DGK Veröffentlichungen der DGK

Ausschuss Geodäsie der Bayerischen Akademie der Wissenschaften

Reihe C | Dissertationen | Heft Nr. 897

Dominik Laupheimer

On the Information Transfer Between Imagery, Point Clouds, and Meshes for Multi-Modal Semantics Utilizing Geospatial Data

München 2023

Bayerische Akademie der Wissenschaften

ISSN 0065-5325 ISBN 978-3-7696-5309-0

Diese Arbeit ist gleichzeitig veröffentlicht in:

OPUS – Online Publikationen der Universität Stuttgart <http://dx.doi.org/10.18419/opus-12668>,

Stuttgart 2022)

DGK Veröffentlichungen der DGK

Ausschuss Geodäsie der Bayerischen Akademie der Wissenschaften

Reihe C Dissertationen Heft Nr. 897

On the Information Transfer Between Imagery, Point Clouds, and Meshes for Multi-Modal Semantics Utilizing Geospatial Data

A thesis
accepted by the Faculty of Aerospace Engineering and Geodesy
of the University of Stuttgart
in partial fulfilment of the requirements for the degree of
Doctor of Engineering Sciences (Dr.-Ing.)

by

Dominik Laupheimer, M.Sc.

born in Laupheim

München 2023

Bayerische Akademie der Wissenschaften

ISSN 0065-5325 ISBN 978-3-7696-5309-9

Diese Arbeit ist gleichzeitig veröffentlicht in:
OPUS – Online Publikationen der Universität Stuttgart
<http://dx.doi.org/10.18419/opus-12668>,
Stuttgart 2022)

Adresse der DGK:

Ausschuss Geodäsie der Bayerischen Akademie der Wissenschaften (DGK)

Alfons-Goppel-Straße 11 ● D – 80 539 München

Telefon +49 – 89 – 23 031 1113 ● Telefax +49 – 89 – 23 031 - 1283 / - 1100

e-mail post@dgk.badw.de ● http://www.dgk.badw.de

main referee: apl. Prof. Dr.-Ing. Norbert Haala

co-referee: Prof. Dr. Ir. George Vosselman

date of defense: September 9th, 2022

ISSN 0065-5325 ISBN 978-3-7696-5309-0

Contents

Acronyms

AAT	Automatic Aerial Triangulation
AL	Active Learning
ALS	Airborne Laser Scanning
BPNet	Bidirectional Projection Network
CAD	Computer-Aided Design
CNN	Convolutional Neural Network
CoG	Center of Gravity
CPU	Central Processing Unit
CRF	Conditional Random Field
DIM	Dense Image Matching
DL	Deep Learning
DSM	Digital Surface Model
DTM	Digital Terrain Model
GPU	Graphics Processing Unit
GSD	Ground Sampling Distance
GT	Ground Truth
GUI	Graphical User Interface
H3D	Hessigheim 3D
ICP	Iterative Closest Point
ImgMA	Image Mesh Association
ISPRS	International Society for Photogrammetry and Remote Sensing
LiDAR	Light Detection and Ranging
LOD	Level of Detail
MBB	Minimum Bounding Box
ML	Machine Learning
MLS	Mobile Laser Scanning
MRF	Markov Random Field
MVS	Multi-View Stereo
nDSM	normalized Digital Surface Model
NeRF	Neural Radiance Field
NN	Nearest Neighbor
OA	Overall Accuracy
OGC	Open Geospatial Consortium
PA	Producer's Accuracy (Recall)
PC	Point Cloud
PCA	Principal Component Analysis
PCImgA	Point Cloud Image Association
PCMA	Point Cloud Mesh Association
PL	Passive Learning
pp	Percent Points
RaDAR	Radio Detection and Ranging
RF	Random Forest
RIU	Reducing Interpretation Uncertainty
SACNN	Structure-Aware Convolutional Neural Network
TLS	Terrestrial Laser Scanning
UA	User's Accuracy (Precision)
UAV	Unmanned Airborne Vehicle
V3D	Vaihingen 3D

Abstract

The semantic segmentation of the huge amount of acquired 3D data has become an important task in recent years. Images and Point Clouds (PCs) are fundamental data representations, particularly in urban mapping applications. Textured meshes integrate both representations by wiring the PC and texturing the reconstructed surface elements with high-resolution imagery. Meshes are adaptive to the underlying mapped geometry due to their graph structure composed of non-uniform and non-regular entities. Hence, the mesh is a memory-efficient realistic-looking 3D map of the real world. For these reasons, we primarily opt for semantic segmentation of meshes, which is a widely overlooked topic in photogrammetry and remote sensing yet. In particular, we head for multi-modal semantics utilizing supervised learning. However, publicly available annotated geospatial mesh data has been rare at the beginning of the thesis. Therefore, annotating mesh data has to be done beforehand. To kill two birds with one stone, we aim for a multi-modal fusion that enables multi-modal enhancement of entity descriptors and semi-automatic data annotation leveraging publicly available annotations of non-mesh data. We propose a novel holistic geometry-driven association mechanism that explicitly integrates entities of modalities imagery, PC, and mesh. The established entity relationships between pixels, points, and faces enable the sharing of information across the modalities in a two-fold manner: (i) feature transfer (measured or engineered) and (ii) label transfer (predicted or annotated). The implementation follows a tile-wise strategy to facilitate scalability to large-scale data sets. At the same time, it enables parallel, distributed processing, reducing processing time. We demonstrate the effectiveness of the proposed method on the International Society for Photogrammetry and Remote Sensing (ISPRS) benchmark data sets Vaihingen 3D and Hessigheim 3D (Niemeyer et al. 2014, Kölle et al. 2021a). Taken together, the proposed entity linking and subsequent information transfer inject great flexibility into the semantic segmentation of geospatial data. Imagery, PCs, and meshes can be semantically segmented with classifiers trained on any of these modalities utilizing features derived from any of these modalities. Particularly, we can semantically segment a modality by training a classifier on the same modality (*direct* approach) or by transferring predictions from other modalities (*indirect* approach). Hence, any established well-performing modality-specific classifier can be used for semantic segmentation of these modalities – regardless of whether they follow an end-to-end learning or feature-driven scheme. We perform an extensive ablation study on the impact of multi-modal handcrafted features for automatic 3D scene interpretation – both for the direct and indirect approach. We discuss and analyze various Ground Truth (GT) generation methods. The semi-automatic labeling leveraging the entity linking achieves consistent annotation across modalities and reduces the manual label effort to a single representation. Please note that the multiple epochs of the Hessigheim data consisting of manually annotated PCs and semi-automatically annotated meshes are a result of this thesis and provided to the community as part of the Hessigheim 3D benchmark. To further reduce the labeling effort to a few instances on a single modality, we combine the proposed information transfer with active learning. We recruit non-experts for the tedious labeling task and analyze their annotation quality. Subsequently, we compare the resulting classifier performances to conventional passive learning using expert annotation. In particular, we investigate the impact of visualizing the mesh instead of the PC on the annotation quality achieved by non-experts. In summary, we accentuate the mesh and its utility for multi-modal fusion, GT generation, multi-modal semantics, and visualizational purposes.

Kurzfassung

Die automatische Interpretation der Menge an erfassten 3D-Daten ist eine elementare Aufgabe. Bilder und Punktwolken (PCs) sind grundlegende Repräsentationen – insbesondere im Bereich der urbanen Kartografie. Texturierte Meshes vereinen beide Repräsentationen, indem sie die PCs vermaschen und die rekonstruierten Oberflächenelemente mit hochauflösendem Bildinhalt texturieren. Meshes passen sich dank ihrer Graphenstruktur bestehend aus nicht-uniformen und nicht-regulären Entitäten an die zugrundeliegende Geometrie des kartierten Bereiches an. Daher ist das Mesh eine speichereffiziente und wirklichkeitsnahe 3D-Karte. Wir widmen uns vorrangig der semantischen Segmentierung von Meshes mittels Methoden des überwachten Lernens – ein in der Photogrammetrie & Fernerkundung noch weitgehend unerforschtes Thema. Hierfür sind annotierte urbane Meshes notwendig, die zu Beginn der Arbeit öffentlich nicht zur Verfügung standen. Um zwei Fliegen mit einer Klappe zu schlagen, streben wir eine multimodale Fusion an, die zum einen eine multimodale Erweiterung von Deskriptoren und zum anderen eine halbautomatische Annotation durch Nutzung öffentlich zugänglicher annotierter Repräsentationen ermöglicht. Wir stellen einen holistischen, geometriegesteuerten Verknüpfungsmechanismus vor, der die jeweiligen Entitäten der Modalitäten Bild, PC und Mesh explizit miteinander assoziiert. Die detektierten Verknüpfungen zwischen Pixeln, Punkten und Oberflächenelementen ermöglichen den intermodalen Informationsaustausch auf zweierlei Weisen: (i) Transfer direkt gemessener oder abgeleiteter Merkmale und (ii) Transfer der prädizierten oder annotierten Klassenzugehörigkeit. Im Sinne der Skalierbarkeit folgt die Implementierung einer kachelweisen Strategie, die eine parallelisierte, verteilte Verarbeitung ermöglicht. Wir demonstrieren die Wirksamkeit der vorgeschlagenen Methode anhand der ISPRS-Benchmark-Datensätze Vaihingen 3D und Hessigheim 3D (Niemeyer et al. 2014, Kölle et al. 2021a). Die multimodale Verknüpfung mit anschließendem Informationsübertrag ermöglicht eine flexible semantische Segmentierung von Geodaten in Form von Bildern, PCs und Meshes. Die Klassifikatoren werden auf einer dieser Modalitäten trainiert und greifen dabei auf Merkmale zurück, die von jener und/oder anderen Modalitäten abgeleitet sind. Die semantische Segmentierung kann durch einen Klassifikator, der auf derselben Modalität trainiert ist (*direkter* Ansatz), oder durch den Übertrag von Prädiktionen anderer Modalitäten (*indirekter* Ansatz) erfolgen. Somit kann jeder etablierte modalitätsspezifische Klassifikator für die semantische Segmentierung dieser Modalitäten verwendet werden – unabhängig davon, ob es ein Ende-zu-Ende-Lernverfahren oder ein merkmalsgesteuertes Verfahren ist. Umfangreiche Tests evaluieren den Einfluss der multimodalen Merkmale auf die 3D-Szeneninterpretation. Außerdem diskutieren wir verschiedene Methoden der Ground Truth (GT)-Erzeugung. Das vorgestellte halbautomatische Verfahren erreicht eine konsistente Annotation über alle Modalitäten hinweg und reduziert den manuellen Annotationsaufwand auf eine einzige Repräsentationsform. Wir weisen darauf hin, dass die Epochen des Hessigheim-3D-Benchmarks, die aus manuell annotierten PCs und halbautomatisch annotierten Meshes bestehen, ein Resultat dieser Arbeit und öffentlich zur Verfügung gestellt sind. Um den Annotationsaufwand weiterhin auf wenige Instanzen einer einzigen Modalität zu reduzieren, kombinieren wir den vorgestellten Informationstransfer mit Active Learning. Des Weiteren beauftragen wir Laien mit der GT-Erzeugung und analysieren insbesondere, welchen Einfluss die visualisierte Modalität (PC oder Mesh) auf die von Laien erreichte Annotationsgenauigkeit hat. Abschließend vergleichen wir die resultierende Klassifikationsgenauigkeit mit dem Ergebnis des herkömmlichen Passive-Learning-Klassifikators, der Experten-Annotationen in der Trainingsphase nutzt. Zusammengefasst demonstrieren wir den Nutzen von Meshes für die multimodale Fusion, GT-Erzeugung, semantische Analyse und Visualisierungszwecke.

Chapter 1

Introduction

1.1 Motivation

In the past decade, 3D data acquisition and data processing has increasingly become feasible and important in the domain of photogrammetry and remote sensing. Over the years, geospatial data acquisition has become more redundant, more complete, faster, and denser – spatially and temporally. Sensors such as cameras, LiDAR scanners, and RaDAR sensors facilitate multi-modal capturing of our world. Throughout this thesis, *multi-modality* refers to multi-sensor acquisition and the respective mapping of the world with imagery, Point Clouds (PCs), and meshes (see Chapter 2). Depending on the application and the desired mapping scale, the respective sensors are mounted on platforms such as satellites, airplanes, Unmanned Airborne Vehicles (UAVs), or autonomous vehicles. The use of multi-sensor systems significantly reduces acquisition time and avoids discrepancies due to time shifts in the captured data (*hybrid data acquisition*). In photogrammetry and remote sensing, particularly for urban mapping, data is commonly acquired via imagery and Airborne Laser Scanning (ALS) at Ground Sampling Distances (GSDs) or footprints, respectively, down to a few centimeters. Nowadays, joint acquisition of photogrammetric and ALS data is state of the art for airborne systems and starts to emerge even for more flexible and lightweight UAV-based systems (Cramer et al., 2018; Haala et al., 2020, 2022; Mandlburger et al., 2017).

Imagery is the fundamental photogrammetric data representation providing (multi-)spectral information. Images project 3D real-world objects into 2D image space. By nature of the projection into grid-like pixel space, images suffer from occlusions, distortions, discretization, and the loss of the third dimension. However, Automatic Aerial Triangulation (AAT) can rectify the intrinsic defects and facilitates 3D reconstruction. As a precondition to proper reconstruction, images must provide unambiguous texture and capture each object point at least twice. Multi-View Stereo (MVS) derives colored dense PCs which can be enhanced to textured meshes. Both modalities map the surface of the captured region (canopy behavior). The accuracy of MVS products is correlated with the GSD which, theoretically, can be scaled arbitrarily.

In contrast, 3D PCs are the immediate output of ALS due to its polar measurement principle. Thus, a single measurement is sufficient to map a 3D point. Its accuracy depends on the accuracy of the trajectory. In comparison to MVS, LiDAR scanning provides multi-target capability and hence penetrates semi-transparent objects such as vegetation. On the other hand, bare LiDAR points do not carry color/texture information like PCs/meshes derived from imagery.

Initially, photogrammetry and laser scanning have been competitive techniques with individual processing pipelines. However, at present, they are seen as complementary systems whose fusion results in more complete, more informative, and more precise products. Recently, Glira et al. (2019) proposed the joint orientation of ALS PCs and aerial imagery, which improves the georeferencing accuracy of ALS data by integrating stabilizing image block geometry into the strip adjustment (*hybrid orientation*). Haala et al. (2020, 2022) proved its potential for UAV data achieving sub-centimeter accuracies. As a side product, the hybrid orientation enables an improved co-registration of imagery and LiDAR data.

The increasing availability of (simultaneously) acquired airborne data with different acquisition methods calls for automated multi-modal processing and scene analysis (*hybrid semantics*). The geospatial community put a big effort into the semantic segmentation of large-scale (LiDAR) PCs over the past few years. However, PCs are not an end-user product in our opinion. From our point of view, enhancing 3D PCs to textured meshes may replace unstructured PCs as the default representation for urban scenes in the future. Intrinsically, meshes facilitate multi-modal data fusion by utilizing LiDAR points and MVS points for the geometric reconstruction while leveraging high-resolution imagery for texturing (*hybrid representation*). Therefore, meshes are realistic-looking 3D maps of the real world and are easily understandable – even for non-experts. In addition to benefits for visualization, textured meshes have other favorable characteristics. Whereas PCs are unordered sets of points, meshes are graphs consisting of vertices, edges, and faces that provide explicit adjacency information. The mesh is adaptive to the underlying geometry due to the non-uniformity and non-regularity of faces (see Chapter 2). This means planar surfaces are represented by a few large faces, whereas vivid areas are reconstructed by many small surface elements. Generally, the adaptiveness embraces noise filtering and results in a less memory-consuming 3D representation compared to a PC. By definition, surface meshes cannot handle multi-target capability like LiDAR PCs, which inevitably leads to a drop in entities to be stored. Furthermore, geometric simplifications based on the desired level of detail and, as the case may be, due to 2.5D mesh geometry may further reduce the memory footprint. Likewise, the high-resolution texture information is stored in compact texture atlases avoiding redundant image content. Therefore, textured meshes provide geometric and textural information in a lightweight fashion. They are util for visualization, spatial analyses, and urban planning. However, the meshing of PCs is a non-trivial task and subject to current research.

Despite their advantageous characteristics, textured meshes are a mostly overlooked topic in photogrammetry and remote sensing (see Chapter 3). For the abovementioned reasons, we strive for semantic segmentation of textured meshes (see Chapter 6). Moreover, we want to account for the current hybridization trend with this work and aim at automated scene interpretation integrating information from PCs and images deploying the mesh as the integrative backbone (*hybrid semantics*). The integrative character of the mesh facilitates data fusion "out-of-the-box" by utilizing LiDAR points and MVS points for the geometric reconstruction while leveraging high-resolution imagery for texturing. However, the information regarding the source modality is neither encoded in the mesh vertices nor the texture atlas and hence, cannot be accessed in further processing steps. Therefore, we propose the explicit linking of imagery, PC, and mesh to recover the relationships between pixels, points, and faces (see Chapter 5). We utilize textured meshes as the core of the multi-modal linking due to their native integrative character. Each face will be associated with several points and pixels. In turn, points and pixels are linked while checking the visibility via the mesh. The established connections on the entity-level enable the information sharing across modalities and joint leveraging of different data sources (*multi-modal fusion*). Representation-specific features and (manually generated) annotations can be shared arbitrarily across the modalities (see Figure 1.1). The top of Table 5.1 depicts the concept of the inter-modal association and the related information transfer in a pictographic manner. We utilize the proposed entity linking a) to enhance modality-specific entities to multi-modal descriptors and b) to reduce the manual annotation effort by transferring labels to other modalities.

We demonstrate the effectiveness of the proposed method on benchmark data sets Vaihingen 3D (V3D) and Hessigheim 3D (H3D) from the International Society for Photogrammetry and Remote Sensing (ISPRS) (see Chapter 4) and compare it to a simple nearest-neighbor transfer. Please note the semi-automatically annotated meshes of Hessigheim are provided to the community as part of the H3D benchmark. The provided annotated urban mesh data may unleash the potential of mesh interpretation at geospatial scale.

The automated semantic analysis booms due to recent developments in Machine Learning (ML), precisely Deep Learning (DL). The quantity and quality of available annotated data, so-called Ground Truth (GT) data, steer the performance of data-driven supervised learning approaches. GT generation, therefore, has become an essential task since the dawn of the DL era. Commonly, experts generate and check GT manually. This process is tedious, time-consuming, and expensive work, specifically for imagery consisting of millions of pixels. Our linking and transferring methodology enables the consistent labeling of various representations while reducing the manual annotation effort to a single modality (semi-automatic labeling).

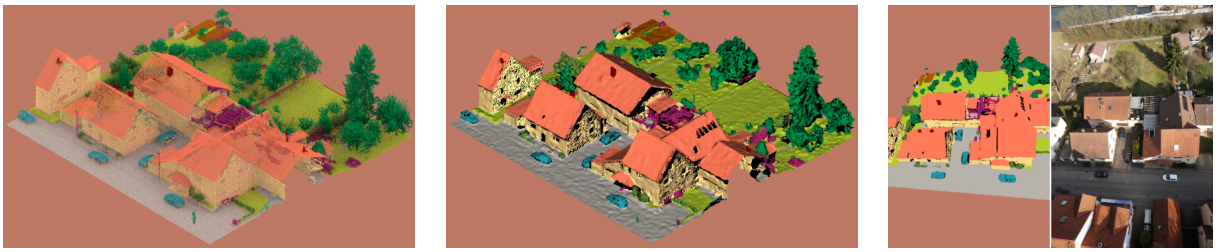


Figure 1.1: The explicit linking of pixels, points, and faces (multi-modal data fusion) enables the information exchange across the three modalities PC (*left*), mesh (*center*), and imagery (*right*). The figure exemplarily depicts the label propagation from the manually annotated PC to the mesh and an oblique image (for a subset of Hessigheim 3D, see Section 4.2). Faces that cannot be linked to points remain without a label (depicted in black). Pixels that are linked to an unlabeled face are colored in black. Background and non-associated pixels are colored in reddish-brown.

The application of Active Learning (AL) reduces the manual labeling effort to a few instances. The key of AL is to label a sparse, informative subset instead of the entire data set. The informative subset is built and annotated in an iterative process, the so-called AL loop. The AL loop iteratively trains a ML classifier on the increasingly extended annotated subset while steering which data points have to be labeled in the next loop iteration. We combine AL with paid crowdsourcing to recruit non-experts for the tedious label work. We analyze the effect of the visualized modality during the annotation process on the annotation quality. In particular, we investigate whether an unknown group of non-experts achieves better annotation quality on the PC when visualizing the mesh instead of the PC. We argue that the realistic-looking appearance of meshes is easier to understand for non-experts than PCs and hence offers better annotation quality. The assigned labels can be shared across all modalities by coupling them via the proposed entity linking. Hence, the integration of the multi-modal entity linking and AL limits the manual labeling effort to a few instances on a single modality. We train ML classifiers on the sparse training sets of 3D modalities PC and mesh and compare their performances to respective classifiers leveraging the entire annotated training data, i.e., Passive Learning (PL).

Apart from reducing the manual labeling effort, we utilize the multi-modal entity linking for enhancing entity descriptors with features from other modalities. We perform an extensive ablation study on the annotated entirety of data sets V3D and H3D for 3D modalities PC and mesh. The ablation study investigates the impact of the measured and engineered features derived from both 3D modalities. In particular, we analyze the benefit of the additional PC features on semantic segmentation of meshes by comparing the classifier performance with/without LiDAR support on the face level. Moreover, we compare *direct* and *indirect* predictions for various feature vector compositions to evaluate whether the circuit to other modalities is beneficial for the semantic analysis. Whereas direct predictions semantically segment a modality by training a classifier on the same modality, the indirect approach transfers predictions from other modalities.

In summary, the proposed multi-modal entity linking across imagery, PC, and mesh is the main contribution and backbone of this thesis. The methodology enables the *Juggling with Representations* and injects flexibility and versatility into the semantic segmentation of geospatial data. The explicit linking of pixels, points, and faces enables the sharing of features and annotations from available data sources. By these means, the multi-modal fusion boosts consistent GT generation across modalities and fosters multi-modal semantic scene analysis in multiple ways. We analyze the semantic segmentation of 3D data by deploying various feature vector compositions and learning strategies, i.e., AL and PL. Besides, we use the mesh to colorize LiDAR PCs and as a visualizer to steer non-expert annotators during PC annotation. Section 1.2 compactly lists the contributions and objectives of the thesis.

1.2 Objectives

- multi-modal linking and data fusion:
  The inter-modal entity linking explicitly associates pixels, points, and faces. The association mechanism utilizes the mesh as core representation and should meet the following requirements:
  - information exchange:
    The established inter-modal connections on the entity level (backbone infrastructure) should enable the information sharing across representations considering the transfer direction and the information type (features or labels).
  - scalability:
    The algorithm should operate in a tile-wise fashion to process large-scale multi-modal real-world data while not being restricted to high-performing computing clusters. The tiled procedure allows for parallel processing, which concurrently reduces processing times. However, optimizing the processing time is out of the scope of this thesis and not a goal per se. Intermediate outputs (relevant for the parallelized computing) should be stored in memory-efficient formats (e.g., sparse pixel cloud, see Chapter 5).
  - genericity and robustness:
    - independence of modality discrepancies:
      The association mechanism has to cope with minor local co-registration residuals and structural discrepancies between modalities as discussed in Section 1.1.
    - independence towards meshing algorithm and mesh dimensionality:
      The method should not impose any constraints on the mesh generation or mesh dimensionality (2.5D or 3D). The linking should work for both 2.5D and 3D meshes regardless of the underlying meshing algorithm/software.
    - platform independence:
      The linking method should not be bound to airborne imagery or ALS data. Any PC no matter the acquisition platform and any imagery following the central perspective may be used. However, we focus on the airborne case and adjust implementation to that scenario.

- concept verification:
  We demonstrate the effectiveness of the proposed method on the ISPRS benchmark data sets V3D and H3D (Niemeyer et al. 2014, Kölle et al. 2021a) by comparing results to a simple nearest-neighbor transfer.

- efficient GT generation:
  The established inter-modal entity relationships enable consistent annotation of the discussed modalities reducing the manual annotation effort to a single modality (see Hessigheim 3D benchmark). The combination with AL further reduces the manual labeling effort to a few instances on a single modality.

- multi-modal semantic segmentation:
  The semi-automatic GT generation enables ML classifier training on each modality (direct semantic segmentation). Made predictions can be shared across modalities, too (indirect semantic segmentation). In both cases, feature sharing may enhance the modality-specific analysis to multi-modal semantic segmentation incorporating information from images, PC, and mesh. We compare direct and indirect results deploying various feature vector compositions. Besides, we compare the classifier performance of the AL pipeline to a conventional classifier that leverages the entire training data.

- mesh utility:
  The mesh is the core of the multi-modal entity linking. Apart from the multi-modal fusion, we analyze the utility of meshes for geospatial data in terms of colorization, visualization, and semantic segmentation. We use the mesh to colorize LiDAR PCs and as a visualizer to steer non-expert annotators during PC annotation.

1.3 Outline

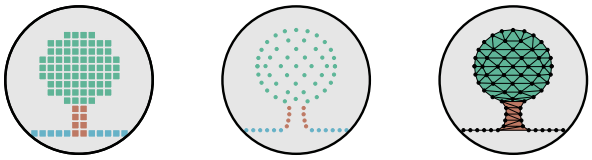
Chapter 2 clarifies terms used throughout this thesis and identifies unique characteristics of the modalities imagery, PC, and mesh by giving a compact overview of the involved representations and their properties. In Chapter 3, we review the literature concerning semantic segmentation, multi-modal fusion, data annotation, and the public availability of annotated geospatial data. In particular, we highlight the 3D scene interpretation utilizing the 3D modalities PC and mesh. Chapter 4 briefly presents the investigated real-world data sets Vaihingen 3D (V3D) and Hessigheim 3D (H3D) (Niemeyer et al. 2014, Kölle et al. 2021a) along with their key parameters. We describe in detail the explicit entity linking and subsequent information transfer by deep-diving its building blocks in Chapter 5. Section 5.1 describes the association of (LiDAR) PCs and meshes in 3D space; Section 5.2 outlines the association of imagery and meshes. Eventually, the combination of both links 3D points and pixels (see Section 5.3). For the sake of good scientific practice, we critically reflect on the preconditions and explore where the proposed approach might be limited due to the mentioned (structural) discrepancies between the mesh, the PC, and imagery (see Section 5.4). In Chapter 6, we highlight the 3D scene interpretation embracing GT generation (see Section 6.1), feature calculation (see Section 6.2), and semantic segmentation of PCs and meshes (see Sections 6.4, 6.5, and 6.6). Section 6.1 discusses annotation methods of varying manual labeling effort. Specifically, we discuss the manual annotation, the semi-automatic labeling utilizing the explicit entity linking, and its sparse counterpart combining the annotation process with AL. The scope of this work is not to design new distinguishing features. Instead, we utilize standard features that are commonly used for (semantic) segmentation tasks (see Section 6.2). We analyze the semantic segmentation of PCs and meshes by training classifiers on each modality directly and by transferring predictions from the other modality. Furthermore, we analyze the benefit of multi-modal descriptors by comparing the classifier performances with/without multi-modal support on the face level and the point level. Section 6.6 compares the achievable classifier performances on PCs and meshes when deploying AL and PL. We summarize the key findings of the performed experiments at the end of each section. Finally, Chapter 7 concludes the core results of the thesis and ventures an outlook of possible developments that might be tackled in the future.

Chapter 2

Background: Modalities and Their Representations

Throughout this thesis, we operate with the three base modalities of photogrammetry and remote sensing: imagery, PC, and mesh. As stated in Section 1.1, photogrammetric processing enables the generation of PCs and meshes from captured images. Likewise, LiDAR PCs or a combination of MVS points and LiDAR points can be enhanced to 3D meshes. Therefore, these three modalities and their representations are related: They are discrete approximations on the underlying continuous manifold. However, they feature different entities which vary in their shape/extension (*uniformity*) and spatial arrangement/density (*regularity*). Table 2.1 compactly contrasts their properties.

Table 2.1: Main properties of the considered modalities imagery (*left*), point cloud (*center*), and mesh (*right*).



	Imagery	Point Cloud	Mesh
Dimensionality	2D/2.5D	3D	2.5D/3D
Entities	Pixel	Point	Vertex, Edge, Face
Regularity	✓	✗	✗
Uniformity	✓	✓	✗
Manifold-Aware Neighborhood	✗	✗	✓

Mathematically, an image $\mathcal{I}(r, c)$ is a structured array/grid of pixels. Pixel positions are denoted by row r and column c. The extension to 3D space is known as voxel space. Any attribute can be represented by an image. Multiple attributes can be stacked onto each other forming a multi-channel image built by per-attribute images. Due to the regular data organization, images commonly define neighborhood relations by 4-connected or 8-connected pixels per channel. However, these adjacency relations are restricted to the grid and do not necessarily mimic adjacency relations on the underlying continuous manifold.

In contrast, PCs provide only implicit neighborhood relations through nearest neighbors in Euclidean space because of their unordered nature. A 3D PC is an unstructured set of uniform 3D points $\mathcal{P} \in \mathbb{R}^{3+d}$. Each point $\boldsymbol{p}$ is represented by coordinates X, Y, and Z and, optionally, an arbitrary number d of attributes.

The mesh $\mathcal{M}$ is a discrete approximation of a 2-manifold built by two-dimensional polygons. Throughout this thesis, we use only surface meshes constructed of 2D triangles. $\mathcal{M}$ can be seen as a graph $\mathcal{M}_G = (\mathcal{F}, \mathcal{E}, \mathcal{V})$ consisting of a set of triangular faces $\mathcal{F}$, a set of edges $\mathcal{E}$, and a set of vertices $\mathcal{V}$. It can be visualized as a wireframe (see Figure 2.1, *left*). The mesh adapts to the local structure due to its non-regular and non-

uniform architecture. In literature, polygon-based meshes are commonly named 2D meshes or surface meshes, whereas meshes constructed of volumes are known as 3D meshes or volume meshes. However, we use the term 3D mesh in this thesis to refer to a meshed three-dimensional point cloud, i.e., each tuple (X, Y) can have multiple Z values. In contrast, a 2.5D mesh is a meshed Digital Surface Model (DSM) where each cell of the discretized XY-plane carries a unique height. 2.5D meshes are also known as DSM meshes. Regardless of dimensionality, the neighborhood relations are explicitly encoded in the graph structure by the adjacency matrix (also known as connectivity matrix). The encoded connectivity allows geodesic neighborhood queries on the manifold discretized by the mesh surface. Thus, the graph representation enables distinguishing geodesic and Euclidean distance.

Automatically reconstructed meshes do not necessarily meet the manifold geometry. Loosely speaking, non-manifold geometry cannot exist in reality and poses a problem in mesh processing tools and operations. Rendering refractive effects, fluid simulations, boolean operations, 3D printing, and graph-based approaches for semantic mesh segmentation require manifold meshes. Common phenomena of non-manifold meshes are isolated points and disconnected parts resulting from occluded acquisition and simplifications. Typically, tree trunks are not reconstructed entirely and cause "floating" treetops. Besides, degenerated faces without area in form of points and edges might occur.

We consider the face as the core entity for our association and transfer operation as well as for the semantic segmentation of meshes (see Chapters 5 and 6). Naturally, the face count is significantly smaller than the number of points or pixels. This fact is beneficial for any follow-on processing. For instance, the number of faces is ~10 times smaller than the number of points for H3D (see Chapter 4). The 2.5D mesh increases the discrepancy in instance count. The number of faces is ~42 times smaller.

Considering the face as core entity, we alternatively express the mesh $\mathcal{M}$ as a set of Centers of Gravity (CoGs): $\mathcal{M}_{\mathrm{PC}} = CoG(\mathcal{F})$. We refer to this representation as *CoG cloud* where each face is represented by its CoG. Representing the mesh as a kind of PC significantly simplifies the mesh handling due to i) evading any problem related to non-manifolds and ii) the reduced memory footprint since only the CoGs have to be processed. Notwithstanding, the CoG cloud differs from an ordinary PC due to the known relation of $\mathcal{M}_{\mathrm{PC}}$ and $\mathcal{M}_G$. The main advantage of $\mathcal{M}_{\mathrm{PC}}$ is to mimic ordinary PCs while still benefitting from mesh-based features like the availability of high-resolution texture. By these means, additional attributes can be attached easily to each face/CoG and classifiers that have been designed for PCs originally can be used. Besides, visualization tools optimized for large-scale PCs can be used to visualize the CoG clouds with a comparably small entity count. Figure 2.1 shows the two representations of the mesh side-by-side by the example of the 2.5D mesh of Hessigheim (see Chapter 4). Whereas $\mathcal{M}_G$ uses the sophisticated *obj* format, the CoG cloud $\mathcal{M}_{\mathrm{PC}}$ is encoded as a plain *ASCII* file enabling simple data handling and data exchange. The *obj* format consists of i) the *obj* file itself describing the geometry and referring to (ii) the *mtl* file encoding material properties which links to the (iii) texture atlas that provides textural information (*jpgs*).

