{\sigma}\right\} \boldsymbol{W}_k \boldsymbol{f}_{j*} \tag{4}$$

3.2. Experimental framework

Our experiments shall be representative of many topographic applications and thus cover urban (Hessigheim) and natural (leaf-wood) contexts for which real data is available. We generate virtual point clouds for those contexts using the VLS software HELIOS++ and assess the performance of deep learning-based semantic segmentation using real or virtual data for the training. We also train models using real data with reflectance for the leaf-wood segmentation cases to compare them with the real and virtual models using only geometric data. For evaluation, we use classification quality assessment metrics, correlation and agreement metrics, expert-based evaluation, and a comparison with a model trained from randomly sampled points from the scene meshes. Moreover, we compare the virtual-to-real generalization of our VLS-DL model with the real-to-real generalization of the FSCT model (Krisanski et al., 2021a) (Fig. 1).

We specialize our VLS-DL model to work with geometric data. Signal strength-related features are discarded because they are not well-standardized for all sensors and are more difficult to compare (Höfle and Pfeifer, 2007; Jutzi and Gross, 2009; Wang et al., 2015). We use the grid subsampling strategy proposed in the original KPConv paper (Thomas et al., 2019) to explore the resolution from the deep learning model perspective.

Moreover, we use the VL3D framework[2] to conduct additional experiments to analyze the VLS-DL approach with different models. This open-source framework allows us to train, use, and evaluate Random Forest, PointNet++, and KPConv for 3D point cloud classification tasks. Thus, we compare the performance of the three models through an ablation study on the amount of training data to find out (1) what the impact of the amount of simulated data is in terms of generalization to unseen real data and (2) how the VLS-DL approach performs with different machine learning models. The ablation strategy is based on removing annotated points representing entire trees rather than downsampling the point cloud. We follow this approach to study the impact of the number of annotated instances, not the point density.

3.2.1. Virtual laser scanning

For the urban classification experiments, we use virtual 3D scenes reconstructed from real 3D data, namely from UAV-borne point clouds and imagery of Hessigheim. Two versions of the virtual Hessigheim scene (Kölle et al., 2021) are created (Table 2). The first version directly uses the original labeled meshes from the Hessigheim 3D benchmark. For the second version, the Hessigheim mesh is modified by replacing mesh faces labeled as the vegetation classes with voxel models created from the Hessigheim3D point cloud (Table 2). Our hypothesis is that voxels, if using an appropriate voxel size, are an adequate object representation and lead to better classification performance because the virtual rays can penetrate through gaps in the vegetation (Weiser et al., 2021b). We use the same training and validation split as in the Hessigheim 3D benchmark.

Separate materials are assigned to the different classes through material library files (MTL). We exploit the HELIOS++ functionality of assigning integer class IDs to the materials by adding the custom *helios_classification* attribute (Winiwarter et al., 2022). In the simulation, the class labels are then transferred from the object intersected by the virtual ray to the point created through the intersection. In the case of multiple intersections, each one will lead to a point with a class coming from the respective intersected object.

With our experiments of leaf-wood classification, we cover two central ways to generate virtual scenes, namely (1) with procedural 3D modeling, where objects are fully computer generated, and (2) by reconstructing real scenes from high-resolution real-world measurements.

[2] https://github.com/3dgeo-heidelberg/virtualearn3d (Accessed on 27 May 2024).

Table 2
Description of the virtual scenes used in the HELIOS++ laser scanning simulations for the leaf-wood classification and the urban classification use cases.

Leaf-wood classification	
(1) Fully Virtual Forest Stand	Up to 30 tree models are created using the algorithm by Weber and Penn (1995) and arranged into a forest stand with overlapping crowns.
(2) Reconstructed Virtual Forest Stand	The full Wytham Woods 3D forest model is used: Trees are closely spaced, their crowns are overlapping and terrain in the form of a meshed digital terrain model is included.
(3) Reconstructed Isolated Trees	Six synthetic datasets are used, each by manually placing eight randomly selected 3D tree models from the Wytham Woods dataset in a circular plot. Each tree is isolated, and the tree crowns do not overlap.
Urban scene classification — Hessigheim	
(1) Original Mesh	The original Hessigheim3D mesh is used as the 3D VLS scene.
(2) Modified mesh with voxel vegetation	Vegetation (Shrub and Tree classes) are removed from the Hessigheim 3D mesh and modeled from the point clouds using small voxels (5 cm × 5 cm × 5 cm). Furthermore, the surface below the vegetation is reconstruction to prevent holes. The resulting hybrid mesh-voxel model serves as the 3D VLS scene.

We refer to point clouds we simulate with the resulting datasets as (1) fully virtual laser scanning (FVLS) and (2) VLS.

For the fully synthetic scenes, 30 tree models with different leaf and needle shapes are generated using the algorithm of Weber and Penn (1995) with different parameter sets. All trees are arranged in a small forest stand, with their crowns clearly overlapping.

For the scenes reconstructed from real data, we use the synthetic Wytham Woods scene of quantitative structure models (QSMs) provided by Calders et al. (2018) and Liu et al. (2022).[3]

We create two versions of the virtual Wytham Woods scene, one with selected isolated trees and one with the full forest plot (closely spaced or "near" trees) (Table 2). The full Wytham Woods forest scene is spatially divided into training and validation.

Two different 3D parametric platforms and scanner models are used for the laser scanning simulations of the different scenes.

For the Hessigheim simulations, we use a model of the *RIEGL* VUX-1LR scanner (RIEGL Laser Measurement Systems, 2022) and similar acquisition settings as in the Hessigheim March 2018 epoch (Cramer et al., 2018). These are summarized in Table A.9 of Appendix A.1. The parametric model linearly approximates the real trajectory, assuming a constant speed of 8 m/s. The o map transforms the point position by composing the platform and the scanner's reference systems. The υ map transforms the direction considering the scanner head expressed in the platform's local reference system, where a rotating mirror deflection model is solved at each simulation step to determine the ray's direction.

The 3D tree scenes for leaf-wood classification are scanned with multiple scan positions using a model of the *RIEGL* VZ-400 TLS (RIEGL Laser Measurement Systems, 2017) and similar scan settings as used in the validation dataset (Table A.10 in Appendix A.2). Since these are tripod-based simulations, any $x_i(t)$ and $\varphi_i(t)$ functions are constants representing the tripod's static position. The o map is the identity function, while the υ map directly solves a polygon mirror deflection model matching the scanner specification.

The process of virtual scene generation and virtual laser scanning for both the urban and the leaf-wood experiments is described in detail in Appendix A.

Finally, a point cloud obtained by sampling points from the mesh surface is also generated for the near trees leaf-wood case (full Wytham Woods forest stand) to quantify the extent to which VLS contributes to better feature extraction on the DL side. The resulting training dataset has 69,999,073 points, which is slightly higher but similar to the 63,297,807 points we use for training with VLS point clouds merged from different scan positions. This dataset is used to train the baseline Mesh-DL model.

[3] https://bitbucket.org/tree_research/wytham_woods_3d_model/src/add_dart/DART_models/ (Accessed on 19 October 2022).

3.2.2. Deep learning

For the details regarding the KPConv network architecture, we would like to refer to Thomas et al. (2019). In the following, we focus on the main particularities characterizing this work.

First, we explain our training procedure. It consists of training a model for a fixed number of epochs, then using the model to classify validation data that has not been seen before and repeating until a maximum number of training processes has been reached. Each training process draws a different randomly selected set of neighborhood centers for training. We use the evaluations on validation data to assess training evolution and compare virtual training with real training exhaustively.

Instead of using an exponential learning rate decay as usually done with KPConv models (Thomas et al., 2019; de Gélis et al., 2023), we use a combination of early stopping and reducing learning rate on the plateau. We do this to decrease our learning rate smoothly on demand for each training process instead of a priori deciding on a fixed number of epochs for the learning rate decay. More concretely, we monitor the sparse categorical cross-entropy loss function with 50 epochs patience for the early stopping of a training process and reduce the learning rate multiplying by $10^{1/3}$ with 20 epochs patience and an additional 10 epochs cooldown for consecutive reductions. This configuration can lead to dividing the learning rate by 10 for every hundred epochs, as in the original proposal, but it will only trigger the reduction if the loss function reaches a plateau. The patience count for the learning rate reduction is preserved among training processes. As in the original model, we use 400 epochs per training process for the Hessigheim3D point clouds. We use 200 epochs for leaf-wood point clouds because they converge much faster. In both cases, we use an initial learning rate of 10^{-3}.

For the experiments on real data that also use reflectance, we normalize all the reflectance values to lie inside the $[0, 1]$ interval in a dataset-dependent way. Generalized approaches are impossible since intensity and reflectance often change between datasets.

As in Thomas et al. (2019), we divide the large point clouds into small subclouds contained in spheres. We distribute the spheres along the scene with a regular spacing of 0.5 times the sphere radius. Separations greater than $2/\sqrt{3}$ times the radius will lead to missing regions for 3D scenes. We selected 0.5 times because it is small enough to increase classification overlapping (which increases reliability) but not too big to lead to intractable classifications. After classification, probabilities for points that appear in multiple spheres are averaged, and the class with the maximum average probability is assigned.

3.2.3. Evaluation

The model evaluation metrics can be classified into aggregated metrics and per-class metrics. For the aggregated metrics, we are computing the Overall Accuracy (OA) (Sokolova and Lapalme, 2009),

Table 3
Aggregated evaluation and agreement metrics for the Hessigheim datasets in March 2018 and March 2019. The KPConv model was trained with real or virtual data. There are two different VLS training datasets. One uses the original triangle mesh, and the other uses a hybrid scene with the original triangle mesh and voxels to represent vegetation. The metrics from left to right are Overall Accuracy (OA), Precision (P), Weighted Precision (WP), Recall (R), Weighted Recall (WR), mean Intersection over Union (mIoU), Weighted mean Intersection over Union (WIoU), F1 score (F1), Weighted F1 score (WF1), Matthews Correlation Coefficient (MCC), and Cohen's Kappa score (K).

Train	Validation	Metrics (%)										
		OA	P	WP	R	WR	mIoU	WIoU	F1	WF1	MCC	K
VLS (mesh)	March 2018	66.1	44.5	70.4	57.1	66.1	33.6	51.9	46.4	67.1	58.4	58.1
VLS (voxel)	March 2018	74.1	52.3	78.5	67.4	74.1	42.6	63.0	55.2	75.7	68.0	67.7
Real	March 2018	82.4	72.9	79.9	59.2	82.4	51.0	71.0	63.1	80.5	78.2	78.0
VLS (mesh)	March 2019	63.9	44.8	64.1	38.1	64.0	28.0	49.0	39.0	63.6	54.8	54.7
VLS (voxel)	March 2019	74.4	47.8	76.1	64.6	74.4	39.1	62.3	50.4	75.0	67.9	67.7
Real	March 2019	81.4	54.7	82.4	66.1	81.4	46.0	71.3	57.9	81.7	76.6	76.6

the Jaccard score or Intersection over Union (IoU) (Jaccard, 1901), the multiclass Precision and Recall, the F1 score (Sokolova and Lapalme, 2009), the Matthews Correlation Coefficient (MCC) (Matthews, 1975), and Cohen's Kappa score (Cohen, 1960). The evaluation metrics can be categorized into classification quality assessment (e.g., OA and F1) and correlation and agreement assessment (e.g., MCC and Kappa score). When aggregating the evaluation metrics, both the weighted and unweighted averages are considered in a non-label-agnostic way to assess the models with and without accounting for class imbalance. For a per-class evaluation, we compute the F1 scores, as used by Kölle et al. (2021) for evaluation of the Hessigheim 3D benchmark. We also carry out an expert-based visual evaluation for the leaf-wood case.

Furthermore, we design a specific evaluation method to compare VLS-sensed point clouds with randomly sampled points on the virtual leaf-wood mesh. For this evaluation, we consider the geometric features of linearity and planarity (Weinmann et al., 2015; Hackel et al., 2016) to characterize a spherical neighborhood defined by a 5 cm radius. Then, we can compare these quantitative features to analyze the difference between VLS and mesh sampling.

Finally, concerning our ablation study on the amount of training data for different models, we compute the experiment five times for each combination of model and training data percentage and consider the mean of the five repetitions as the final result. We also compute the standard deviation because it allows us to understand how stable a training process is, i.e., how different the model performance can be depending on the stochastic initialization of the weights. We select the IoU to evaluate these experiments as a function of the percentage of training data. Thus, we can fit a line $y = \alpha x + \beta$, where y is the mean IoU (as a percentage), and x is the percentage of training data (where 100% represents the full training dataset with no ablation). Then, analyzing the slope of this line will reveal whether using more training data yields better results ($\alpha > 0$) or not ($\alpha <= 0$). We also compute the Pearson correlation coefficient (Pearson, 1895; Stigler, 1989) r to quantify the correlation between more training data and a better IoU. If $r \to 1$, the correlation is confirmed; if $r \to -1$, the correlation is the opposite; and if $r \to 0$, there is no correlation.

4. Results

In this section, we present the results of our experiments. We start with an aggregated comparison between models. Then, we provide detailed results for both the Hessigheim and the leaf-wood experiments. Finally, we present the results of the comparison between mesh sampling, VLS, and real-world point clouds.

4.1. Aggregated comparison

We compared different versions of the KPConv model, varying the initial cell size for the grid subsampling, which defines the receptive field of the finest grain layers. Fig. 3 shows a summary of these results for the Hessigheim datasets (a, b, c, d) and the near trees leaf-wood case (e, f, g, h). We used one more training process for the leaf-wood segmentation experiments because their training requires fewer epochs than semantic segmentation in urban contexts.

For both the March 2018 and the March 2019 real-world Hessigheim datasets, OAs above 80% and MCCs around 75% were achieved. These scores derived with real data act as a baseline against which we evaluate the results of the VLS-DL model. While the OA of the VLS-DL model based on the original Hessigheim mesh data is only 66.1%, we can improve this value by 8% up to 74.1% by replacing the mesh representation of vegetation (C06: shrub and C07: tree) with a voxel model, which results in a more realistic synthetic point cloud for these classes. Larger initial cell sizes in the neural network work better for real and virtual Hessigheim point clouds. Small receptive fields (2 and 3 cm) perform poorly in these datasets because they barely contain any information about the global context.