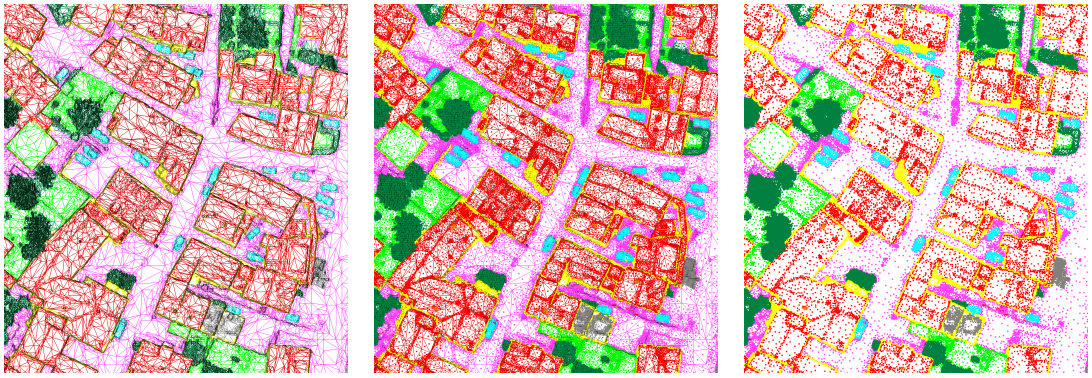


Figure 2.1: The two representations of the mesh modality by the example of the semantically segmented 2.5D mesh of Hessigheim (see Chapter 4): The graph structure as wireframe (*left*) and the CoG cloud (*right*). The *center* shows the combination of both.

Chapter 3

Related Work

The semantic segmentation of geospatial data has become a standard task in photogrammetry and remote sensing. Generally, the semantic analysis deals with various representations such as (RGB-D) imagery, voxels, PCs, and meshes. In particular, the geospatial community put a big effort into the semantic segmentation of large-scale (LiDAR) PCs over the past few years. In contrast, mesh interpretation has hardly been explored by the community, although recent years show increasing interest in meshed 3D models – particularly for applications like smart city models (Boussaha et al., 2018). Virtual city models are util for gaming, urban planning, navigation, disaster management, and others. Commonly, virtual 3D cities are encoded as structured boundary representation following CityGML, the Open Geospatial Consortium (OGC) standard with multiple pre-defined Levels of Detail (LODs) (Gröger et al., 2012; Kolbe et al., 2005). In comparison, the graph-like meshes are a default data representation in computer graphics and computer vision. For instance, Google Earth uses the textured mesh to represent geospatial data in 3D enriched with geotagged information (Google, 2001). However, only a few works deal with the semantic segmentation of real-world geospatial meshes (Gao et al., 2022; Knott and Groenendijk, 2021; Rouhani et al., 2017; Tutzauer et al., 2019). Feng et al. (2018) state that research focuses on volumetric grids, multi-view representations, and PCs due to the complexity of meshed models (see Chapter 2). Another reason is that annotated urban meshes have not been publicly available before 2021. The recent rise of annotated urban mesh data and advances in sophisticated end-to-end learning models may unleash the potential of automated mesh interpretation.

The mesh modality plays an important role in the presented thesis. However, automated mesh generation is not the scope of our work. We are aware that the meshed 3D reconstruction of the real world is a non-trivial task and an active research field. Conventionally, surface meshes are derived from PCs in post-processing. Linsen and Prautzsch (2001) compare local and global triangulation methods. Kazhdan et al. (2006) present the Poisson surface reconstruction, which poses the surface reconstruction task as a spatial Poisson problem based on oriented PCs. Labatut et al. (2007) and Vu et al. (2009) investigate the meshed 3D reconstruction from images of large-scale urban scenes. Bláha et al. (2017) present an approach that jointly refines the geometry and the semantic segmentation. Recently, end-to-end learning schemes have emerged as a counterweight to conventional reconstruction methods that stringently exploit geometric constraints between 3D space and acquired imagery. Neural Radiance Fields (NeRFs) are a famous representative of such DL-based methods that have been originally intended to synthesize novel views of 3D scenes encoded in a fully-connected non-convolutional neural network (Mildenhall et al., 2020). However, these approaches struggle with reconstruction accuracy for the time being and are yet not reasonably applicable to complex, large-scale urban scenes due to memory constraints. Han et al. (2019) provide a survey of the recent developments in image-based 3D reconstruction and texturing using DL techniques.

Section 3.1 reviews important works for automated 3D scene interpretation – both for PCs and meshes. ML classifiers, particularly DL methods, rely on a huge amount of annotated data, wherefore GT generation is a crucial task. We compactly review available GT of geospatial data and GT generation strategies in Section 3.2.

3.1 Semantic Segmentation of Geospatial 3D Data

The automatic semantic segmentation picked up speed quickly with the success of DL approaches in image space. The successful implementation of AlexNet, a Convolutional Neural Network (CNN), on a Graphics Processing Unit (GPU) marks a turning point in image classification (Krizhevsky et al., 2012). CNNs apply convolutional operations on the regular and uniform image data (see Chapter 2) and learn domain knowledge from the data itself. Since then, data-driven end-to-end learning approaches dominate image interpretation and have replaced traditional ML approaches that exploit handcrafted features encoding the domain knowledge (Garcia-Garcia et al., 2017; Minaee et al., 2020). Particularly, fine-tailored CNN-inspired architectures like the Fully Convolutional Network (Long et al., 2014), the U-Net (Ronneberger et al., 2015), and the SegNet (Badrinarayanan et al., 2017) revolutionized the semantic segmentation of images. Image analysis and its advances can be seen as the cradle of semantic segmentation setting the trends for 3D scene interpretation. However, the transition from the regularly and uniformly structured images to irregular and non-uniform 3D structures like PCs and meshes is not trivial as CNNs require grid-like input data.

Therefore, many approaches by-pass the structural demands by using a transitional representation, e.g., by projecting 3D data into image space. The circuit to image space allows using well-performing CNN architectures and back-projecting the results to 3D space (He and Upcroft, 2013; Wu et al., 2015).

The computer vision community worked hard to adapt DL architectures to the non-grid-like structure of PCs and meshes, making the circuit to transitional representations obsolete. PointNet and PointNet++ are the first networks that directly operate on 3D PCs (Qi et al., 2017a,b). Griffiths and Boehm (2019) review the current state-of-the-art DL architectures for processing unstructured PCs. Ahmed et al. (2018) show advances of DL on different 3D data representations. They discuss representation-specific challenges and highlight differences between Euclidean and non-Euclidean data. The emerging field of geometric DL extends basic DL operations to non-Euclidean domains such as graphs and manifolds in order to integrate topological information (Bronstein et al., 2016). MeshCNN is a representative of end-to-end learning approaches that manage to process 3D meshes directly (Hanocka et al., 2019). However, networks adapted to non-grid-like 3D structures are often studied on data covering single objects or indoor scenes (see Section 3.2.2).

In recent years, conventional feature-driven ML may have lost popularity due to the rise of in-vogue end-to-end learning approaches, although expressive feature engineering is feasible in 3D space as the underlying geometry can be accessed directly. While DL approaches do not require handcrafted features, they rely on a large amount of training data and tedious fine-tuning of hyperparameters (Gao et al., 2021). In contrast, traditional ML depends on handcrafted features and, therefore, provides better interpretability and explainability. Designing and selecting appropriate features depends on domain knowledge (Duda et al., 2001) and is subject to research. Features may embrace engineered quantities and sensor-intrinsic attributes such as pulse characteristics (Eitel et al., 2016). A wide variety of features proved to be well-performing for the semantic segmentation of ALS PCs (Chehata et al., 2009; Mallet et al., 2011; Weinmann et al., 2015). The used features describe the local geometry or refine measured sensor data. Several works have revealed that the height above ground is the most expressive feature (Chehata et al., 2009; Guo et al., 2011; Kölle et al., 2019). Similarly, geometric contextual features can also be derived on the mesh (George et al., 2017; Kalogerakis et al., 2010; Rouhani et al., 2017). Schindler (2012) states that any form of contextuality improves the performance of a point-based classifier. Notably, contextual features implicitly smooth the predicted labels and reduce the need for computationally expensive spatial regularizations.

Random Forest (RF) models (Breiman, 2001) offer a good trade-off between accuracy and efficiency and have been deployed successfully in several works dealing with semantic scene interpretation (Chehata et al., 2009; Guo et al., 2011; Kölle et al., 2019; Weinmann et al., 2015).

In sum, automated scene interpretation leverages feature-driven strategies, end-to-end learning schemes, and a combination of both – regardless of the underlying modality. In the following, we review advances in semantic segmentation of 3D PCs in more detail (see Section 3.1.1). Subsequentially, we present the state of the art for semantic segmentation of meshes (see Section 3.1.2). In particular, we highlight feature engineering as we use RF models as off-the-shelf classifiers within this work, relying on established features. Finally, we show approaches that fuse data from multiple modalities (see Section 3.1.3).

3.1.1 Semantic Segmentation of Point Clouds

Meanwhile, there are several review papers that present the state of the art for semantic segmentation of PCs focusing on end-to-end learning approaches (Bello et al., 2020; Griffiths and Boehm, 2019; Liu et al., 2019; Xie et al., 2020; Zhang et al., 2019). The sheer amount emphasizes the current interest of academia in the semantic segmentation of PCs.

End-to-End Learning Approaches. The unstructured nature of 3D PCs prevents applying CNNs directly to them. Therefore, PCs are commonly structured into grid-like 3D or 2D representations by voxelization or multi-view rendering, respectively, to overcome the non-gridded design.

Several works migrate the PC to voxel space and train a supervised classifier. The predicted labels for the voxels are transferred to all contained points afterward (Hackel et al., 2016; Huang and You, 2016; Maturana and Scherer, 2015; Roynard et al., 2018). Voxelization comes along with memory overhead and suffers from discretization and the curse of dimensionality. Therefore, much effort is put into networks that use sparse 3D convolutions (Graham et al., 2017). This approach has been successfully applied to urban PCs (Schmohl and Sörgel, 2019). Similarly, OctNets (Riegler et al., 2017; Tatarchenko et al., 2017; Wang et al., 2017) and Kd-Networks (Klokov and Lempitsky, 2017) use octrees and k-d trees to accelerate sparse 3D convolutions by replacing hashed regular grids.

Detouring via image space, multi-view approaches leverage established networks dedicated to the semantic segmentation of images. The per-pixel predictions are back-projected to 3D space (Boulch et al., 2017; He and Upcroft, 2013; Lawin et al., 2017). The grid-like proxy enables the use of CNNs but comes along with information loss due to discretization, occlusions, and projection.

Özdemir et al. (2021) downsample aerial PCs with a voxel-based filter and generate 2D patches for the downsampled points, gathering sensor-intrinsic and handcrafted features of neighboring points. They unfold the 2D patches into 3D volumes and apply 2D and 3D CNN classifiers to the respective patches to semantically segment aerial PCs.

The rise of PointNet and its hierarchical successor PointNet++ constitutes a milestone in semantic PC segmentation since they operate directly on unstructured 3D PCs for the first time (Qi et al., 2017a,b). The gist of PointNet is to use a symmetric function to be independent of set permutation. The PC is encoded by a global feature vector, which is attached to each per-point feature vector. Operating only on a global scale, PointNet misses to leverage local context. Its extension PointNet++ hierarchically applies PointNets to the iteratively subsampled PC and, hence, operates on several scales. This procedure mimics hierarchical feature learning with increased contextual information similar to CNNs in image space. Winiwarter et al. (2019) successfully applied PointNet++ to geospatial PCs. Meanwhile, several dedicated architectures have been developed that operate directly on the PCs (Boulch, 2019; Hu et al., 2020; Thomas et al., 2019).

PCs do not provide topological information per se. Therefore, many works resort PCs into graphs aiming at better capturing the topology and the local geometry of the underlying 3D shapes (Landrieu and Simonovsky 2017, Chang et al. 2018, Ali Khan et al. 2020).

Feature-Driven Learning Approaches. The adaption of (geometric) DL methods contributed to substantial progress in semantic segmentation of PCs in the last decade. Xie et al. (2020) review semantic segmentation of PCs comparing DL and traditional ML approaches.

Several works highlight the semantic segmentation of geospatial LiDAR data exploiting a wide spectrum of handcrafted features (Blomley and Weinmann, 2017; Chehata et al., 2009; Gross and Thoennessen, 2006; Guo et al., 2011; Hackel et al., 2016; Jutzi and Gross, 2009; Landrieu et al., 2017; Mallet et al., 2011; Niemeyer et al., 2014; Weinmann et al., 2015). Commonly, features are grouped into eigenvalue-based features, plane-based features, height-based features, echo-based features, full-waveform-based features, and color features (Chehata et al., 2009; Guo et al., 2011; Mallet et al., 2011). Eigenvalue-based features are the most common among them to describe the shape of the local neighborhood and the local point distribution. The Principal Component Analysis (PCA) derives the eigenvalues from the covariance matrix of coordinates (Maas and Vosselman, 1999) also known as structure tensor.

A multitude of eigenvalue-based features – sometimes called covariance features – have been designed. Pauly et al. (2003) approximate the *change of curvature* for PCs as eigenvalue-based quantity and use the *sum of eigenvalues* to describe the total variation of the local neighborhood. West et al. (2004) introduce comprehensive refinements of the eigenvalues like *linearity*, *planarity*, *sphericity*, *anisotropy*, *omnivariance*, and *eigenentropy*. Mallet et al. (2011) adapt these quantities by normalizing the eigenvalues prior to calculating the features. Demantké et al. (2012) derive a *verticality* score based on the eigenvector corresponding to the smallest eigenvalue.

The structure tensor depends on the definition of a local neighborhood. The so-called neighborhood recovery is a crucial research topic in feature-driven ML. Conventionally, the local neighborhood of a point is defined by a) a spatially fixed-sized sphere with radius r (Brodu and Lague, 2012; Gross and Thoennessen, 2006; Jutzi and Gross, 2009), b) a spatially fixed-sized cylinder with radius r (Chehata et al., 2009; Filin and Pfeifer, 2005; Guo et al., 2011), c) a numerically fixed-sized vicinity of k closest points (Linsen and Prautzsch, 2001; Weinmann et al., 2013), or d) a locally adaptive vicinity (Demantké et al., 2011; Weinmann et al., 2015, 2014). Whereas variants a) – c) require to set the scale parameters (i.e., r or k) empirically or heuristically, the locally adaptive methods automatically determine the optimal neighborhood size. Combinations of various neighborhood types are implemented in (Blomley and Weinmann, 2017; Mallet et al., 2011; Niemeyer et al., 2014; Weinmann et al., 2015).

To reduce the importance of scale parameter choice while increasing contextuality, the developed features are commonly evaluated on multiple scales (Blomley and Weinmann, 2017; Brodu and Lague, 2012; Jutzi and Gross, 2009; Niemeyer et al., 2014). Calculating multi-scale features increases the computational burden. Therefore, Weinmann et al. (2015) use k-d trees to determine the local neighborhood of k closest points efficiently. Moreover, they select the most relevant features by analyzing their importance and thus reduce the feature calculation effort. Hackel et al. (2016) generate a scale pyramid via voxel-grid filter for high-density terrestrial LiDAR data. The pyramid downsamples the point density with increasing scale keeping a constant number of Nearest Neighbors (NNs) for each scale. Apart from contextual features, Niemeyer et al. (2014) avoid noisy results utilizing a statistical context model, namely a Conditional Random Field (CRF). Likewise, Landrieu et al. (2017) extend the point-based pipeline by structured regularization, a graph-based contextual strategy to avoid noisy predictions. Vosselman et al. (2017) segment the data by a combination of segmentation approaches and subsequentially perform the semantic segmentation with handcrafted features.

3.1.2 Semantic Segmentation of Meshes

The mesh structure is more complex than a PC due to the topological nature. Its semantic segmentation, particularly for large-scale urban scenes, is still in its infancy.

End-to-End Learning Approaches. By analogy to semantic segmentation of PCs, common approaches for semantic mesh segmentation make a circuit to transitional modalities to take advantage of established (DL) classifiers. Common transitional spaces are voxels (Wu et al., 2015), 2D image space (Huang et al., 2019; Kalogerakis et al., 2017; Su et al., 2015), and sampled PCs (Gao et al., 2021). Gao et al. (2021) sample a PC from their publicly available meshed benchmark data set and compare several 3D classifiers regarding training costs and performance. They compare a feature-driven segment-based RF with state-of-the-art DL architectures for the semantic segmentation of PCs such as PointNet, PointNet++, SPG, KPConv, RandLA-Net (Hu et al., 2020; Landrieu and Simonovsky, 2017; Qi et al., 2017a,b; Thomas et al., 2019). They found that the segment-based RF outperforms most DL-based classifiers while exhibiting significantly faster training time (including feature calculation) although operating on the Central Processing Unit (CPU).

So-called geometric DL avoids the transition to another representation. Instead, geometric DL approaches operate directly on the irregular and non-uniform mesh, exploiting the mesh topology. To the best of our knowledge, almost all approaches that directly operate on meshes merely proved their effectiveness with non-urban data sets covering indoor scenes or single objects. Besides, these meshes do not necessarily provide textural information – one of the main properties of MVS-generated meshes. Fundamental research in the field of DL on triangular meshes is done by (Masci et al., 2015; Tatarchenko et al., 2018; Verma et al.,

2018). Precisely, they address the mesh structure and introduce specific convolution operations on the mesh vertices incorporating the encoded topology. Chang et al. (2018) propose the Structure-Aware Convolutional Neural Network (SACNN), a neural network that uses generalized filters, which aggregate local inputs of different learnable topological structures. By that, SACNNs work with both Euclidean and non-Euclidean data. Generally, graph convolutions operate on edge relationships of local patches and are invariant to the shape of these patches in Euclidean space. In contrast, Euclidean convolutions capture deformations of the surface patches. Therefore, Schult et al. (2020) propose the DualConvMesh-Net that combines geodesic and Euclidean convolutions on 3D meshes. Geodesic convolutions utilize the underlying mesh structure and help to separate spatially adjacent but disconnected surfaces. To complement this, Euclidean convolutions establish connections between nearby disconnected surfaces. Feng et al. (2018) propose the MeshNet, a classification network that considers the face as the core entity encoding the face and its adjacent faces. The fixed-sized vicinity overcomes the irregularity of mesh data, enabling the processing with convolutional blocks. The non-uniformity is handled by providing structural features for each face. However, they do not encode textural features and test the network only on the small-scale ModelNet data set (see Section 3.2.2).

Similarly, Hanocka et al. (2019) present a CNN-inspired network that handles the irregularity and non-uniformity of meshes. The gist of their proposed MeshCNN is to consider the edge as the central unit generating a fixed-sized neighborhood of 5 edges (including the current edge itself). The regularized neighborhood definition facilitates the application of convolutional operations. MeshCNN mimics the conventional CNN architecture by sequential execution of convolutions and pooling operations. The specialized layers operate on the edges and leverage the intrinsic topological information.

The key of neural networks for semantic segmentation is the encoder-decoder principle, where data is encoded in a few parameters and then upsampled to the original resolution. However, the connectivity of the non-uniform and non-regular mesh makes pooling and unpooling difficult. For instance, MeshCNN considers the edge as the core unit to utilize the edge collapse algorithm for downsampling (Hoppe, 1997). Generally, the encoded topological information complicates the application of DL and increases the computational burden during the representation learning. The majority of DL-based approaches proved their capacity on non-geospatial data sets (Chang et al., 2018; Feng et al., 2018; Hanocka et al., 2019; Hu et al., 2021b; Qiao et al., 2019; Schult et al., 2020).

Knott and Groenendijk (2021) are the first who successfully adopted a well-performing DL approach from the computer vision community. They achieved to apply MeshCNN to a real-world mesh covering the village of Vaihingen, Germany, at the expense of a reduced class catalog and blocked processing, splitting the mesh into smaller parts of constant edge count. To that end, they had to restructure the spatial mesh tiling to use the local vicinity information reasonably. Besides, they enhance feature encoding with radiometric features improving performance significantly.

To reduce the computational complexity, Gao et al. (2022) present PSSNet, a two-stage framework, that first over-segments the mesh into homogeneous segments in terms of geometric and photometric characteristics and subsequentially applies a graph convolutional network. The classifier encodes local geometric and photometric features per segment and spatial inter-segment relationships.

Feature-Driven Learning Approaches. Graph neural networks exploit the topology in an end-to-end fashion, but require manifolds as input. Automatically reconstructed meshes may not fulfill this constraint due to imperfections (e.g., flying tree crowns), complicating the application of these networks (Knott and Groenendijk, 2021). Besides, they feature high computational complexity, which demands some sort of complexity reduction of large-scale meshes (Gao et al., 2022; Knott and Groenendijk, 2021). Conventional ML approaches circumvent these issues by engineering features in advance. The separation of feature calculation and classifier training decreases the complexity of the training process and copes with non-manifolds.

Kalogerakis et al. (2010) lists a plethora of commonly used features for mesh reasoning. Other geometric features are presented by Günther (1989), Douros and Buxton (2002), Dyn et al. (2001), and Verdie et al. (2015). Zhou (2018) provides a code base for geometric processing of meshes, including feature calculation.

The computer vision community exploited geometric properties of their small-scale meshes (like Princeton Shape Benchmark) for years, mainly aiming at mesh segmentation instead of semantic segmentation. As

texture is not an inherent characteristic of these meshed models, the vast majority of standard features covers geometric and topological attributes. To incorporate texture, Valentin et al. (2013) extract textural features from images and train a cascaded boosting classifier along with geometric features extracted from the mesh. Taime et al. (2018) perform semantic segmentation solely based on color information and compare the results to purely geometry-driven results.

In contrast, photogrammetric meshes provide texture "out-of-the box". Therefore, geometric and radiometric features can be calculated per vertex, edge, and face. Rouhani et al. (2017) derive geometric and photometric features from a photogrammetric mesh, gather faces into so-called superfacets, and train a RF. For each superfacet, the class and its similarity to the neighboring superfacet are predicted. The similarity information is used to assign weights to the pairwise potential of a Markov Random Field (MRF) and accounts for contextual information between the classes.

In (Tutzauer et al., 2019), we represent the mesh by its CoGs. The CoG cloud reduces the complexity and simplifies further processing. Furthermore, it is robust against non-manifolds occurring in automatically generated meshes. The CoG cloud differs from a common PC as it benefits from inherent mesh properties like the availability of high-resolution texture and adjacency knowledge. We adopted and enhanced the multi-branch 1D CNN (George et al., 2017), which is a mixture of feature-engineering and feature-learning, and applied it to a 2.5D mesh. For each face, a multi-scale feature vector is computed and serves as input for the 1D CNN. Different scales are fed to respective branches. We compare the achieved results to a RF.

One of the main differences between meshes and PCs is the availability of high-resolution texture. Therefore, we have investigated the feature importance and show that color information is beneficial for the semantic segmentation of 2.5D meshes. More precisely, in Laupheimer et al. (2020a), we attest that per-face color (i.e., texture) outperforms per-vertex color by evaluating several radiometric feature qualities. We achieved to double the performance gain by utilizing color information of the entire face instead of per-vertex only. However, we also show the inherent limitations of texture due to occlusions, the absence of imagery, and the quality of the geometric reconstruction. Besides, in (Laupheimer et al., 2020a), we took up the concept of CoG cloud and compared the performance of RF to PointNet++ for semantic segmentation of a 2.5D mesh. The previous research on 2.5D meshes presented in (Laupheimer et al., 2020a; Tutzauer et al., 2019) is the foundation for this thesis dealing with 3D meshes. These results will not be discussed in this thesis but have paved the way for it.

3.1.3 Multi-Modal Fusion and Semantic Segmentation

In the light of the recent hybridization trend, multi-modal processing is expected to improve automated 3D reconstruction and scene interpretation. Multi-modal fusion has the potential to generate more complete and more detailed mapping products due to the complementary properties of different modalities. In turn, the increased geometric quality is likely to improve the semantic analysis.

Glira et al. (2019) propose a methodology to jointly orientate imagery and ALS data, which simplifies the fusion of the derived and co-registered PCs as a side effect. Recently, there are software solutions that enable data fusion of multi-modal PCs and refine the fusion on the mesh-level. For instance, software `SURE` by nFrames (Rothermel et al., 2012) produces meshes as generated from LiDAR and MVS (see Figure 4.4).

Multi-modality confronts structural differences across modalities as discussed in Chapter 1 and huge amounts of data due to redundant multi-modal capturing. Therefore, many works dealing with semantic segmentation involve only one representation and perform individual reasoning for 2D or 3D data. Dedicated architectures tailored to the structural properties of the underlying modality operate solely in 2D or 3D (see Section 3.1.1 and Section 3.1.2). Generally, these approaches keep multi-modality at a minimum, scratching only the surface of multi-modal processing. For instance, the fusion of imagery and ALS data on the point-level is commonly confined to the colorization of ALS points or its derived voxel space.

In most cases, multi-modality is a means to an end that allows abusing well-performing classifiers of another modality. However, these approaches involve only one modality in the narrow sense. To give an example, the well-established and fast semantic segmentation of images is mostly abused as a proxy for 3D scene analysis (Boulch et al., 2017; He and Upcroft, 2013; Kalogerakis et al., 2017; Lawin et al., 2017; Su

et al., 2015). Theoretically, any quantity can be projected into image space, adding another channel to the image. In practice, the curse of dimensionality prevents the projection of an arbitrary number of quantities. Often, 3D scenes are projected into RGB-D images encoding the 3D structure as view-dependent depth (Chang et al., 2017; Dai et al., 2017; Hua et al., 2016).

The detour via image space leverages well-performing and effective semantic image segmentation methods but comes along with information loss due to projection and discretization. Therefore, other approaches invert the information transfer by augmenting 3D entities instead of 2D image space. By these means, 3D entities are equipped with features derived from high-resolution images and fed into a 3D classifier. In this way, the semantic segmentation jointly leverages the geometrical 3D structure and high-resolution imagery instead of simply back-projecting per-pixel predictions to 3D space (Chiang et al., 2019; Dai and Nießner, 2018; Jaritz et al., 2019). The 3D space may be represented as voxel (Dai and Nießner, 2018), PC (Jaritz et al., 2019), or mesh vertices (Chiang et al., 2019). Deploying trainable networks for the 2D/3D feature extraction converts the entire approach into an end-to-end 2D-3D learning pipeline. These multi-modal approaches are known as unidirectional 2D-3D learning, as information is first aggregated on one modality and then processed there by a dedicated architecture. Hu et al. (2021c) enhance this sequential processing of the unidirectional 2D-3D learning approaches with parallelized and entangled 2D and 3D classifiers. They propose the Bidirectional Projection Network (BPNet) that simultaneously trains 2D and 3D classifiers next to each other. Two dedicated encoder-decoder networks for 2D and 3D space are the core of their symmetric network. The encoder-decoder networks are coupled with the intermediate levels of the respective decoders. Thereby, 2D and 3D space benefit from each other during training and generate predictions at a stroke in both spaces during inference.

3.2 Ground Truth Generation and Ground Truth Availability

Gathering and annotating real-world data is tedious, time-consuming, and error-prone. Human annotators manually assign an application-dependent semantic label to each entity during the annotation process (conventional GT generation). Commonly, experts annotate the domain-specific data with task-specific labels, which makes the annotation process an expensive affair. In particular, DL methods require a huge amount of annotated data. However, GT data is not publicly available for many tasks, hindering the training of ML/DL systems. The lack of annotated data in combination with data-hungry DL methods demands efficient strategies for GT generation. Section 3.2.1 reviews several strategies that boost the annotation process. In Section 3.2.2, we present publicly available GT of 3D data.

3.2.1 Ground Truth Generation Strategies

To cure the annotation gap and to reduce the manual involvement of experts, several strategies have emerged in recent years next to the conventional labeling, such as a) synthetic GT generation, b) GT generation deploying AL, c) semi-automatic annotation, and d) GT generation by recruiting non-experts via paid crowdsourcing or gamification incentives.

a) Synthetic Ground Truth Generation. Synthetic data may replace or complement real-world data, where data acquisition is not feasible due to economic or logistic constraints. Procedural modeling facilitates the artificial creation of 3D scenes, which may be enhanced with an arbitrary detail degree regarding textures and contextual scene embedding (e.g., trees, light poles). Synthetic data has the advantage that co-registration issues and inter-modal discrepancies do not exist by design. Therefore, huge data volumes can be generated with pixel-perfect annotations by rendering artificial 3D scenes. Hence, procedural modeling enables the fast and simple GT generation at an arbitrary scale for multiple modalities. Besides, the artificial scenes provide users with a simple yet fast way of switching between textures, reflectance properties, lighting conditions, viewing poses, and many others. Conceptually, synthetic data can avoid class imbalance which is a common issue of annotated real-world data. However, pure synthetic data cannot mimic the complexity

and versatility of the real world. Their appearance is limited to the diversity of the artificial design. Cabezas et al. (2015) provide the SynthCity data set consisting of synthetic urban scenes created with the procedural software CityEngine by ESRI (2021). Liu et al. (2021) present a simulator and a large-scale urban scene data set based on Unreal Engine 4 by Epic Games (2019) and AirSim (Shah et al., 2017). The so-called UrbanScene3D contains artificial and real-world scenes on different scales. Likewise, Biljecki et al. (2016) provide a procedural modeling engine for synthesizing buildings in multiple LODs in CityGML. Similarly, Fedorova et al. (2021) present a generator that synthesizes buildings as meshed models in *OBJ* format. Apart from these urban-level and street-level scenes, there are several synthetic indoor scenes publicly available such as Structured3D (Zheng et al., 2020), SceneNet (Handa et al., 2015), and InteriorNet (Li et al., 2018). Zhou et al. (2020) review available synthetic 3D data sets of outdoor and indoor scenes.

The availability of surface representations – regardless of being real-world reconstructions or artificial designs – allows for simulating PCs. For instance, HELIOS++ mimics measured LiDAR PCs based on surface representations by simulating laser scanning on different platforms (Winiwarter et al., 2022). Uggla and Horemuz (2021) procedurally generate synthetic PCs by sampling the 3D mesh models.

b) Ground Truth Generation Deploying Active Learning. Settles (2009) gives an overview of the AL literature. AL binds the annotation to the classifier training. The iterative AL process aims to detect those instances which are sufficient for training a ML model. Hence, AL reduces the number of instances that have to be annotated. Learning from a small pool of labeled points has been proven successful for semantic segmentation of PCs (Kölle et al., 2021c; Li and Pfeifer, 2019; Lin et al., 2020). Kölle et al. (2021c) test various selection strategies in their AL pipeline to detect the most informative points for the annotation process. The AL pipeline proposed in this work is based on their AL pipeline and their findings regarding the selection strategies (see Section 6.1.3).

c) Semi-Automatic Ground Truth Generation. Semi-automatic annotation is an umbrella term for several techniques that avoid labeling each entity manually. Gao et al. (2021) propose a semi-automatic mesh labeling framework that incrementally trains a classifier that is abused as a pre-labeler to steer and alleviate the manual annotation process. The interactive approach processes the mesh data tile-wisely. First, one of the tiles is annotated manually to initialize the classifier. Its predictions on another tile are checked and – if necessary – relabeled by a human operator in a refinement step. The refined labels augment the training set, and the classifier is retrained. This process is repeated until all mesh tiles have been exhausted. By these means, the classifier improves per iteration, and manual interaction reduces in later refinement steps.

In our work, we propagate manually assigned labels from one modality to other modalities (see Section 6.1.2). This approach requires known inter-modal entity relationships (see Chapter 5). This sort of semi-automatic annotation limits the tedious and time-consuming label work to a single modality. Hence, the semi-automatic labeling boosts the annotation of various representations for large-scale data. Specifically, we utilize the publicly available annotated LiDAR PC of V3D to generate a labeled mesh. Likewise, we propagate labels from the annotated LiDAR PC of H3D to the respective mesh. However, in that case, the manual annotations have to be provided by ourselves (see Section 6.1.1).

d) Ground Truth Generation by Recruiting Non-Experts. Establishing massive annotated data corpora is expensive when employing experts. So-called paid crowdsourcing recruits non-expert workers through respective platforms like *Amazon Mechanical Turk* or *Microworkers* (Buhrmester et al., 2011; Hirth et al., 2011). Paid crowdsourcing reduces labeling costs since non-experts take the place of expensive experts. For instance, the approximately 14 million images of the famous *ImageNet* data set have been annotated by non-experts (Deng et al., 2009). Similarly, the ModelNet data set providing annotated CAD models has been labeled by deploying paid crowdsourcing. Wu et al. (2015) hired human workers on Amazon Mechanical Turk to manually assign each CAD model to specified categories.

Commonly, crowdworkers are not familiar with the kind of data they should label. The unfamiliarity holds particularly for geospatial data featuring unfamiliar representations and perspectives like oblique airborne

images or PCs. Therefore, the annotation process must be steered by simple and comprehensible interfaces. Herfort et al. (2018), Walter and Soergel (2018) and Kölle et al. (2021c) demonstrate that crowdworkers are generally capable of interpreting 3D scenes represented by ALS PCs. Kölle et al. (2020) exploit the combination of crowdsourcing and AL to minimize manual labeling effort.

The annotation task is tedious work. Ramirez et al. (2019) present a virtual reality tool that gamifies the manual labeling of meshes and PCs to make tedious annotation work more fun.

Apart from dedicated annotation tasks, there are also volunteered crowdsourcing projects aiming at a more generic information supply. For example, OpenStreetMap provides a map database filled and maintained entirely by volunteers (OpenStreetMap contributors, 2017).

3.2.2 Ground Truth Availability of 3D Data

Several works review available GT of RGB-D, multi-view, volumetric, point and mesh representations for different applications and use-cases (Garcia-Garcia et al., 2017; Griffiths and Boehm, 2019; Minaee et al., 2020; Xie et al., 2020; Ye et al., 2020). In the following, we highlight publicly available GT data focusing on 3D PCs and meshes covering urban scenes. Table 3.1 lists currently available labeled urban data sets acquired by an airborne system (plane or UAV) categorized by the sensor, modality, and extension. The table proves that the majority of annotated urban data are LiDAR PCs. Meshes have been a widely overlooked topic in automated scene interpretation for years. The first annotated urban meshes arose in 2021. Besides, we see that available annotations are limited to a single representation. Our H3D data set is an exception that provides annotated 3D PCs and meshes of multiple epochs. The data sets used throughout the thesis are described in more detail in Chapter 4.

Table 3.1: Comparison of publicly available annotated real-world urban data sets acquired by an airborne platform (plane or UAV). #Entities denotes the number of points and faces for PCs and meshes, respectively. We do not claim the list to be exhaustive, but it denotes the most famous and most commonly used data sets to the best of our knowledge.
*AHN and H3D cover several epochs of varying entity counts.

Name	Platform	Sensor	Modality	Annotation	Classes	#Entities	Area	Year
V3D	plane	LiDAR	PC	manually	9	1.2 M	0.6 km^2	2014
AHN	plane	LiDAR	PC	manually	4	*	41.5 km^2	2019
DublinCity	plane	LiDAR	PC	manually	13	260 M	2 km^2	2019
DALES	plane	LiDAR	PC	manually	9	505 M	10 km^2	2020
LASDU	plane	LiDAR	PC	manually	5	3 M	1 km^2	2020
Campus3D	UAV	Camera	PC	manually	14	937 M	1.6 km^2	2020
SensatUrban	UAV	Camera	PC	manually	31	2847 M	7.6 km^2	2020
Swiss3DCities	UAV	Camera	PC	manually	5	226 M	2.7 km^2	2020
SUM-Helsinki	plane	Camera	Mesh	semi-automatic	6	19 M	4 km^2	2021
H3D (ours)	UAV	LiDAR + Camera	PC + Mesh	semi-automatic	11	*	0.2 km^2	2021

Ground Truth Availability for Point Clouds. The vast majority of annotated geospatial data are LiDAR PCs captured with ALS, Mobile Laser Scanning (MLS), and Terrestrial Laser Scanning (TLS). Apart from outdoor scenes, there are also annotated indoor scenes which are provided and investigated by the computer vision community (Armeni et al., 2016; Uy et al., 2019).

There are several ALS data sets covering outdoor urban-level PCs such as V3D (Niemeyer et al., 2014), DublinCity (Zolanvari et al., 2019), DALES (Varney et al., 2020), LASDU (Ye et al., 2020), and H3D (Kölle et al., 2021a). While the previously listed data sets feature rather generic classes, Wichmann et al. (2018) present the RoofN3D benchmark dedicated to learning different roof types in ALS data. Actueel Hoogtebestand Nederland (AHN) (2021) provides annotated ALS data at a country-scale covering the Netherlands.

TLS is well-suited to capture urban scenes from the street level. Due to the static and close-distance acquisition, TLS data generally features higher point densities than ALS PCs. Hackel et al. (2017) provide the Semantic3D.net benchmark consisting of large-scale annotated TLS data of diverse urban scenes. Matrone et al. (2020) present the ArCH data set dedicated to cultural heritage scenarios.

Semantic segmentation is also a hot topic in the context of autonomous driving. A plentitude of annotated roadway-level MLS data meet the needs for this active research field: Oakland 3D (Munoz et al., 2009), Paris-rue-Madame (Serna et al., 2014), iQumulus (Vallet et al., 2015), Paris-Lille-3D (Roynard et al., 2018), KITTI (Geiger et al., 2012), SemanticKITTI (Behley et al., 2019), and Toronto3D (Tan et al., 2020).

Recently, there have emerged annotated MVS clouds like Campus3D (Li et al., 2020), SensatUrban (Hu et al., 2021a), and Swiss3DCities (Can et al., 2021). They all feature pointwise annotation of UAV data. Whereas the first covers the campus of the National University of Singapore, the others cover two UK cities and three Swiss cities, respectively.

Ground Truth Availability for Surface Reconstructions. At the commencement date of this thesis, annotated large-scale real-world 3D meshes of urban scenes have not been accessible. The pioneering work from Rouhani et al. (2017) is based on annotated MVS mesh data that is not publicly available. Recently, two annotated meshes have been provided to the community to foster semantic segmentation of urban meshes: SUM-Helsinki (Gao et al., 2021) and H3D (Kölle et al., 2021a). The latter is a result of this thesis.

The SUM data set provides an annotated large-scale photogrammetric mesh covering the city of Helsinki, Finland. The MVS mesh has been reconstructed from aerial oblique images with ContextCapture (Bentley, 2017). The data has been labeled semi-automatically by human-verified predictions achieved with an iteratively trained classifier (see Section 3.2.1).

H3D meshes have been annotated semi-automatically, too, but without abusing a trained classifier as prelabeler. Instead, we followed an inter-modal approach and propagated manual annotations from the PCs to the respective meshes. In comparison to the pure photogrammetric mesh of Helsinki, the H3D mesh has been reconstructed from aerial oblique images and LiDAR points with `SURE` (Rothermel et al., 2012). Besides, H3D provides multiple epochs featuring a more fine-grained class catalog than the SUM-Helsinki benchmark. Throughout this thesis, we utilize the epoch March 2018, which is detailed in Chapter 4. At the time of writing, H3D is the only geospatial data set that provides annotated multi-temporal PCs and meshes covering outdoor urban scenes.

To the best of our knowledge, SUM-Helsinki and H3D are the only available annotated meshed models that cover urban scenes. 3DCityDB makes urban-scale city models publicly available in the CityGML format (Yao et al., 2018). Riemenschneider (2014) provides street-level meshes reconstructed from mobile MVS data within the ETHZ CVL RueMonge 2014 data set. Selvaraju et al. (2021) provides annotated building instances in the BuildingNet data set.

Meshes have been a default data representation in the domain of computer vision for decades. Therefore, the development of (DL) mesh classifiers is mainly driven by that community (see Section 3.1). However, they typically deal with 3D shapes covering indoor scenes or single objects represented as meshes or CAD models. Famous data sets covering indoor scenes are SceneNN (Hua et al., 2016), ScanNet (Dai et al., 2017), Matterport3D (Chang et al., 2017), and the 2D-3D-S data set (Armeni et al., 2017). The famous Princeton Shape Benchmark (Shilane et al., 2004) and ShapeNet (Chang et al., 2015) provide annotated objects represented as meshes. However, CAD models are more common on the object scale. Annotated CAD models are provided in the ModelNet (Wu et al., 2015), ObjectNet3D (Xiang et al., 2016), ABC (Koch et al., 2019), PartNet (Mo et al., 2019), and MCB (Kim et al., 2020).

3.3 Demarcation of Existing Research

The 2D-3D methods are characterized by integrating 3D data and imagery. However, simple projection may map occluded points into imagery. Generally, point projection suffers from occlusions and time-shifts during acquisition (Peters and Brenner, 2019). Therefore, we interpose the mesh as the visibility model to tackle the

occlusion problem. Besides, the linking of 3D modalities takes second place in existing works. The related information exchange between them is commonly handled by simple interpolation.

We aim at an explicit linking of pixels, points, and faces (backbone infrastructure) enabling a flexible information exchange considering structural discrepancies across modalities of real-world data (see Chapter 5). Our work differs from existing research since it explicitly aims at a holistic multi-modal data fusion of imagery, PCs, and meshes. One could say that we extend common 2D-3D methods, which strictly connect one 3D modality with images, to a 2D-3D-3D method, which also explicitly connects modalities in 3D space. In this thesis, we focus on the 3D-3D linking of urban PCs and meshes and prove its superiority over simple cross-modality interpolation (see Figure 6.4).

Apart from the linking and the provision of information, we investigate the semantic segmentation of meshes (see Chapter 6). As discussed in Chapter 1, the mesh representation inherently facilitates multi-modal data fusion. However, the mesh is widely overlooked for semantic analyses yet. We enhance the textured mesh with features derived from the PC by leveraging the proposed linking method. To this end, the semantic segmentation uses the 3D geometrical structure of both 3D modalities and texture from image data while benefitting from mesh characteristics, i.e., fewer entities and redundant-free texture. Accessing the lightweight texture atlas facilitates the efficient handling of imagery. In contrast, 2D-3D approaches ignore the condensed information encoded in the texture atlas and tap into multi-view imagery.

We perform an extensive ablation study for the deployed features using a rather simple RF classifier. Precisely, we utilize the CoG representation presented in (Tutzauer et al., 2019) and mimic the feature-driven pipeline presented by Weinmann et al. (2015) enhanced with multi-modal entity linking. Most similar to our work is the research presented in (Gao et al., 2022; Rouhani et al., 2017) operating on large-scale urban scenes, too. Compared to their research, we incorporate more features from multiple modalities and consider a more complex semantic segmentation task with more classes. Whereas Rouhani et al. (2017) apply computationally expensive regularization via MRF, Gao et al. (2022) first segment the mesh to achieve spatially smooth results. In contrast, we use contextual features at multiple scales as implicit smoothing operation (Schindler, 2012).

To emphasize, the goal of our work is not the development of a new classifier or an ablation study of classifiers. Instead, we focus on the information provision and analyze the utility of shared information across modalities. As a product, we publish the multi-modal H3D benchmark (Kölle et al., 2021a). The community is encouraged to test any developed classifier on the benchmark. By these means, we implicitly foster a sustainable ablation study of classifiers that is not limited to the period of this thesis and keeps up with the time.

Chapter 4

Data

We utilize the publicly available ISPRS benchmark data sets Vaihingen 3D (Cramer, 2010; Niemeyer et al., 2014) and Hessigheim 3D (Epoch March 2018, Kölle et al. (2021a)). The used data sets differ significantly in radiometry, geometry, resolution/density, and the covered annotated area. Besides, the manual annotations are assigned based on distinct class catalogs. Please note the publicly available mesh labels of the Hessigheim 3D (H3D) epochs are an immediate result of this thesis and have been generated in a semi-automatic manner (see Section 6.1.2). The manual point annotations have been transferred from the LiDAR PC to the mesh leveraging the proposed multi-modal linking (see Chapter 5).

Although being already captured in 2008, Vaihingen 3D (V3D) may still be representative of large-scale country-wide mapping with moderate GSD of some centimeters and a considerable time shift between ALS and image data collection. On the contratry, H3D provides extremely high-resolution data with (at least partially) synchronous data capture from a hybrid sensor system and is representative of data collection at small-scale complex built-up areas.

For this reason, the geometric and radiometric quality of H3D outperforms V3D. Accordingly, the number of entities and the memory footprint are higher for H3D although covering a smaller area.

For both data sets, LiDAR data and imagery are registered and georeferenced via strip adjustment and AAT respectively. Hence, the LiDAR PCs and imagery (and MVS clouds respectively) are co-registered implicitly through the georeference. However, the implicit co-registration between these clouds may not be optimal due to independent registration procedures. We utilize the Iterative Closest Point (ICP) algorithm for fine-registering the LiDAR PCs and MVS clouds to reduce co-registration discrepancies between the PCs for both data sets. Specifically, we determine the transformation parameters by deploying ICP on the MVS cloud and a filtered subset of the annotated LiDAR PC. The subset consists of points belonging to flat classes such as *Impervious Surface* and *Roof*. The determined transformation is applied to the entire LiDAR PC afterwards. The MVS cloud is used as reference in the ICP algorithm because it is natively co-registered with imagery and is input to the meshing algorithm. Therefore, aligning the LiDAR PC with the MVS cloud results in fine-registered modalities PC, mesh, and imagery for both data sets.

We generated textured and tiled meshes with the `SURE` software from nFrames for both data sets utilizing either imagery only or both data sources (Rothermel et al., 2012). By these means, the data sets feature imagery, a 3D LiDAR PC, and a textured mesh – each split into train set $\mathcal{T}$ and test set $\mathcal{E}$.

Figure 4.1 shows the workflow for data preparation highlighting the modality-specific and hybrid processing steps. Table 4.1 offers a compact overview of the three modalities for V3D and H3D.

For both project areas, we utilize Digital Terrain Models (DTMs) with grid widths of $1\,\mathrm{m} \times 1\,\mathrm{m}$ to derive height-based features (see Section 6.2). In case of Vaihingen, the DTM is derived by software *SCOP++* (Pfeifer et al., 2001) based on the publicly available LiDAR data filtered by non-ground classes. The Hessigheim DTM is provided by the State Office for Spatial Information and Land Development Baden-Wuerttemberg (Landesamt für Geoinformation und Landentwicklung Baden-Württemberg).

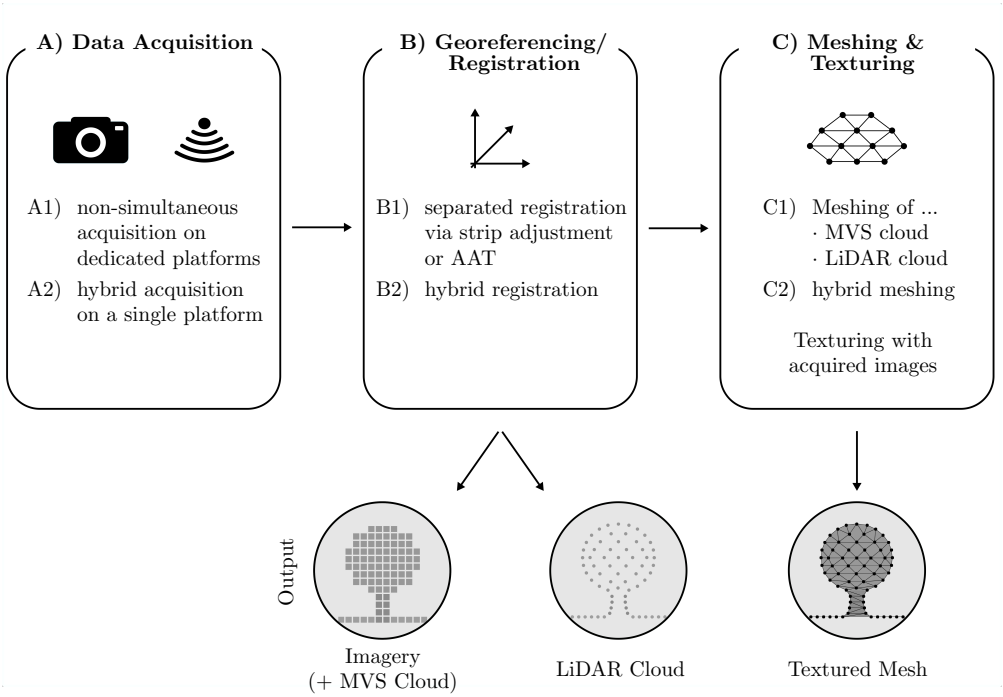


Figure 4.1: Flow chart of data preparation. The output in form of oriented imagery, registered LiDAR data, and textured mesh is input for the inter-modal entity linking and the automatic semantic segmentation. The data is prepared in three steps: A) Data acquisition with camera and LiDAR sensor. B) Individual or joint georeferencing/registration. C) Meshing and texturing of the derived georeferenced mesh.

Table 4.1: Properties of data sets V3D and H3D used throughout this thesis (see Figures 4.2 and 4.5). The project area is given approximately by the agglomeration of mesh tiles that intersect with the labeled LiDAR cloud. The face count is adapted to the overlapping LiDAR cloud. The class catalogs are given in Figures 4.2 and 4.5 respectively.

Data Set	Imagery	Point Cloud	Mesh
V3D 875 m × 700 m	$GSD_{nadir} = 8$ cm 15 images @ 14 430 px × 9420 px	4–8 points/m^2 1.2 M points	16 tiles 970 k faces source: MVS_{nadir}
Manual Annotations	✗	✓	✗
H3D 700 m × 250 m	$GSD_{oblique} = 2.5$ cm 1979 images @ 6000 px × 4000 px $GSD_{nadir} = 3.7$ mm 524 images @ 11 608 px × 8708 px	400–800 points/m^2 125.6 M points	74 tiles 12.6 M faces source: $MVS_{oblique}$ + ALS
Manual Annotations	✗	✓	✗

4.1 Vaihingen 3D (V3D)

V3D covers Vaihingen an der Enz, Germany, and is a well-established benchmark data set for ALS classification in photogrammetry and remote sensing. The official contest submission was closed in 2018. Therefore, the labels of the test set are publicly available, too.

The manually annotated LiDAR PC is provided along with return number, number of returns, and intensity (Niemeyer et al., 2014). The 1.2 M annotated points are assigned to the following 9 classes: *Power Line*,

Low Vegetation, *Impervious Surface*, *Car*, *Fence/Hedge*, *Roof*, *Facade*, *Shrub*, *Tree*. We neglected class *Power Line* for our experiments as it consists of few points and is hardly recovered in the mesh.

The data has been captured asynchronously from airplanes equipped with an Ultracam-X camera and an ALS 50 LiDAR scanner (Cramer, 2010). The time-shift between nadir imagery (GSD = 8 cm) and ALS acquisition (4–8 points/m^2) is several weeks.

We generated a textured mesh for V3D based on the provided nadir images. We opted for a pure photogrammetric mesh since the time-shift of imagery and ALS data is roughly one month. The nadir images impose a demanding challenge for reconstructing and texturing vertical faces which will hamper an explicit linking of faces and points. However, the ALS data maps only scarcely the vertical faces, too. The purely photogrammetric character of the V3D mesh trivializes the entity linking between imagery and mesh.

The total MVS mesh covers an area of 1750 m × 2000 m and is partitioned into tiles of size 175 m × 175 m. The mesh tiles that overlap with the labeled PC cover roughly 875 m × 700 m. In total, the mesh consists of 17.5 M faces where roughly 5 % of faces overlap with the labeled cloud. Figure 4.2 shows the annotated ALS cloud and the overlapping mesh tiles in textured fashion. As can be seen by the attached overview map, the labeled cloud is close to the fringe area of the reconstructed mesh where the geometric quality of the mesh is limited inevitably (see Figure 6.5). Despite the time shift, we utilize the textured mesh to colorize the ALS PC like described in Section 4.2.

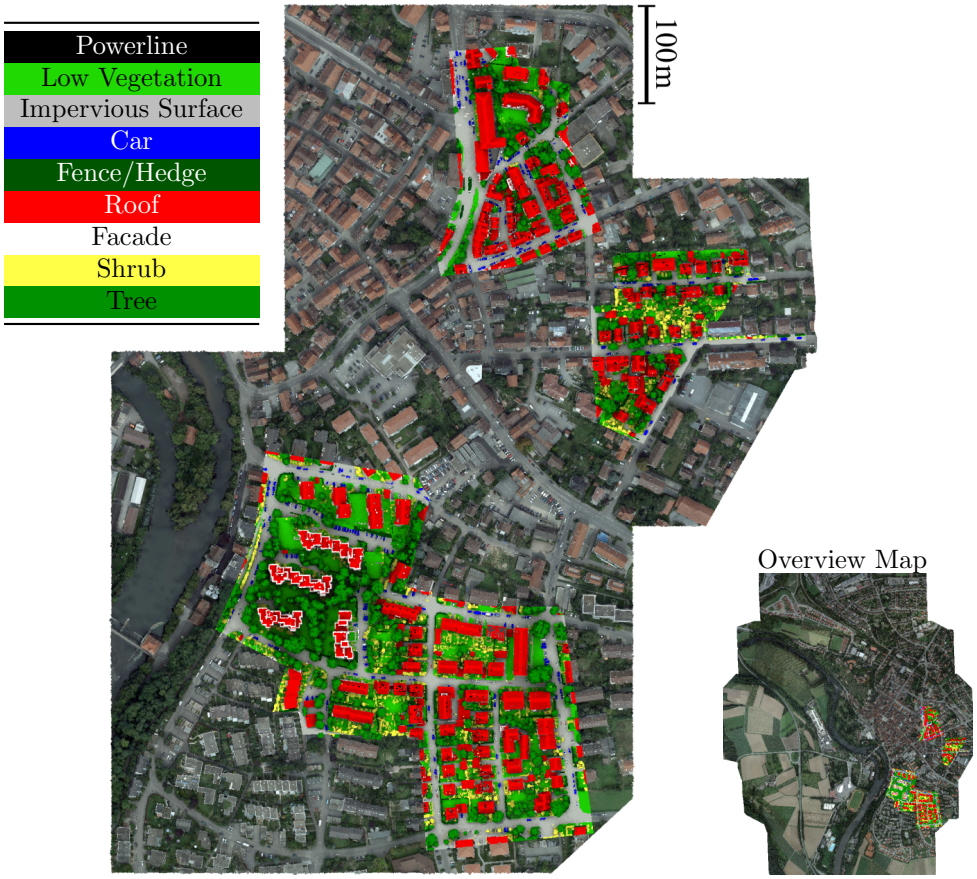


Figure 4.2: Top view of V3D depicting the annotated LiDAR point cloud and the respective overlapping mesh tiles in textured fashion (*center*). The annotated ALS data is color-coded utilizing the labeling scheme depicted on the *left*. The southern part forms the train set $\mathcal{T}$. The disjoint parts in the north form the test data $\mathcal{E}$. The overview map on the *lower right* shows the extension of the entire MVS mesh.

4.2 Hessigheim 3D (H3D)

The original purpose of H3D aims at the deformation monitoring of the ship lock and its surrounding in Hessigheim, Germany (Cramer et al., 2018). The data has been captured by two UAVs in March 2018, i.e., in leaf-off season. Oblique imagery (GSD = 2.5 cm) has been acquired simultaneously along with ALS data (400–800 points/m^2) from a RiCopter multi-copter platform equipped with a Riegl VUX-1LR LiDAR scanner and two Sony Alpha 6000 oblique cameras. Nadir images (GSD = 3.7 mm) have been acquired from the CopterSystems CS-SQ8 copter with a PhaseOne iXU-RS 1000 camera. The time-shift between the data capturing with these two drones ranges from several hours to one day.

The LiDAR data has been manually annotated using a fine-grained class catalog, which is based on the V3D class catalog but refined due to H3D's higher point density (Kölle et al., 2019, 2021a; Laupheimer et al., 2020a). This allows differentiating more details than in V3D. We added classes *Urban Furniture*, *Gravel/Soil*, *Vertical Surface* (e.g., found at the ship lock), and *Chimney* totaling up to 11 classes. *Urban Furniture* features a high intra-class variance, since essentially it serves as quasi-class *Other*. Labels of the test set remain sealed because H3D is an active benchmark provided and administrated by the Institute for Photogrammetry, University of Stuttgart.

Apart from the XYZ coordinates and labels, the LiDAR data provides inherent per-point attributes such as the return number, number of returns, and reflectance. Unlike the publicly available V3D PC, we additionally provide color information in form of RGB tuples for all LiDAR points. The per-point RGB tuples are extracted from the mesh texture by nearest neighbor transfer. The textured mesh allows to artificially sample an arbitrarily dense PC with fine-grained colorization, which is input to the interpolation. The artificially sampled PC benefits from the superior geometric reconstruction quality of the hybrid mesh (compared to a pure MVS mesh, see Figure 4.4) for urban canyons and semi-transparent objects such as vegetation. The nearest neighbor interpolation in 3D space is a simple approximation of an occlusion-aware projection of 3D LiDAR points to image space. Figure 4.3 compares the mesh-driven colorization with a interpolation from the MVS PC. The ALS PC colorization based on the respective MVS PC suffers from interpolation artifacs due to geometric discrepancies in the PCs. The most obvious manifestation of this phenomenon is visible for vegetational objects, mainly trees. In this way, the interpolated RGB tuples from the MVS cloud cannot compete with colors transferred from the textured mesh. The mesh-driven PC colorization appears more realistic and visually appealing, supporting the human viewer. Therefore, we opted for this technique to colorize the PC of the H3D benchmark (see Section 6.2).

(a) MVS PC *(b)* Textured mesh *(c)* ALS PC colored by MVS PC *(d)* ALS PC colored by mesh

Figure 4.3: Colorization of the ALS cloud of H3D (*c* and *d*) with RGB tuples as interpolated from the MVS PC (*a*) or the hybrid mesh (*b*). The *sky-blue* ellipses highlight differences in geometry and colorization.

The textured mesh is generated in a hybrid manner by fusing the simultaneously acquired ALS data and oblique imagery in software `SURE` (Rothermel et al., 2012) to benefit from their complementary properties. The fusion of both data sources results in a more complete mesh compared to a mesh derived from images only. For instance, urban canyons are hard to reconstruct from imagery (due to required visibility in at least two images), but their reconstruction works smoothly for LiDAR data (where one received echo is already

sufficient). The oblique images ensure a proper and realistic texturing of vertical faces (e.g., facades). Figure 4.4 compares the geometric qualities of a pure MVS mesh and a hybrid mesh as generated from imagery and LiDAR data.

Figure 4.4: Comparison of the geometric qualities of a pure MVS mesh (*left*) and a hybrid mesh as generated from imagery and LiDAR data (*right*). Both meshes have been generated with `SURE` (Rothermel et al., 2012). The *sky-blue* ellipses highlight differences in the geometry.

The mesh is split into tiles of 50 m × 50 m covering a total area of approximately 700 m × 250 m. Figure 4.5 shows the annotated ALS cloud and the overlapping mesh tiles in textured fashion.

For comparative purposes, we generated pure MVS meshes with `Metashape` software (Metashape, 2021) and `SURE` (see Figure 6.6). Additionally, we generated a 2.5D mesh with `SURE` for the Hessigheim data.

To validate quantitatively the semi-automatic annotation of the 3D mesh and the entity linking in general, we have manually annotated a subset of the test split of the H3D hybrid mesh (see Section 6.1.2). The respective part is marked in sky-blue in Figure 4.5. Besides, we have manually annotated the 3 M faces of the 2.5D mesh. We have used the 2.5D mesh to learn about efficient manual annotation (see Section 6.1.1) and to gain first insights into automated semantic segmentation of meshes (Laupheimer et al., 2020b; Tutzauer et al., 2019). In particular, we tested the robustness of the linking method deploying asymmetric thresholding on the 2.5D mesh (see Section 5.1, Laupheimer et al. (2020b)). The gained knowledge is inserted into the processing of 3D meshes with roughly 4.2 times more faces.

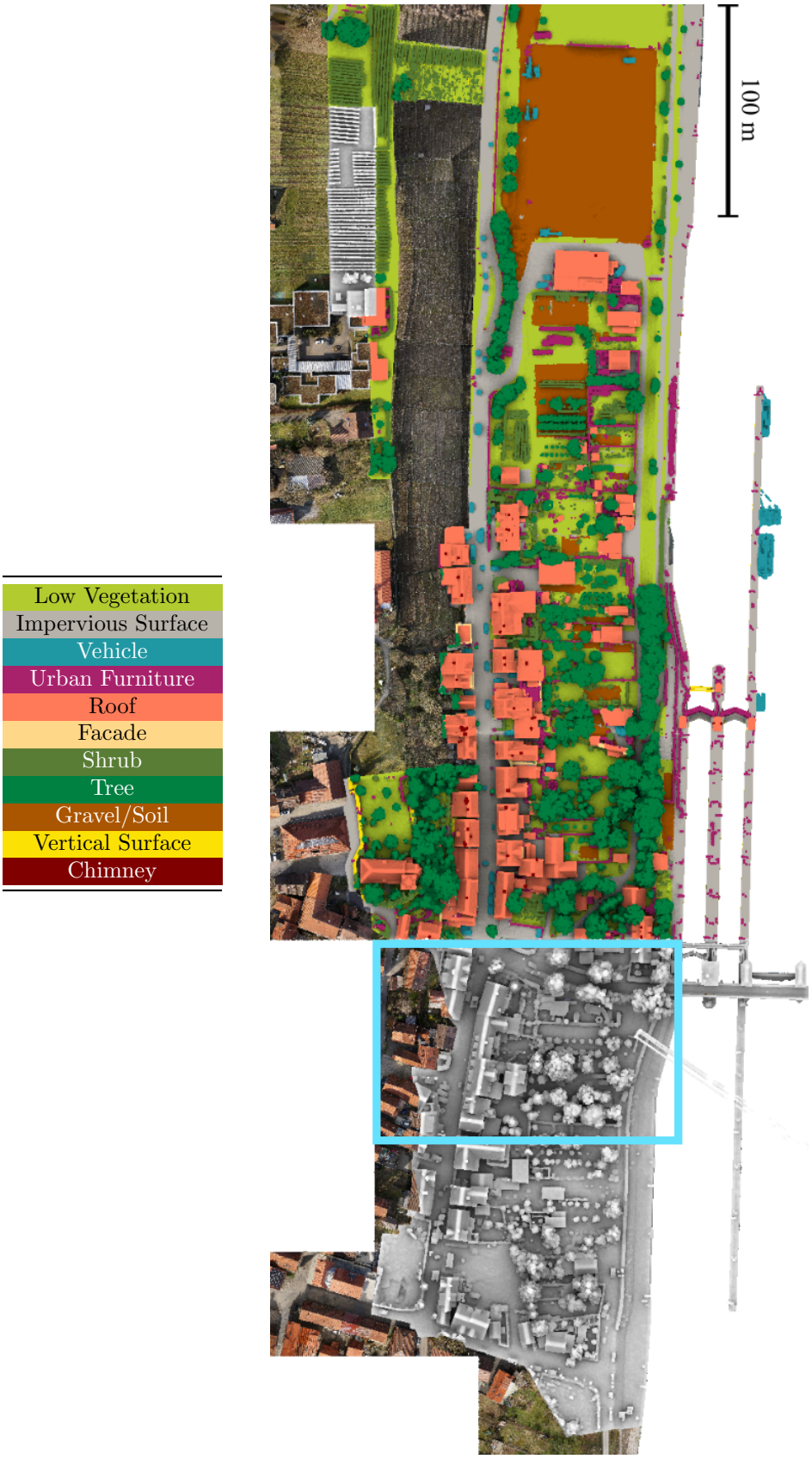


Figure 4.5: Top view of H3D depicting the annotated LiDAR point cloud and the respective overlapping mesh tiles in textured fashion (*right*). The annotated ALS data is color-coded utilizing the labeling scheme depicted on the *left*. The northern part forms the train set $\mathcal{T}$. The disjoint parts in the northwest and south form the test set $\mathcal{E}$. Annotations of the test set are kept sealed as it is an active benchmark at the time of writing (marked in *gray*). The *sky-blue* box indicates the part of the test set that has also been annotated manually.

Chapter 5

Multi-Modal Entity Linking

We aim for an explicit holistic linking of the three common data types in the domain of photogrammetry and remote sensing: imagery, PC, and mesh. The backbone of the proposed conflation consists of two geometry-driven inter-modal linking parts: (a) Point Cloud Mesh Association (PCMA) which links faces and points (see Section 5.1) and (b) Image Mesh Association (ImgMA) which links faces and pixels in multiple images (see Section 5.2). Linking both association mechanisms yields to (c) Point Cloud Image Association (PCImgA) (see Section 5.3). The PCImgA establishes a connection between points and imagery via the mesh as a mediator. Loosely speaking, the mesh acts as adhesive tape between parts (a) and (b) resulting in (c). Point visibility is implicitly given through the mesh. The top of Table 5.1 illustrates the total multi-modal association mechanism with iconic pictograms.

The established connections between the entities across the distinct representations enable an information transfer that allows features and labels to be shared. We refer to this as *Juggling with Representations*. Table 5.1 compactly lists the information transfer operations depending on information type (feature or label) and transfer direction for each part of the entire association mechanism. The linking is the key for a joint semantic segmentation (see Chapter 6).

The presented methodology is a generic approach that works with any photographic image following the central perspective and PC data regardless of the acquisition platform and PC type. However, we focus on the linking of aerial RGB imagery and ALS PCs along with the respective textured mesh (see Chapter 4) and test the pipeline on airborne data (see Chapter 6). Section 5.4 discusses the preconditions and limitations of the association mechanism in detail.

Concerning the scalability of the proposed association approach, we process data tile-wise in a parallelized fashion. The tile-wise association significantly reduces the memory footprint of each association step while accelerating the association due to parallel processing. We couple the information transfer with the linking methodology for the same reason. Therefore, the association inherently is a label transfer and feature calculation tool (aggregation of features). However, the explicit linking information will be stored and can be accessed later on, too.