The best models from the Hessigheim experiments are compared in Table 3. For the geometric KPConv case, this is the model with an initial cell size of 9 cm, and for the virtual case, this is the model with an initial cell size of 10 cm (Fig. 3). Using the original mesh as the VLS input scene leads to a 16.3% lower OA for March 2018 and 17.5% lower OA for March 2019 compared to the real data. Improving the scene with voxels for vegetation reduces this difference to 8.3% in OA for March 2018 and 7.0% for March 2019. Compared to the real geometric model, the original mesh VLS-DL model results in an MCC reduction of 19.8% for March 2018 and 21.8% for March 2019. With the virtual scene with voxels, the MCC reduction is about 10.2% for the March 2018 epoch and 8.7% for the March 2019 epoch. These results indicate that improved scene modeling is fundamental for VLS-based multiclass semantic segmentation in urban point clouds. There are no significant differences between March 2018 and March 2019 epochs.

For the leaf-wood datasets, Fig. 3 shows that for real models, a smaller initial cell size (1 and 2 cm) works better than a bigger one. These results suggest that fine-grain information is important to separate the wood components (trunk and branches) from the leaves. However, VLS-DL models perform better on real datasets when trained on bigger initial cell sizes (5 and 6 cm), while they perform better on virtual datasets with smaller cell sizes. These results suggest that fine-grain vegetation modeling in the input scene might lead to even better VLS-DL models for leaf-wood segmentation.

Table 4 presents a quantitative comparison between the best leaf-wood models (Geometric, Reflectance, VLS-DL, and FVLS-DL) and the FSCT model (Krisanski et al., 2021a).

For the case of the isolated trees, the model trained on geometric data achieves only 0.3% more OA and 1.1% higher MCC than the FVLS-DL model, while considering reflectance leads to 2.2% greater OA and 6.5% greater MCC. The virtual-to-real generalization of the FVLS-DL model is 9.7% better in OA and 19.7% better in MCC than the real-to-real generalization of the FSCT model trained on a different dataset than ours (Krisanski et al., 2021a). For the near trees case, the real model trained on reflectance is not considerably better than the model trained on just the geometry. Here, the VLS-DL model (trained on the Wytham Woods synthetic point cloud) has higher accuracy than the FVLS-DL model, and achieves just 1% lower OA and 3.5% lower MCC than the real models.

Table 4
Aggregated evaluation and agreement metrics for the leaf-wood datasets with isolated and near trees, respectively. The KPConv models (KPC) were trained with real or virtual data. The FSCT model (Krisanski et al., 2021a) is used to quantify the real-to-real generalization with a different model than ours. In the features column, G means geometric features, and R means reflectance. The VLS model is trained on point clouds simulated with meshes derived from real trees, while the FVLS model is trained on simulated point clouds of fully synthetic trees. The metrics from left to right are Overall Accuracy (OA), Precision (P), Weighted Precision (WP), Recall (R), Weighted Recall (WR), mean Intersection over Union (mIoU), Weighted mean Intersection over Union (WIoU), F1 score (F1), Weighted F1 score (WF1), Matthews Correlation Coefficient (MCC), and Cohen's Kappa score (K).

Model	Features	Validation	Metrics (%)										
			OA	P	WP	R	WR	mIoU	WIoU	F1	WF1	MCC	K
FVLS KPC	G	Isolated	93.2	88.2	93.8	92.4	93.2	82.3	87.8	90.0	93.3	80.4	80.1
VLS KPC	G	Isolated	87.0	84.1	86.7	81.6	87.0	71.5	77.4	82.7	86.8	65.7	65.5
Real KPC	G	Isolated	93.5	93.2	93.5	88.5	93.5	83.1	87.8	90.5	93.3	81.5	81.1
Real KPC	G+R	Isolated	95.4	91.0	96.0	96.3	95.4	87.6	91.7	93.3	95.6	87.1	86.6
FSCT	G	Isolated	83.5	77.6	86.3	83.4	83.5	66.9	73.6	79.5	84.3	60.7	59.3
FVLS KPC	G	Near	92.6	87.4	92.7	87.8	92.6	78.8	86.9	87.6	92.7	75.2	75.2
VLS KPC	G	Near	93.7	91.0	93.5	87.0	93.7	80.6	88.4	88.8	93.5	77.8	77.6
Real KPC	G	Near	94.7	96.0	94.9	86.0	94.7	82.5	89.8	90.0	94.4	81.3	80.0
Real KPC	G+R	Near	94.8	94.8	94.8	87.0	94.8	83.0	90.0	90.3	94.5	81.4	80.7
FSCT	G	Near	88.7	80.3	89.5	83.9	88.7	70.9	81.1	81.9	89.0	64.1	63.9

Table 5
Evaluation of different submitted models on the test data from the Hessigheim3D benchmark (https://ifpwww.ifp.uni-stuttgart.de/benchmark/hessigheim/results.aspx). The aggregated metrics are overall accuracy and mean F1 score. The F1 score is also calculated on a per-class basis. The geometric KPConv and VLS-DL models are our real and virtual models. The classes from left to right are low vegetation (C00), impervious surface (C01), vehicle (C02), urban furniture (C03), roof (C04), facade (C05), shrub (C06), tree (C07), soil/gravel (C08), vertical surface (C09), and chimney (C10).

Model	OA(%)	F1(%)	F1 per-class (%)									
			C00	C01	C02	C03	C04	C05	C06	C07	C08	C09	C10
WHU221118	89.8	79.0	92.9	90.2	78.5	57.9	95.7	80.4	68.5	97.2	62.4	73.1	72.5
Shi220705	84.2	63.5	87.6	85.6	52.4	36.7	95.5	69.3	47.4	94.3	25.1	66.0	38.6
Zhan221025	79.7	65.3	84.3	77.9	58.1	42.3	93.3	65.4	53.5	95.3	23.7	59.9	64.7
Gao-PN++210422	68.5	41.2	78.1	72.1	31.8	13.7	74.0	47.8	28.3	71.8	9.7	21.7	4.4
jiabin221114	58.3	43.4	66.2	18.0	34.2	38.0	72.0	69.0	47.7	78.7	9.8	35.9	8.3
VLS-DL	68.8	54.2	71.1	55.8	2.7	25.9	93.7	73.3	39.0	93.2	20.7	70.4	49.8
Geometric KPConv	81.7	63.8	84.8	82.0	19.5	42.4	94.8	77.8	59.1	94.5	3.6	73.8	69.2

Table 6
Evaluation and agreement metrics per class for the Hessigheim datasets in March 2018 and March 2019. The real geometric KPConv model was trained with real data. There are two different VLS-DL models. One uses the original triangle mesh, and the other uses a hybrid scene with the original triangle mesh and voxels to represent vegetation. The classes from left to right are low vegetation (C00), impervious surface (C01), vehicle (C02), urban furniture (C03), roof (C04), facade (C05), shrub (C06), tree (C07), soil/gravel (C08), vertical surface (C09), and chimney (C10).

Model	Validation	F1 per-class (%)									
		C00	C01	C02	C03	C04	C05	C06	C07	C08	C09	C10
VLS-DL Original	March 2018	65.2	68.3	12.1	31.9	81.1	61.4	25.9	74.4	3.7	46.8	39.6
VLS-DL Voxel	March 2018	71.0	70.7	18.7	33.9	90.5	72.7	44.1	88.8	5.1	50.8	61.1
Real Geometric	March 2018	83.3	84.2	41.0	50.2	91.6	74.7	51.0	92.9	1.0	52.6	71.3
VLS-DL Original	March 2019	62.8	59.7	1.3	29.5	85.4	59.9	35.7	59.5	0	27.3	8.1
VLS-DL Voxel	March 2019	70.0	61.8	16.8	35.7	92.9	69.1	69.0	88.8	1.5	25.2	24.1
Real Geometric	March 2019	80.7	83.6	32.4	43.0	93.3	78.5	53.5	88.5	1.0	46.3	35.7

4.2. Hessigheim results

We evaluated our models in detail on the Hessigheim3D (Kölle et al., 2021) public test benchmark. The results of the Hessigheim3D benchmark in Table 5 show that our real models provide average results despite using geometric information only. While using real data leads to similar evaluation on validation and test, the VLS-DL model generalizes worse to the test dataset than to the validation dataset (12.9% decrease in OA compared to 8.3% decrease with validation data). However, it still provides results that match the performance of PointNet++-based models such as Gao-PN++210422 trained on real data.

Returning to the validation dataset, Table 6 offers the quantitative evaluation of our models on a per-class basis. While low vegetation and impervious surfaces give acceptable results, all models have problems with soil/gravel points. This can be explained by the difficulty of distinguishing between types of ground points using geometric data only and no spectral/radiometric information. These results can also be analyzed in confusion matrices, shown in Fig. 4 for the real geometric KPConv model and in Fig. 5 for the VLS-DL model. The confusion matrices in Fig. 6 are aggregated on the main categories ("ground" for low vegetation, impervious surface, and soil/gravel; "object" for vehicles, urban furniture, and vertical surface; "building" for roof, facade, and chimney; "non-low vegetation" for shrub and tree). Both models – the geometric KPConv model trained on real data and the VLS-DL model – offer outstanding performance on point-wise ground classification when there is no need to distinguish between types of ground. Quantitatively, the geometric KPConv model trained on real-world data achieves 92.2% OA and 87.5% MCC in the aggregated semantic segmentation, while when trained on virtual data, it achieves 90.2% OA and 84.5% MCC, respectively.

The point-wise classification of building and non-ground vegetation offers promising results for real and virtual models. The many objects in the Hessigheim point clouds (vehicles, urban furniture, and vertical surfaces) are the main problem for both models. An explanation could

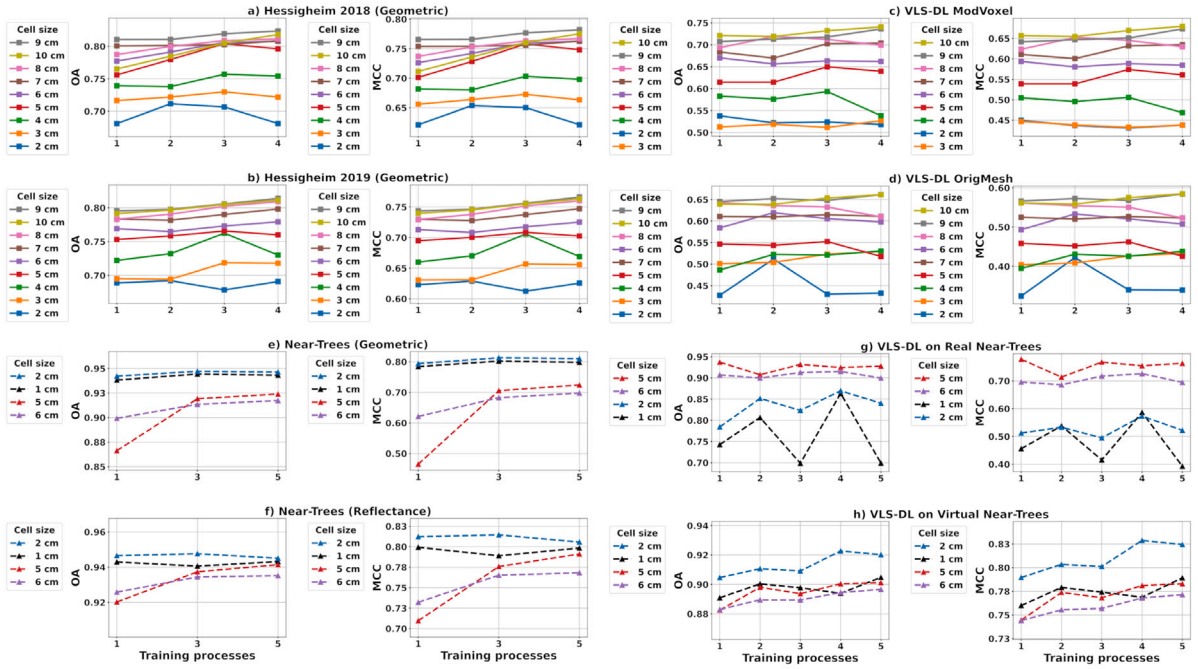

Fig. 3. Comparison of many models with different initial cell size configurations leading to different receptive fields. For each model, the overall accuracy (OA) and the Matthews Correlation Coefficient (MCC) are plotted on the *y*-axis while the *x*-axis represents the number of training processes. Each training process picks a new sample of neighborhood centers. Solid line graphs refer to the Hessigheim datasets, while dashed line graphs refer to the leaf-wood dataset. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

be that these classes highly benefit from reflectance and RGB information, which was not included in our VLS-DL models. Moreover, some of these classes are also underrepresented and have more heterogeneous geometries than others. With 29% of the points of the class "object" correctly classified with the VLS-DL model, compared to 44% with the geometric KPConv model (trained on real data), the results indicate a suboptimal 3D representation of objects (especially vehicles and urban furniture). For urban contexts, the VLS scene model can be significantly improved to fix overly irregular surfaces, missing triangles, and non-smooth orientation changes on some surfaces.

A top view of the classified validation point clouds is shown in Fig. 7. Visual inspection reveals some coarse grain issues (e.g., the problematic roof at the top), which are similar between real and virtual models. However, the misclassification of small urban objects is a bigger problem for the VLS-DL model (Fig. 7a). In a post-processing evaluation, all models perform well when grouping the semantic classes into aggregated categories (Fig. 7b). The conclusions from the visual inspection agree with those from the confusion matrices.

We conducted an additional experiment that analyzes the domain adaptation capabilities of the VLS-DL model from unoccupied laser scanning (ULS) datasets (Hessigheim3D) to TLS datasets (Semantic3D). The main result from this experiment is that both real and VLS-DL achieve high scores for domain adaptation, with average OAs of 90.26% (real) and 88.53% (VLS-DL). Further details can be found in Appendix B.

4.3. Leaf-wood results

Table 7 shows that the results of our FVLS-DL and VLS-DL models are quantitatively within the state-of-the-art (SOTA) interval for leaf-wood segmentation. The FVLS-DL model is 3% lower in OA than the top SOTA model (Han and Sánchez-Azofeifa, 2022) whether considering isolated or near trees, while the VLS-DL model is only 2% below the SOTA considering the near trees experiments. Thus, the performance of both virtual models is near the state-of-the-art when properly tuned.

A general visual impression of point-wise classification on near trees is given in Fig. 8 together with the corresponding confusion matrices. All the models give a good approximation of the label reference. The model trained with reflectance gives the best results and shows the least confusion. However, the differences are small. More concretely, the reflectance-based model correctly classifies 3% more wood points than the virtual model and just 1% more leaf points. The real geometric KPConv and the VLS-DL model differ when studying their confusion. More concretely, the geometric KPConv misclassifies more wood points than the VLS-DL model, but the latter misclassifies more leaf points.

Finally, Table 8 represents the quantitative assessment of the FVLS-DL, VLS-DL, and real (geometric only) models on different datasets. In addition to the point clouds created from Weiser et al. (2023) (results reported in Tables 4 and 7), the classification performance of the models (Real, VLS-DL and FVLS-DL) was assessed with point clouds from Wang et al. (2021), for both the isolated and the near trees case, and for point clouds used in Xi et al. (2020) and Hopkinson (2020) for just the near trees case (Appendix B). The results reveal that all models generalize well when tested on different datasets. More specifically, the FVLS-DL model generalizes as well as the real model to isolated trees with around 93% OA for Weiser and 95% OA for Wang, while the VLS-DL model generalizes to near trees with just 1% less OA than the real model for Weiser and with a negligible difference of 0.4% OA for Wang. Concerning the Hopkinson point cloud, the virtual models have around 4% higher OA than the real model. This result can be explained by the fact that the real model learns more fine-grain details than the virtual ones, and these details are not present in the Hopkinson point cloud, which is less dense and has some gaps in the wood areas compared to the others. These results suggest that using VLS to generate targeted training data can lead to better classifications by providing training point clouds that mimic the particular characteristics of the target dataset.

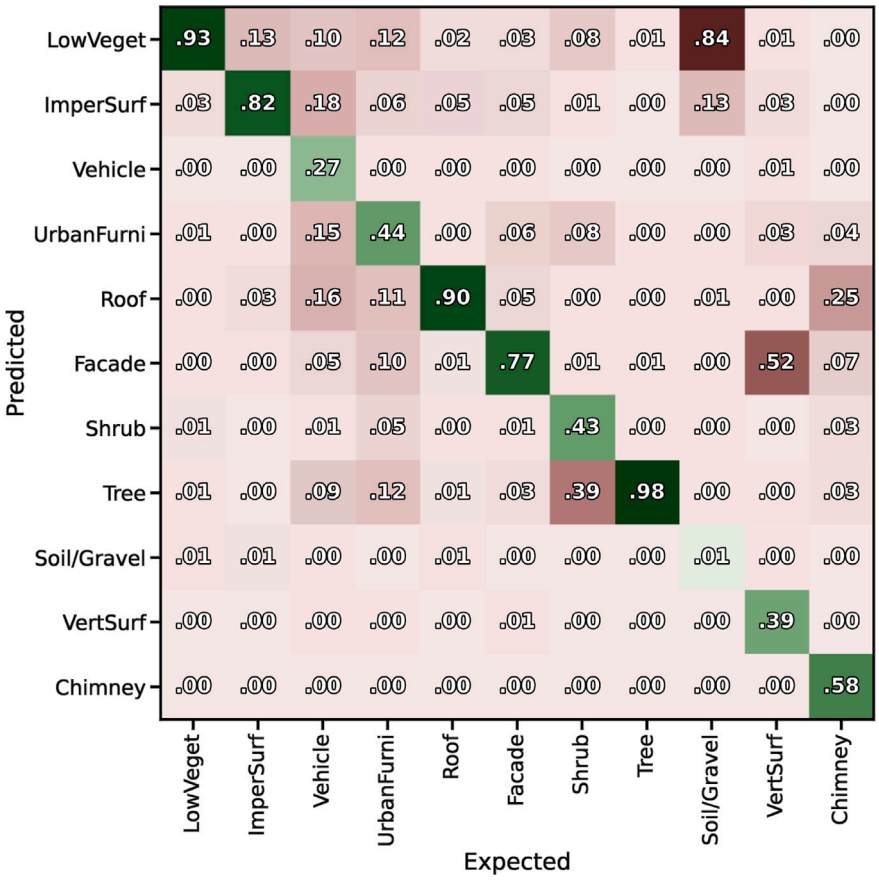

Fig. 4. Confusion matrix normalized by expected class for the geometric KPConv model trained on real point clouds from the Hessigheim dataset.

Table 7
Comparison between VLS-DL and selected machine learning models from the literature using distinct approaches for the leaf-wood separation problem. The FVLS-DL model is the VLS-DL model trained on fully synthetic trees. The overall accuracy is the aggregated evaluation metric, and the F1 score for leaf and wood is the per-class evaluation metric. The different metrics come from the different datasets used in the corresponding publications. We evaluated our models on the real datasets built from a subset of the Weiser et al. (2023) dataset (Appendix B). The Mesh-DL model is used as a baseline solution to quantify to which extent VLS improves the results of a deep learning model compared to feeding the points from the scene's meshes directly to the neural network.

Authors	Input	Model	OA	F1 (leaf)	F1 (wood)
Krishna Moorthy et al. (2020)	Multiscale geometric features	Random Forest	94%	97%	81%
		XGBoost	94%	97%	81%
		LightGBM	94%	96%	80%
Vicari et al. (2019)	Geometric features	Unsupervised	89%	92%	73%
Han and Sánchez-Azofeifa (2022)	Geometric features (deep learning)	FCN	92%	92%	92%
		LSTM-FCN	96%	96%	96%
		ResNet	96%	96%	96%
Ours (validated on isolated trees)	FVLS point cloud	FVLS-DL	93%	96%	84%
	VLS point cloud	VLS-DL	87%	91%	74%
Ours (validated on near trees)	FVLS point cloud	FVLS-DL	93%	96%	80%
	VLS point cloud	VLS-DL	94%	96%	81%
	Points from mesh	Mesh-DL	52%	60%	39%

4.4. Mesh sampling, VLS, and real-world point cloud comparison

To determine the realism of the physically-based laser scanning simulations, we investigate the difference in geometric features between the real leaf-wood point clouds, VLS point clouds of the Wytham Woods scene, and point clouds generated by randomly sampling points on the Wytham Woods mesh. Fig. 9 summarizes the main findings using 2D histograms to show the density of feature values for different heights above ground.

First, looking at Fig. 9 (a), (c), and (e), note that the distribution shown in the two-dimensional histograms characterizing the training point clouds is significantly different when comparing real or virtual point clouds with mesh sampling. Visually, the distribution of planarity and linearity is more dispersed for mesh sampling, while it has some relatively high point concentrations for virtual and real point clouds. While the differences between the real and virtual training datasets can be explained by the fact that they represent different scenes (mixed forests in Baden-Württemberg, Germany, and the Wytham Woods research forest in Oxfordshire, United Kingdom, respectively), the VLS and mesh sampling cases come from the same scene. Furthermore, the white line representing a linear estimator fitting the feature as a function of height (z coordinate) has very similar slopes and values for virtual and real point clouds, while it is notably different for the points generated by mesh sampling.

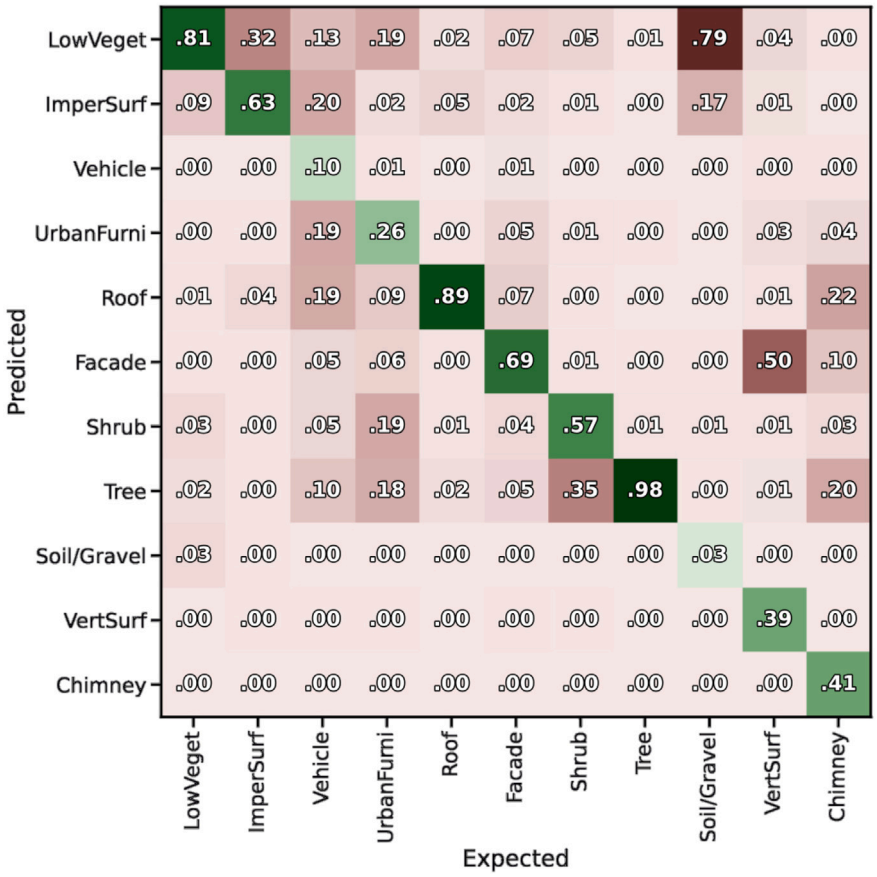

Fig. 5. Confusion matrix normalized by expected class for the VLS-DL model applied to the Hessigheim dataset.

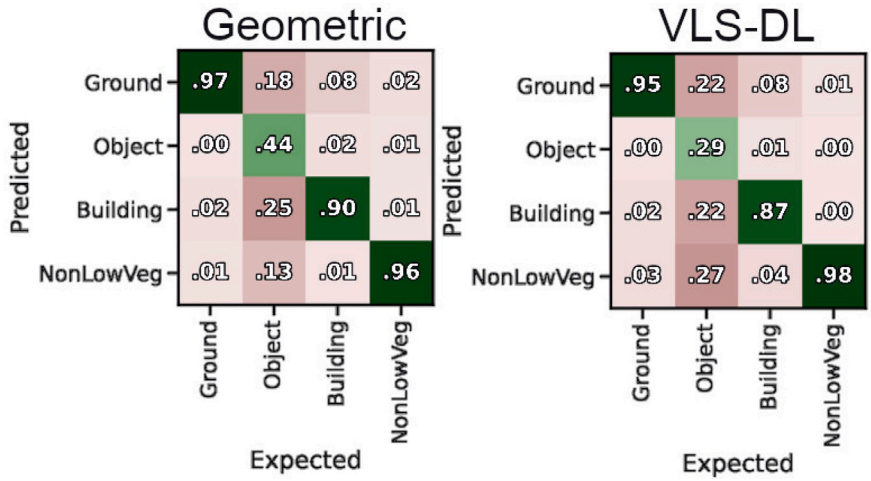

Fig. 6. Confusion matrices normalized by expected class with aggregated categories for the geometric KPConv and VLS-DL models on the Hessigheim dataset.

Second, the misclassification of geometric KPConv models is shown with respect to the geometric features in the 2D histograms of incorrectly labeled points in Fig. 9 (b), (d), and (f). The distributions for virtual and real point clouds are similar, especially for linearity and planarity. The distributions of the mesh sampling case have more obvious misclassification concentrations. The conclusions also hold for the linear least-squares fit. These findings match the quantitative evaluation of the deep learning models shown in Table 7. In terms of performance, the Mesh-DL model is 42% lower in OA than the VLS-DL model. Moreover, the Mesh-DL model is 36% worse at classifying leaf points than the VLS-DL model, while it is 42% worse at classifying wood points, when measured in terms of F1 score.

4.5. Ablation study on different models

The results of our ablation study to investigate the impact of the amount of simulated data and the performance of different models are summarized in Fig. 10. We can clearly see that all models yield worse results when using only 16% of the available training data. All the FVLS-DL models achieve the best result when using the full simulation

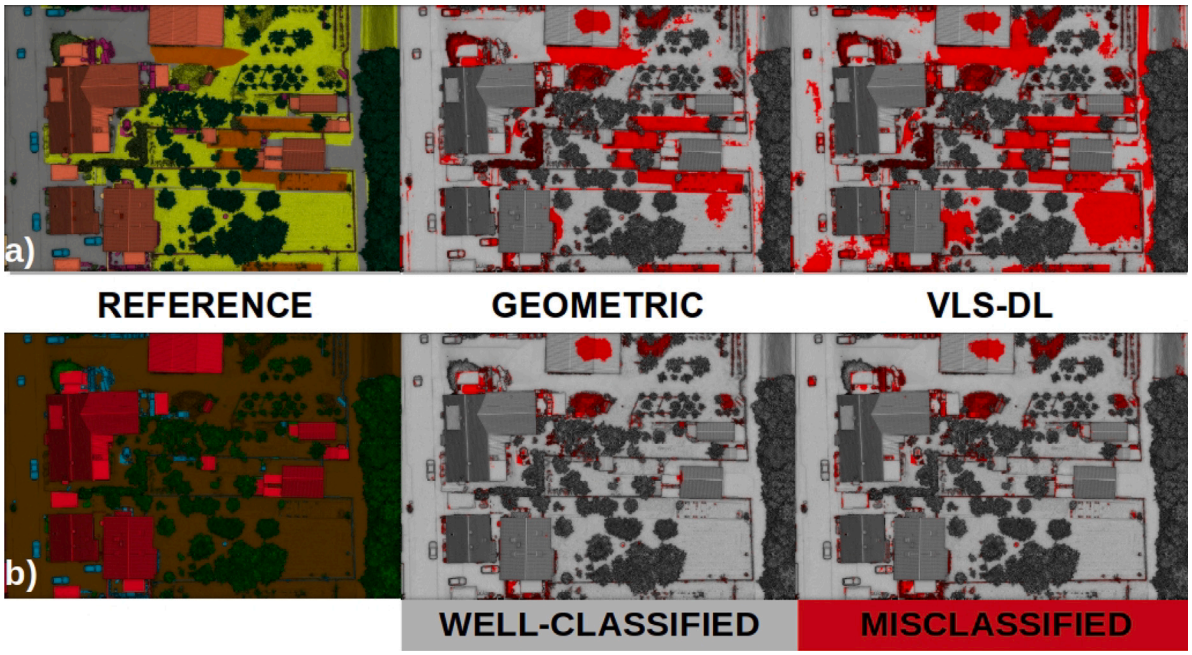

Fig. 7. Top view of the reference labels and classification confusion on the validation point cloud. The top row (a) shows the original eleven classes and the confusion from the Hessigheim3D benchmark (Kölle et al., 2021). The bottom row (b) represents our reduced classes: ground, building, vegetation, and object. The gray color represents successfully classified points, while the red represents misclassified points. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