5.1 Point Cloud Mesh Association (PCMA)

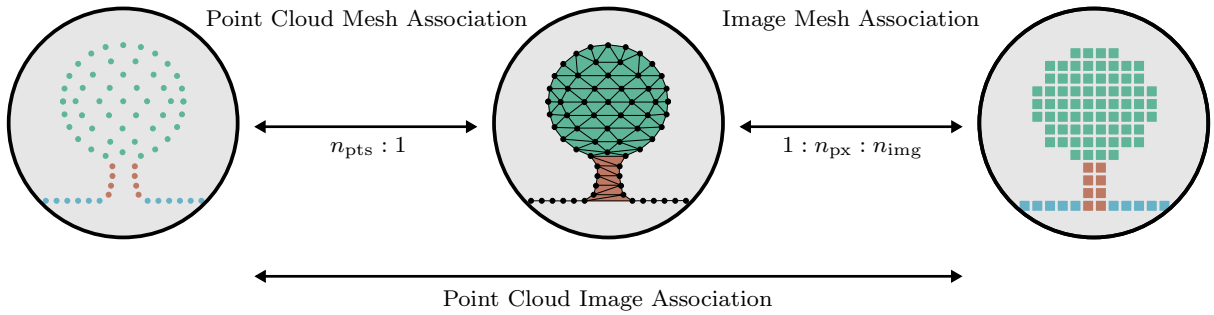
The Point Cloud Mesh Association (PCMA) explicitly links faces and points in a face-centered geometry-driven approach. Each face f (represented by its CoG) is associated with n_{pts} points that represent the same surface by following the three steps:

(i) clipping of the point cloud $\mathcal{P}$ to a spherical vicinity of the CoG ($\mathcal{P} \rightarrow \mathcal{P}'_f$),

(ii) filtering of *out-of-face* points ($\mathcal{P}'_f \rightarrow \mathcal{P}''_f$),

(iii) filtering of *off-the-face* points ($\mathcal{P}''_f \rightarrow \mathcal{P}'''_f$).

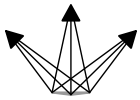
Table 5.1: Overview of the proposed method that links imagery, point cloud (PC), and mesh via inter-modal subprocesses a) Point Cloud Mesh Association (PCMA), b) Image Mesh Association (ImgMA), and c) Point Cloud Image Association (PCImgA). For each association mechanism, the transfer operations depend on the information type (feature or label) and the transfer direction. The pictograms on the right depict the linking of the respective entities. PCImgA provides two association modes: implicit and explicit linking (see Section 5.3). The relationship of implicit PCImgA is described by $n_{\text{pts}} : 1 : n_{\text{px}} : n_{\text{img}}$ ($n_{\text{pts/px/img}}$: number of points/pixels/images). The explicit version creates pixel-point pairs for each image. The points may be covered by multiple images.



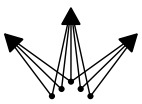
(a) Point Cloud Mesh Association (PCMA)

	Mesh $\mapsto$ PC ($1 : n_{\text{pts}}$)	PC $\mapsto$ Mesh ($n_{\text{pts}} : 1$)	
Feature Transfer	Copy Value	Median Aggregation	
Label Transfer	Copy Value	Majority Vote	

(b) Image Mesh Association (ImgMA)

	Mesh $\mapsto$ Img ($1 : n_{\text{px}} : n_{\text{img}}$)	Img $\mapsto$ Mesh ($n_{\text{img}} : n_{\text{px}} : 1$)	
Feature Transfer	Copy Value	Median Aggregation	
Label Transfer	Copy Value	Majority Vote	

(c) Point Cloud Image Association (PCImgA)

	PC $\mapsto$ Img	Img $\mapsto$ PC	
Feature Transfer	Copy Value	Median Aggregation	
Label Transfer	Copy Value	Majority Vote	

Out-of-face points are points that are not enclosed by the face borders when projected orthogonally onto the face plane. *Off-the-face points* do not coincide with the face plane, i.e., they are below or above the face surface. A manually set threshold θ decides whether a point coincides with a face or not. Both point types are not mutually exclusive and exist due to the co-registration residuals, simplification during the meshing, the representation type differences as discussed in Chapter 1, and geometry differences (e.g., in case of 2.5D mesh geometry or due to asynchronous data acquisition).

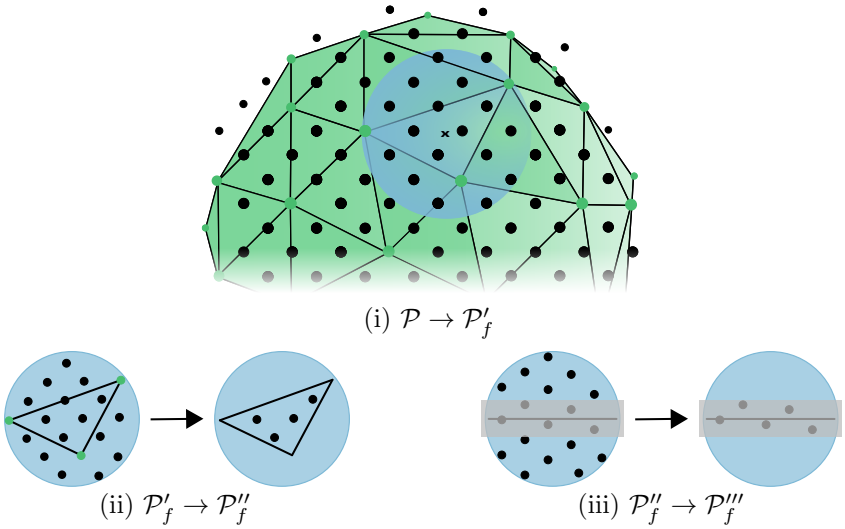


Figure 5.1: Sequential association steps (i) – (iii) of the PCMA to filter point cloud $\mathcal{P}$ for each face f: $\mathcal{P} \rightarrow \mathcal{P}'_f \rightarrow \mathcal{P}''_f \rightarrow \mathcal{P}'''_f$. *(i)*: Clipping of $\mathcal{P}$ (*black dots*) to the spherical vicinity (*blue*) of the considered face. Its CoG is marked with a *black cross*. The mesh surface and its vertices are depicted in *green*. *(ii)*: Filtering of *out-of-face points* based on the clipping result $\mathcal{P}'_f$ (orthogonal view concerning the triangular face surface depicted by its edges marked in *black*). *(iii)*: Filtering of *off-the-face points* from $\mathcal{P}''_f$ (side view with respect to the triangular face). The face is depicted as *black* line. The threshold band is marked in *gray*.

At first, we roughly reduce the search space for each face in order to accelerate the association. To this end, we build a kD tree for the PC (tile) and query the built tree with CoGs of all faces. Thereby, we detect all PC points within distance r for each face (ball query). The *association sphere* contains all association candidates. The query parameter r is set in dependence of the manually set association threshold θ and the maximum distance $t_{\max}$ of the CoG to the respective face vertices. $t_{\max}$ equals $\frac{2}{3}$ of the largest median of the triangular face. Geometrically, r is set to the length of the hypotenuse of the triangle as defined by θ and $t_{\max}$. In simple terms, the query parameter r is set to the minimum distance that guarantees the manually set threshold θ to be effective for the entire face while enclosing the entire face (see Figure 5.2). The ball query delivers a subset of points $\mathcal{P}'_f$ for each face f, which may contain *off-the-face points* and *out-of-face points*. Figure 5.3 showcases scenarios where radius r is set too small (i.e., r equals either $t_{\max}$ or θ) resulting in a ball query that erroneously filters possible association candidates.

Second, we filter *out-of-face points* by neglecting points of $\mathcal{P}'_f$ whose orthogonal projections on the face plane are not enclosed by the face outline (Laupheimer et al., 2020b). We filter these points with the help of barycentric coordinates, which enable a fast execution based on vector algebra. For each face, we parameterize the respective face vertices $\mathbf{v}_i$ and the potentially associated LiDAR points with barycentric coordinates. Each face is defined by its vertices $\mathbf{v}_1 = (1, 0, 0)$, $\mathbf{v}_2 = (0, 1, 0)$ and $\mathbf{v}_3 = (0, 0, 1)$. Any point $\mathbf{p}$ on the face plane is parameterized with barycentric coordinates b_i by $\mathbf{p} = b_1\mathbf{v}_1 + b_2\mathbf{v}_2 + b_3\mathbf{v}_3$. For details on barycentric coordinates, we refer the interested reader to (Ericson, 2005). A point $\mathbf{p}$ is inside the face (or on the edges of the face) if $0 \leq b_i \leq 1\ \forall i$. We can easily exclude points on the edges, e.g., vertices $\mathbf{v}_i$, by replacing operator $\leq$ with $<$. Figure 5.4 shows a face parameterized in barycentric coordinates. The green marked

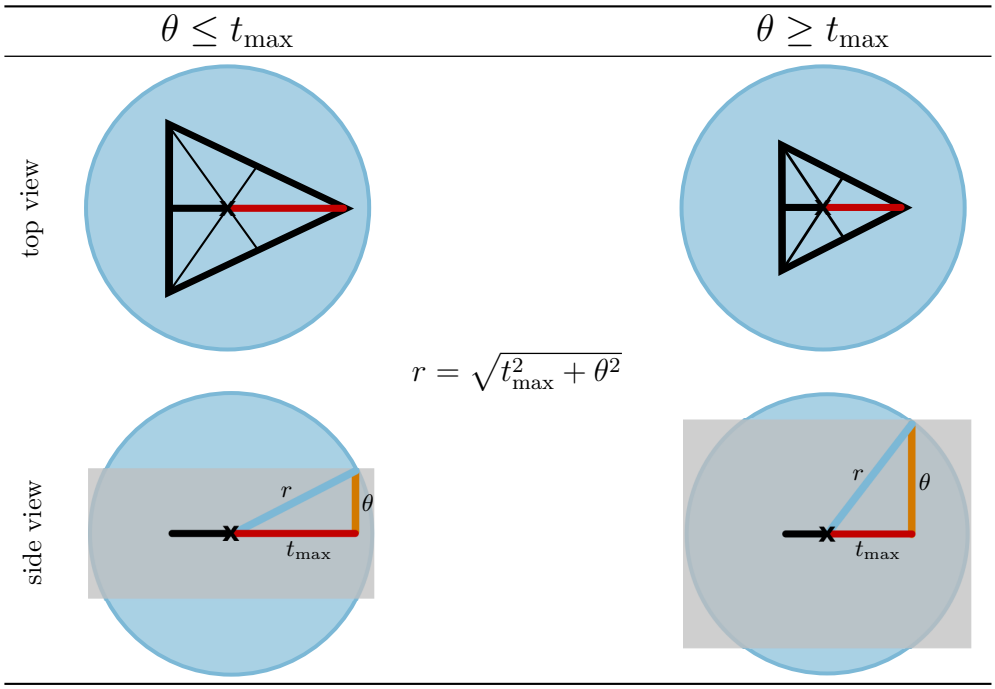


Figure 5.2: Definition of the association sphere with radius r (*blue*) for the PC clipping (step (i) of PCMA) shown as a) top view with respect to the triangular face surface (*top*) and b) side view with respect to the largest median of the triangular face (*bottom*). The maximum distance $t_{\max}$ (*red*) from the CoG (*black cross*) to face vertices equals $\frac{2}{3}$ of the maximum median of the triangular face. The threshold θ is marked in *orange*; the respective threshold band is marked in *gray*. The figure depicts both scenarios: $\theta \leq t_{\max}$ (*left*) and $\theta \geq t_{\max}$ (*right*).
The defined radius guarantees to enclose the entire face by enclosing the maximum distance $t_{\max}$ (see top view). At the same time, r avoids prefiltering points by enclosing the threshold band across the entire face (see side view). Figure 5.3 showcases scenarios where r is set too small and already prefilters possible association candidates.

area shows the inside of the face that fulfills the above condition (with operator $<$). We exclude points on the edges since technically, they belong to two adjacent faces and cannot be assigned unambiguously. By these means, we sacrifice edge points in the linking process to avoid ambiguity in the subsequent information transfer. As a side-effect, the few neglected points accelerate the information transfer. Having in mind the many-to-one relationship between points and face, neglecting those points is of minor importance for the subsequent multi-modal feature vector composition on the mesh (see Section 6.5.1). Since the filter conditions work only on points in the plane, we orthogonally project all points of $\mathcal{P}'_f$ onto the plane before filtering. The orthogonal projection is only relevant to the association mechanism. The original PC remains unchanged. Visually, the result of (ii) is the intersection of the association sphere and an infinite triangular prism as defined by the face and its normal vector. We refer to this as *sphere-limited association prism* enveloping the remaining subset of points $\mathcal{P}''_f$. $\mathcal{P}''_f$ equals $\mathcal{P}'_f$ filtered by *out-of-face points*.

Finally, we filter the remaining *off-the-face points*. For this purpose, we inspect the distance for each remaining point and its orthogonal projection on the face, which has already been calculated for filtering of *out-of-face points*. If the distance exceeds a chosen association threshold θ, the point is considered to be an *off-the-face point* and is not associated with the face. Visually, the sphere-limited association prism is pruned by θ ending up with a *threshold-limited association prism* ($r > \theta$, see Figure 5.2).

Since we have to compensate several discrepancies between PC and mesh, we use a more sophisticated adaptive thresholding with an arbitrary user-defined number of filter levels $n_l \in \mathbb{N}$. Each level l consists of two independent thresholds θ_l^+ and θ_l^-. The absolute threshold values increase with increasing level. Thresholds θ_l^+ and θ_l^- limit the orthogonal distance to the face plane in the normal direction or the opposite

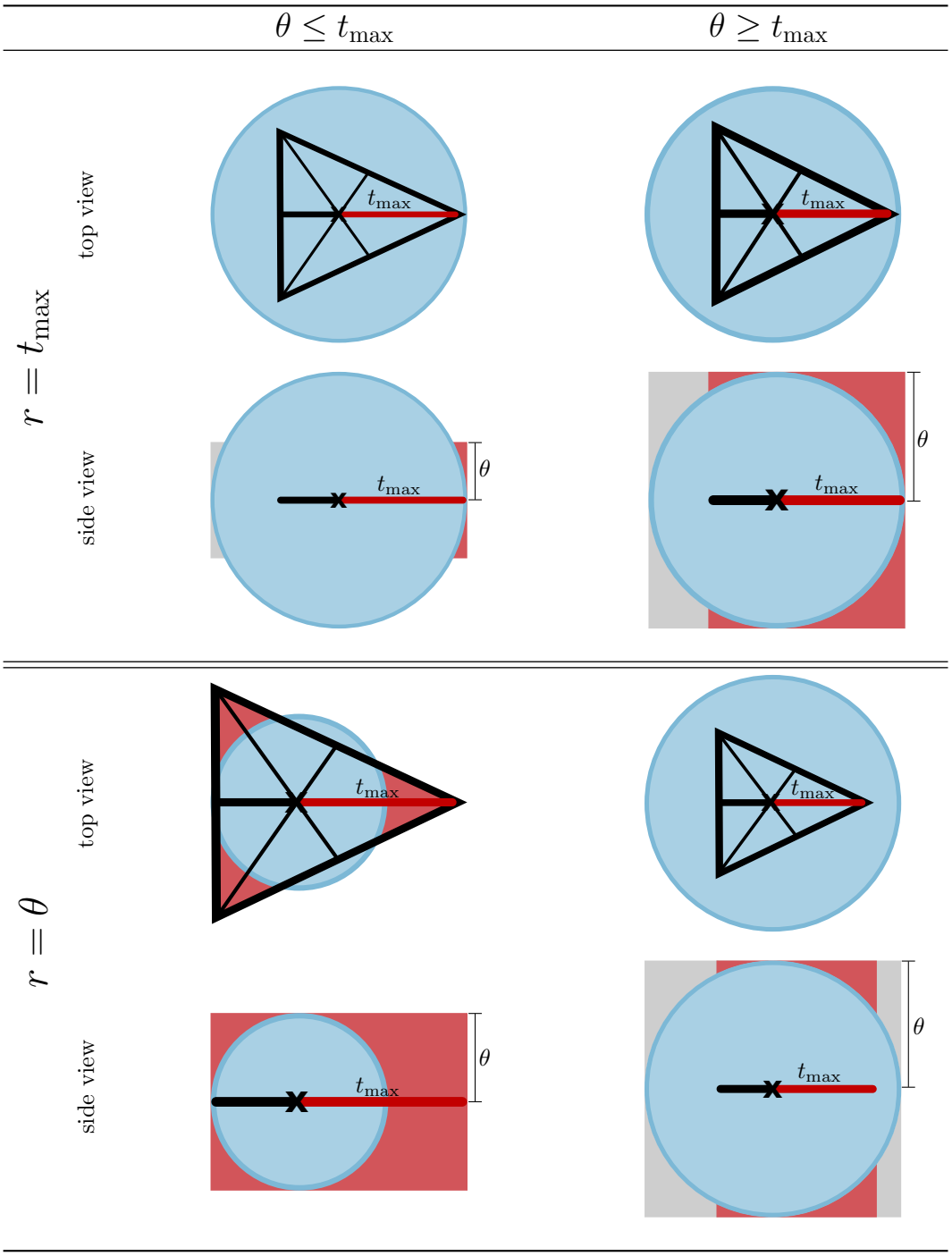


Figure 5.3: Demonstration of association sphere (*blue*) for the PC clipping (step (i) of PCMA) when radius r is set too small: a) $r = t_{\max}$ (*top*) and b) $r = \theta$ (*bottom*). The maximum distance $t_{\max}$ (*dark red*) from the CoG (*black cross*) to face vertices equals $\frac{2}{3}$ of the maximum median of the triangular face. The threshold band marked in *gray* is defined by threshold θ. The figure depicts both scenarios: $\theta \leq t_{\max}$ (*left*) and $\theta \geq t_{\max}$ (*right*). The figure shows top views with respect to the triangular face surface (*upper rows*) and side views with respect to the largest median of the triangular face (*bottom rows*).
Too small radii do not guarantee to enclose the entire face nor the threshold band across the entire face. Areas marked in *light red* depict zones that should be enclosed by the association sphere. In those areas, possible association candidates are prefiltered mistakenly.
$r = t_{\max}$ guarantees to enclose all association candidates on the face, but may miss *off-the-face* points. Likewise, $r = \theta$ does not guarantee to enclose all *off-the-face* association candidates. Besides, this scenario may even miss association candidates on the face. Figure 5.2 shows a proper choice of r.

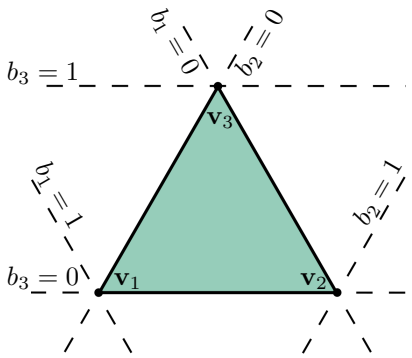


Figure 5.4: Barycentric coordinates b_i of the face $\triangle \mathbf{v}_1\mathbf{v}_2\mathbf{v}_3$ with vertices $\mathbf{v}_1 = (1, 0, 0)$, $\mathbf{v}_2 = (0, 1, 0)$ and $\mathbf{v}_3 = (0, 0, 1)$.

direction. Starting from level 1, the algorithm tries to associate points with the respective thresholds θ_1^+ and θ_1^-. If points have been associated, the association stops for the corresponding face ("early stopping"). Otherwise, the next level l is activated. This practice accelerates the entire association process including the information transfer. On the other hand, by nature of our approach, not all points might be associated (see Section 5.4). Here, our reasoning is to favor near-surface points at the cost of missing to link a few points (small margin association). The remaining subset $\mathcal{P}_f'''$ is the set of n_{pts} points linked to face f. Figure 5.5 depicts an excerpt of the H3D PC colorized by the linked face indices.

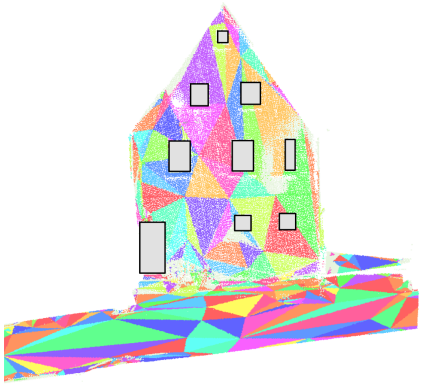


Figure 5.5: The example depicts the dense LiDAR point cloud of H3D randomly colored by the indices of the associated faces (of the respective 2.5D mesh). Non-associated points are marked by the black-framed grayish polygons. Those points are not linked to faces due to exceeding the given thresholds (e.g., measured interior LiDAR points inside the building) or due to "early stopping" (the 2.5D mesh geometry does not properly reconstruct the facade geometry, e.g., faces trespass borders of windows and facade).

The adaptive thresholding facilitates an arbitrary degree of freedom by the set magnitude of thresholds and their inter-level spacing. Hence, it balances the strictness of the point-face linking. Small-valued thresholds along with small-spaced levels (i.e., a small $\theta_{\max} = \max(\{|\theta_l^+|, |\theta_l^-|\} \forall l)$) enforce a tight coupling where only points close to the mesh surface are associated. Large values along with large level spacing (i.e., a large $\theta_{\max}$) loosen the coupling. Levels $l > 1$ can be seen as a fall-back for scenarios where a proper association is not possible. Moreover, the asymmetric two-fold thresholding per level enables non-symmetric filtering improving flexibility and adaptiveness. For instance, the asymmetric adaptive thresholding allows associating faces with points where 2.5D mesh geometry and 3D geometry of the PC differ significantly while favoring the association of near-surface points (e.g., at facades or tree stems, see Figure 5.14). Laupheimer et al. (2020b) discuss in detail the particular challenges for the PCMA using a 2.5D mesh. Therefore, our association approach is agnostic to the dimensionality of the mesh: 2.5D or 3D meshes can be processed.

To accelerate the association, the linking is done in tile-wise and parallelized fashion. Therefore, we impose the given mesh tiling (see Chapter 4) on the PC. For a proper entity linking close to the tile fringes, we pad the PC tiles by some margin (see Figure 5.6). The per-tile padding depends on the maximum of the user-defined thresholds $\theta_{\max}$ and the maximum distance of the CoGs and their respective face vertices $\max(\{t^f_{\max}\} \forall \text{faces } f)$. Rephrased, the padding is defined by the maximum radius $r_{\max}$ of all association spheres (see Figure 5.2). As a simple alternative, we implemented the option to use a hard cut-off distance as a global padding value, which therefore is uniform across tiles and accelerates the processing (at least for reasonable cut-off values).

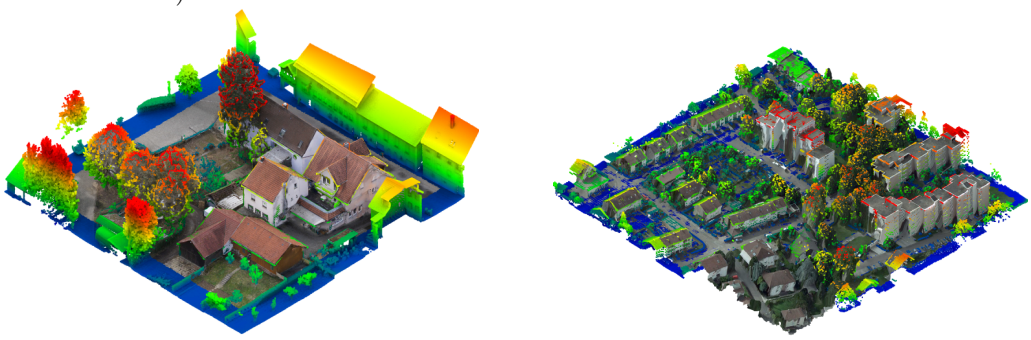


Figure 5.6: Tile dimensions of the height-coded point cloud (*blue*: low, *red*: high) and the textured mesh to guarantee face-point linking at the fringe of the mesh tile. Here, the mesh tile extension is imposed to the point cloud with a hard-coded padding of 5 m (*left*: H3D, *right*: V3D).

The association information is stored as a per-point attribute. For each associated point, the respective face index is attached to its attributes. Non-associated points are marked with -1. The stored indices trivialize the transfer of features and labels from the mesh to the PC (direction Mesh $\mapsto$ PC in Table 5.1). In this case, we copy the desired values to the PC at a stroke due to the one-to-many relationship. Reversely, the stored face indices can also be used to transfer features and labels from the PC to the mesh (direction PC $\mapsto$ Mesh in Table 5.1). To speed up the transfer, we couple the information transfer directly with the association mechanism. The many-to-one relationship calls for information aggregation. For each face, we derive robust median features from the linked subset $\mathcal{P}'''_f$. Features may embrace sensor-intrinsic and hand-crafted features such as pulse characteristics and derived quantities (see Section 6.2). Analogously, majority votes determine the per-face labels as transferred from the associated points. Therefore, the association inherently is a label transfer and feature calculation tool (median features). The aggregated features and labels are attached to the CoG cloud. Non-associated faces are marked with -1 labels and receive zeroed median features (Laupheimer et al., 2020b).

Figure 5.7 shows the PC and the respective 3D mesh for the same subset of H3D side-by-side colorized by labels and various attributes. The PC appears to be a closed surface like the mesh due to the high point density and the chosen point size for visually appealing illustration. The annotations and visualized features have been automatically transferred from the manually annotated PC deploying PCMA (except for texture and relative height). The relative height is derived for each entity utilizing the respective DTM (see Section 6.2). Faces that cannot be linked with points are depicted in black (background color). Obviously, there are almost no differences notable between the modalities. This shows that the information transfer works reasonably – both for discrete and continuous quantities. Moreover, the filtering effect of the median aggregation becomes obvious for the transferred features in the marked regions (only highlighted in the colored/textured subfigure). For instance, roofs and the curved street close to the ship lock appear to be more monotonous on the mesh than on the PC – particularly for the colorized reflectance. This effect is further intensified by the "early stopping" of the geometric-driven linking, which favors near-surface points and might eliminate PC noise. Please note the colorization on the two modalities does not match perfectly due to different shading on PC and mesh. The normal-depending shading prevents a perfectly matching color comparison but is necessary to uncover structure, i.e., to generate an illusion of depth.

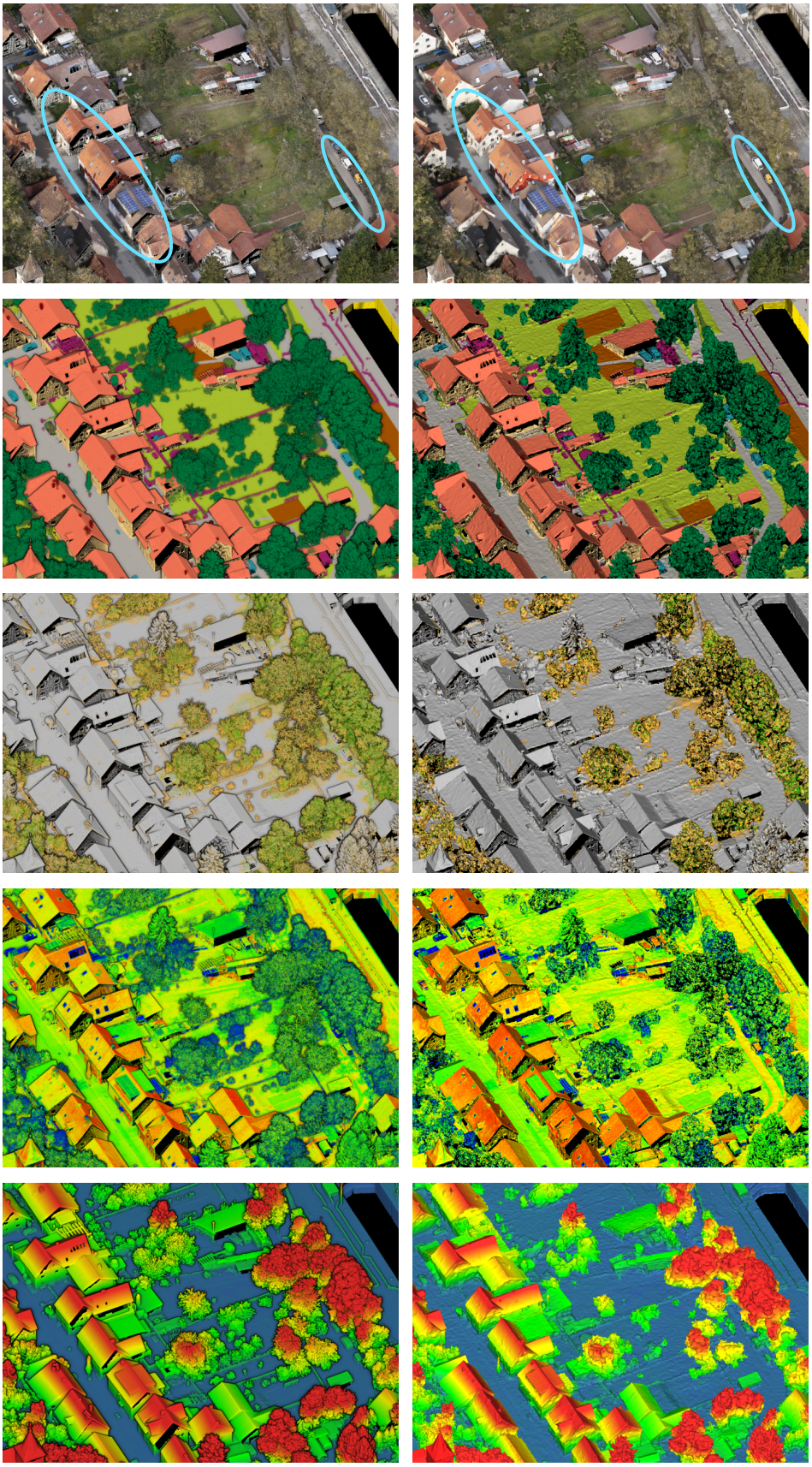


Figure 5.7: Side-by-side comparison of the 3D PC (*left*) and the 3D mesh (*right*) of the center of the Hessigheim village close to the ship lock. The snapshots depict various quantities (from *top* to *bottom*): color/texture, label, number of returns, reflectance, relative height. Label, number of returns, and reflectance have been propagated by PCMA-steered information transfer. The *sky-blue* ellipses highlight parts that are discussed in the text.

5.2 Image Mesh Association (ImgMA)

The Image Mesh Association (ImgMA) explicitly links faces and pixels across various images in a geometry-driven approach. Each pixel is associated with the respective visible face as detected by the following three steps:

(I) preselection of potentially visible mesh tiles per image utilizing their Mininum Bounding Boxes (MBBs),

(II) ray tracing per image and preselected tile (image-tile-pair),

(III) fusion of ray tracing results per image via depth filtering across tiles.

We end up with n_{px} associated pixels across n_{img} images for each face. Figure 5.8 shows the workflow using two oblique example images and two vertically separated mesh tiles for steps (II) and (III).

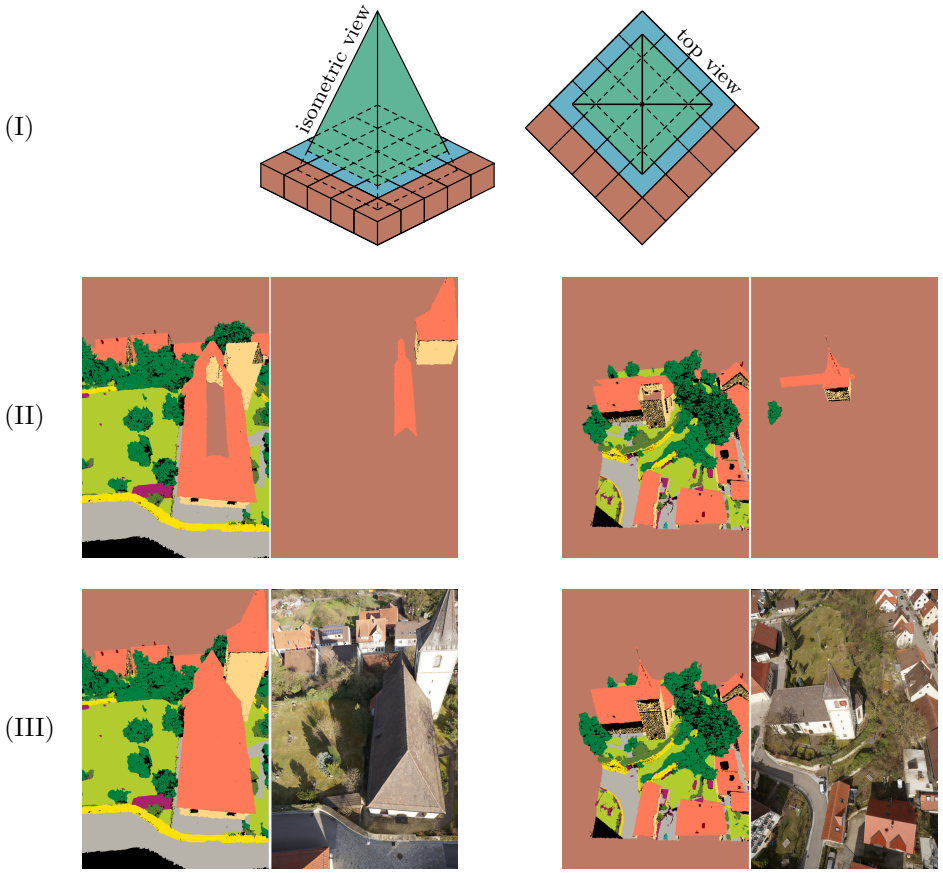


Figure 5.8: Steps (I) – (III) of the ImgMA shown in accordance with the information transfer Mesh ↦ Img (see Table 5.1). The depicted real-world examples of steps (II) and (III) show the association of two oblique images with two vertically split mesh tiles of H3D. *(I)*: Preselection of potentially visible mesh tiles per image shown schematically in isometric (*left*) and top view (*right*). The stretched camera pyramid (*green*) intersects with MBBs of potentially visible mesh tiles (*blue*). Non-intersecting MBBs are marked in *reddish-brown*. *(II)*: Image-wise ray tracing results for the image-tile pairs of previously detected visible tile candidates. The *reddish-brown* area shows where ray tracing fails due to missing intersections of image rays and considered mesh tiles. *Black* indicates unlabeled faces. *(III)*: Fusion of ray tracing results per image via depth filtering across tiles.