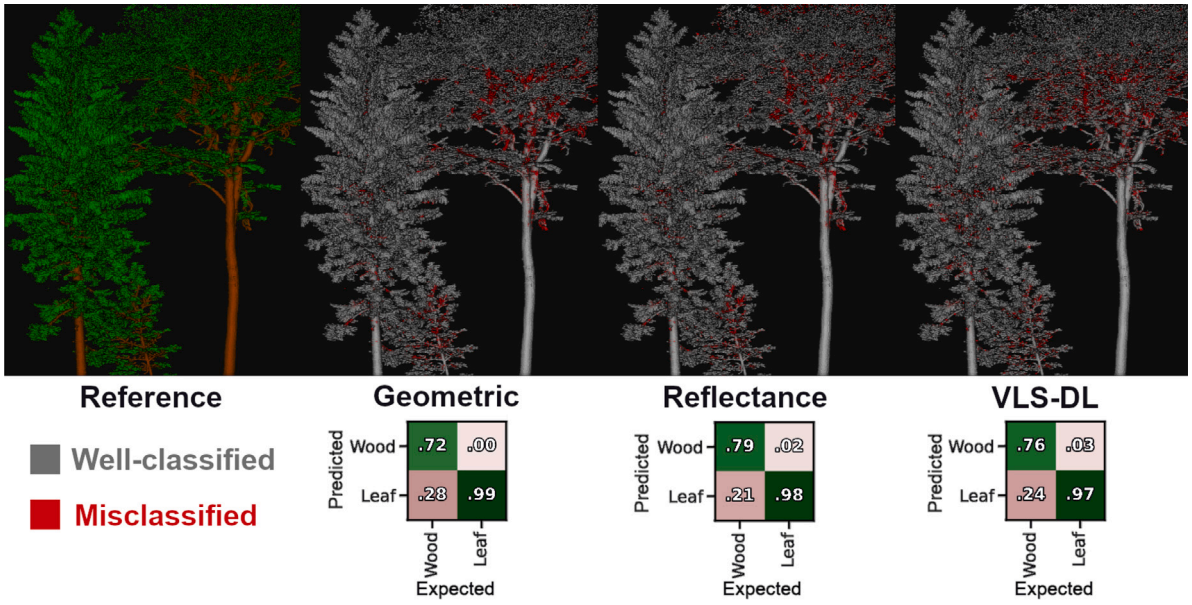

Fig. 8. Visualization of leaf-wood segmentation results on real-world tree point clouds (near trees) for the manual labeling (reference), the geometric KPConv model, the reflectance KPConv model, and the VLS-DL model. Points labeled as leaves are colored green, and points labeled as wood are colored brown. The gray points represent successful classifications, while the red ones are misclassifications. The given confusion matrices are normalized by expected class. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

dataset. For the real models, there is a small exception with KPConv that achieves slightly better results (1.1% better) when using 82% of the training data, probably due to a slight overfit to the fine-grain details of the training dataset when using all the data. The standard deviations are very small for the Random Forest models, which can be explained because they are an ensemble of many decision trees, making them robust to noise. Also, Random Forest is a classical machine learning model that is not as data-hungry as neural networks, so the difference between the worst and best IoU is smaller than for neural networks.

All the experiments have a linear tendency with a positive slope ($\alpha > 0$), implying that using more training data yields better results no matter the model. This claim is also verified by looking at the correlation coefficients because all of them are clearly positive, i.e., $r \to 1$. The most noticeable result is that the PointNet++ model trained on 82% of the data yields worse results than using 44% or 61% of the data. The

Table 8
Evaluation metrics for the generalization experiments with isolated and near trees, respectively. The real model was trained on real labeled point clouds from the dataset of Weiser et al. (2022), also used for our other experiments. The VLS-DL and FVLS-DL models were trained on simulated point clouds. The VLS-DL models used meshes derived from real data for simulation, while the FVLS-DL model used fully synthetic trees. The validation point clouds are taken from Weiser et al. (2021a), from Hopkinson (2020), and from Wang et al. (2021) and are named by the first authors of the data publications.

Validation	Real			VLS-DL			FVLS-DL		
	OA (%)	F1 (%)		OA (%)	F1 (%)		OA (%)	F1 (%)	
		Leaf	Wood		Leaf	Wood		Leaf	Wood
Isolated Weiser	93.5	95.9	85.2	87.0	91.3	74.1	93.2	95.6	84.4
Isolated Wang	95.2	96.5	92.5	87.1	89.8	82.5	95.4	96.5	93.0
Near Weiser	94.7	96.9	83.1	93.7	96.2	81.3	92.6	95.5	79.7
Near Wang	94.7	96.4	90.2	95.1	96.6	91.3	93.8	88.7	96.4
Near Hopkinson	90.1	81.6	93.3	94.5	96.3	88.7	94.5	95.6	89.2

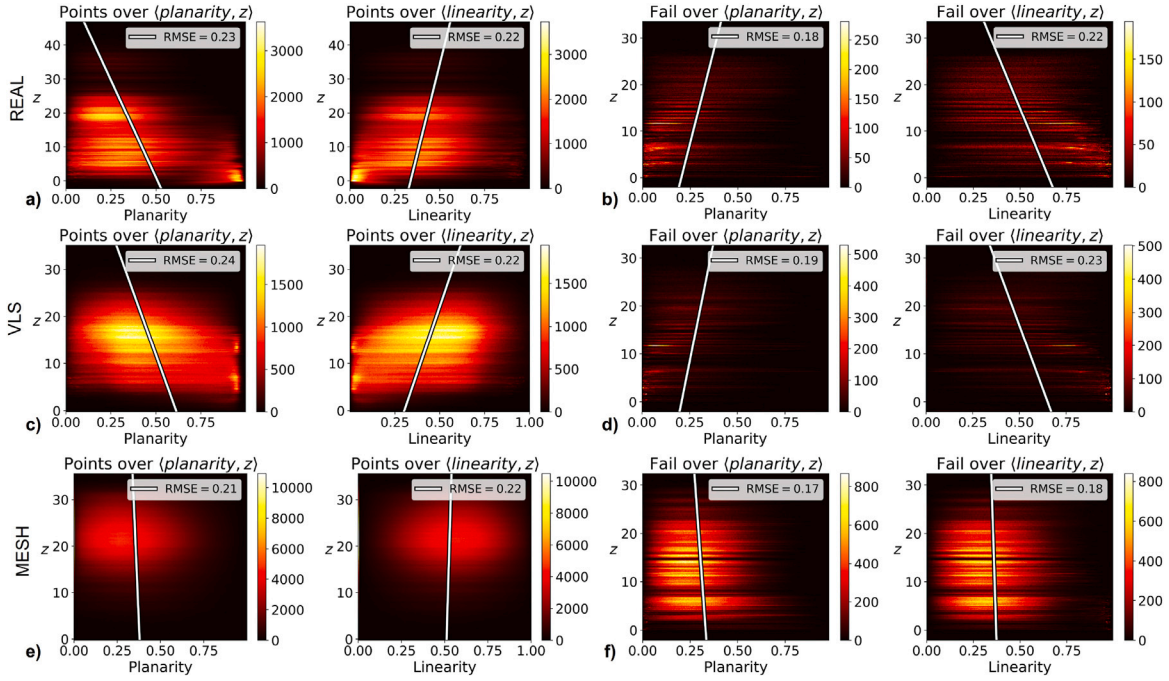

Fig. 9. Distribution of geometric features along the z axis for mesh sampling, VLS, and real point clouds. The top row shows (a) the feature distribution in the real point cloud and (b) the feature distribution of misclassified real points for a model trained on real data. The middle row shows (c) the feature distribution in the VLS point cloud and (d) the feature distribution of misclassified real points for a model trained on VLS data. The bottom row shows (e) the feature distribution in the mesh sampling point cloud and (f) the feature distribution of misclassified real points for a model trained from the mesh sampling point cloud. The white line is the best linear fit of the feature as a function of the height (z).

standard deviation for this case is quite high too (14.3%). However, even for this case, we have a positive slope and correlation coefficient.

5. Discussion

The results shown in this paper prove that VLS can be used to train DL models without the need for real labeled reference data. These VLS-based models then generalize to real point clouds. Thus, VLS-DL can reduce the time and cost associated with industrial and research projects involving machine learning-based classification of point clouds. The degree to which costs can be reduced depends on the complexity of the 3D scene modeling task. Simple scenes will significantly reduce time and cost, while complex scenes involving many object types will yield a less significant reduction due to the more sophisticated 3D modeling required.

The VLS-DL model has been tested for the two broader domains in the point cloud processing community: urban and natural contexts. Each study case requires identifying and improving the key components related to these categories.

5.1. VLS-DL applied to urban contexts

From the results of the urban experiments, we understand that the 3D input scene is a critical component for VLS simulations and that it is affecting the specific classification task. Scene representation problems, such as oversimplification, lead to lower performance as the realism of the virtual point clouds is reduced. From the results in Table 3, we can see that a voxelized representation of vegetation reduced the OA gap between real and virtual training data by 8% compared to using the original mesh-based scene model for VLS. Further improvements are expected to close this gap even more.

First, we propose improving VLS with more accurate surface roughness simulation to improve differentiation between natural (e.g., low vegetation) and artificial (e.g., impervious surface) ground. Second, we propose two improvements for urban object differentiation: (1) the simulation of more scene parts with under-represented objects (e.g., applying VLS many times with different scene rotations and slight scale changes), and (2) a more accurate representation of vehicles and urban furniture such as fences and streetlights. Lack of detail problems were also mentioned by Wu et al. (2018) when using the default GTA-V

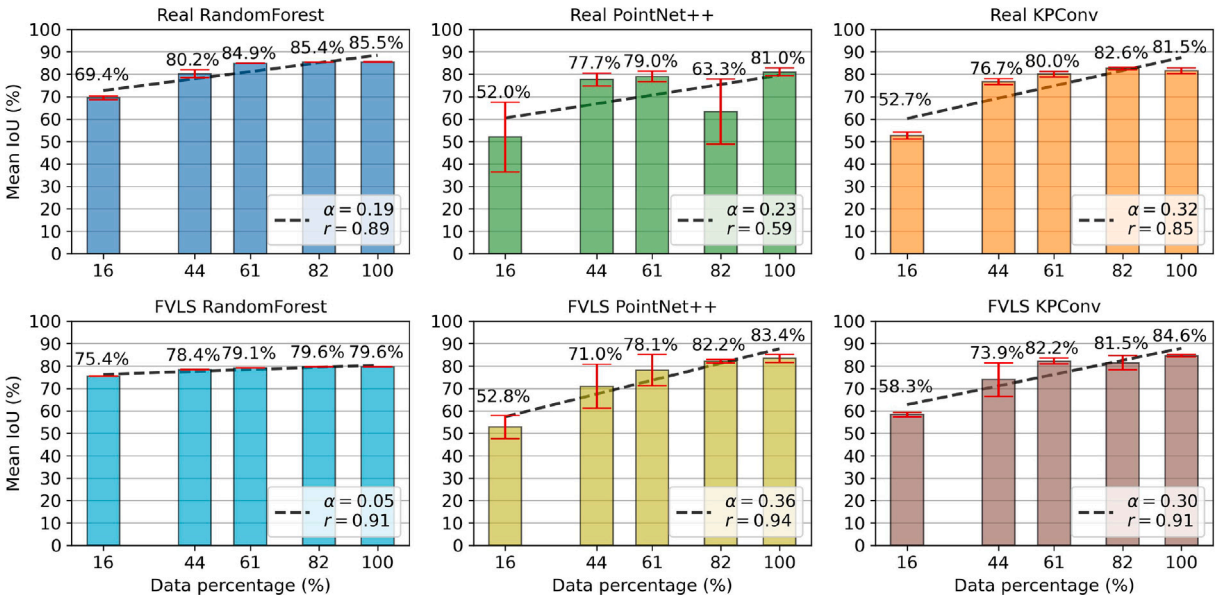

Fig. 10. The results of the ablation experiments on the amount of training data for different models using the VL3D framework. The percentages correspond to the amount of annotated points in the training dataset with a maximum of 35,149,732 points for the real dataset and 53,671,632 for the simulated one, with a fixed number of 86,804,388 real points for the validation dataset. Each represented data percentage is the mean between real and simulated data percentages, which, on average, differ around 3.88%. The top row shows the mean IoU for the models trained on real data from Weiser et al. (2022), and the bottom row for the models trained on fully virtual laser scanning (FVLS) data. All the models are validated on real data from Wang et al. (2021). Each column bar represents the mean IoU calculated by training and validating the model five times. The red caps represent the standard deviation of the five repetitions on the same subset of data. The black dashed line represents the linear tendency. Also, α is the slope of the line, and r is the Pearson correlation coefficient. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

physics, leading to simplified representations where pedestrians were treated as cylinders.

Furthermore, some problems must be addressed from the DL side (e.g., using different neighborhood definitions). For example, differentiating building facades and vertical surfaces is more related to global context than local neighborhood analysis since both are planar surfaces with similar orientation.