To accelerate the ImgMA we make use of the given mesh tiling. For each image, we first detect tile candidates that may be captured by the image. The subsequent ray-tracing procedure considers only the

detected subset of tiles and is executed in a parallelized fashion per image. We define a tile candidate to be visible if its MBB intersects with the stretched camera pyramid (MBB visibility check). The stretched camera pyramid is defined by the projection center and the projection rays crossing the corner pixels of the respective image. The lowest of all MBB faces limits the stretched camera pyramid, i.e., there is no mesh data below the stretched camera pyramid. The stretched camera pyramid is depicted in green in Figures 5.8, 5.9, 5.10, 5.11, and 5.12. In 3D computer graphics, the stretched camera pyramid is called view frustum. Since the MBBs are not fully occupied by the enclosed mesh tiles, some detected tile candidates might contain merely faces that are not visible in the image. Nonetheless, this approach significantly reduces the number of tiles that have to be processed by the ray-tracing procedure for each image. We detect intersections of the stretched camera pyramid and a MBB by a three-stage check. Once a check succeeds, the respective enclosed tile is marked as visible and the residual checks are omitted for that tile (check omission). Figures 5.9, 5.10, and 5.11 sketch the performed checks demonstrating a successfull and failing example each. Figure 5.12 schematically depicts their specific use-cases to detect visible MBBs per image.

At first, we check for each tile if any corner point $\boldsymbol{p}$ of the respective MBB is inside the stretched camera pyramid (point-in-polyhedron test). The left of Figure 5.12 depicts the scenario where all corner points are inside the stretched camera pyramid. Figure 5.9 showcases the passing and failing of the point-in-polyhedron test for two distinct MBBs: a) at least one corner point is inside the stretched camera pyramid (*left*) and b) the MBB is entirely outside the stretched camera pyramid (*right*). As a precondition, we define all normals $\boldsymbol{n}_\varphi$ of the faces φ of the stretched camera pyramid to point outwards. A point $\boldsymbol{p}$ is inside or on the pyramid if $\langle \boldsymbol{p}'_\varphi | \boldsymbol{n}_\varphi \rangle \leq 0 \, \forall \varphi$, i.e., if $\boldsymbol{p}$ is on the negative side of the plane for all φ. Vector $\boldsymbol{p}'_\varphi$ starts at an arbitrary point $\boldsymbol{p}_\varphi$ that belongs to face φ and ends at $\boldsymbol{p}$, i.e., $\boldsymbol{p}'_\varphi = \boldsymbol{p} - \boldsymbol{p}_\varphi$. For all triangular faces, we choose the projection center as the starting point $\boldsymbol{p}_\varphi$, i.e., triangular faces share the same $\boldsymbol{p}'_\varphi$. For the ground face, a randomly picked corner point of the ground polygon is used. Geometrically, the negative or vanishing dot products indicate obtuse or orthogonal angles between $\boldsymbol{p}'_\varphi$ and $\boldsymbol{n}_\varphi$. Similarly, the dot products show for each face φ which side point $\boldsymbol{p}$ is on. Since face normals point outwards, the point is inside or on the pyramid when all dot products are ≤ 0. To speed up the MBB visibility check, the algorithm breaks once a corner point is detected to be inside the stretched pyramid (omission within check).

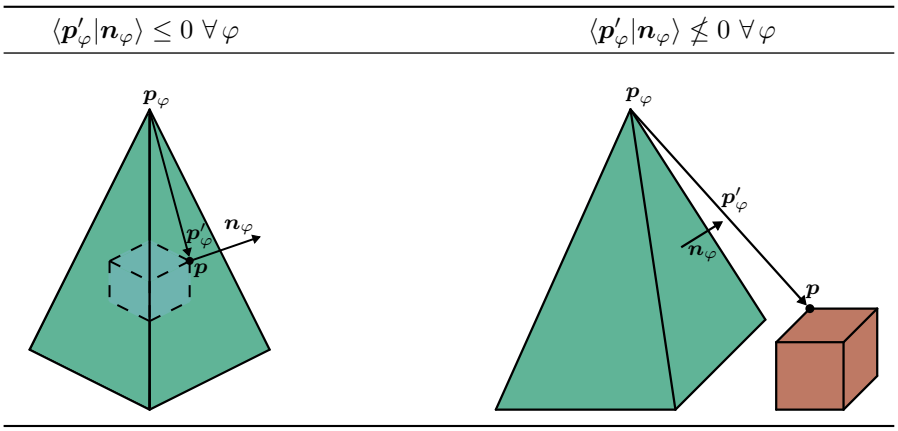


Figure 5.9: First MBB visibility check: Point-in-polyhedron test to check whether any corner point $\boldsymbol{p}$ of the MBB of the mesh tile is inside the stretched camera pyramid (*green*). The necessary quantities for the scalar product are depicted for a single face φ with its normal vector $\boldsymbol{n}_\varphi$. Vector $\boldsymbol{p}'_\varphi$ starts at an arbitrary point $\boldsymbol{p}_\varphi$ that belongs to face φ and ends at $\boldsymbol{p}$, i.e., $\boldsymbol{p}'_\varphi = \boldsymbol{p} - \boldsymbol{p}_\varphi$.
Left: At least one corner point is inside the stretched camera pyramid. The MBB will be marked as visible (*blue*).
Right: The MBB (*red*) is entirely outside the stretched camera pyramid and passed on to the subsequent checks.

This check does not cover the case where the MBB and the pyramid intersect but all MBB corner points are outside the stretched camera pyramid (see Figure 5.12, *center* and *right*). This case is covered by the second and the third check. For them, please note the difference between a face and a plane, and accordingly

an edge and a line/ray. A face is a closed polygon of finite area, whereas a plane as defined by mathematics is infinite. Similarly, an edge is a finite closed line segment, whereas a line/ray is infinite.

The second check tests if the pyramid edges starting from the projection center (i.e., not the ground face edges) intersect with any MBB face (edge face intersection). Mathematically, intersection candidates are calculated by intersecting lines as defined by the pyramid edges and planes as defined by MBB face normals and an arbitrary point of the respective MBB face. Valid intersection points are part of the pyramid edges and the MBB faces. For each edge-face-pair, we first filter calculated intersection candidates that are not part of the pyramid edges. Since the stretched camera pyramid is limited by the lowest MBB plane, intersection candidates can be filtered by their height. Likewise, we neglect intersection candidates whose heights exceed the projection center's height (for airborne acquisition). The remaining intersection candidates are checked to be within the respective face polygon of the MBB. Due to the rectangular shapes of the MBB faces and their parallelism to the coordinate axes, the point-in-polygon test is trivial (comparison via coordinates). Figure 5.10 showcases both the succeeding and failing of the second MBB visibility check.

any intersection candidate inside MBB face	all intersection candidates outside MBB face

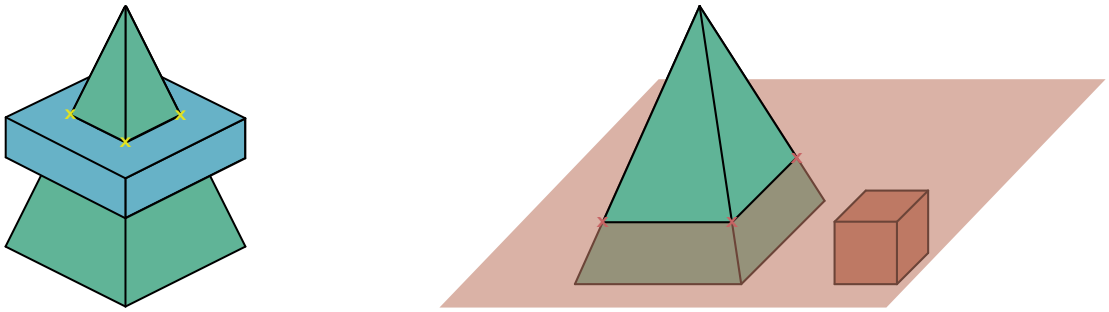


Figure 5.10: Second MBB visibility check: Detection of valid intersection points of lines defined by pyramid edges and planes defined by MBB faces. Valid/invalid intersection points are marked as *yellow*/*red* crosses. The stretched camera pyramid is depicted in *green*.
Left: At least one intersection point is part of a pyramid edge and a MBB face. The MBB is marked as visible (*blue*).
Right: The intersection points are only part of the plane (*transparent red*) defined by the MBB face, but not of the MBB (*red*). The MBB cannot be marked as visible and is passed to the third check.

The second check does not cover the case where the MBB and the pyramid intersect but none of the pyramid edges intersect with MBB faces (see Figure 5.12, *right*). To cover this case, we check if any MBB edge intersects with the pyramid faces (edge face intersection). This is the reverse situation of the second check. Mathematically, intersection candidates are calculated by intersecting lines as defined by the MBB edges and planes as defined by the face normals and an arbitrary point on the respective face. Each line might deliver multiple intersection candidates. However, we retain only the intersection points closest to an arbitrary MBB corner point $\boldsymbol{p}_i$ of the respective edge for further analysis. Valid intersection points are part of one of the pyramid faces and one of the MBB edges. For the airborne scenario, we first prefilter intersection candidates not matching the valid height range (see second check). The remaining intersection candidates are input to a point-in-polygon test to filter candidates that are not part of pyramid faces. For the residual intersection candidates, we check whether they are part of the MBB edges. Each edge is defined by an arbitrary MBB corner point $\boldsymbol{p}_i$ as the starting point and a direction vector $\boldsymbol{r}_{ij}$ as defined by the respective residual corner point $\boldsymbol{p}_j$ on the same edge, i.e., $\boldsymbol{r}_{ij} = \boldsymbol{p}_j - \boldsymbol{p}_i$. The respective intersection candidate $\boldsymbol{q}$ is part of the MBB edge if $\langle \boldsymbol{r}_{ij} | \boldsymbol{q} - \boldsymbol{p}_i \rangle \geq 0 \wedge ||\boldsymbol{r}_{ij}|| \geq ||\boldsymbol{q} - \boldsymbol{p}_i||$. Graphically, the spanned vector $\boldsymbol{q} - \boldsymbol{p}_i$ points in the same direction and its length is smaller than or equal to the edge. Figure 5.11 demonstrates the fulfilling and not-fulfilling of the constraint for two distinct MBBs. A MBB that failed all three checks is marked as

not visible for the respective image. To speed up the MBB visibility check, the algorithm breaks once a valid intersection point is detected by the second or third check (omission within check).

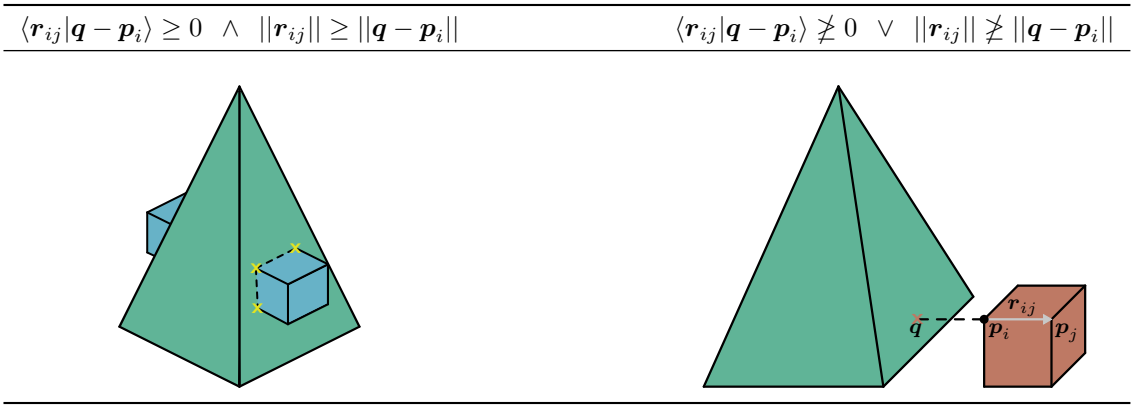


Figure 5.11: Third MBB visibility check: Detection of valid intersection points of lines defined by MBB edges and planes defined by stretched pyramid faces. Valid/invalid intersection points are marked as *yellow*/*red* crosses. The stretched camera pyramid is depicted in *green*.
Left: at least one intersection point is part of a pyramid face and a MBB edge. The MBB is marked as visible (*blue*).
Right: the intersection candidate is only part of the pyramid face, but not of the MBB edge. The MBB cannot be marked as visible since it failed all visibility checks.

The combination of check omissions and omission within checks accelerates the preselection of tiles. The check cascade starts with the fastest check. The first check is one order of magnitude faster than the other checks. It is also the most likely case for the investigated data sets. We argue that an image footprint almost certainly covers several mesh tiles and therefore envelopes several MBB corner points. Similarly, it is unlikely that all corner points of all visible MBBs are outside the stretched camera pyramid – particularly when oblique imagery is used. Therefore, the cases covered by the second and the third check are less likely.

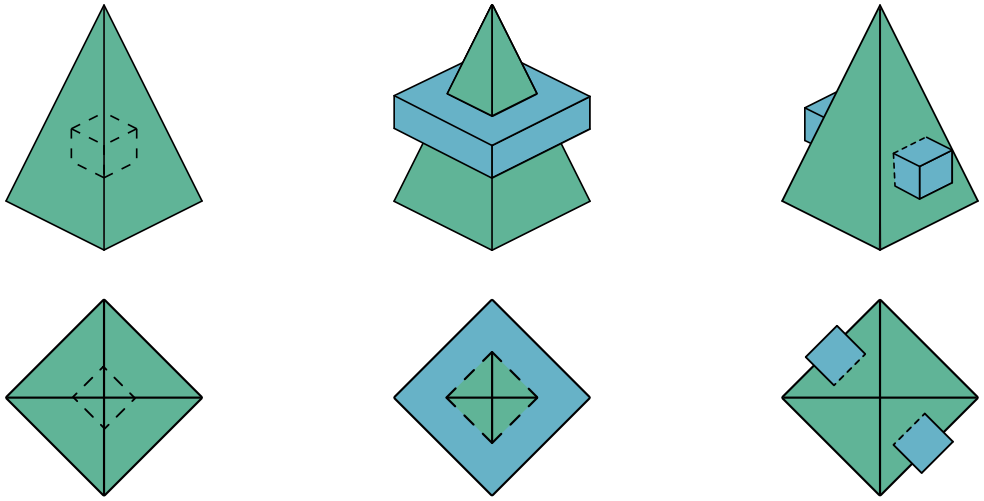


Figure 5.12: Image-wise visibility checks for MBBs (*blue*) as performed by intersections of the stretched camera pyramid (*green*) and MBB (step (I) of ImgMA). The depicted cases pass the visibility check (*top*: isometric view, *bottom*: top view). Please note that the lowest of all MBB faces limits the stretched camera pyramids.

As a result of the three-stage preselection of mesh tiles (step I), we receive a list of potentially visible tiles for each image, i.e., each image $\mathcal{I}$ is linked to $n^{\mathcal{I}}_{\text{tiles}}$ tiles. Hence, there are $n^{\mathcal{I}}_{\text{tiles}}$ image-tile-pairs for

each $\mathcal{I}$. Vice versa, a list of images that potentially map the mesh tile is stored for each tile. For each image-tile-pair, visible faces are determined via backward ray tracing. For this purpose, a 3D projection ray is created for each pixel and intersected with the mesh faces of all detected tiles. The intersected faces are candidates for the final association result. Concerning a single image-tile-pair, all candidate faces are truly visible. However, an image probably covers multiple tiles, and hence, some faces might be occluded by faces of another tile (see church in Figure 5.8). Particularly, tiling in the height direction and oblique capturing cause images to cover multiple tiles. Consequently, the tile-wise process causes an incomplete ray tracing concerning the entire set of mesh tiles.

To cure the incompleteness, we fuse the $n^{\mathcal{I}}_{\text{tiles}}$ ray tracing results across the visible tiles into one final ray tracing result per image. Implementationally, the fusion across the $n^{\mathcal{I}}_{\text{tiles}}$ image-tile-pairs is done implicitly by depth updates (directly coupled to step II). Initially, each pixel is associated with a near-infinite depth value. The association information is updated whenever a candidate face reduces the depth value. Hence, the final association information is steered by faces of minimum depth that are truly visible (i.e., faces that mark the first intersection along the respective ray). To speed up the implementation, steps (II) and (III) are coupled and parallelized with respect to images. Each processing step handles $n^{\mathcal{I}}_{\text{tiles}}$ image-tile-pairs per image. The coupled implicit fusion reduces the memory footprint of the algorithm since only the final ray tracing result has to be stored. Otherwise, the face indices and depth values would have to be stored for each pixel of each image-tile pair.

The final association information is stored as a binary file (.h5) per image. The association information consists of the depth and the tile-dependent face index for each linked pixel. To minimize the memory footprint, we avoid storing the association information as image matrices. Particularly for oblique images, only a small part of the image may be associated due to a limited reconstruction area of the mesh (see Chapter 4). We store the association information as a *sparse pixel cloud* per image containing only pixels that have been linked with faces. More specifically, the sparse pixel cloud is split per tile into subclouds since face indices are tile-dependent starting from zero for each mesh tile. The subdivision into subclouds avoids the redundant storing of the tile ID. Each subcloud contains tuples of associated pixel positions (r_i, c_i), the depth d, the face index f, and optionally, other attributes $\{a_i\}$ (e.g., labels) as transferred from the mesh: $(r_i, c_i, d_i, f_i, \{a_i\})$. For convenience, any associated information may be stored as an image channel. This might be beneficial for fast access, fast visualization, and non-sparse DL methods, but at the cost of a large memory footprint. In this case, non-associated pixels carry default values for any quantity (marked with -2). Pixels that are linked to unlabeled faces are marked with -1 (i.e., the label of faces that have been missed during manual annotation or could not have been labeled by PCMA-steered annotation).

If the desired quantities have not been encoded in the sparse pixel cloud, the stored face indices trivialize the subsequent transfer of features and labels from the mesh to the images (direction Mesh $\mapsto$ Img in Table 5.1). In this case, we copy the desired quantities to the linked pixels of the associated images at a stroke due to the one-to-many-to-many relationship. Reversely, the stored face indices (and the list of linked images for each mesh tile) can also be used to transfer features and labels from the imagery to the mesh (direction Img $\mapsto$ Mesh in Table 5.1). The many-to-many-to-one relationship calls for information aggregation across associated images for each face. Aggregated features may embrace sensor-intrinsic color values, handcrafted features, and features as derived by DL pipelines. Analogously, the per-face labels have to be aggregated by majority votes from the linked pixels.

5.3 Point Cloud Image Association (PCImgA)

The Point Cloud Image Association (PCImgA) aims for the linking of 3D points and pixel locations. Theoretically, the collinearity equations establish an explicit relationship between 3D points and pixel locations. Each 3D point can be projected into the image space given the exterior and interior orientation of the respective image. However, the bare projection cannot solve the visibility problem and hence links visible and non-visible points with imagery. Therefore, point visibility has to be checked prior to the linking of 3D points and pixel locations. To this end, we leverage the mesh representation as a visibility model by combining

mechanisms PCMA and ImgMA. The detour via the mesh largely solves the visibility problem for 3D points in image space (see Section 5.4). The PCMA and ImgMA can be combined implicitly and explicitly.

The implicit linking couples the association mechanisms PCMA and ImgMA by simply executing them sequentially caching the information to be shared on the mesh. By these means, the information from the starting representation is gathered per face and transduced to the target representation. Specifically, each face is associated with several points and several pixels across several images. The per-face label and features derived from the starting representation are determined via majority vote and median aggregation respectively (see Table 5.1). Subsequentially, the gathered per-face quantities are copied to the target modality. The implicit linking associates points and pixels apparently by the joint face as a proxy. In reality, n_{px} pixels of n_{img} images and n_{pts} points are exclusively linked to the respective common face and the information transfer utilizes the per-face aggregations (one-to-many relationship). As described in Sections 5.1 and 5.2, the information transfers PC $\mapsto$ Mesh and Mesh $\mapsto$ Img are coupled to the respective association mechanisms and therefore speed-up the transfer PC $\mapsto$ Img. We argue that the transfer direction from PC to image space is more relevant – at least, this holds for the considered data sets V3D and H3D. For the reversed situation, the image information has to be gathered per face from the stored sparse pixel clouds and copied to the points after the linking process (requiring another loop over tiles).

On the contrary, the explicit linking couples the association mechanisms PCMA and ImgMA by leveraging the stored association information and utilizing the collinearity equations. As a result of ImgMA, visible faces of a tile are known for each image. At the same time, mechanism PCMA delivers the associated points for each face. Consequently, associated points of visible faces are marked as visible for each image. Therefore, the explicit combination of PCMA and ImgMA results in a visible subset of the PC per image. The collinearity equations explicitly link the visible points and the pixel locations across all imagery. Hence, explicit linking truly associates points and pixel locations.

5.4 Preconditions and Limitations

The proposed method connects PCs, photographic imagery (following the central perspective), and textured meshes oriented in the same coordinate system and covering the same area. The multi-modal linking is designed generically. In particular, PCMA works with any mesh and PC data regardless of the acquisition platform (aerial, terrestrial, mobile), the mesh type (photogrammetric meshes, LiDAR meshes, or hybrid meshes), and PC type (MVS cloud, LiDAR cloud, persistent scatterer cloud) as it incorporates only modality-specific properties. PCMA is not constrained to a specific mesh generation algorithm or mesh geometry (see Figure 6.6). Furthermore, 2.5D and 3D meshes can be processed and linked with PCs (see Figure 5.14).

In this work, we focus on the linking of aerial RGB imagery and ALS PCs along with the respective textured meshes. Therefore, we fine-tune ImgMA to the constraints of an airborne acquisition. Precisely, the preselection of potentially visible mesh tiles (i.e., the three-stage visibility check of step I of ImgMA) incorporates knowledge about the airborne acquisition to accelerate the linking process (see Section 5.2). However, neglecting the airborne-constrained preselection enables the processing of terrestrial images, too, at the cost of a longer processing time. Any image following the central perspective can be processed regardless of the image type (panchromatic, RGB, multi-spectral).

The main constraint of the total association mechanism is the co-registration of the three modalities. Geospatial data is co-registered implicitly through its georeference. Particularly, the image-steered meshing process guarantees the co-registration of imagery and mesh data. Inherently, MVS PCs and meshes are aligned with imagery. Ideally, imagery and PC data are co-registered simultaneously in a joint adjustment like proposed by Glira et al. (2019). Consequently, the derived mesh is aligned with both data sources. However, joint adjustment of both data sources is not state-of-the-art in practice yet. The utilized data and preprocessing steps in this thesis reflect state-of-the-art data processing with two separated registration pipelines for ALS data and imagery. We fine-register both data sources after their disjoint registration by applying the ICP algorithm to cure co-registrations discrepancies globally (see Chapter 4). Local registration discrepancies in 3D space might be covered by the adaptive thresholding during the PCMA.

A good relative orientation of 2D and 3D data is crucial to the linking of pixels with points or faces. The linking of 3D data and image space via PCImgA and ImgMA depends entirely on the co-registration quality. Therefore, pixel-accurate co-registration is necessary. However, the fine co-registration of both data sources is not the focus of this work. We expect joint adjustment of both sources soon to be state of the art.

The entire association benefits from good geometric reconstruction as the mesh is the core of the proposed linking mechanism. We are aware that automated meshing is a non-trivial process and object of current research. Proper reconstruction ensures proper entity linking, enabling proper information transfer – even for thin structures like street lights or antennas. As a general rule, the better the mesh represents the true 3D structure, the better works the proposed association and the subsequent information transfer.

To the best of our knowledge, state-of-the-art commercial meshing algorithms scarcely incorporate semantics. Therefore, reconstructed faces do not necessarily represent semantic borders. For instance, consider the transition of a planar *Impervious Surface* to *Low Vegetation*, like a lawn, in the real world. The mesh representation may simplify this scenario to a single large face. On the contrary, the same scenario is captured properly in the PC and imagery. Consequently, too large triangles at class borders will associate points and pixels of different classes (see Figure 6.8).

Assuming a proper co-registration, entities across representations can only be associated when underlying real-world objects are captured or reconstructed in all representations. In other words, each face should at least enclose one point or one pixel respectively representing the same 3D object. Reversely, each point or pixel should be mapped onto a corresponding face. However, these relationships do not always exist due to inter-representation differences as discussed in Chapter 1 and deficiencies during acquisition. Whereas imagery and the mesh map the surface, LiDAR systems with multiple echos also capture sub-surface points which lack a correspondence in the other modalities (e.g., semi-transparent objects such as vegetation). Fine-structured objects may not be reconstructed entirely in the mesh, but the objects are fully captured in the PC and imagery. The missing reconstructions in the mesh omit to link points of the respective 3D object and even causes wrong associations between imagery and 3D space as the visibility check via the incomplete mesh is violated. Additionally, there might be inter-representation discrepancies due to asynchronous acquisition (see Figure 5.13). Besides, a sensor may miss capturing specific areas due to unfavorable path planning, sensor failure, and occlusions. For instance, a facade may be captured by images but not by the LiDAR system. Hence, pixels and reconstructed mesh faces cannot link any 3D points (see Figures 5.13 and 6.4).

Since the relationship of 2D and 3D space is strictly defined by the collinearity equations, we highlight the discussion of PCMA in the following. Figure 5.14 and Table 5.2 sketch discrepancies of PC and mesh despite representing the same real-world objects. The adaptive thresholding largely covers these discrepancies.

Despite and due to the adaptive thresholding, not all points are associated with faces by the PCMA. There are three groups of unassociated points (see Table 5.2). In Section 5.1, we presented the association methodology. Visually, the association is done with the help of the per-face association prism as spanned by the face and its normal. The association prism is pruned by the range of manually set thresholds $\{\theta_l^+, \theta_l^-\}$. The threshold-based approach filters all points whose orthogonal projections to the face plane exceed a specific value (see Table 5.2, A1). The reasoning behind this non-association is that the respective entities most likely represent different surfaces (e.g., cars and street in case of asynchronous data acquisition, see Figure 5.13). Furthermore, the adaptive thresholding breaks the association once a point is associated at any threshold level l. Hence, points that are closer to the face than the maximum threshold may not be associated (see Table 5.2, A2). The adaptive thresholding favors near-surface points by ensuring the association of points fulfilling the tightest threshold.

The set of association prisms does not necessarily enclose all points of the PC. Particularly, points above and below the mesh surface may fall into *dead zones* that are not covered by diverging association prisms. The prisms of adjacent faces diverge when they form convex or concave surfaces, i.e., their normal vectors are not parallel. Non-perfect reconstructions of planar surfaces artificially introduce convex or concave structures (see Table 5.2, B1). Naturally, points above convex surfaces or below concave surfaces cannot be linked. Points above the reconstructed roof ridge are a prime example for that scenario (see Table 5.2, B2). For this reason, co-registration discrepancies in combination with reconstruction imperfections increase the number of non-associations.

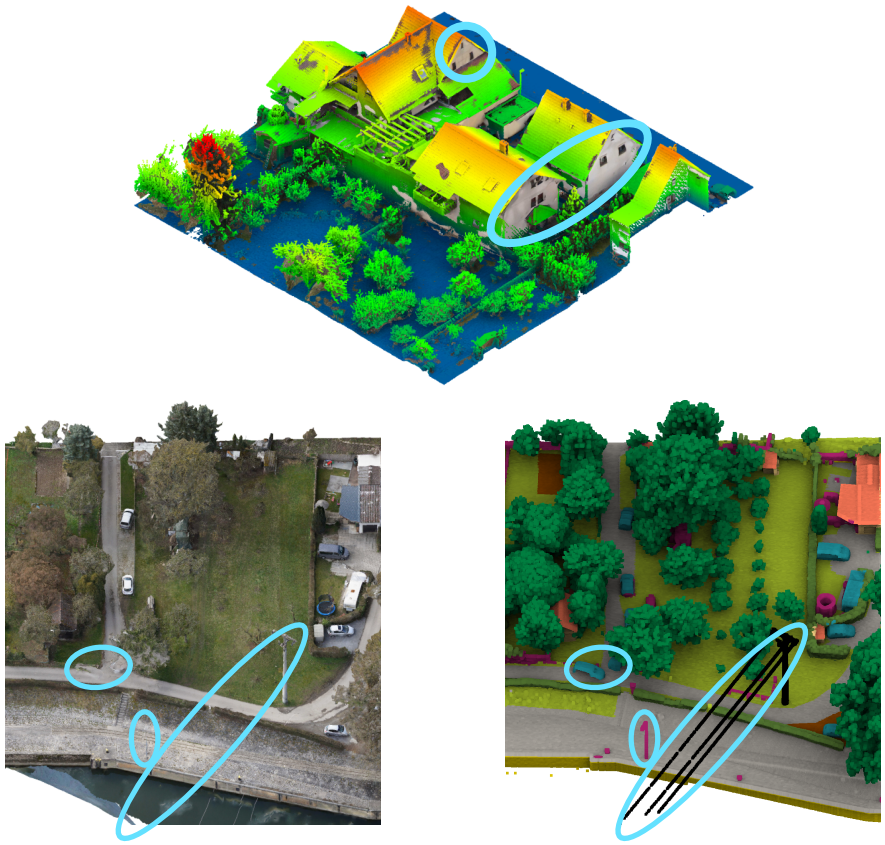


Figure 5.13: Discrepancies (*sky-blue ellipses*) between 3D models due to missed acquisition or reconstruction by the example of H3D data. *Top*: Urban area. Overlay of textured mesh and ALS PC (height-coded). Facades are reconstructed entirely in the mesh but are captured only partially by the LiDAR sensor. *Bottom*: Ship lock area. The mesh (*left*) partially reconstructs the river but misses to fully reconstruct thin structures like light poles and power lines, which are captured entirely in the PC (*right*). The asynchronous data acquisition of images and ALS data causes inconsistencies between mesh and PC (e.g., for cars).

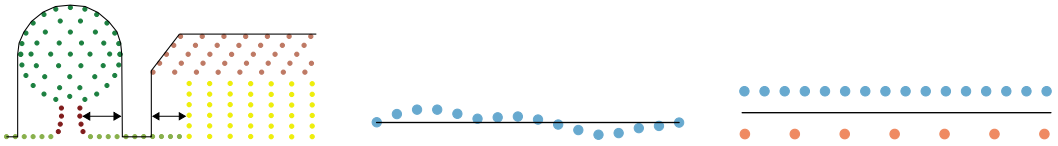
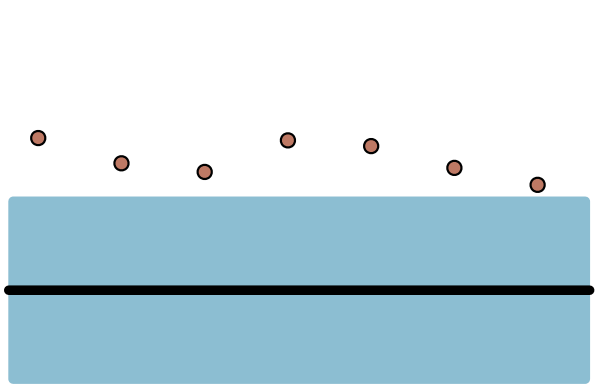


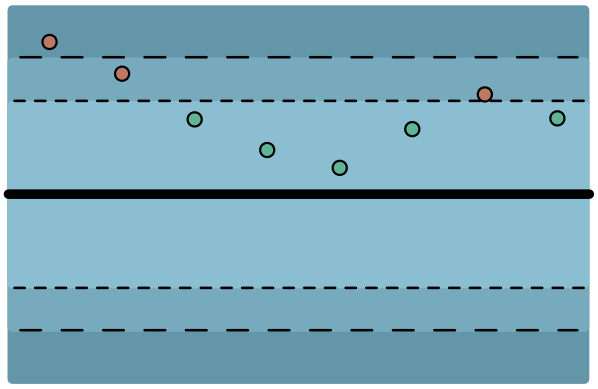
Figure 5.14: Discrepancies between PCs and the mesh (*black line*) caught by adaptive thresholding. *Left*: *Black arrows* indicate the discrepancy between the 2.5D mesh and the annotated 3D PC. *Center*: The noisy PC (*blue*) oscillates about the reconstructed mesh. *Right*: There might be misalignment between PCs (*blue*: MVS, *orange*: LiDAR) and the mesh as generated of a single source or complementary sources.

Table 5.2: The schematic drawings illustrate the three cases where faces (*black* lines, separated by *black* strokes) and points are not linked by PCMA (side view with respect to the faces). Non-associated points are marked in *red*; associated points in *green*. The pruned association prisms are marked in *blue*. Adaptive thresholds are omitted for accessibility reasons – except for case A2, which depicts the increasing thresholds with increasing blueness. Diverging association prisms create *dead zones* like depicted in B (*hatched in red*). B1 mimics a perfectly planar surface as *dashed black* line.

A) points outside the threshold range

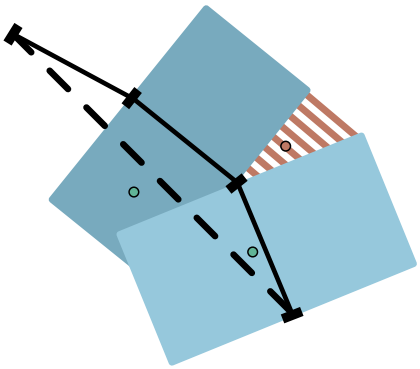


A1: points outside the maximum threshold

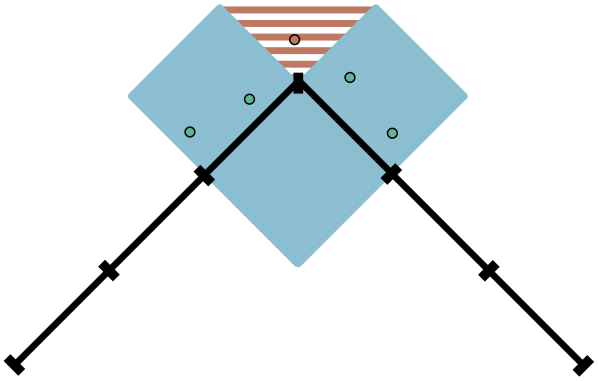


A2: points in different threshold bands ("early stopping")

B) points outside the association prisms

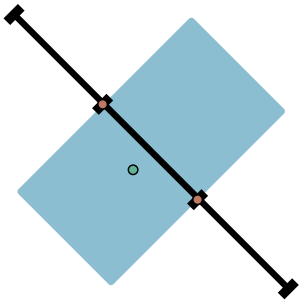


B1: noisy measurements and noisy reconstruction

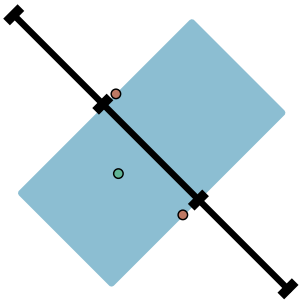


B2: misalignment

C) points on the association prism (optional)



C1: points coincide with face vertices



C2: points coincide with association prism boundary

Optionally, we reduce the association rate by declaring points that are projected onto the face edges and vertices to be *out-of-face points* (see Table 5.2, C). By these means, we avoid their association by choice. Technically, these points belong to at least two adjacent faces. Hence, it is hard to decide to which face they should be assigned. Therefore, such points cannot be associated unambiguously and may cause ambiguity in the information transfer. Here, our reasoning is to associate not all but unambiguous points. As a side-effect, neglecting these points accelerates the subsequent information transfer. However, if desired, our implementation allows us to associate those points, too (see Section 5.1).

Naturally, the PCImgA inherits the limitations of PCMA and ImgMA as it is a combination of both. Besides, there are specific issues regarding the PC visibility as determined via the mesh as a visibility model. Figure 5.15 depicts point visibility and showcases apparently visible points. Faces are marked as visible once a pixel is associated with them. Points, in turn, are marked as visible if they are associated with a visible face. However, a face marked as visible does not have to be entirely visible. Moreover, a face marked as visible does not have to be associated with points that are truly visible as well. The adaptive thresholding links points and faces along the normal directions, whereas the line-of-sight is relevant for the point visibility. The composition of PCMA and ImgMA does not guarantee a correct visibility check for each point-pixel relationship (explicit PCImgA) although taken individually, the made associations by the subpipelines may be correct (implicit PCImgA). For the explicit linking of the PCImgA, truly non-visible points are linked with pixels through collinearity equations causing incorrect information transfer. As discussed before, the implicit PCImgA makes use of the inherent co-registration of the mesh and imagery while overcoming minor local co-registration residuals in 3D space utilizing the adaptive thresholding. Therefore, the visibility checks for faces are only affected by the reconstruction quality of the mesh in case of deploying implicit PCImgA and MVS-steered reconstructions. Hence, implicit PCImgA is more robust than its explicit counterpart. In addition, the implicit PCImgA provides extensive transfer (per face) instead of point-wise propagation only.

Generally, small faces represent the geometry better (e.g., fine structures) and reduce the impact of truly non-visible points and non-class aware borders. However, large triangles are beneficial concerning processing time and noise reduction. We are aware of these issues regarding the association of 3D space and imagery. However, the thesis focuses on the linking of modalities in 3D space and the association of mesh with imagery (i.e., implicit PCImgA).

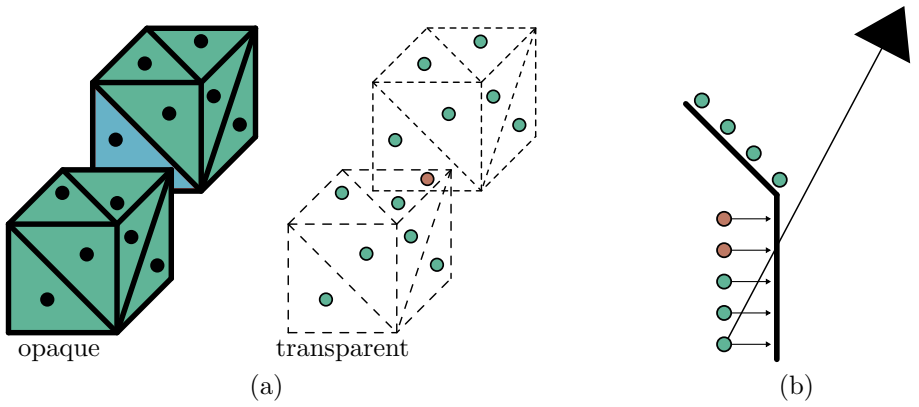


Figure 5.15: The schematic sketches indicate truly (*green*) and apparently visible points (*red*) as detected by explicit PCImgA utilizing the mesh (*black wireframe*) as the visibility model.
(a) showcases a scene from camera perspective in opaque (*left*) and transparent (*right*) fashion. Entirely visible faces are depicted in *green* in the opaque mode. The truly but non-fully visible face (*blue*) causes an apparently visible point. (b) showcases a scenario where the mesh surface and point cloud geometry differ (e.g., for buildings in 2.5D meshes). The entire mesh surface is visible in the depicted camera (*black triangle*). Hence, the associated points are marked as visible. However, only a few points are truly visible. The divergence of the surface normal (i.e., association direction for PCMA) and line-of-sight (i.e., association direction for 2D-3D linking) cause apparently visible points. The *red* marked points are not truly visible in the camera due to the roof overhang. However, they are linked to a visible face making them apparently visible.

5.5 Summary

In this section, we have presented in detail the explicit entity linking with the mesh as core modality (see Table 5.1). The holistic linking of imagery, PCs, and meshes explicitly associates their entities pixels, points, and faces. Their association allows flexible inter-modal information sharing. Measured and engineered features can be shared arbitrarily. Besides, predictions and manual annotations can be propagated across the modalities. The injected flexibility allows *Juggling with Representations*. Table 5.1 sketches the association mechanism denoting the entity relationships and summarizes the transfer operations dependent on the information type (feature or label).

The explicit entity linking and subsequent label transfer boost the GT generation by limiting the manual annotation to a single modality. Besides, the process guarantees consistent annotation across modalities as it avoids disjoint manual annotation processes. Section 6.1.2 gives a qualitative and quantitative analysis for the automatically transferred GT propagated from the manually annotated PC to the mesh and compares the results to a simple NN interpolation. Combining the proposed inter-modal linking with AL further reduces the manual annotation effort to few entities on a single modality (see Section 6.1.3).

By these means, the information transfer allows classifier training on modalities that have not been annotated manually and thus boosts semantic segmentation in general (see classifier training on the mesh in Section 6.5). Moreover, the explicit entity linking allows incorporating multi-modal information in the semantic segmentation processes. Modality-specific descriptors can be enhanced with information from other sources. For instance, we leverage the inter-modal entity linking to transfer PC features to the mesh (see Figure 5.7), enhancing per-face descriptors to multi-modal feature vectors.

Taken together, the proposed entity linking and subsequent information transfer inject great flexibility into the semantic segmentation of geospatial data. Imagery, PCs, and meshes can be semantically segmented with classifiers trained on any of these modalities utilizing features derived on any of these modalities. Figure 6.1 summarizes the versality of the semantic segmentation utilizing the entity linking as the backbone. Particularly, we can semantically segment a modality by training a classifier on the same modality (*direct* approach, see Sections 6.4.1 and 6.5.1) or by transferring predictions from other modalities (*indirect* approach, see Sections 6.4.2 and 6.5.2). Hence, any established well-performing modality-specific classifier can be used for semantic segmentation of these modalities – regardless of whether they follow an end-to-end learning or feature-driven scheme.

The bilateral association processes PCMA, ImgMA, and PCImgA are detailed in Sections 5.1, 5.2, and 5.3 respectively. In particular, we highlighted the implementation and tile-wise strategy enabling parallel, distributed processing. We put a big effort into the tile-based implementation to guarantee scalability to large-scale data sets. Chapter 6 validates its applicability on data sets V3D and H3D featuring different scales and resolutions of involved modalities.

We discussed the preconditions and limitations of the association process focusing on PCMA as the relationship of 3D space and imagery is well-defined by collinearity equations. Besides, the subsequent chapter focuses on the semantic segmentation of 3D modalities. The main condition for the entity linking and the label transfer is the co-registration of modalities which is implicitly given for geospatial data. Inter-modal discrepancies and the reconstruction quality of the mesh affect the association mechanisms (see Section 5.4).

Chapter 6

Multi-Modal Semantic Segmentation

The goal of semantic segmentation is to assign a class label to each entity, i.e., to each pixel, point, and face/CoG for imagery, PCs, and meshes, respectively. The mesh-centered multi-modal entity linking allows the sharing of features (inherent and engineered) and labels (predictions or GT) across modalities (see Chapter 5). Hence, the linking can be applied as a backbone for multi-modal joint semantic segmentation. Joint semantics refers to considering features from other modalities and consistent labeling across modalities. Figure 6.1 depicts the concept of multi-modal joint semantics utilizing the entity linking as the backbone.

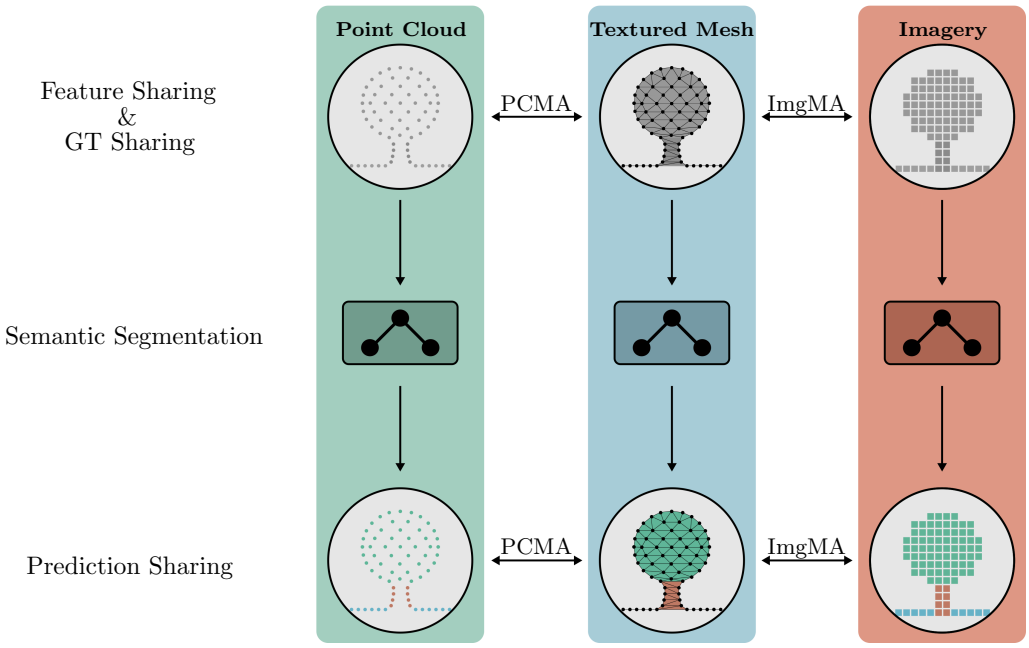


Figure 6.1: Compact overview of the joint semantic segmentation of the modalities imagery (*right*), PC (*left*) and mesh (*center*) utilizing the multi-modal entity linking as the backbone (see Chapter 5). The multi-modal entity linking enables the flexible sharing of features (inherent or engineered) and labels (GT or predictions) – before and after the semantic segmentation with a trained ML classifier. ImgMA establishes explicit links between pixels and faces (see Section 5.2). PCMA associates points with faces (see Section 5.1).

The proposed association method injects flexibility into automated scene interpretation. The backbone allows the semantic segmentation of the mentioned modalities in a *direct* and *indirect* manner. For both versions, modality-specific feature vectors can be enhanced to multi-modal descriptors with the feature

transfer as described in Chapter 5. Direct semantic segmentation refers to the conventional way with classifier training and inference on the same modality. The indirect approach entails classifier training and inference on another modality, followed by a subsequent prediction sharing causing a consistent multi-modal annotation. For instance, the mesh can be semantically segmented in a direct manner utilizing supporting features from the PC and imagery. Afterward, the prediction results can be transferred to the supporting modalities leveraging the established connections on the entity level. The prediction sharing leads to an indirect semantic segmentation of the PC and imagery. Please note the modality of the direct semantic segmentation can be chosen arbitrarily thanks to the integrative backbone. Consequently, best-performing classifiers for any modality might be plugged into the pipeline.

Nowadays, ML classifiers are the tool of choice to interpret the huge amount of acquired data. The training process of supervised ML classifiers requires GT data. We highlight various GT generation methods in Section 6.1. Feature-driven traditional ML approaches call for feature engineering. Section 6.2 lists the engineered and inherently available features for colorized PCs and textured meshes.

The availability of GT and features enables the training of both feature-driven traditional ML classifiers and end-to-end learning approaches. In this work, we deploy well-established and fast RF models in favor of large-scale data sets. We argue that their performance can compete with end-to-end learning approaches on 3D data since informative feature engineering is feasible in 3D space[1]. Moreover, one of our goals is an extensive ablation study to evaluate the impact of the deployed features and multi-modality. In our opinion, the comparison has a higher level of validity when features are kept fixed in the training process. End-to-end learning approaches would give up control by feature learning and may affect comparability. However, we emphasize that any (end-to-end learning) classifier may replace the RF models. In previous work, we analyzed the performance of RF models, multi-branch 1D CNN as adopted from George et al. (2017), PointNet and PointNet++ on the 2.5D mesh of Hessigheim data (Laupheimer et al., 2020a; Tutzauer et al., 2019). Besides its rapidity, the RF classifier captivates due to its interpretability and explainability leading to insights on feature relevance. Thanks to the bootstrapping scheme, off-the-shelf RF models are robust against outliers and label noise. Furthermore, RF classifiers can be plugged easily into the AL pipeline of Section 6.1.3 since the required features can be computed at once in advance of the AL loop (in contrast to end-to-end learning approaches). For all experiments, we utilize RF models as off-the-shelf supervised ML classifiers with a fixed parametrization for the sake of fair comparison. The RF hyper-parameters have been determined empirically by grid search on a small subset of the H3D(PC) train set beforehand, considering PC-inherent features only. The deployed RF models consist of 100 binary decision trees with a maximum depth of 18.

The current chapter concentrates on 3D scene interpretation. The utilized evaluation metrics tailor-made for the mesh are quickly reviewed in Section 6.3. We focus the analyses on the direct semantic segmentation of meshes due to their favorable properties (see Chapter 1) and due to being the core of the multi-modal entity linking. Hence, the mesh is the main modality of the underlying thesis. In Section 6.1 and Section 6.4, we show the utility of the mesh texture for manual and automatic PC labeling. Section 6.4.1 discusses the direct semantic segmentation of PCs for V3D and H3D (see Chapter 4). In Section 6.5.1, we deep-dive the direct semantic segmentation of meshes with an in-depth analysis of various feature vector compositions. Table 6.1 gives an overview of the studied feature sets $\mathcal{FS}$. Section 6.4.2 and Section 6.5.2 discuss the indirect semantic segmentation of PCs and meshes respectively and compare the results to the respective direct semantic segmentation (see Section 6.4.1 and Section 6.5.1). We want to stress that the thesis does not cover the broad research field of semantic image segmentation with its specific particularities. However, the proposed entity linking still allows the indirect semantic image segmentation and provides GT data for images in an automatic manner (see Figure 6.7).

The coupling of the PC and the mesh via the multi-modal entity linking allows for an entangled annotation, classifier training, and evaluation. Section 6.6 analyzes the performance of AL classifiers on the PC and the mesh. The classifiers are trained on a variety of sparse, iteratively refined training pools, which are annotated by crowdworkers (see Section 6.1.3).

Editor's note: We summarize the key findings of the performed experiments at the end of each section.

[1] see H3D benchmark results (April 2022) `https://ifpwww.ifp.uni-stuttgart.de/benchmark/hessigheim/results.aspx`

Table 6.1: Overview of the feature vector compositions $\mathcal{FS}$ sorted into $\mathcal{FS}$ groups "mesh-only", "PC-only" and "multi-modal". The computed features are listed in Table 6.3 and Table 6.4 respectively and are derived for both data sets V3D and H3D. The listed $\mathcal{FS}$ incorporate contextual features derived from spherical vicinities of radii $r \in \{1\,\text{m}, 2\,\text{m}, 3\,\text{m}, 5\,\text{m}\}$. For H3D, we additionally calculate contextual PC features on small-scale radii $r \in \{0.125\,\text{m}, 0.25\,\text{m}, 0.5\,\text{m}, 0.75\,\text{m}\}$ to account for the high point density (see Section 6.2). Configurations extended by features based on the small-scale radii are marked with a prime in the following sections. Owing to PCMA, each $\mathcal{FS}$ can be realized on both the mesh and the PC.

$\mathcal{FS}$	**Description**
"mesh-only"	
a	Mesh-intrinsic geometric mesh features (without relative height)
b	Geometric mesh features (mesh-intrinsic + relative height)
c	Radiometric mesh features (texture)
d	Geometric & radiometric mesh features (all mesh features)
"PC-only"	
e	PC-intrinsic geometric PC features (without relative height)
f	Geometric PC features (PC-intrinsic + relative height)
g	Radiometric PC features (reflectance)
h	Geometric & radiometric PC features
i	Geometric & radiometric PC features and echo characteristics (all PC features)
"multi-modal"	
j	Geometric & radiometric features from both modalities and echo characteristics (fusion of "mesh-only" and "PC-only" features)
k	Geometric & radiometric PC features, echo characteristics, and radiometric mesh features (fusion of "PC-only" features and mesh texture; sparse version of $\mathcal{FS}_j$)

6.1 Ground Truth Generation

As discussed in Section 3.2, (modality-wise) GT availablility is a crucial issue wherefore GT generation is a highly relevant task. We discuss the tedious manual annotation by example of PCs and meshes of the Hessigheim data (see Section 6.1.1). To reduce the manual labeling effort, we utilize a) the multi-modal entity linking for a semi-automatic consistent multi-modal labeling (see Section 6.1.2) and b) AL for limiting the manual labeling effort to a small number of entities (see Section 6.1.3). This is in contrast to PL where commonly all entities are annotated. Integrating the entity linking into the AL pipeline further limits the manual labeling effort to a few entities on a single representation.

Regardless of the deployed annotation method, we attach the assigned labels to the *las* file or *ASCII* file in case of PC annotation. Likewise, we extend the respective *ASCII* file of the CoG cloud $\mathcal{M}_{\text{PC}}$ with the annotated class labels. Its graph-based counterpart $\mathcal{M}_G$ attaches a specific material with a unique RGB tuple to each face. Accordingly, the labeled $\mathcal{M}_G$ consists of an *obj* file (containing the same geometry as the respective textured version) and a *mtl* file, which encodes the class pseudo-texturing of the mesh. The semantic per-face label can be retrieved from the RGB tuple by parsing the labeled mesh (*obj* + *mtl* file) with a data-set-specific lookup table.

6.1.1 Manual Annotation by Experts

Manual annotation is inevitably necessary to train supervised ML classifiers. Commonly, experts who are familiar with the kind of data fully annotate the data since GT determines the effectiveness of the data-driven trained classifiers.

The main issue of manual GT generation is that the annotation is very tedious and time-consuming work. We refrained from manual GT generation for imagery since we are interested in 3D semantics and omit the intensive label work correlating with overlapping images. In this work, we manually annotated PCs and meshes with the support of student assistants. The data has been annotated based on the corresponding class catalogs presented in Chapter 4. Students received explicit instructions on how to label the data and distinguish between different semantic classes. To handle the huge amount of data, we first split the data into tiles and provided them to studentish annotators. Subsequent quality control was accomplished in a two-stage procedure. First, student assistants cross-checked each other's labels for each tile, and finally, various Ph.D. candidates verified and harmonized the annotations as the last instance after the tile merging. The harmonization phase particularly ensures consistent labeling close to former tile borders. Figure 6.2 sketches the workflow of the manual annotation by experts.

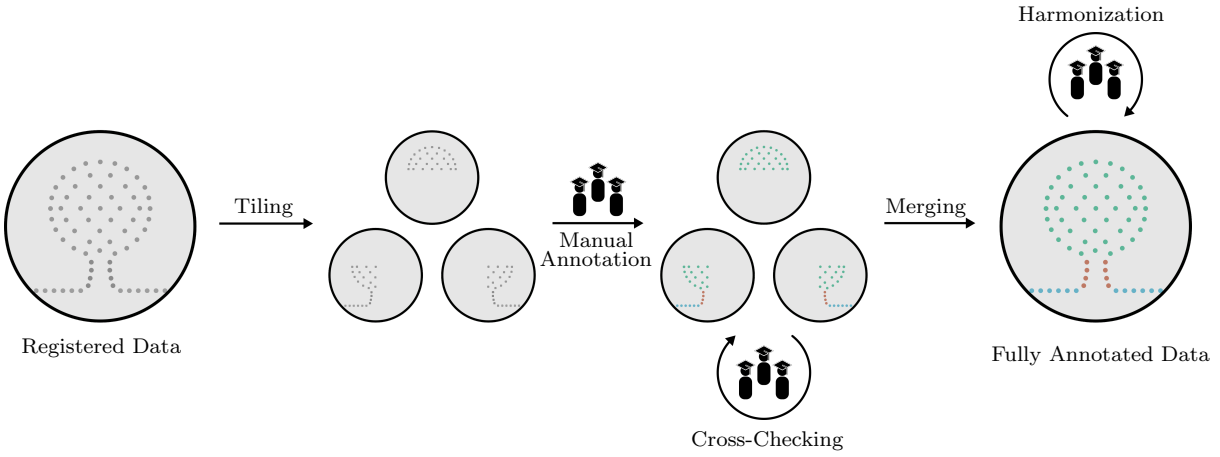


Figure 6.2: Workflow of the manual GT generation exemplarily depicted for PC data. The workflow can be applied to 3D and 2.5D meshes likewise. Experts provide labels for each instance, cross-check, and harmonize the annotations. The cross-checking may eliminate label noise. The harmonization may avoid label ambiguity.

The complexity of the real world makes it hard to map the data onto the defined class catalogs. Eventually, the label assignment is a subjective decision. The harmonization phase tries to remove the subjectiveness or, at the least, tries to provide consistent subjective decisions. For instance, overgrown paths might be annotated as *Low Vegetation* or *Impervious Surface* (see Figure 6.3, *center, right*). Similarly, greened flat roofs can be labeled as *Roof* or *Low Vegetation* (see Figure 6.3, *left*). The harmonization tackles label ambiguity and strives for consistent intra-modal labeling. Our goal was to provide an annotated data set offering a broad application range. However, the final decision on a label is always application-dependent, wherefore the generic labels might not be adequate for specific applications. For instance, one might be interested in the total amount of greened spaces in cities. In that particular case, annotating the greened flat roof as *Low Vegetation* is more appropriate than labeling it by its functionality. Despite the comprehensive quality check, we are aware of the fact that manual labeling is an error-prone endeavor and that operators might miss or misclassify some entities during labeling due to subjective perception or occlusions. Errors cannot be avoided, although the manually attached labels are counter-checked. The so-called label noise might complicate the training procedure and certainly affects performance evaluation. The cross-checking and harmonization are dedicated to eliminating label noise and label ambiguity.

The label process of the 3D PC of Hessigheim was carried out with software `CloudCompare` (Girardeau-Montaut, 2021). The PC was manually segmented into subsets of homogeneous class membership. These segments have been merged afterward into subsets that carry points of a single class only. The respective class labels associated with distinct RGB tuples have been assigned to the points accordingly. The gross labeling time is several months with 4 students working in parallel (cost: approx. 22.5k €). The annotated PC is provided to the community as part of the H3D benchmark (Kölle et al., 2021a).

Figure 6.3: Example snapshots taken from the textured mesh that demonstrate the complexity of the real world resulting in label ambiguity. The assigned label depends on the annotator's perception and class catalog. The flat greened roof on the *left* might be annotated as *Roof* or *Low Vegetation* – both labels are reasonable. The figures in the *center* and *right* depict human-created structures that have been overgrown and eroded over time. Should they still be annotated as *Impervious Surface*?

Apart from the manually labeled PC, we also provide annotated meshes of the Hessigheim data. Particularly, we manually annotated the entire project area for the 2.5D mesh and a small subset of the H3D test set. Please note only a small subset of the 3D mesh has been labeled due to time restrictions (see Figure 4.5). We claim the manually annotated subset is sufficient to independently evaluate the accuracy of the semantic segmentation on the 3D mesh (see Section 6.5). Besides, we validate the multi-modal entity linking by comparing the assigned labels with the automatically transferred labels from the PC (see Section 6.1.2).

The 2.5D mesh was used to analyze the robustness of the entity linking against the 2.5D mesh geometry (Laupheimer et al., 2020b) and to gain first insights into the semantic mesh segmentation (Tutzauer et al., 2019). For the labeling process of the 2.5D mesh, the software `Autodesk 3ds Max 2019` (Autodesk, 2019) dedicated to 3D modeling and rendering has been used. In total, the manual labeling of the 2.5D mesh lasted approximately 300 working hours (net working time).

We noted that student assistants coped better with `CloudCompare` than `Autodesk 3ds Max 2019`. The students familiarized themselves faster with `CloudCompare`, and they needed less assistance when working with this software. Based on this observation, we utilized the CoG representation to manually annotate the 3D mesh of Hessigheim, adopting the class catalog and the manual labeling process carried out on the Hessigheim PC. The textured 3D mesh has been used as visual support during the CoG annotation. Both the CoG cloud and graph representation are annotated after the labeling process. The manual annotation of the subset took approximately 60 h (net working time) in total. Please note that the labeling times between the 2.5D mesh and the 3D mesh are difficult to compare as the granularity of the 3D mesh is significantly higher – both regarding the classes and the face density.

6.1.2 Semi-Automatic Annotation Utilizing Multi-Modal Entity Linking

The multi-modal entity linking allows the sharing of (manually attached or automatically predicted) labels across modalities. Hence, modalities can be labeled consistently in a semi-automatic manner once one of the modalities has been manually annotated. In other words, the multi-modal entity linking and the subsequent automatic propagation of manually attached labels boost the annotation process by limiting the time-consuming and expensive labeling effort to a single representation. In particular, tedious and redundant labeling of overlapping images can be reduced once one of the 3D modalities has been annotated.

To the best of our knowledge, there is no geospatial real-world data set that provides annotations for PC,

mesh, and imagery at the same time. The majority of geospatial benchmark data provides annotated LiDAR PCs (see Section 3.2). Therefore, we showcase the semi-automatic labeling by transferring manual annotations from the PC to the mesh and images as indicated for a dedicated tile of H3D by Figure 1.1 (Laupheimer and Haala, 2021). However, we emphasize that any modality can be the annotated starting point for the semi-automatic labeling process (see Table 5.1, *top*). For instance, information can be transferred from a manually annotated 2.5D mesh to the respective 3D LiDAR PC (Laupheimer et al., 2020b). Figure 5.7 shows the PC and the respective 3D mesh of H3D side-by-side highlighting various attributes including labels. The annotations have been automatically transferred from the manually annotated PC (2^{nd} *row*). The same holds for the visualized features, i.e., number of returns and reflectance (3^{rd} *and* 4^{th} *row*). There are almost no differences notable between the modalities showing that the information transfer works reasonably – both for discrete and continuous quantities. The semi-automatically annotated H3D mesh data is provided to the community as part of the H3D benchmark data to fill the void of labeled meshes (Kölle et al., 2021a). Figure 6.7 shows a variety of labeled oblique and nadir images covering Hessigheim or Vaihingen after the indirect label transfer from the PC via the mesh as mediator (implicit PCImgA).

In the following, we discuss the semi-automatic labeling of mesh and imagery by deploying PCMA and implicit PCImgA by the example of H3D and V3D. We use the established entity connections to transfer the manual annotations from the PCs to the meshes and images. We validate the label propagation qualitatively and quantitatively, which in turn validates the feature sharing at the same time. We contrast labels propagated with our mechanism to a simple nearest (inter-modal) neighbor interpolation for comparison. Qualitative results of the annotated images and meshes are discussed in the follow-up paragraph *Qualitative Analysis* and illustrated in several figures (see Figures 1.1, 6.4, 6.5, 6.6, 6.7).

The paragraphs *Consistency Analysis* and *Quantitative Analysis* focus on the threshold-depending PCMA since the relationship of image space and 3D space is well-known and strictly defined by collinearity equations. Hence, the relationship does not have to be validated. The adaptive thresholds of PCMA facilitate an arbitrary degree of freedom by their set magnitudes and inter-level spacing. Small thresholds and small inter-level spacing enable tight coupling of mesh and PC. In contrast, loose coupling uses large thresholds and large inter-level spacing. We heuristically adapt the multi-level set of thresholds to the investigated data sets and their respective circumstances by considering the point-to-face distances between the meshes and the PCs. In particular, we derive the median of absolute point-to-face distances for a randomly sampled subset of tiles before calling PCMA for the entirety of tiles in a parallel fashion. The derived median absolute differences are 16.8 cm for V3D and 4.0 cm for H3D. We use these values as yardsticks to heuristically set the first threshold. For H3D, the inter-level spacing equals the first threshold resulting in a tight coupling of PC and mesh. For V3D, we loosely couple both modalities by doubling the threshold in each level. For the used data sets consisting of 3D PCs and 3D meshes, we use symmetric thresholding and set values heuristically as follows: $\theta_1 = \pm 15\,\text{cm}$, $\theta_2 = \pm 30\,\text{cm}$, $\theta_3 = \pm 60\,\text{cm}$ (V3D) and $\theta_1 = \pm 5\,\text{cm}$, $\theta_2 = \pm 10\,\text{cm}$, $\theta_3 = \pm 15\,\text{cm}$ (H3D). By these means, we account for the co-registration quality and reconstruction quality of both data sets (see paragraph *Qualitative Analysis*, Figure 6.5). Please note the comparably large thresholds θ_2 and θ_3 are used for less than 30 % of associated faces in data set V3D and thus enable reasonable association. The vast majority of these faces (67 %) associates sub-surface vegetational points. For associating the 2.5D mesh and the 3D PC of Hessigheim, we utilized asymmetric thresholds to better adapt to the inter-modal discrepancies (Laupheimer et al., 2020b).

We perform a forward and backward passing of labels to check the consistency of the transferred information and to validate the concept (label consistency check). For this reason, we transfer labels from the PC to the mesh (forward pass) and therefrom back to the PC (backward pass). During forward passing, we aggregate labels on the face level via majority voting. The backward pass is a straightforward copy operation utilizing the stored association information. Hence, the consistency analysis merely requires GT for the labeled starting modality to validate the effectiveness of PCMA (Laupheimer and Haala, 2021). The consistency of forward and backward passed labels can be abused as a proxy measure for correctness when working with co-registered modalities. Geospatial data feature implicit co-registration of different data sources due to their georeference. However, the proxy measure has to be taken with care when operating with non-co-registered data or coarsely co-registered data. In that case, forward and backward passing can-

not detect false associations based on coarse co-registration (see Section 5.4). For example, an erroneous link of a facade in the PC with a tree in the mesh will cause a wrongly labeled tree in the mesh, but the back-propagated label on the PC will be correct again. For this reason, the consistency check has to be combined with a visual quality check for good measure (see paragraph *Qualitative Analysis*).