5.2. VLS-DL applied to leaf-wood segmentation

The FVLS-DL and VLS-DL models achieve near state-of-the art results for the leaf-wood case (Table 7) and achieve satisfactory virtual-to-real generalization on point clouds from different datasets (Table 8). It is expected that training with VLS data from trees of many different growth forms, leaf shapes and sizes will help to improve the results. Thus, creating large and diverse training datasets is an advantage of VLS over real data, for which massive datasets are often unavailable due to the high effort of point cloud acquisition and annotation. Taking advantage of fully virtual data (FVLS-DL) improved the leaf-wood segmentation of isolated trees by 6.2% OA compared to scene meshes derived from real point clouds (VLS-DL), as shown in Table 4. This may be due to the higher diversity of the trees in the FVLS scene, which also includes conifers, compared to the VLS scene, which has only deciduous trees (mostly sycamore maple, and some ashes and oaks). Furthermore, some applications lack even small labeled datasets due to the complexity of manually annotating real point clouds. In these cases, VLS is a cost-effective approach to obtaining labeled training data for deep learning models.

5.3. VLS-DL versus mesh sampling

The distribution analysis of geometric features (Fig. 9) represents an empirical validation of the theoretical VLS-DL model, as VLS provides a better representation space for feature extraction than sampling points on meshes. More formally, we show that the interaction of the parametric model described in Section 3.1, the feasible regions described in Eq. (1), and the corresponding physical and convenience constraints leading to the set of points P described in Eq. (2) give rise to models that generalize better to real data than those trained using points randomly sampled from the 3D meshes. This clearly underlines the importance of laser scanning simulation and the creation of realistic geographic point clouds.

5.4. VLS-DL hyperparameter tuning

Models trained solely with geometric data have easy-to-understand and easy-to-tune hyperparameters, as illustrated in the aggregated model comparisons of Fig. 3. Optimizing the receptive field can be addressed with a simple linear search governed by the cell size of the smallest grid subsampling, yet leading to a significant improvement of the model. The validity of this method holds for both urban and tree point clouds. Moreover, neighborhood topology can be altered to cover a wider area when a more global context is required. This is why de Gélis et al. (2023) used a cylinder-like neighborhood to improve their Siamese KPConv model. These straightforward modifications have a drastic impact on the performance of the DL model.

Recall that the input matrix $\boldsymbol{P}$ from Eq. (3) can be expressed as a function of continuous variables governing the virtual data. Some variables might be the components of the vector $\boldsymbol{\theta}$ defining the simulation constraints in Eq. (2). Others might be the variables governing the parametric ray generation model or transformations and constraints on the feasible regions defining the scene. Consequently, it must be possible to optimize the VLS model parameters too.

We argue that the feature extraction operator, especially when considering classes separable from geometric information, can link the VLS and the DL models. Therefore, fitting the simulator to maximize the class separability of the feature extraction operator inside the VLS model should improve the performance of a DL model using the same operator. To do this, first, a realistic simulation is needed as a baseline. Then, the realistic simulation could be optimized to maximize the adequacy of the VLS point cloud as training data for a particular task. Loosing too much realism will break the relationship between virtual training data and real validation data, so the optimization must be constrained to stick to slight transformations.

It is possible to quantify and visualize the differences between virtual and real point clouds using continuous metrics derived from the geometry of a neighborhood, as shown in Fig. 9. These measurements are similar to the classification error distribution. Thus, tuning the VLS model to improve the class-separability of the extracted features while keeping them realistic should improve the DL model's performance.

5.5. VLS-DL cost and automation

If an appropriate model of a labeled scene is available, VLS generates perfectly labeled training data for DL without costly equipment other than a computer. The performance of a classifier trained with VLS data can be improved in different ways. On the one hand, improving the ray-generation model can be done by looking at manufacturer specifications, often offered as open-access documents at no cost. Also, some open-access simulators like HELIOS++ provide ways to automatically derive an accurate parametric platform model through interpolation from raw trajectory files that can be simulated or reused at no extra cost. On the other hand, scene modeling is potentially the most expensive cost for the VLS-DL model, requiring more time and often human involvement. Nevertheless, it is also crucial to simulate a proper training dataset, and poor scenes lead to poor point clouds, which have little to no benefit in training DL models. More particularly, when complex scene modeling relies on hand-crafted meshes, buying high-quality meshes or commercial software or hiring a 3D modeling professional will significantly increase the cost. However, this increased cost can be amortized for those cases where the same scene can be used to generate different point clouds.

Fortunately, there are several ways in which the scene modeling problem can be tackled. Sometimes, high-quality meshes can be automatically derived from real data. For example, using high-resolution and accurate LiDAR or photogrammetry sensors to obtain reliable input data for meshing algorithms. Another alternative is to use procedurally generated meshes. The first benefit of these approaches is that they keep the cost of VLS-DL low. More importantly, they also open up the gate to fully automatic workflows. Specifically, when the scene representing the object for study can be procedurally generated, the whole VLS-DL model can be fully automated because all the other parts of the workflow are already available as interaction-free tools. In general, we believe that the benefits of VLS-DL outweigh the challenges identified.

6. Future challenges

6.1. VLS-DL for regression

Machine learning problems can be divided into two broad categories: classification and regression. The VLS-DL model has proven good enough to solve classification problems on real point clouds. Now, regression problems are a relevant next milestone. Exploring regression on point clouds will allow us to study how the VLS-DL model performs when it must compute continuous values on a point cloud instead of assigning a discrete class value for each point. For instance, the output of the leaf-wood segmentation model can be used to train a regression model on the wood points to compute biomass estimations. Also, the leaf points can be used to estimate the leaf area index, an important ecological parameter that can be used to estimate the energy flow in the leaves and the expected productivity of a tree.

6.2. Tuning virtual laser scanning

In this work, we showed that the VLS-DL model could be seen as a combined model such that both the simulator and the neural network can be tuned to improve the overall model performance. We also showed that changes in VLS imply changes in DL performance. These changes were characterized using continuous geometric features and linear fitting. Future efforts to improve the VLS-DL model should aim at integrating the feature extractor operator into the simulator to perform fine-grain tuning to maximize the class separability of the features or to minimize the difference to a reference point cloud. If the previously described integration is achieved, starting the optimization algorithm directly in the VLS model should work as long as (1) the VLS optimization starts from a realistic simulation and (2) the VLS optimization is constrained to avoid drastic transformations that deviate too far from the realistic baseline.

Furthermore, when procedurally generated 3D scenes are realistic enough, the VLS tuning can be fully automated. Exploring procedural generation algorithms for 3D scenes is relevant to future research towards fully automatic VLS-DL. In this case, hyperparameter tuning could be fully automated through random and grid search strategies, genetic algorithms, or particle swarm optimization, as usually done in automated machine learning (Das and Cakmak, 2018). In this work, we explored the combination of VLS-DL with procedurally generated trees with successful results. Other works have explored artificial intelligence models like Generative Adversarial Networks (GANs) for point cloud processing in the context of robotics and autonomous driving (Goodfellow et al., 2014; Caccia et al., 2019; Triess et al., 2022).

6.3. Dynamic virtual laser scanning

Until now, VLS has typically been computed assuming a static scene. However, laser scanning is often used in real-world contexts with changing conditions (e.g., moving vehicles and pedestrians or moving trees with leaf flutter and branch buffeting due to wind). The VLS-DL model should also be made compatible with dynamic VLS, i.e., with simulations in which the scene changes over time. By extending VLS to support dynamic scenes, the VLS-DL model can be trained on point distributions explained by motion, including certain scanning artifacts and motion-caused occlusions. For example, the open-source software HELIOS++ has recently included support for dynamic VLS, which opens up the gate for future research in this direction.

7. Conclusions

In this paper, we showed that deep learning models trained solely with virtual laser scanning (VLS) data can be used to semantically segment real point clouds with high accuracy. Training with real data leads to an overall accuracy (OA) 1% higher than training with virtual data in our leaf-wood experiments and 7% to 8.3% higher in our urban experiments. We almost closed the gap between VLS and reality for the leaf-wood segmentation problem, achieving near state-of-the-art results with full VLS-based training. For semantic segmentation in urban contexts, we started to close the gap between VLS and reality, reducing the difference in OA from 16.3% and 17.5% to 8.3% and 7% for the 2018 and 2019 point clouds, respectively, by improving the 3D scene with voxel-based vegetation modeling.

Also, we show that fully automatic scene generation and model training are possible with our FVLS-DL model, which is based on a fully computer-generated virtual scene. Further experiments comparing the FVLS-DL approach against real data for different models (Random Forest, PointNet++, and KPConv) show that the proposed method works for different neural networks and classical machine learning models. The ablation study used to compare these models confirmed that simulating more data leads to better model performance in terms of intersection over union for all investigated models.

The theoretical description of VLS-DL and the empirical validation from our experiments suggest that a deeper integration of both models is possible. Moreover, while our results are satisfactory, improving them requires human-based manual work to tune the hyperparameters and design realistic 3D scenes. While this work is not so laborious and prone to errors as manual labeling, better integration of VLS-DL could alleviate this burden by automatizing fine-grain tuning and bringing

optimization from DL to VLS. Furthermore, if procedurally generated scenes can be computed with sufficient realism, the entire workflow can be optimized automatically.

We consider there is enough evidence to claim that VLS is a convenient solution for training a wide variety of point cloud classifiers based on supervised training. Furthermore, the time and cost-saving potential of VLS-DL makes it a viable option for point cloud research and industrial applications. Typical problems, such as insufficient or imbalanced data, can be addressed using VLS-generated data. Thus, taking advantage of virtual laser scanning can revolutionize deep learning applied to point clouds.

CRediT authorship contribution statement

Alberto M. Esmorís: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing, Data curation. **Hannah Weiser:** Conceptualization, Data curation, Investigation, Validation, Visualization, Writing – review & editing, Methodology, Software, Writing – original draft. **Lukas Winiwarter:** Conceptualization, Funding acquisition, Validation, Writing – review & editing, Investigation, Software. **Jose C. Cabaleiro:** Formal analysis, Funding acquisition, Investigation, Supervision, Validation, Writing – review & editing, Conceptualization. **Bernhard Höfle:** Conceptualization, Funding acquisition, Investigation, Project administration, Supervision, Validation, Writing – review & editing, Methodology, Software.

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.

Acknowledgment

This research was funded by the Deutsche Forschungsgemeinschaft (DFG), German Research Foundation, by the projects SYSSIFOSS (Grant Number: 411263134) and VirtuaLearn3D (Grant Number: 496418931). It was furthermore supported by the BMBF (Federal Ministry for Education and Research, Germany) in the frame of the AImon5.0 project (Funding code: 02WDG1696).

This work has also received financial support from the Consellería de Cultura, Educación e Ordenación Universitaria (accreditation ED431C 2022/16 and accreditation ED431G-2019/04) and the European Regional Development Fund (ERDF), which acknowledges the CiTIUS-Research Center in Intelligent Technologies of the University of Santiago de Compostela as a Research Center of the Galician University System, and the Ministry of Economy and Competitiveness, Government of Spain (Grant Number PID2019-104834GB-I00 and PID2022-141623NB-I00).

The many deep learning experiments computed on the FinisTerrae-III supercomputer were possible thanks to the CESGA (Galician supercomputing center). Diverse experiments were also possible thanks to the data curators of the Hessigheim and Wytham Woods datasets and the manually labeled leaf-wood datasets.

The authors gratefully acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant INST 35/1597-1 FUGG. Furthermore, the authors acknowledge the data storage service SDS@hd supported by the Ministry of Science, Research and the Arts Baden-Württemberg (MWK) and the German Research Foundation (DFG) through grant INST 35/1503-1 FUGG.

Open Access funding enabled and organized by Projekt DEAL.

Appendix A. Virtual scene generation and virtual laser scanning

A.1. Urban scene classification

Creating the virtual scenes

The Hessigheim 3D (H3D) benchmark dataset (Kölle et al., 2021) is already split into training and validation. We used the training subsets of the meshes and the point clouds to create two versions of the 3D scene for our laser scanning simulations. For both versions, a modified material library (.MTL file) with an added `helios_classification` value was used. An example for one material is shown below:

```
newmtl Low Vegetation
Ka 0.698039 0.796078 0.184314
Kd 0.698039 0.796078 0.184314
Ks 0.698039 0.796078 0.184314
illum 2
Ns 136.430000
helios_classification 0
```

In this way, the simulated returns are automatically assigned the `helios_classification` value of the material of the surface which they hit.

Version 1 of our H3D virtual scene uses the original mesh tiles from the H3D training dataset. Version 2 of our H3D virtual scene uses a modified scene representation, where vegetation (classes "shrub" and "tree", as well as all faces labeled as "unlabeled" in the close vicinity of vegetation classes) have been removed from the mesh and instead modeled using voxels. For this, vegetation points were filtered from the H3D training point clouds and transformed into voxel models with 5 cm × 5 cm × 5 cm voxels using the HELIOS++ xyzloader.[4] After removing faces classified as vegetation from the mesh, the 3D model contains holes (below the now semi-transparent voxel vegetation representations). To fill these holes, we added a digital terrain model (DTM) to the scene, which we generated from the ground points of the H3D point cloud (classes low vegetation, impervious surface, soil/gravel). This DTM was shifted downwards by 0.2 m to ensure that it does not cover or intersect with other parts of the scene and assigned the material (i.e., classification) "low vegetation". Fig. A.11 compares the two versions.

Virtual laser scanning configuration

A trajectory of one of the Hessigheim ULS campaigns (columns: X, Y, Z, roll, pitch, yaw) was provided by colleagues at the Institute for Photogrammetry and Geoinformatics at the University of Stuttgart. It consists of three separate flights. The trajectory points were converted into a line feature and then simplified using the Douglas Peucker algorithm. The waypoints of the simplified lines were then used in the survey XML file for the X–Y "leg" positions. The heights of the waypoints were obtained from the original trajectory point file. For the 2019 surveys for training, some waypoints were moved further out along the Y-direction to ensure that the entire mesh is scanned.

Acquisition settings were selected to match the real data acquisition and are shown in Table A.9.