Another option to verify the transferred labels (and features) without independent GT on another modality is to extend the consistency check with a training process. The indirect verification encompasses the transfer of features and GT labels to the mesh, the training of a classifier on the mesh, and the back transfer of its predictions to the PC. Instead of checking the consistency of forward and backward passed GT labels, the indirect analysis checks the consistency of GT and backward passed predictions (see Section 6.4). The reasoning behind the indirect analysis is that only classifiers trained on a proper linking will achieve good performance on the starting modality. Erroneous connections will transfer wrong GT labels and features to the mesh that do not match with the radiometric and geometric features calculated on the mesh itself. As a consequence, the training and feature mapping is impaired as feature vectors lose distinctiveness.

However, true quantitative analysis is only possible with independently acquired GT (see paragraph *Quantitative Analysis*). We use the manually labeled submesh of H3D to quantify the transfer accuracy.

Qualitative Analysis. We qualitatively verify the effectiveness of the explicit entity linking by visualizing the transferred labels. In our view, human inspection is a powerful qualitative analysis step that does not depend on tediously generated GT data. Human interaction is inevitable for the task of GT generation at some point. Nonetheless, visual inspection is often underestimated due to its non-automatability.

Figure 5.7 visualizes features and labels on the mesh that have been transferred from the PC. The side-by-side comparison with the PC hints at a reasonable entity linking and correlating information transfer. Figure 6.4 depicts the H3D mesh annotated by two different label transfer strategies. Our mechanism is contrasted with a simple nearest (inter-modal) neighbor interpolation. The quantitative analysis of both strategies is discussed in the subsequent paragraph *Quantitative Analysis*.

A striking difference is that NN interpolation generates fully annotated meshes leading to visually appealing results at first glance. In particular, faces that are not linked by the PCMA benefit from the interpolation when they are surrounded by correctly annotated data points. In that case, the interpolated labels match reality, and data gaps are closed reasonably, as can be seen for some facades, e.g., the facade of the church tower (2^{rd} *row*). The facade is covered by a single LiDAR strip, wherefore the scan lines are visible for the PC – and the mesh when propagating labels with our approach.

However, the interpolation causes wrong labeling when NNs carry label noise. The interpolation even increases the label noise on the mesh since a single wrongly labeled point is treated as a seed point for the interpolation process. The 4^{th} *row* shows the influence of label noise in the PC on the mesh. A single point in between the two buildings on the left-hand side has been annotated erroneously as *Roof*. The label noise is limited to a single entity and therefore almost not visible on the PC and the mesh annotated with our approach (in between the two buildings in the black area). In contrast, the label noise causes an area labeled as *Roof* for the interpolated version, which is even visible in the snapshot of Figure 6.4. As the label noise is located outside the annotated PC, the respective mislabeled area is surrounded by another mislabeled area due to interpolation. The example shows that the NN interpolation introduces errors for data gaps – even when surrounding points might carry correct labels. Likewise, facades receive wrong labels as visible in the examples of the 1^{st} and 3^{rd} *row*. Hence, the interpolation further increases label noise. In both cases, our mechanism transfers information only where an explicit connection is given. Therefore, our approach does not increase label noise and provides annotations consistent with the manually annotated starting modality.

The *last row* depicts that our approach manages to connect complex structures reasonably, e.g., fine structures in the lock area. Therefore, our approach benefits from continuously improving meshing approaches and the respective improved geometry.

The example in the 1^{st} *row* illustrates that both transfer options manage to provide the correct label despite differing geometry in both modalities. Please note that the PC contains more cars than the reconstructed mesh. The reason for that is the different asynchronous data acquisition of PC and oblique imagery. The LiDAR PC is composed of several LiDAR strips. Each strip captures stationary and moving

cars, which form part of the final LiDAR PC. The hybrid mesh uses both data sources: camera (MVS PC) and LiDAR data (LiDAR PC). During meshing, the MVS cloud has been detected as more reliable, wherefore the movable cars form not part of the mesh. Both transfer strategies provide correct labels since some LiDAR strips cover the street below the car. Therefore, the annotated PC provides labels below the car, too. The combination of adaptive thresholding and "early stopping" guarantees to link faces with the close-by points only carrying the relevant labels.

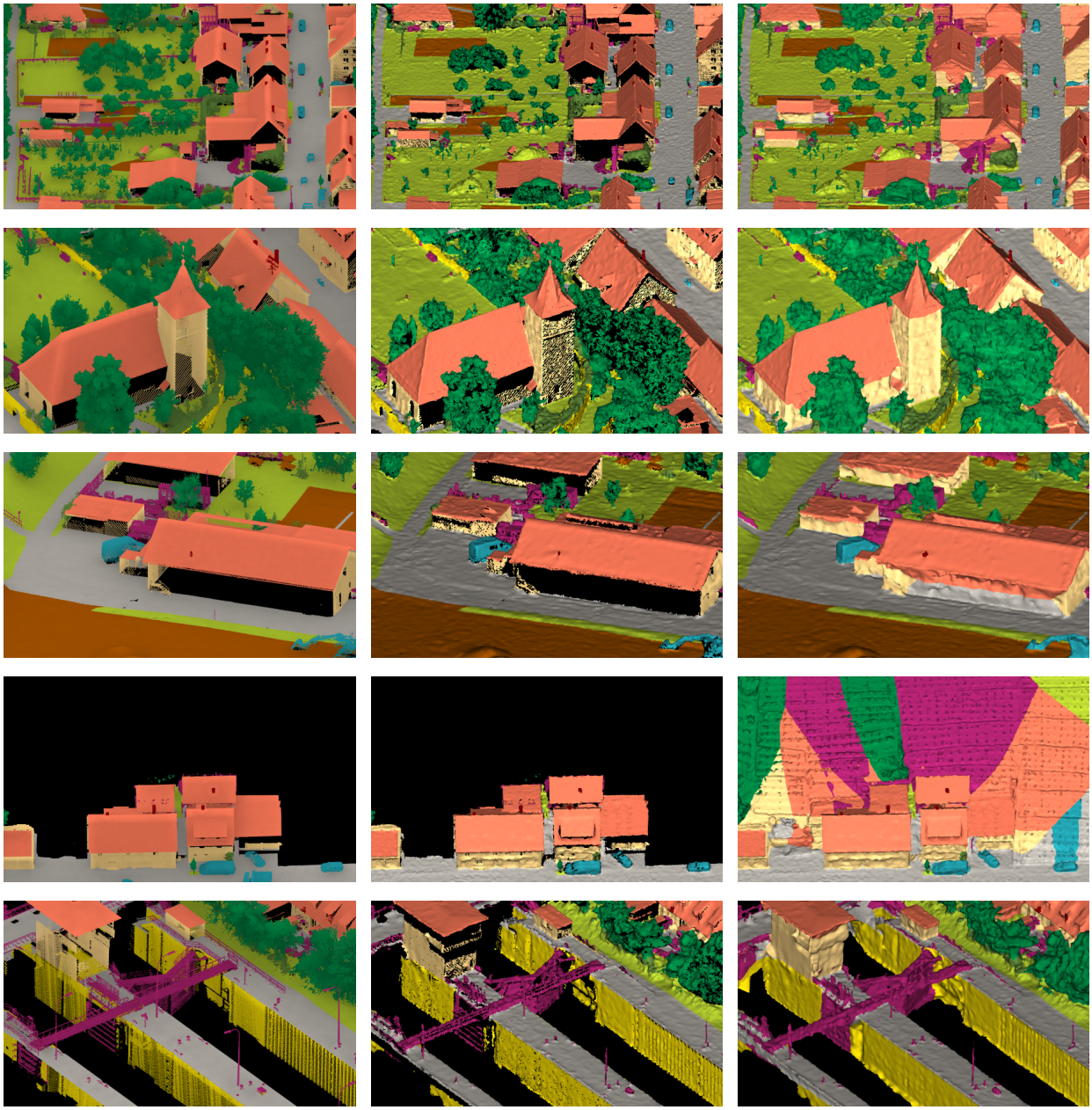


Figure 6.4: The *left* column shows the manually labeled LiDAR cloud of H3D. The other columns show the automatically annotated mesh as a result of the information transfer via a) our multi-modal entity linking (*center*) and b) NN interpolation (*right*).

Figure 6.5 shows the effectiveness of this combination by the example of the fringe area of V3D. The height-coding indicates reconstruction errors in the leftmost part of the photogrammetric mesh: the building and tree are not reconstructed. The linking falls back on higher-level thresholds when PC and mesh geometry differ significantly. The falsely reconstructed faces at the ground remain unlabeled, showing that the comparably large thresholds link PC and mesh still reasonably. At the same time, the adaptive thresholding guarantees a correct linking of faces and points where geometry is reconstructed correctly.

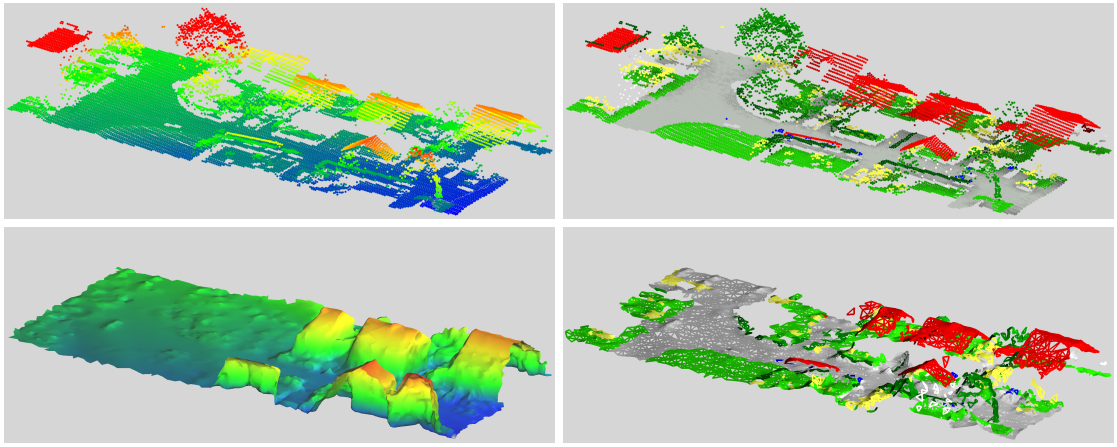


Figure 6.5: Fringe area of V3D represented by the PC (*top*) and the MVS mesh (*bottom*). The left column shows 3D data in height-coded fashion (*blue*: low, *red*: high). The top right shows the manually annotated PC; the bottom right shows the automatically annotated mesh as a wireframe. Faces that do not map the real geometry are not linked to the PC, and hence, remain unlabeled (see holes in the leftmost part for tree and building).

We already demonstrated the effectiveness of the non-symmetric adaptive thresholding for the challenging 2.5D mesh geometry in (Laupheimer et al., 2020b). The asymmetric thresholds per level allow adaption to the 2.5D geometry and the respective discrepancy between PC and 2.5D mesh (see Figure 5.14). To this end, PCMA works for both 2.5D and 3D meshes.

Figure 6.6 shows 3D meshes of different reconstruction qualities generated from different data sources and different meshing software. The hybrid mesh generated with `SURE` (*left*, Rothermel et al., 2012) reconstructs better vegetation and fine details, e.g., the church spire. The MVS mesh generated with `Metashape` (*right*, Agisoft Metashape, 2018) has significantly fewer faces resulting in smoother surfaces (e.g., the street). Therefore, both meshes feature different face sizes, face counts, and class distributions. The comparison demonstrates that the established entity connections are reasonable, and the transfer works for both kinds of meshes. The qualitative inspection verifies that the association mechanism is not confined to a specific meshing procedure and mesh type.

Figure 6.7 shows a selection of automatically annotated images for both data sets as derived via the implicit linking of PCImgA. We opt for the implicit linking to create densely labeled images as faces are projected to image space instead of single points. Furthermore, implicit PCImgA makes use of the perfect co-registration of MVS mesh and imagery for V3D. At the same time, the enclosed PCMA can cope with the co-registration discrepancy in 3D space by leveraging the adaptive thresholding. Hence, the implicit linking avoids inconsistent point-pixel pairs. Besides, Figure 6.7 reveals the dependence of mesh quality. The picture on the center-right depicts a ship (class *Vehicle*) in the ship lock surrounded by water. Since the mesh does not properly reconstruct the ship, the transferred labels do not cover the entire ship in image space.

Consistency Analysis. The comparison of forward and backward passed labels between co-registered modalities allows the validation of the semi-automatic labeling and may be considered as proof of concept.

For V3D, the adaptive thresholding associates 40.9% of faces covering 53.8% of the surface area with 75.6% of LiDAR points. The proxy analysis reveals that 98.9% of associated points show coincidence in

Figure 6.6: Annotated meshes as achieved by the label transfer with PCMA. The hybrid mesh generated with `SURE` from nFrames (*left*) used oblique images and LiDAR data for the reconstruction. The MVS mesh generated with `Metashape` from Agisoft utilized oblique images only. Whereas the hybrid mesh features more details, the pure photogrammetric mesh provides smoother surfaces. The *first* row shows an overview, the *second* row a more detailed view. The *third* row shows the wireframe of the *second* row to give a better impression of the varying face counts, distributions, and sizes. The *last* row shows two close-ups of the wireframe.

Figure 6.7: Automatically annotated images with labels as transferred from the PCs to the meshes and, therefrom, to image space (implicit PCImgA) for H3D (*top*: oblique, *center*: nadir) and V3D (*bottom*). Unlabeled pixels are depicted with RGB values. Pixels remain unlabeled when the respective 3D space is unlabeled or suffers from reconstruction artifacts as depicted for the ship in the 2^{nd} *row*. For a better assessment of the annotation quality, RGB images are shown on the left except for the nadir image of V3D which covers the entire labeled area.

manual and backtransferred labels. 2.0% of associated faces are linked to points carrying various labels causing label inconsistencies for 1.1% of associated points. The achieved weighted average precision of the label consistency check is 98.9%.

For H3D, the adaptive thresholding associates 67.3% of faces covering 71.2% of the surface area with 55.9% of LiDAR points. 99.6% of the associated points pass the label consistency check. Vice versa, 0.9% of associated faces are linked to points carrying different labels. The weighted average precision is 99.9%.

Different extensions of modalities and structural differences among 3D modalities prevent full association of points and faces for both data sets (see Figure 5.13 and facades in Figure 6.4 or Figure 6.5). Please note the apparently low association rates on the mesh level refer to the overlapping area between mesh tiles and PCs (see Figures 4.2 and 4.5). Particularly semi-transparent objects reduce the association rates on the PCs as sub-surface points lack a partner entity in the mesh. The majority of non-associated points belong to vegetational classes (V3D: 68%, H3D: 74%). For the latter, 45% of non-linked points belong to class *Tree*. The high LiDAR density of H3D causes a low association rate on the point level. In contrast, the comparatively high association rate on the face level indicates proper co-registration and high-quality mesh. For V3D, 15.5% of non-linked points belong to *Facade* and *Roof* pointing out the minor MVS mesh quality.

The proxy analysis reveals that the majority of established point-face connections is consistent for both data sets (98.9%/99.6%). Likewise, the forward-backward-pass shows that common meshing does not incorporate semantic borders. Consider the transition of a planar *Impervious Surface* to *Low Vegetation*, which may be simplified to one large face in the mesh. Figure 6.8 shows inconsistencies for the entire H3D data set and as close-up. The overview (*top*) exhibits 0.4% of points failing the label consistency check. These points represent semantic borders and indicate improper mesh reconstruction and difficulties in annotating points on semantic borders. We are aware of the fact that co-registration discrepancies in 3D space increase this effect. Due to the high mesh quality and small co-registration discrepancy, H3D is less affected than V3D.

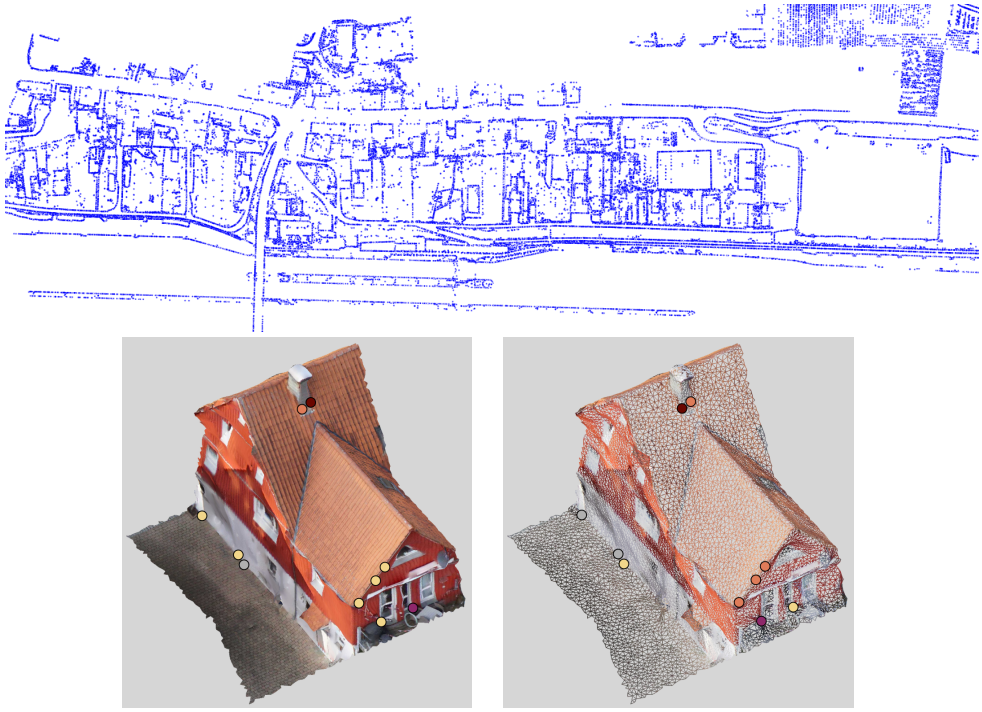


Figure 6.8: The *top* depicts the overview of points that show inconsistencies between manual annotations (assigned to the point cloud) and back-transferred labels (propagated from the mesh, annotated with the multi-modal information transfer) for data set H3D. North points towards the right. The close-up at the bottom depicts inconsistently labeled points marked by the manually annotated GT on the textured mesh (*left*) and the back-transferred labels on the wireframe (*right*). The classes are depicted with the H3D color scheme.

As a side effect, the proxy analysis of forward and backward passing labels helps to detect label noise on the PC. Figure 6.9 depicts a building in V3D as PC (*left*) and wireframe (*right*). The transferred labels on the mesh (*lower right*) seem not to match the given GT on the PC (*upper left*). For instance, some faces on the facade are marked as *Roof*. Consequently, the backtransferred labels to the PC (*lower left*) do not match with the initial labeling. Here, the appearance of false labels hints at label noise, since inconsistencies cannot be explained by georeference issues. Figure 6.10 shows the GT of the building along with its class-wise GT for classes *Facade* and *Roof*. The figure discloses that few points carry labels of both classes.

Quantitative Analysis. The quantitative analysis of the information transfer PC → Mesh is performed on the manually annotated part of the test set of the H3D mesh (see Section 4.2). We compare two automated label transfer options quantitatively: i) the label transfer leveraging PCMA (*ours*) and ii) the label transfer achieved by a simple NN interpolation across modalities. Figure 6.4 depicts the results of both transfer options next to each other. Figure 6.11 compares the corresponding normalized confusion matrices where the manually assigned face labels are used as GT (*Subfigures a* and *b*). Additionally, *Subfigure c* analyzes the NN-interpolated labels on faces where our method does not transfer any labels. The surface-aware evaluation metrics are highlighted in Section 6.3.

Please note that GT generation is prone to label noise and label ambiguity (see Section 6.1.1). Therefore, the manually assigned labels on the PC and mesh do not match perfectly. The differing visual perception of both modalities during manual annotation might further increase inter-modal GT discrepancies. The mismatching GT is a systematic defect that restricts the achievable transfer performance. However, that natural limitation holds for any label transfer method as they rely on the same GTs on the mesh and the PC. Hence, the comparison still points out which method performs better.

The chosen threshold set achieves a $mF1$-score of 93.24 % and an area-aware Overall Accuracy (OA) of 96.71 % for the transferred labels. In contrast, NN interpolation correctly annotates 86.76 % of the surface area with a $mF1$-score of 80.23 %. The global comparison suggests that our method outperforms the NN interpolation. Both propagation methods have significant problems on a similar level in separating *Shrub* from *Tree*. This is a direct effect and a prime example of label ambiguity. Separating a tree from a bush highly depends on the annotators' perception. Figure 6.3 illustrates real-world scenarios that may result in label ambiguity.

The confusion matrices of Subfigures a and b indicate that the NN interpolation injects new label noise. For example, 20.38 % of *Vertical Surface* receive the interpolated label *Facade*. The PCMA-based transfer outperforms the interpolated labels for the per-class precision values – particularly for classes *Soil/Gravel* and *Vertical Surface*. The dedicated performance analysis on faces that do not receive a label by our transfer method (*Subfigure c*) shows that NN interpolation introduces label noise for adjacent classes (e.g., *Facade* & *Low Vegetation*, *Roof* & *Facade*). Moreover, the interpolation of labels in data-gap regions show increased confusion of non-adjacent classes (e.g., *Chimney* and *Soil/Gravel*). In general, 42.12 % of the non-linked surface area receives a wrong label by the NN interpolation. The analysis underlines the superiority and effectiveness of our approach as linking method.

Please note that the comparison of Subfigures a and b bases on differently sized subsets of the manually annotated mesh as NN interpolation annotates all faces whereas the PCMA-steered transfer is limited by the detected point-face associations. The filtering of the NN-interpolated result to faces that have been linked to the PC allows evaluating the pure impact of the label propagation methods, i.e., the per-face aggregation. In other words, we can compare the impact of the majority vote and the simple NN interpolation. The majority of linked faces receive the same label from both approaches. Differences are only observed at class borders. We see that the simple NN interpolation is 1.21 pp and 0.59 pp better for $mF1$ and OA respectively than the majority vote. We assume that the non-class aware meshing and remaining co-registration effects are the reason for this behavior.

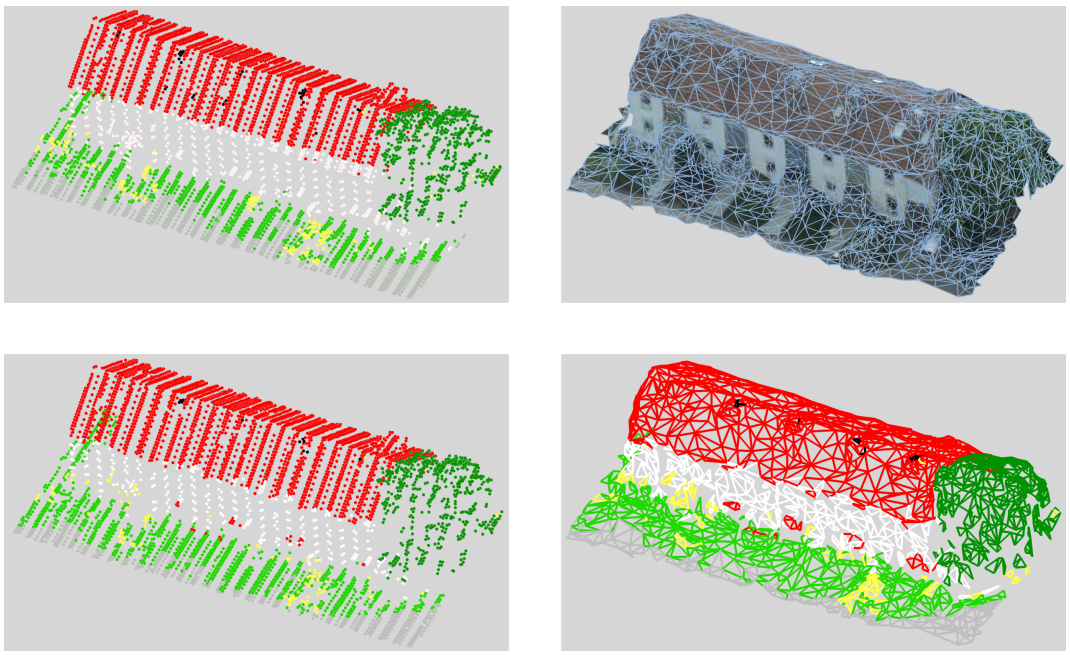


Figure 6.9: Visualization of the forward and backward pass of manual GT between the PC (*left*) and the mesh (*right*) for a specific building in data set V3D. The top row shows the manually annotated PC and the unlabeled, textured mesh overlaid with its wireframe. The bottom row shows the automatically labeled mesh (PC $\mapsto$ Mesh) and the respectively labeled PC with back-transferred labels from the mesh (Mesh $\mapsto$ PC). The emerging inconsistencies are discussed in the text and detailed in Figure 6.10.

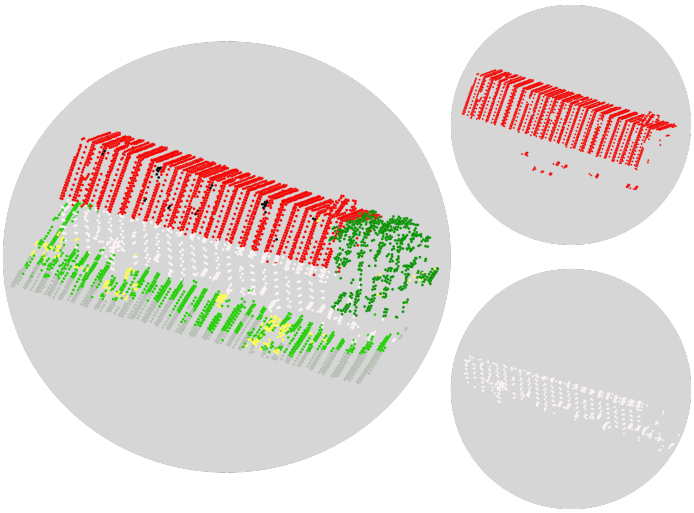
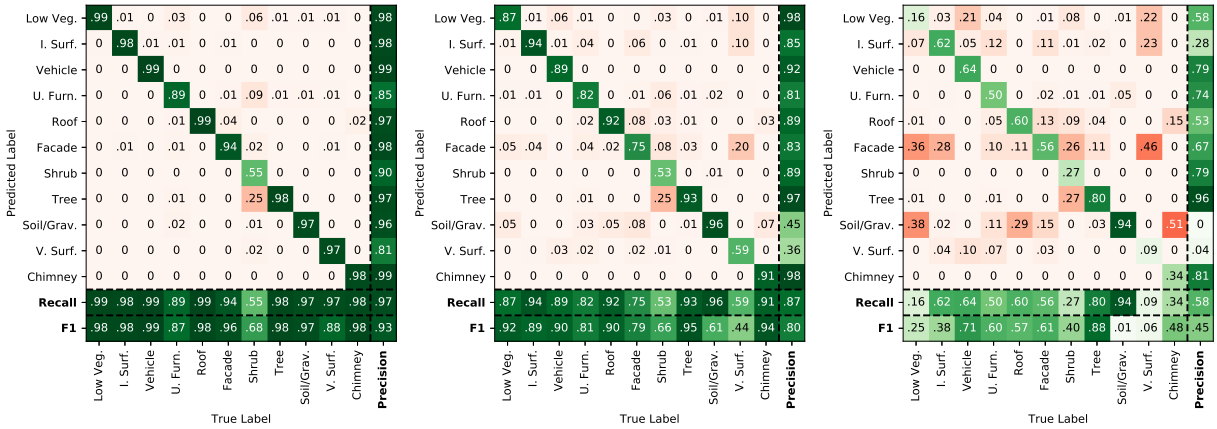


Figure 6.10: Label noise in form of duplicates in data set V3D for the investigated building of Figure 6.9. The right side shows the GT separated by classes *Roof* (*top*) and *Facade* (*bottom*). The duplicate annotation causes a wrong label transfer and thus inconsistencies in the forward-backward-pass.



(a) Manual GT vs. *Ours* *(b)* Manual GT vs. *NN* *(c)* Manual GT vs. NN (where *Ours* prevents the label transfer)

Figure 6.11: Comparison of manually assigned labels (*Manual GT*) with automatically transferred labels from the manually annotated PC on the mesh. The normalized confusion matrices are weighted by the face areas. The considered automated label transfer options are: i) the label transfer leveraging PCMA (*Ours*) and ii) the label transfer achieved by a simple NN interpolation across modalities (*NN*). *Subfigures a* and *b* compare the corresponding normalized confusion matrices where the manually assigned face labels are used as GT. The transferred labels are considered as predicted labels in all scenarios. *Subfigure c* analyzes the NN interpolated labels on faces where our method does not transfer any labels.

6.1.3 Focused Annotation Utilizing Active Learning and Crowdsourcing

As exemplified in Section 6.1.1, manual labeling is tedious and time-consuming work. Therefore, we strive to keep the labeling effort to a minimum by deploying AL. The iterative AL process intertwines GT generation and classifier training. An iteratively trained classifier repeatedly proposes instances for labeling that may contribute most to its training gain in the next iteration step. Hence, instead of annotating the entire data set, only these informative entities are annotated by a human operator (*focused annotation*). In contrast, GT generation and classifier training are two separated and sequentially executed processes in PL. First, the underlying data is fully annotated and afterward used for classifier training. The intertwined concept of AL reduces the labeling effort to a sparse but expressive subset sufficient for training.

The annotation has to be done by a human operator. The human-in-the-loop typically is an expert (like discussed in Section 6.1.1) but may even be a non-expert. Recruiting a group of unknown non-experts for accomplishing a specific task is known as crowdsourcing. Paid crowdsourcing reduces labeling costs since cost-intensive experts are not involved in the annotation procedure. Consequently, the combination of AL and crowdsourcing enables a fast and comparably inexpensive generation of annotated data.

For these reasons, we propose a human-in-the-loop AL pipeline where a ML classifier is trained on an iteratively increasing sparse training set with labels received via paid crowdsourcing. Theoretically, this AL pipeline can be applied to each modality individually. However, we extend the vanilla AL pipeline from Kölle et al. (2020) with coupled modality-specific branches to limit the annotation effort to a single AL loop that jointly annotates multiple modalities. We couple branches (and modalities respectively) with the proposed multi-modal entity linking (see Chapter 5). To this end, the enhanced AL procedure enables entangled iterative refinement of ML classifiers and semantic segmentation of multiple modalities in a parallelized fashion (Kölle et al., 2021b).

Figure 6.12 depicts the coupled multi-modal AL pipeline for processing the 3D modalities encompassing the respective branches *PC Branch* and *Mesh Branch*. PCMA entangles the two branches (see Section 5.1). The illustrated pipeline misses an *Image Branch* as we focus on 3D semantics. However, extending the pipeline with an *Image Branch* is trivial thanks to PCImgA and/or ImgMA (see Sections 5.3 and 5.2).

The core of the coupled AL pipeline is the *PC Branch* which features the AL loop. Generally, AL loops are composed of three interdependent components (Settles, 2009): i) the ML model, ii) the strategy for selecting the most informative entities and iii) the employed oracle $\mathcal{O}$ for manual annotation. The proposed extended AL pipeline entails these components only for the *PC Branch*. Other branches do not contribute to selecting the most informative entities and do not directly interact with the employed oracle/crowd. To emphasize, the decision on which instances have to be labeled is confined to the PC modality. However, the acquired GT is transferred automatically to other modalities utilizing the established multi-modal entity linking. By these means, the coupling implicitly acquires labels for multiple modalities at a stroke and avoids independent modality-wise labeling processes. To summarize, the coupling reduces the annotation costs and facilitates iterative and parallel training of the respective ML models for each branch/modality fed with manual or the respectively transferred annotations. The PC-driven AL loop steers the iterative refinement of ML models of other branches covering other modalities.