Resulting VLS point clouds for model training

Fig. A.12 shows the real and the two different VLS point clouds. The number of intermediate returns is several hundred times lower when using the original mesh in the simulations than when using the modified 3D scene with voxel vegetation. Unlike the mesh, the simulated laser beam can penetrate the voxelized canopies and thus generate multiple returns, making the simulation more realistic. This can also be seen visually in Fig. A.12. The number of intermediate

Fig. A.11. Comparison of (a) the original Hessigheim 3D mesh and (b) the modified Hessigheim 3D mesh with voxelized vegetation. Colored by classification. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

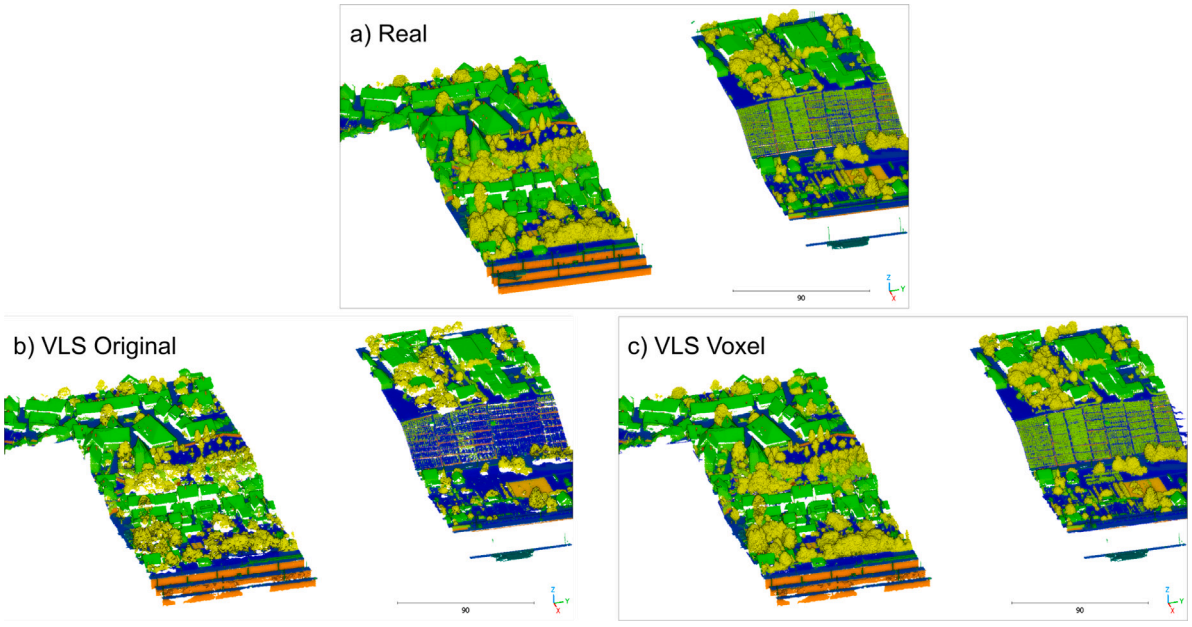

Fig. A.12. Images of (a) the training subset of real Hessigheim 3D point cloud, (b) the simulated point cloud using the original Hessigheim3D mesh, and (c) the simulated point cloud using the modified Hessigheim 3D scene with vegetation represented by voxels (all March 2019 epoch). Colored by classification. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table A.9
ULS scan settings used for the Hessigheim HELIOS++ simulations.

Setting	Value
Scanner	*RIEGL* VUX-1LR
Scan angle	±35° off nadir
Pulse frequency	820 kHz
Scan frequency	133 Hz
UAV speed	8 m/s

returns in the VLS point cloud with voxel vegetation is in the same order of magnitude as in the real point cloud.

Pulse density is lower in the VLS point clouds than in the real point clouds in both 2018 (VLS original: 720 pts/m^2, VLS voxel: 740 pts/m^2, real: 1050 pts/m^2) and 2019 epochs (VLS original: 820 pts/m^2, VLS voxel: 870 pts/m^2, real: 910 pts/m^2). This may be due to different reasons: (1) While the publication states 8 m/s as the speed of the UAV, the UAV does not move with constant speed in reality and might have been slower due to wind or due to the necessary deceleration and acceleration in the corners when turning. Due to a lack of the GPS Time attribute in the real point clouds, or an identifier for the flight strip, in-depth investigations of these differences were not feasible. (2) The mesh contains unlabeled faces. As we are not training on the "unlabeled" class, we removed it from the simulated point cloud, which lowers the number of points.

A.2. Leaf-wood classification

Creating the virtual scenes

With our experiments on leaf-wood classification using VLS training data, we cover two central ways of generating virtual scenes: (a) procedural 3D modeling with no real data at all, and (b) the reconstruction of a real scene from 3D measurements such as photogrammetric and/or laser scanning point clouds.

The fully synthetic scenes were assembled using procedurally generated 3D tree models of trees. These were generated using the algorithm of Weber and Penn (1995), implemented in the add-on "Sapling Tree

[4] https://github.com/3dgeo-heidelberg/helios/wiki/Scene#xyz-point-cloud-loader (Accessed on 2 October 2023).

Table A.10
TLS scan settings used for the HELIOS++ simulations for the leaf-wood experiments. For the fully virtual trees (FVLS) and the isolated Wytham Woods trees (VLS isolated), a higher resolution was used than for the full Wytham Woods forest stand (VLS near).

Setting	Value	
Scanner	*RIEGL* VZ-400	
Vertical field of view	−40°– 60°	
Pulse frequency	300 kHz	
Effective measurement rate	122 kHz	
	FVLS, VLS isolated	VLS near (Wytham Woods forest)
Horizontal resolution	0.017°	0.04°
Vertical resolution	0.017°	0.04°
Vertical point spacing (10 m range)	3 mm	7 mm
Horizontal point spacing (10 m range)	3 mm	7 mm

Gen"[5] in the open-source 3D modeling software Blender (Blender Online Community, 2023). 10 different trees were created by modifying the various parameters. Three were conifers with needles and seven were broadleaf trees with different leaf shapes. For each tree, two more trees were created by changing the random seed, resulting in a total of 30 trees. The trees were arranged in a small forest stand, with their crowns clearly overlapping (Fig. A.13a).

For the second option, scenes reconstructed from real data, we used the Wytham Woods 3D model, which is openly available[6] (Calders et al., 2018; Liu et al., 2022). We used the positions and tree mesh models (.OBJ files) in the `DART_models/3D-explicit model` folder in the branch `add_dart`.

It is important to note that due to the conversion of the cylindrical quantitative structure models (QSMs) to triangular meshes, the trunks and branches of the tree models are angular rather than round. Leaves are modeled as flat elongated hexagons.

We have modified the material library (.MTL) file to add the `helios_classification`, 0 to the material "TrunkAndBranches", and 1 to the material "Leaves". The trees were split spatially into training and test by manually extracting a quarter of the area to be used for testing.

Besides the full Wytham Woods forest scene ("near trees"), we created six scenes, in which eight trees were randomly drawn from the Wytham Woods dataset (without replacement) and assembled into a common scene with plenty of space between them (Fig. A.13c). We call this version "isolated trees" because the crowns of the trees do not overlap and no input neighborhood of one tree contains any points of another tree.

Virtual laser scanning configuration

The acquisition settings for all three leaf-wood experiments are summarized in Table A.10. The synthetic forest stand of fully computer-generated trees was scanned from six scan positions using a virtual terrestrial laser scanner of the model *RIEGL* VZ-400. The scan positions were regularly distributed on a 60 m radius circle around the trees and were oriented toward the forest plot center, scanning a horizontal field of view (FOV) of 90°. The Wytham Woods scene of the full forest stand was virtually scanned from 15 scan positions. These were manually distributed on the boundaries and inside the forest plot. The horizontal FOVs were defined based on the positions, so that a full 360° scan is performed for the scan positions within the forest plot and smaller FOVs were used for positions at the boundaries. The simulated training point cloud is shown in Fig. A.13b. Simulations were carried out the same way for the smaller validation scene but using only four scan positions. The scenes with isolated Wytham Woods trees were virtually scanned from six positions, evenly spaced on a circle of 35 m radius around the trees. Each scan had a horizontal FOV of 90°.

[5] https://docs.blender.org/manual/en/latest/addons/add_curve/sapling.html (Accessed on 11 August 2023).

[6] https://bitbucket.org/tree_research/wytham_woods_3d_model/src/add_dart/DART_models/ (Accessed on 19 October 2022).

Resulting VLS point clouds

Resulting simulated VLS point clouds for training the VLS-based leaf-wood classifiers are displayed in Fig. A.13. We refer to the point clouds of the procedurally modeled trees (Fig. A.13a) as "fully virtual laser scanning (FVLS)" point clouds and to the models trained with them as FVLS-DL models.

The point clouds of the Wytham Woods tree models (Fig. A.13b and c) are referred to as VLS point clouds (near trees and isolated trees, respectively) and the models are referred to as VLS-DL models.

Appendix B. Real point clouds for training and validation

Data for virtual-to-real generalization studies

The real urban classification models were trained on the real training point clouds of the Hessigheim 3D benchmark. All models for urban classification, real, VLS original and VLS voxel, were evaluated using the validation point clouds in the Hessigheim 3D benchmark.

For leaf-wood classification, two real models were trained, one on a point cloud of isolated trees and one on a point cloud with near trees (Fig. B.14) with eight labeled tree point clouds from Weiser et al. (2023). Each tree was individually normalized by subtracting the ground elevation at the location of the tree trunk and then re-positioned in a common point cloud in a local coordinate system.

The isolated trees training point cloud was composed of the tree point clouds with the IDs `AcePse_SP02_04`, `FagSyl_BR01_01`, `FagSyl_BR05_P8T4`, `PicAbi_BR02_14`, `PinSyl_KA10_03`, `PseMen_BR04_02`, `QuePet_BR01_01`, and `QueRub_KA09_T053` and the point clouds were spread out with a lot of space in between each tree.

The near trees training point cloud was composed of the tree point clouds with the IDs `AcePse_SP02_04`, `FagSyl_BR01_01`, `PicAbi_BR02_14`, `PinSyl_KA09_T048`, `PinSyl_KA10_03`, `PseMen_BR04_02`, `QuePet_BR01_01`, and `QueRub_KA11_09`. They were so closely spaced, that their crowns may touch or overlap (Fig. B.14a).

The main validation datasets for this study were generated from the remaining trees of the labeled tree point cloud dataset (Weiser et al., 2023), respectively. For the isolated trees, these were `PicAbi_BR08_01`, `PinSyl_KA09_T048`, and `QueRub_KA11_09`.

In addition, we used subsets of the datasets by Wang et al. (2021) and Hopkinson (2020) as real validation point clouds (Table 8). The "Isolated Wang" dataset was composed of the point clouds with the IDs 2, 11, 18, 68, 96, 97, and 102. The "Near Wang" dataset was composed of the point clouds with the IDs 4, 5, 12, 18, 31, 32, 57, 59, 60, 63, 73, 75, 83, 87, 88, 89, 92 93, 95, 96, 100, 101, and 104. Finally, the "Near Hopkinson" validation dataset consisted of the point clouds named `MDD04_012`, `MDD06_007`, `MDD07_010`, `MDD08_006`, and `MDD09_007`.

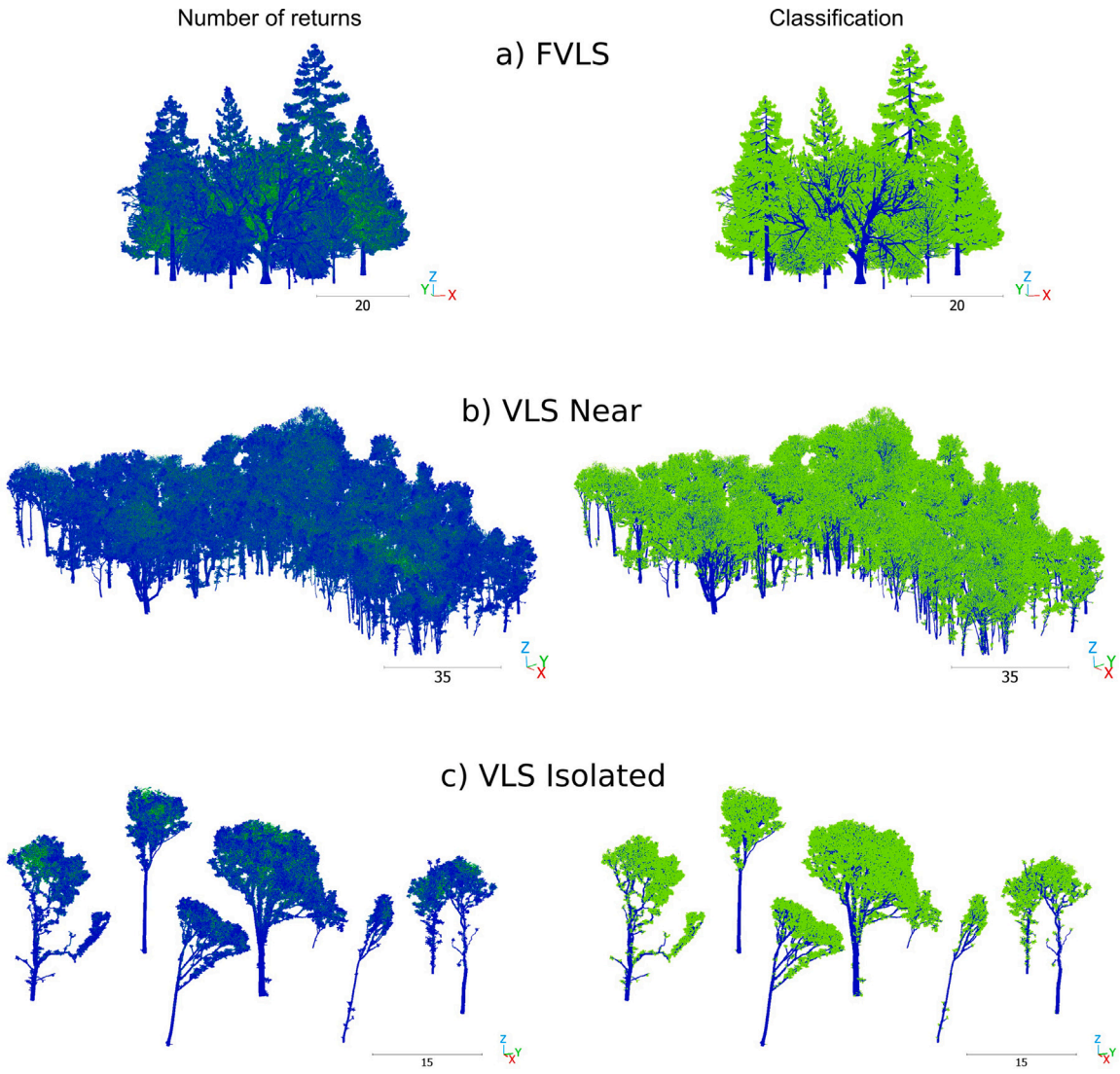

Fig. A.13. Simulated point clouds of (a) the fully virtual synthetic scenes with procedurally generated tree models, (b) the Wytham Woods scene, and (c) the isolated Wytham Woods scene. Colored by number of returns (left) and classification (right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

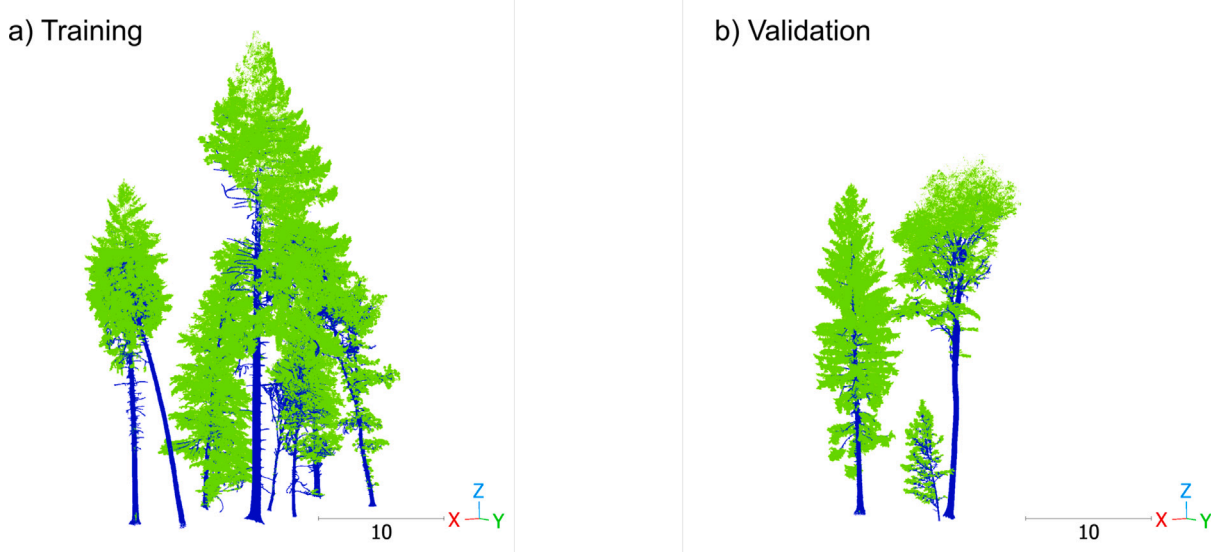

Fig. B.14. Real training and validation point clouds for leaf-wood separation with trees from Weiser et al. (2023). Colored by classification: green=leaf and blue=wood. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

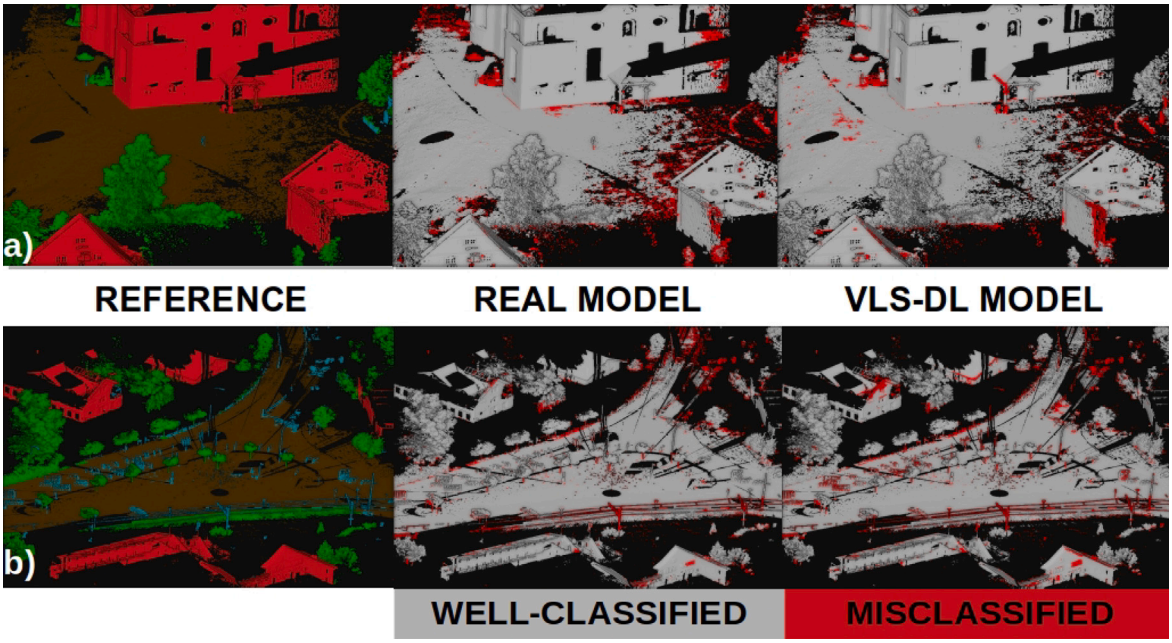

Fig. B.15. Visualization of the domain adaptation results, including the top view of the reference labels and classification confusion on the validation point clouds from the Semantic3D dataset (Hackel et al., 2017). The top row (a) shows the misclassified points for the Bildstein station 5 point cloud. The bottom row (b) corresponds to the SG27 station 5 point cloud. The classes are ground, building, vegetation, and object. The gray color represents successfully classified points, while the red represents misclassified points. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Domain adaptation experiment

The quantification of the domain adaptation capabilities of a deep learning model is a relevant question that arises in the context of representation learning. Domain adaptation refers to the performance of a model on a given task when applied to a different input distribution (Goodfellow et al., 2016b). Thus, we designed an experiment to explore how well the VLS-DL model performs regarding domain adaptation compared to a neural network trained on real data.

Our experiment consisted of training two different KPConv models, the first with real and the second with simulated point clouds from the Hessigheim dataset (Kölle et al., 2021). Then, we evaluated their performance on the 15 labeled point clouds of the TLS Semantic3D dataset (Hackel et al., 2017). Since the two datasets have different classes, we used the reduced classes of the Hessigheim dataset introduced in the main manuscript. Besides, we also reduced the classes from the Semantic3D dataset into terrain (man-made terrain and natural terrain), vegetation (high vegetation and low vegetation), buildings (buildings), and objects (hardscape, scanning artifacts, and cars). We argue that this experiment allows us to evaluate domain adaptation because ULS and TLS correspond to different input distributions.

Table B.11 shows the quantitative evaluation of the experiments with the two different KPConv models. We computed the OA and the MCC for each combination of DL model and validation point cloud. In doing so, we used the official training dataset for validation because we needed labeled data to quantify the results. Generally, the domain adaptation capabilities of the VLS-DL model match those of the real model, which is, on average, only 1.73% better in terms of OA. The point clouds in Fig. B.15 provide a visual representation of the results. The results of these experiments demonstrate the great potential of VLS-DL models to generalize to unseen real data, even when domain adaptation is required.

Table B.11
Evaluation metrics for the domain adaptation experiments. One model was trained on real point clouds (Real KPConv), and the other was trained with VLS point clouds (VLS-DL KPConv) corresponding to the ULS-based Hessigheim dataset (Kölle et al., 2021). The evaluations are calculated on the real TLS-based Semantic3D dataset (Hackel et al., 2017). For each KPConv model and each point cloud in the validation dataset, the overall accuracy (OA) and the Matthews correlation coefficient (MCC) are provided. The rows highlighted in bold correspond to the scenes with the best domain adaptation results.

Validation	Real KPConv		VLS-DL KPConv	
Point cloud	OA (%)	MCC (%)	OA (%)	MCC (%)
Bildstein station 1	89.0	83.5	87.7	81.2
Bildstein station 3	91.0	85.7	89.6	83.7
Bildstein station 5	**96.0**	**91.4**	**94.4**	**88.0**
Domfountain station 1	90.1	79.4	89.8	79.8
Domfountain station 2	83.5	71.5	83.3	70.1
Domfountain station 3	95.6	92.3	92.1	86.5
Untermaederbrunnen s1	93.8	90.3	89.6	84.3
Untermaederbrunnen s3	88.3	82.1	78.9	71.3
SG27 station 1	97.1	84.2	95.2	76.6
SG27 station 2	93.1	88.0	91.3	85.1
SG27 station 4	93.1	89.5	93.1	89.3
SG27 station 5	**97.3**	**93.9**	**96.5**	**91.9**
SG27 station 9	94.3	86.1	93.1	83.1
Neugasse station 1	80.4	69.2	71.1	70.2
SG28 station 4	71.3	60.4	82.2	72.8

References

Ao, Z., Wu, F., Hu, S., Sun, Y., Su, Y., Guo, Q., Xin, Q., 2022. Automatic segmentation of stem and leaf components and individual maize plants in field terrestrial LiDAR data using convolutional neural networks. Crop J. 10 (5), 1239–1250. http://dx.doi.org/10.1016/j.cj.2021.10.010.

Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S., 2016. 3D semantic parsing of large-scale indoor spaces. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 1534–1543. http://dx.doi.org/10.1109/CVPR.2016.170.

Blender Online Community, 2023. Blender - a 3D Modelling and Rendering Package. Blender Institute, Amsterdam, URL https://www.blender.org/.

Boni Vicari, M., Disney, M., Wilkes, P., Burt, A., Calders, K., 2018a. Leaf and Wood Classification Framework for Terrestrial LiDAR Point Clouds: Field Data Validation Dataset. Zenodo, http://dx.doi.org/10.5281/zenodo.1324156.

Boni Vicari, M., Disney, M., Woodgate, W., 2018b. Leaf and Wood Classification Framework for Terrestrial LiDAR Point Clouds: Simulated Data Validation Dataset. Zenodo, http://dx.doi.org/10.5281/zenodo.1324158.

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press.

Bryson, M., Wang, F., Allworth, J., 2023. Using synthetic tree data in deep learning-based tree segmentation using LiDAR point clouds. Remote Sens. 15 (9), http://dx.doi.org/10.3390/rs15092380.

Caccia, L., Hoof, H.v., Courville, A., Pineau, J., 2019. Deep generative modeling of LiDAR data. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS, pp. 5034–5040. http://dx.doi.org/10.1109/IROS40897.2019.8968535.

Calders, K., Origo, N., Burt, A., Disney, M., Nightingale, J., Raumonen, P., Åkerblom, M., Malhi, Y., Lewis, P., 2018. Realistic forest stand reconstruction from terrestrial LiDAR for radiative transfer modelling. Remote Sens. 10 (6), http://dx.doi.org/10.3390/rs10060933.

Cohen, J., 1960. A coefficient of agreement for nominal scales. Educ. Psychol. Meas. 20 (1), 37–46. http://dx.doi.org/10.1177/001316446002000104.

Cramer, M., Haala, N., Laupheimer, D., Mandlburger, G., Havel, P., 2018. Ultra-high precision uav-based lidar and dense image matching. Int. Arch. Photogramm. Rem. Sens. Spatial Inform. Sci. XLII-1, 115–120. http://dx.doi.org/10.5194/isprs-archives-XLII-1-115-2018.

Das, S., Cakmak, U.M., 2018. Hands-On Automated Machine Learning: A Beginner's Guide to Building Automated Machine Learning Systems Using AutoML and Python. Packt Publishing.

de Gélis, I., Lefèvre, S., Corpetti, T., 2023. Siamese KPConv: 3D multiple change detection from raw point clouds using deep learning. ISPRS J. Photogramm. Remote Sens. 197, 274–291. http://dx.doi.org/10.1016/j.isprsjprs.2023.02.001.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V., 2017. CARLA: An open urban driving simulator. In: Proceedings of the 1st Annual Conference on Robot Learning. pp. 1–16.

Esmorís, A.M., Yermo, M., Weiser, H., Winiwarter, L., Höfle, B., Rivera, F.F., 2022. Virtual LiDAR simulation as a high performance computing challenge: Toward HPC HELIOS++. IEEE Access 10, 105052–105073. http://dx.doi.org/10.1109/ACCESS.2022.3211072.

Ferrara, R., Virdis, S.G., Ventura, A., Ghisu, T., Duce, P., Pellizzaro, G., 2018. An automated approach for wood-leaf separation from terrestrial LIDAR point clouds using the density based clustering algorithm DBSCAN. Agric. Forest Meteorol. 262, 434–444. http://dx.doi.org/10.1016/j.agrformet.2018.04.008.

Gao, F., Yan, Y., Lin, H., Shi, R., 2022. PIIE-DSA-net for 3D semantic segmentation of urban indoor and outdoor datasets. Remote Sens. 14 (15), http://dx.doi.org/10.3390/rs14153583.

Gastellu-Etchegorry, J.-P., Yin, T., Lauret, N., Grau, E., Rubio, J., Cook, B.D., Morton, D.C., Sun, G., 2016. Simulation of satellite, airborne and terrestrial LiDAR with DART (I): Waveform simulation with quasi-Monte Carlo ray tracing. Remote Sens. Environ. 184, 418–435. http://dx.doi.org/10.1016/j.rse.2016.07.010.

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

González-Collazo, S.M., Balado, J., González, E., Nurunnabi, A., 2023. A discordance analysis in manual labelling of urban mobile laser scanning data used for deep learning based semantic segmentation. Expert Syst. Appl. 230, 120672. http://dx.doi.org/10.1016/j.eswa.2023.120672.

Goodfellow, I., Bengio, Y., Courville, A., 2016a. Deep Learning. The MIT Press.

Goodfellow, I., Bengio, Y., Courville, A., 2016b. Deep Learning. The MIT Press, pp. 526–531, Chapter 15.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K. (Eds.), In: vol. Advances in Neural Information Processing Systems, 27, Curran Associates, Inc, URL https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Graham, B., 2015. Sparse 3D convolutional neural networks. In: Xie, X., Jones, M.W., Tam, G.K.L. (Eds.), Proceedings of the British Machine Vision Conference 2015. BMVC 2015, Swansea, UK, September 7-10, 2015, BMVA Press, pp. 150.1–150.9. http://dx.doi.org/10.5244/C.29.150.

Graham, B., Engelcke, M., Maaten, L., 2018. 3D semantic segmentation with submanifold sparse convolutional networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, IEEE Computer Society, Los Alamitos, CA, USA, pp. 9224–9232. http://dx.doi.org/10.1109/CVPR.2018.00961.

Griffiths, D., Boehm, J., 2019. A review on deep learning techniques for 3D sensed data classification. Remote Sens. 11 (12), http://dx.doi.org/10.3390/rs11121499.

Gschwandtner, M., Kwitt, R., Uhl, A., Pree, W., 2011. BlenSor: Blender sensor simulation toolbox. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Wang, S., Kyungnam, K., Benes, B., Moreland, K., Borst, C., DiVerdi, S., Yi-Jen, C., Ming, J. (Eds.), Advances in Visual Computing. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 199–208.

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M., 2021. Deep learning for 3D point clouds: A survey. IEEE Trans. Pattern Anal. Mach. Intell. 43 (12), 4338–4364. http://dx.doi.org/10.1109/TPAMI.2020.3005434.

Hackel, T., Savinov, N., Ladicky, L., Wegner, J.D., Schindler, K., Pollefeys, M., 2017. SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In: ISPRS Ann. Photogram. Rem. Sens. Spat. Inform. Sci. Information Sciences, vol. IV-1-W1, pp. 91–98.

Hackel, T., Wegner, J.D., Schindler, K., 2016. Fast semantic segmentation of 3D point clouds with strongly varying density. ISPRS Ann. Photogram. Rem. Sens. Spat. Inform. Sci. III-3, 177–184. http://dx.doi.org/10.5194/isprs-annals-III-3-177-2016.

Han, T., Sánchez-Azofeifa, G.A., 2022. A deep learning time series approach for leaf and wood classification from terrestrial LiDAR point clouds. Remote Sens. 14 (13), http://dx.doi.org/10.3390/rs14133157.

Hildebrand, J., Schulz, S., Richter, R., Döllner, J., 2022. Simulating LiDAR to create training data for machine learning on 3D point clouds. ISPRS Ann. Photogram. Rem. Sens. Spat. Inform. Sci. X-4/W2-2022, 105–112. http://dx.doi.org/10.5194/isprs-annals-X-4-W2-2022-105-2022.

Höfle, B., Pfeifer, N., 2007. Correction of laser scanning intensity data: Data and model-driven approaches. ISPRS J. Photogramm. Remote Sens. 62 (6), 415–433. http://dx.doi.org/10.1016/j.isprsjprs.2007.05.008.

Hopkinson, C., 2020. Data for: See the forest and the trees: Effective machine and deep learning algorithms for wood filtering and tree species classification from terrestrial laser scanning. http://dx.doi.org/10.17632/4gbzk9sy24.1, Mendley Data, V1.

Hurl, B., Czarnecki, K., Waslander, S., 2019. Precise synthetic image and LiDAR (PreSIL) dataset for autonomous vehicle perception. In: 2019 IEEE Intelligent Vehicles Symposium. IV, IEEE Press, pp. 2522–2529. http://dx.doi.org/10.1109/IVS.2019.8813809.

Jaccard, P., 1901. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bull. del la Société Vaudoise des Sciences Naturelles 37, 547–579.

Jutzi, B., Gross, H., 2009. Normalization of lidar intensity data based on range and surface incidence angle. ISPRS - Int. Arch. Photogramm. Rem. Sens. Spatial Inform. Sci. 38.

Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J.D., Ledoux, H., 2021. The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and multi-view-stereo. ISPRS Open J. Photogramm. Rem. Sens. 1, 11. http://dx.doi.org/10.1016/j.ophoto.2021.100001.

Krisanski, S., Taskhiri, M.S., Gonzalez Aracil, S., Herries, D., Muneri, A., Gurung, M.B., Montgomery, J., Turner, P., 2021a. Forest structural complexity tool—An open source, fully-automated tool for measuring forest point clouds. Remote Sens. 13 (22), http://dx.doi.org/10.3390/rs13224677.

Krisanski, S., Taskhiri, M.S., Gonzalez Aracil, S., Herries, D., Turner, P., 2021b. Sensor agnostic semantic segmentation of structurally diverse and complex forest point clouds using deep learning. Rem. Sens. 13 (8), 1413. http://dx.doi.org/10.3390/rs13081413.

Krishna Moorthy, S.M., Calders, K., Vicari, M.B., Verbeeck, H., 2020. Improved supervised learning-based approach for leaf and wood classification from LiDAR point clouds of forests. IEEE Trans. Geosci. Remote Sens. 58 (5), 3057–3070. http://dx.doi.org/10.1109/TGRS.2019.2947198.

Liu, C., Calders, K., Meunier, F., Gastellu-Etchegorry, J.P., Nightingale, J., Disney, M., Origo, N., Woodgate, W., Verbeeck, H., 2022. Implications of 3D forest stand reconstruction methods for radiative transfer modeling: A case study in the temperate deciduous forest. J. Geophys. Res.: Atmosp. 127 (14), http://dx.doi.org/10.1029/2021JD036175, e2021JD036175.

Matthews, B., 1975. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. Biochim. Biophysica Acta (BBA) - Protein Struct. 405 (2), 442–451. http://dx.doi.org/10.1016/0005-2795(75)90109-9.

Momo Takoudjou, S., Ploton, P., Sonké, B., Hackenberg, J., Griffon, S., de Coligny, F., Kamdem, N.G., Libalah, M., Mofack, G.I., Le Moguédec, G., Pélissier, R., Barbier, N., 2018. Data From: Using Terrestrial Laser Scanning Data to Estimate Large Tropical Trees Biomass and Calibrate Allometric Models: a Comparison with Traditional Destructive Approach. Dryad, http://dx.doi.org/10.5061/dryad.10hq7.

Otepka, J., Ghuffar, S., Waldhauser, C., Hochreiter, R., Pfeifer, N., 2013. Georeferenced point clouds: A survey of features and point cloud management. ISPRS Int. J. Geo-Inf. 2 (4), 1038–1065. http://dx.doi.org/10.3390/ijgi2041038.

Pearson, K., 1895. Note on regression and inheritance in the case of two parents. Proc. R. Soc. Lond. Ser. I 58, 240–242.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017a. PointNet: Deep learning on point sets for 3D classification and segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 77–85. http://dx.doi.org/10.1109/CVPR.2017.16.

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017b. PointNet++: Deep hierarchical feature learning on point sets in a metric space. http://dx.doi.org/10.48550/arXiv.1706.02413, arXiv preprint arXiv:1706.02413.

RIEGL Laser Measurement Systems, 2017. RIEGL VZ-400, data sheet. URL http://www.riegl.com/uploads/tx_pxprieglddownloads/10_DataSheet_VZ-400_2017-06-14.pdf. (Accessed: 13 Feb 2023).

RIEGL Laser Measurement Systems, 2022. RIEGL VUX-1LR, data sheet. http://www.riegl.com/uploads/tx_pxprieglddownloads/RIEGL_VUX-1LR-22_Datasheet_2022-09-14.pdf. (Accessed: 22 Feb 2023).

Rousseau, D., Turgut, K., Dutagaci, H., 2022. Robustness of 3D point-based deep learning for plant organ segmentation against point density variation and noise. http://dx.doi.org/10.22541/au.166497086.66500223/v1.

Schmohl, S., Sörgel, U., 2019. Submanifold sparse convolutional networks for semantic segmentation of large-scale als point clouds. ISPRS Ann. Photogram. Rem. Sens. Spat. Inform. Sci. IV-2/W5, 77–84. http://dx.doi.org/10.5194/isprs-annals-IV-2-W5-77-2019.

Shan, J., Toth, C., 2018. Topographic Laser Ranging and Scanning: Principles and Processing, Second Edition. CRC Press, http://dx.doi.org/10.1201/9781315154381.

Singer, N.M., Asari, V.K., 2021. DALES objects: A large scale benchmark dataset for instance segmentation in aerial LiDAR. IEEE Access 1. http://dx.doi.org/10.1109/ACCESS.2021.3094127.

Sokolova, M., Lapalme, G., 2009. A systematic analysis of performance measures for classification tasks. Inf. Process. Manage. 45 (4), 427–437. http://dx.doi.org/10.1016/j.ipm.2009.03.002.

Solow, D., 2014. Linear programming: An introduction to finite improvement algorithms: Second edition. In: Dover Books on Mathematics, Dover Publications.

Stewart, J., 2012. Calculus : Early Transcendentals. Brooks/Cole, Cengage Learning, Belmont, Cal.

Stigler, S.M., 1989. Francis Galton's account of the invention of correlation. Statist. Sci. 4 (2), 73–79. http://dx.doi.org/10.1214/ss/1177012580.

Thomas, H., Qi, C.R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L.J., 2019. KPConv: Flexible and deformable convolution for point clouds. In: Proceedings of the IEEE International Conference on Computer Vision.

Triess, L.T., Rist, C.B., Peter, D., Zöllner, J.M., 2022. A realism metric for generated LiDAR point clouds. Int. J. Comput. Vision 130 (12), 2962–2979. http://dx.doi.org/10.1007/s11263-022-01676-8.

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 Ecol. Evol. 10 (5), 680–694. http://dx.doi.org/10.1111/2041-210X.13144.

Wang, C., Li, Q., Liu, Y., Wu, G., Liu, P., Ding, X., 2015. A comparison of waveform processing algorithms for single-wavelength LiDAR bathymetry. ISPRS J. Photogramm. Remote Sens. 101, 22–35. http://dx.doi.org/10.1016/j.isprsjprs.2014.11.005.

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 Ecol. Evol. 11 (3), 376–389. http://dx.doi.org/10.1111/2041-210X.13342.

Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M., 2019. Dynamic graph CNN for learning on point clouds. ACM Trans. Graph. 38 (5), 1–12. http://dx.doi.org/10.1145/3326362.

Wang, D., Takoudjou, S.M., Casella, E., 2021. LeWoS: A Universal Leaf-Wood Classification Method to Facilitate The 3D Modelling of Large Tropical Trees Using Terrestrial LiDAR. Dryad, http://dx.doi.org/10.5061/dryad.np5hqbzp6.

Weber, J., Penn, J., 1995. Creation and rendering of realistic trees. In: Mair, S.G., Cook, R. (Eds.), Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '95, ACM Press, New York, New York, USA, pp. 119–128. http://dx.doi.org/10.1145/218380.218427.

Weinmann, M., Urban, S., Hinz, S., Jutzi, B., Mallet, C., 2015. Distinctive 2D and 3D features for automated large-scale scene analysis in urban areas. Comput. Graph. 49, 47–57. http://dx.doi.org/10.1016/j.cag.2015.01.006.

Weiser, H., Schäfer, J., Winiwarter, L., Krašovec, N., Seitz, C., Schimka, M., Anders, K., Baete, D., Braz, A.S., Brand, J., Debroize, D., Kuss, P., Martin, L.L., Mayer, A., Schrempp, T., Schwarz, L.-M., Ulrich, V., Fassnacht, F.E., Höfle, B., 2021a. Terrestrial, UAV-Borne, and Airborne Laser Scanning Point Clouds of Central European Forest Plots, Germany, with Extracted Individual Trees and Manual Forest Inventory Measurements. PANGAEA, http://dx.doi.org/10.1594/PANGAEA.933426.

Weiser, H., Schäfer, J., Winiwarter, L., Krašovec, N., Fassnacht, F.E., Höfle, B., 2022. Individual tree point clouds and tree measurements from multi-platform laser scanning in German forests. Earth Syst. Sci. Data 14 (7), 2989–3012. http://dx.doi.org/10.5194/essd-14-2989-2022.

Weiser, H., Ulrich, V., Winiwarter, L., Esmorís, A.M., Höfle, B., 2023. Manually Labeled Terrestrial Laser Scanning Point Clouds of Individual Trees for Leaf-Wood Separation. heiDATA, http://dx.doi.org/10.11588/data/UUMEDI.

Weiser, H., Winiwarter, L., Anders, K., Fassnacht, F.E., Höfle, B., 2021b. Opaque voxel-based tree models for virtual laser scanning in forestry applications. Remote Sens. Environ. 265, 112641. http://dx.doi.org/10.1016/j.rse.2021.112641.

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 Sens. Environ. 269, http://dx.doi.org/10.1016/j.rse.2021.112772.

Winiwarter, L., Mandlburger, G., Schmohl, S., Pfeifer, N., 2019. Classification of ALS point clouds using end-to-end deep learning. PFG – J. Photogramm. Rem. Sens. Geoinform. Sci. 87 (3), 75–90. http://dx.doi.org/10.1007/s41064-019-00073-0.

Wu, B., Wan, A., Yue, X., Keutzer, K., 2018. SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud. In: 2018 IEEE International Conference on Robotics and Automation. ICRA, IEEE Press, pp. 1887–1893. http://dx.doi.org/10.1109/ICRA.2018.8462926.

Wu, B., Zheng, G., Chen, Y., 2020. An improved convolution neural network-based model for classifying foliage and woody components from terrestrial laser scanning data. Rem. Sens. 12 (6), 1010. http://dx.doi.org/10.3390/rs12061010.

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 J. Photogramm. Rem. Sens. 168, 1–16. http://dx.doi.org/10.1016/j.isprsjprs.2020.08.001.

Zahs, V., Anders, K., Kohns, J., Stark, A., Höfle, B., 2023. Classification of structural building damage grades from multi-temporal photogrammetric point clouds using a machine learning model trained on virtual laser scanning data. Int. J. Appl. Earth Obs. Geoinf. 122, 103406. http://dx.doi.org/10.1016/j.jag.2023.103406.

Zhang, X.-D., 2017. Matrix Analysis and Applications. Cambridge University Press, http://dx.doi.org/10.1017/9781108277587.