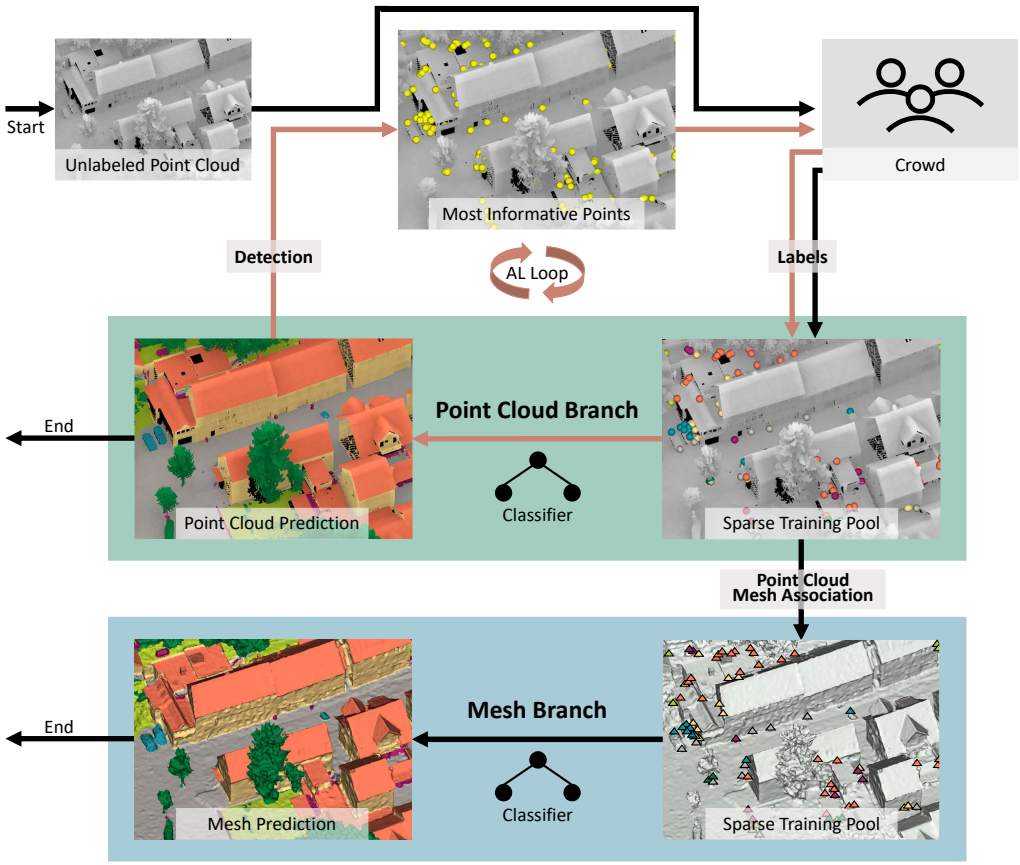


Figure 6.12: Crowd-based AL pipeline presented in (Kölle et al., 2021b) for coupled 3D semantic segmentation of both the PC (*Point Cloud Branch*) and the mesh (*Mesh Branch*). The AL loop (*red*) operates on the PC steering the manual labeling. Paid non-experts (i.e., the *Crowd*) manually assign GT annotations to the PC which is associated with the mesh via PCMA (see Section 5.1). The manually attached labels are automatically transferred to the mesh leveraging the established entity linking. Each branch iteratively trains a ML classifier on the iteratively updated sparse training pool during the AL loop. Most informative points are selected in the *Point Cloud Branch* based on the prediction probabilities (*Detection*). The detected points are presented to the crowd for labeling.

In the following, we detail the proposed AL pipeline sticking to its realization as depicted in Figure 6.12. Starting with completely unlabeled train sets U_{PC} and U_{Mesh} for the PC and the mesh respectively, the

oracle $\mathcal{O}_C$, embodied by the crowd C, has to provide few labeled PC instances for each class to actuate the PC-driven AL loop (iteration $i = 0$). These annotated points form a first small training pool $T_{\mathrm{PC}}(0)$ which is fed into the *PC Branch*. Its automatically derived counterpart $T_{\mathrm{Mesh}}(0)$ is directed into the *Mesh Branch*.

Based on the initialized sparse training pools $T_{\mathrm{PC}}(0)$ and $T_{\mathrm{Mesh}}(0)$, any arbitrary ML classifier can be trained for semantic segmentation of the PC and the mesh. We opted for fast RF models because of their simple adaption to the AL setting. The required features can be computed at once in advance of the AL loop. This is in contrast to feature-learning approaches, where features are learned at run time. Employed features are discussed in Section 6.2 for both the mesh and the PC.

Afterward, the trained classifiers are applied to the remaining unlabeled pools $R_{\mathrm{PC}}(0)$ and $R_{\mathrm{Mesh}}(0)$. This procedure is repeated for each iteration i. It holds: $R_{\{\mathrm{PC,Mesh}\}}(i) = U_{\{\mathrm{PC,Mesh}\}} \setminus T_{\{\mathrm{PC,Mesh}\}}(i)$.

Since the AL loop is PC-centered, the inferences on $R_{\mathrm{PC}}(i)$ are used to select instances that have the most positive influence on the classifier's performance in the subsequent iteration step $(i + 1)$ and therefore justify manual labeling effort. To detect the most informative points, we leverage the per-point entropy H as an intrinsic measure for determining the uncertainty of the model's per-point prediction. The entropy is derived for each point (represented by its feature vector $\boldsymbol{x}$) from the posterior probabilities $\boldsymbol{p}(\boldsymbol{c}|\boldsymbol{x})$ as derived from the RF inference. For each iteration step i, we declare the point represented by feature vector $\boldsymbol{x}_H(i)$ to be most informative if its entropy amounts to the maximum within the inference set $R_{\mathrm{PC}}(i)$.

For efficiency reasons, we select multiple points featuring the top-n entropy scores in each iteration step. Otherwise, each iteration step would add merely a single annotated point to $T_{\mathrm{PC}}(i + 1)$ (and $T_{\mathrm{Mesh}}(i + 1)$ respectively).

ALS PCs generally tend to suffer from class imbalance and therefore the subset of selected points, too. To alleviate severe class imbalance, we calculate weighted entropy scores by dynamically derived class-dependent factors $w_c(i)$ for each i (see Equation 6.1). The per-class weights are determined as ratio of the total number of labeled points $N_{T_{\mathrm{PC}}}(i)$ and the number of representatives of each class N_c currently present in $T_{\mathrm{PC}}(i)$ at iteration step i, i.e., $w_c(i) = \frac{N_{T_{\mathrm{PC}}}(i)}{N_c(i)}$ (Kölle et al., 2021c). The reasoning is to enforce class diversity by down-weighting posterior probabilities of predicted classes that match with the predominant classes in the sparse training pool. As a consequence, the respective entropy scores will be down-weighted and the corresponding points are unlikely to be selected as the most informative points.

$$\boldsymbol{x}_H(i) = \underset{\boldsymbol{x}|\boldsymbol{x}\in R_{\mathrm{PC}}(i)}{\arg\max}\ H(\boldsymbol{x}, i) = \underset{\boldsymbol{x}}{\arg\max}\ -\sum_c w_c(i) \cdot p(c|\boldsymbol{x}) \cdot \log p(c|\boldsymbol{x}) \quad \text{with } w_c(i) = \frac{N_{T_{\mathrm{PC}}}(i)}{N_c(i)} \tag{6.1}$$

Naive selection of points featuring top-n weighted entropy scores results in selecting points with similar feature vectors (probably of the same class). To avoid such misconception of the annotated training pool, we use a more sophisticated approach as proposed by Zhdanov (2019). We detect k clusters in the feature space of the inference set $R_{\mathrm{PC}}(i)$ by running a k-means clustering (Lloyd, 1982). The number of clusters k is set to the number of points n selected in each iteration. For each cluster, we sample the point with the maximum weighted entropy. The cluster-driven sampling increases the diversity of selected points concerning their feature vectors and hence, boosts the convergence of the AL loop. We opted for the weighted entropy enforcing diversity in features space (DiFS) since it proved to be the best-performing selection method on V3D (Kölle et al., 2021c). The AL loop is repeated for N_i iteration steps until convergence or until exhaustion of budget.

The interpretation of geospatial data is demanding for non-experts due to unfamiliar perspectives and data types. Most probably, the crowd has never dealt with 3D data before. For this reason, we pursue several strategies to achieve high-quality labels from the crowd nevertheless. The strategies include crowd guidance by easy-to-use software, simple class catalog, suitable data visualization, reasonable point selection, motivating the crowd through bonus payments, and quality checks of the annotations.

As the annotation quality has a great impact on the training process, the annotation task has to be designed as easily understandable as possible. Therefore, we adapt the web-based labeling tool presented in Kölle et al. (2020) which has been designed explicitly for labeling by non-experts. Figure 6.13 depicts the Graphical User Interface (GUI) of the annotation tool.

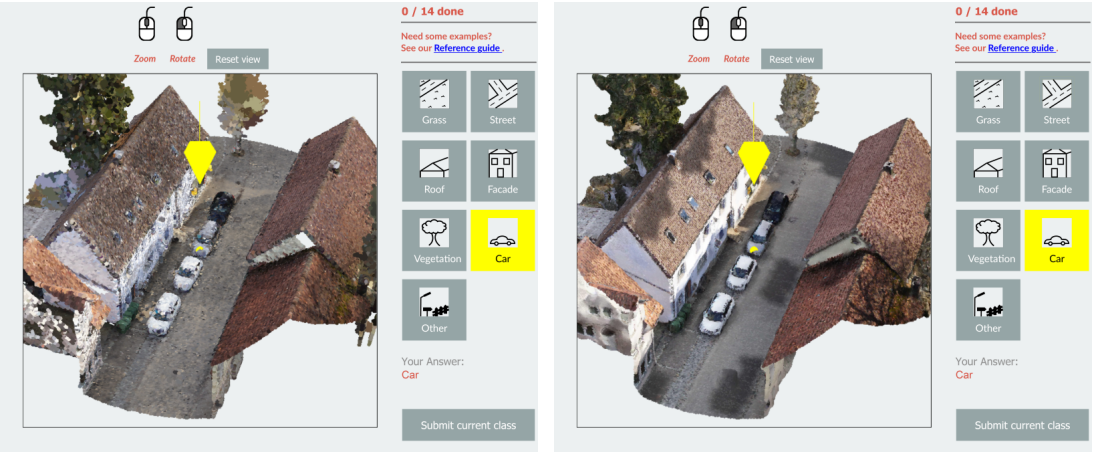


Figure 6.13: Web-based PC annotation tool developed by Kölle et al. (2020) for a crowd-based labeling of a point in question (highlighted by a *yellow* point and arrow) within the AL loop (see Figure 6.12). The neighborhood of the highlighted point is presented either as colorized PC (*left*) or textured mesh (*right*) (Kölle et al., 2021b).

To initialize the training pool, we run a first labeling campaign, where a random subset of the entire data set is presented to each crowdworker. The random subset is defined by a vertically oriented cylinder around a randomly sampled point. For each class, a representative has to be annotated. Multiple subsets may be inspected since the presented subset may not contain representatives of all classes. The per-class annotation is done by simply clicking an arbitrary representative point of the displayed subset(s). After the initialization, we run another labeling campaign in conjunction with the AL loop, where each point in question, i.e., one of the selected most informative points, has to be annotated. For better interpretation, all entities within a vertically oriented cylindrical neighborhood are presented to the crowd. A yellow dot and an arrow of the same color pointing to it highlight the point in question. The annotation is done by simply clicking the respective class button (see Figure 6.13). At all times, the annotators can interact with the shown data by zooming and rotating before annotation. We run both labeling campaigns twice, visualizing the neighborhood either by the colorized 3D PC or the textured mesh. In the subsequent paragraph *Analysis – Crowd Performance*, we evaluate the impact of the visualization modality on the crowd's labeling accuracy.

During the AL loop, crowdworkers are not free to choose the points to label. They are asked to label the most informative points as detected by the ML classifier and the applied selection strategy. Generally, these points are situated on class borders where the classifier is most uncertain, e.g., at the boundary of *Facade* and *Roof* (Ertekin et al., 2007). The true class is ambiguous for such points and their labeling strongly depends on the annotator's class understanding. To ease the interpretability of points in question, we increase the distance to class borders by Reducing Interpretation Uncertainty (RIU) (Kölle et al., 2021c). The initially detected most informative points are considered as seed points for a subset of points within distance d_{RIU}. The point with the lowest weighted entropy within this spherical vicinity is picked and presented to the crowd instead of the initially selected point. The method assumes that a lower entropy score is closely related to a larger distance to the class boundary and therefore increased interpretation certainty. As a result, the actual most informative points remain unlabeled but presumedly easier to interpret close-by points are annotated. We analyze the RIU-induced labeling accuracy in paragraph *Analysis – Crowd Performance*. In the subsequent paragraph *Experimental Setup*, we give more details about the enrolled experiments.

Apart from simplifying the annotation process by a simple task design and easier to interpret points, we encourage the crowdworkers to work carefully and try to maximize the annotation quality by deploying paid crowdsourcing. Crowdworkers are paid for each successfully accomplished job and may additionally receive a bonus payment if the task has been fulfilled perfectly. On the contrary, poorly executed jobs lead to rejection and denial of payment.

In our experiments, we initialize the training set $T_{\mathrm{PC}}(0)$ by conducting a labeling campaign where n_{start} crowdworkers are asked to mark one point for each class c. Therefore, the campaign ends with $n_{\mathrm{start}} \cdot n_c$ annotated points. To determine the correctness of provided labels, we run a checking campaign after the initial labeling campaign. In the checking campaign, n_{check} independent crowdworkers have to decide for each point whether the given label is correct or not (binary decision). Each checking crowdworker checks the made n_c annotations of a specific annotator. Points form part of the initial training pool $T_{\mathrm{PC}}(0)$ only if the majority vote of the binary decisions indicates a correct annotation.

To check the crowdworkers of the checking campaign in a background process, we extend the subset of points to be checked by 4 points of known GT (checkpoints). Therefore, each crowdworker of the checking campaign has to decide upon n_c+4 points. The checkpoints do not contribute to the training pool. However, they determine the payment of the checking crowdworkers. If 3 out of 4 reviewed checkpoints are correct, the crowdworker is paid. If all 4 checkpoints are reviewed correctly, the crowdworker additionally receives a bonus payment. In all other cases, the crowdworker is not paid and their reviews are rejected. Since crowdworkers strive to get the most out of the accomplished tasks, the checkpoints implicitly control the quality (and can be seen as quality control points).

During the AL loop, we batch the most informative points into jobs of $n_{\mathrm{AL}} < n$ points for the labeling campaign. We extend the batches by 4 additional checkpoints to check the annotation quality and steer the payment analogously to the checking campaign of the initialization step. Each crowdworker has to annotate $n_{\mathrm{AL}}+4$ points. By these means, the checking is integrated into the labeling campaign during the AL loop. If more than one checkpoint has been annotated wrongly, the crowdworker is not paid and their provided labels are rejected. By this means, correct annotation of checkpoints is forced. If all 4 checkpoints are annotated correctly, the crowdworker additionally receives a bonus payment. The reasoning behind this approach is that provided labels are likely to be correct when all checkpoints have been annotated correctly. Due to the closed world assumption, each selected point has to be assigned to one of the given classes. The result is often that one class is used as a "fallback" class which poses the risk that crowdworkers assign this class for most points to complete the task as fast as possible (Gadiraju et al., 2015). The injection of checkpoints allows the exclusion of crowdworkers following this pattern. To keep the expert label work small, the same checkpoints are used for all batches within each iteration step. In total, the checking requires $N_i \cdot 4$ points with manually annotated GT.

The annotation quality of the checkpoints is used as a proxy measure for the annotation quality and it steers the payment of crowdworkers. However, the annotation quality of n_{AL} points in question cannot be validated. Therefore, each batch is annotated by n_{check} independent crowdworkers passing the annotation test of checkpoints. For each selected point, the provided labels are aggregated and a majority vote determines the label. In the case of a draw, the label is picked randomly from the aggregated labels. We refer to the crowd-provided GT data as *crowd truth*.

Experimental Setup. Utilizing the proposed multi-modal AL pipeline, we investigate on 3 research questions by running several experiments:

i) How does the visualized modality influence the quality of the PC annotations provided by the crowd? Which representation is suited best for presenting to crowdworkers – colorized PC or textured mesh? We determine which data representation is easier to understand for the crowd by analyzing the labeling accuracy as a measure for the crowd performance.

ii) How does RIU influence the quality of the PC annotations provided by the crowd? We check whether deploying RIU improves the labeling accuracy as a measure for the crowd performance.

iii) Can AL models compete with PL models? We evaluate the performances of the conducted AL loops and compare them to respective PL results (see Section 6.6).

We run 6 different AL loops to answer the stated research questions (see Table 6.2). The AL loops deploy oracles that are shown different modalities and different most informative points depending on the selection strategy. $\mathrm{AL}(\mathcal{O}_O)$ and $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_O)$ denote the simulated AL loops using an omniscient oracle $\mathcal{O}_O$ that always provides the correct labels for the selected points. Therefore, $\mathcal{O}_O$ is invariant to the visualization of the point's vicinity (colored PC or textured mesh). The index RIU indicates the usage of the method RIU which increases the distance to class borders of selected points. $\mathrm{AL}(\mathcal{O}_{C|\mathrm{PC}})$ and $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{PC}})$ refer to the same two AL approaches (with and without RIU) to annotate PCs using a crowd C of non-experts as oracle, where the PC is presented to crowdworkers ($\mathcal{O}_{C|PC}$). Analogously, the 3D textured mesh visualizes the point's vicinity for $\mathrm{AL}(\mathcal{O}_{C|\mathrm{Mesh}})$ and $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$.

Table 6.2: Overview of AL configurations deployed for the proposed crowd-based coupled AL pipeline (see Figure 6.12). The fundamental selection strategy uses the weighted entropy H (see Equation 6.1) and enforces diversity in feature space (DiFS) by utilizing the approach of (Zhdanov, 2019). RIU may be applied additionally to avoid the selection of points on class borders (and to improve the interpretability of selected points for non-experts). The neighborhood of a point in question may be visualized by colorized PC or textured mesh.

Configuration	**Oracle $\mathcal{O}$**	**Visualized Modality**	**Selection Strategy**
$\mathrm{AL}(\mathcal{O}_O)$	omniscient $\mathcal{O}_O$	-	H + DiFS
$\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_O)$	omniscient $\mathcal{O}_O$	-	H + DiFS + RIU
$\mathrm{AL}(\mathcal{O}_{C\|\mathrm{PC}})$	crowd $\mathcal{O}_C$	PC	H + DiFS
$\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C\|\mathrm{PC}})$	crowd $\mathcal{O}_C$	PC	H + DiFS + RIU
$\mathrm{AL}(\mathcal{O}_{C\|\mathrm{Mesh}})$	crowd $\mathcal{O}_C$	Mesh	H + DiFS
$\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C\|\mathrm{Mesh}})$	crowd $\mathcal{O}_C$	Mesh	H + DiFS + RIU

For comparability, the same initialization set $T_{\mathrm{PC}}(0)$ (and the derived $T_{\mathrm{Mesh}}(0)$ respectively) is used for all AL runs. The initialization set is generated by $n_{\mathrm{start}} = 100$ crowdworkers. Each annotated point is checked by $n_{\mathrm{check}} = 3$ independent crowdworkers (minimum configuration for an unambiguous majority vote for a binary decision). After the initialization, $n = 300$ most informative points are selected for labeling by the top-n entropy scores in each AL loop iteration. The most informative points are batched into jobs of $n_{\mathrm{AL}} = 10$ points (plus the checkpoints). Each batch is annotated by $n_{\mathrm{check}} = 3$ independent crowdworkers resulting in 90 jobs per iteration. Crowdworkers earn 0.10 \$ for each successfully completed job regardless of the conducted campaign. The paid bonus amounts to 0.05 \$. As a compromise between total costs and labeling accuracy, each run is conducted for $N_i = 10$ iteration steps.

For all experiments, d_{RIU} is set to 1.5 m according to the findings of Kölle et al. (2020). We provide a point's cylindrical neighborhood of radius $r = 20$ m either as colorized dense 3D PC or textured mesh. Figure 6.13 shows the annotation of the same point while its vicinity is illustrated either by the colorized PC (*left*) or the textured mesh (*right*).

In the conducted experiments, we use the Hessigheim data in a slightly modified fashion compared to their description in Section 4.2. We mimic a commonly colorized LiDAR PC by interpolating colors from the respective photogrammetric PC derived from oblique Sony imagery (rather than from the textured mesh as described in Section 4.2). The colorization emphasizes the visual difference of the modalities PC and mesh. Particularly for vegetation, color information may be mapped insufficiently to the LiDAR PC since Dense Image Matching (DIM) fails to generate accurate 3D PCs there (see upper tree in Figure 6.13). Furthermore, we simplify the class catalog, as some classes are hard to interpret for non-experts. Precisely, we keep classes *Urban Furniture*, *Low Vegetation* (including *Gravel/Soil*), *Impervious Surface*, *Vehicle*, *Roof* (including *Chimney*), *Facade* (including *Vertical Surface*) and *Vegetation* (fusion of *Shrub* & *Tree*). Besides, we simplify class names for crowdworkers as can be seen in Figure 6.13 (e.g., *Urban Furniture* → *Other* and *Impervious Surface* → *Street*).

For efficiency reasons, we spatially subsample the dense H3D PC with 30 cm point distance. We further exclude all points from labeling which cannot be associated with a face via PCMA. Hence, we guarantee

clear correspondences between points and faces and enable the label transfer to the mesh during the AL loops. Section 5.4 discusses why few points and faces remain without any associated entities. The mentioned filter steps reduce the number of points in the unlabeled training pool U_{PC}. In the case of the dense H3D data set, many close-by points likely share similar feature vectors. The spatial subsampling not only boosts processing speed but also helps to guarantee diversity of selected points (Kölle et al., 2021c).

The first two research questions refer to the achievable annotation accuracy of the crowd truth. The labeling accuracy determines the performance of the crowd by evaluating the made annotations against the GT given for the H3D PC (with modified class catalog). We analyze the impact of both visualization modalities and RIU on the labeling accuracy in the subsequent paragraph *Analysis – Crowd Performance*. Comparability is given as the conducted AL campaigns share the same initial training pool, i.e., they are coupled by $T_{\mathrm{PC}}(0)$. Figure 6.23 depicts the true class distributions of selected points for each iteration and configuration. The distributions diverge because of the provided annotations based on the different selection strategies and visualized modalities. It has to be noted that the labeling accuracy is analyzed on the PC only since manually annotated GT is given for the PC and the crowd consistently interacts with the PC, even if the neighborhood of the selected points may be presented as textured mesh.

The achieved labeling accuracy influences the performance of the trained ML models for semantic PC segmentation. As the PC-driven AL also steers the coupled *Mesh Branch*, the semantic segmentation of meshes depends on the labeling accuracy achieved on the PC, too. The coupled approach is faster and cheaper than a decoupled approach covering each modality individually. Besides, the coupling of mesh and PC via PCMA allows for comparing classifier performances on PC and mesh since the respective models are trained on correlated instances sharing the same location. Running independent AL loops for the mesh and the PC would avoid a valid comparison of the classifier performances on the two modalities as decoupled branches would select independent entities in both modalities and generate decoupled ML models. However, we are aware of the fact that the comparison is not entirely fair since the selection of entities is purely determined by the PC-driven AL loop. Eventually, the entity selection is optimized for the semantic segmentation of PC depending on corresponding PC features and the iteratively trained RF model for the PC. In Section 6.6, we discuss the achievable classifier performances of the conducted AL configurations and compare the achieved accuracies to plain PL on both modalities.

Analysis – Crowd Performance. In this paragraph, we focus on the annotation quality of the labels provided by crowdworkers $\mathcal{O}_C$. We analyze the crowd performance for the conducted AL configurations (see Table 6.2) considering different visualization modalities (textured mesh vs. colorized PC) and different selection strategies for detecting informative points (with/without applying RIU). It should be recalled that labeling accuracy is evaluated on the PC as the crowd truly annotates points and manual annotated GT is available for the PC (see Section 4.2).

In our experiments, the crowd completed the batched jobs of one AL iteration in less than 11 h. The initialization stage encompassing the labeling and checking campaign lasts roughly 16 h. In total, a complete AL run is finished in about 5 d annotating 3700 points (approx. 11 h·10 iterations + approx. 16 h for initialization = 126 h = 5.25 d). We stress that we ask the crowd to label 0.25 % of available training points only, which causes costs of 190 $ at most:

$$\text{Campaign Costs} = \underbrace{100\,\text{jobs} \cdot 0.10\,\$}_{\substack{\text{Labeling Campaign}\\ \text{Initialization}}} + \underbrace{3\,\text{workers} \cdot 100\,\text{jobs} \cdot 0.15\,\$}_{\substack{\text{Checking Campaign}\\ \text{Initialization}}} + \underbrace{10\,\text{iterations} \cdot 3\,\text{workers} \cdot 30\,\text{batches} \cdot 0.15\,\$}_{\substack{\text{Labeling/Checking Campaign}\\ \text{AL Loop}}}$$

The focused annotation requires negligible costs and labeling effort compared to the full annotation by experts (see Section 6.1.1).

Figure 6.14 displays the confusion matrices derived from the provided annotations for completed AL runs, i.e., for the final labeled training pools $T_{\mathrm{PC}}(10)$.

Comparing $\mathrm{AL}(\mathcal{O}_{C|\mathrm{PC}})$ with $\mathrm{AL}(\mathcal{O}_{C|\mathrm{Mesh}})$ and $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{PC}})$ with $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$ respectively, we observe that crowd oracles perform better when spatial context is visualized by the textured mesh instead of colorized PC. The *OA* is approximately 3 pp better in both scenarios. The $mF1$-score is increased by 3–4 pp.

Inspecting the basic configurations not using RIU, we observe that the crowd confuses adjacent classes

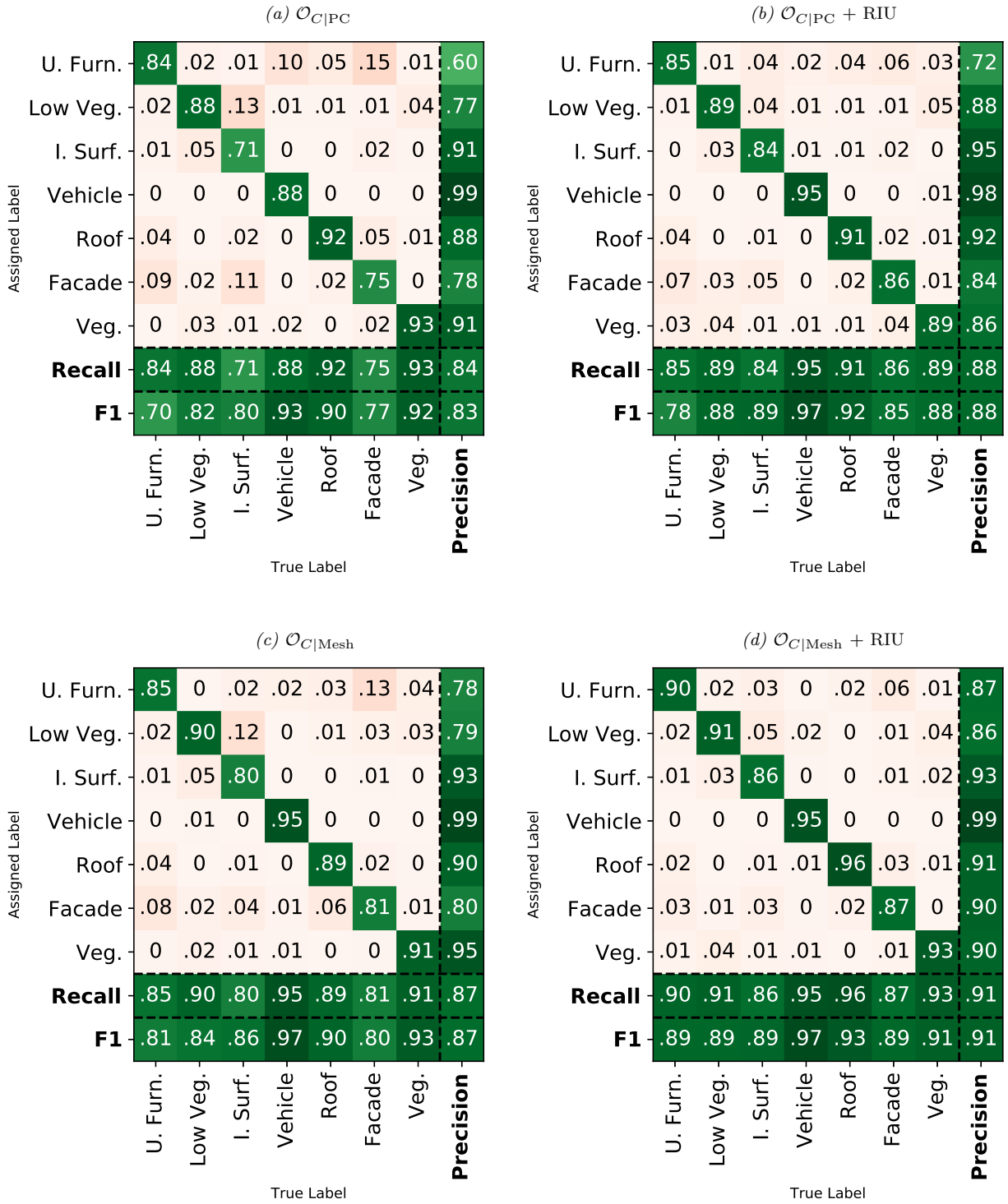


Figure 6.14: Labeling accuracy of the *crowd truth* ("Assigned Label") on the PC achieved after the AL loop (10 iterations) by various crowd oracles $\mathcal{O}_C$ utilizing different representations for visualization and selection strategies (see Table 6.2). The vicinity of a point is either presented as colorized PC (a and b, *top row*) or as 3D textured mesh (c and d, *bottom row*). Points are selected based on the weighted entropy while enforcing diversity in feature space as default (*left column*). The selection strategy is modified by additionally applying RIU (*right column*).

in real-world 3D scenes (*Low Vegetation* vs. *Impervious Surface*, *Roof* vs. *Facade*, *Facade* vs. *Impervious Surface* and *Urban Furniture* vs. *Facade*). We note that $\mathcal{O}_{C|\mathrm{PC}}$ is more confused than $\mathcal{O}_{C|\mathrm{Mesh}}$ for the listed class pairs. In both cases, the crowd struggles the most regarding class *Urban Furniture* causing low per-class precision and $F1$-score. Precisely, the crowd tends to label a point as *Urban Furniture* abusing this class as "fallback" (named *Other* in the GUI of the web-tool, see Figure 6.13). Repeatedly choosing class *Other* might indicate that crowdworkers have severe difficulties in interpreting many points – particularly when point vicinity is given as colorized PC. The mesh representation seems to alleviate interpretation difficulties resulting in fewer provided *Other* labels. We observe that the mesh modality causes less inter-class confusion. We assume that the mesh is easier to interpret for non-experts due to its realistic-looking closed surface.

Since AL focuses on selecting points the classifier is most uncertain about, eventually, points lying on class borders are selected. Such points are ambiguous for interpretation and might even be mixed by experts. The confusion matrices in Figure 6.14 indicate that applying RIU decreases inter-class confusions and increases the labeling accuracy on the PC regardless of the visualized modality. RIU improves the interpretability of selected points for non-omniscient oracles. For both visualization modalities, RIU increases the *OA* by about 4 pp and $mF1$-score by 4–5 pp compared to the respective counterparts not using RIU-modified most informative points. We deduce that RIU minimizes boundary-close mislabeling by amplifying the distance to the decision boundaries. Hence, the crowd can assign correct class labels for more points. Class *Vegetation* is an exception, where RIU seems to be counterproductive for proper recognition when visualizing the colorized PC. In that case, the increased distance of the point in question to the class border results in a selected point that may not be colorized reasonably (see Figure 6.13). As a consequence, the per-class recall drops in this particular case. However, when visualizing the textured mesh, RIU improves the recall for this class, too.

The crowd yields top labeling results when enhancing the query function with RIU and relying on the textured mesh as presentation modality. $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$ achieves the best performance with 90.96 % and 90.77 % for $mF1$-score and *OA* respectively (see d in Figure 6.14).

To summarize, the analysis reveals that a) understanding meshes is significantly easier for non-experts (i.e., the crowd) and b) RIU improves interpretability substantiating our initial hypotheses. The findings demonstrate that minding the employed oracle and adjusting the query functions to their needs is profitable.

While Figure 6.14 indicates the performance of the oracles concerning the annotated train pool after the last loop iteration, Figure 6.15 tracks the OAs of the labeling accuracy at each iteration step (considering only the points that have been labeled in the same iteration). The courses of the tracked OAs per iteration depict differences due to the different representation methods and selection strategies. We note that the iteration-specific *OA* decreases in the course of the AL loop for all configurations. A possible explanation might be that the selected points become more complex with increased duration of the AL loop. In the beginning, the classifier might be confused for all points with feature vectors dissimilar to the already presented entities. While being complex for the hardly trained classifier, the points might be simple to interpret for the non-experts. With increasing loop iteration, the ML model will become more confident on points belonging to typical objects of the considered PC. Consequently, sampled points become more demanding for labeling. We emphasize this holds for the iteration-specific performance of the crowd. In contrast, accuracies gradually increase for the iteratively trained models (see Figure 6.22). $\mathcal{O}_{C|\mathrm{Mesh}}$ + RIU shows only a slight decrease of *OA*, which underlines its effectiveness.

6.1.4 Summary

Supervised ML classifiers depend on the availability of GT data. Therefore, GT generation is an important task. The process of providing GT data requires manual interaction inevitably. We presented three methods of varying manual effort to generate GT data on various modalities: i) pure manual annotation, ii) semi-automatic annotation (manual annotation coupled with automatic label transfer), and iii) focused manual annotation. Whereas i) requires full labeling by hand on each modality (see Section 6.1.1), semi-automatic annotation reduces the manual labeling effort to a single representation by leveraging multi-modal entity linking (see Section 6.1.2). The focused annotation incorporates AL to limit the manual labeling effort on each modality to a small subset of entities. The combination of ii) and iii) minimizes the labeling effort to

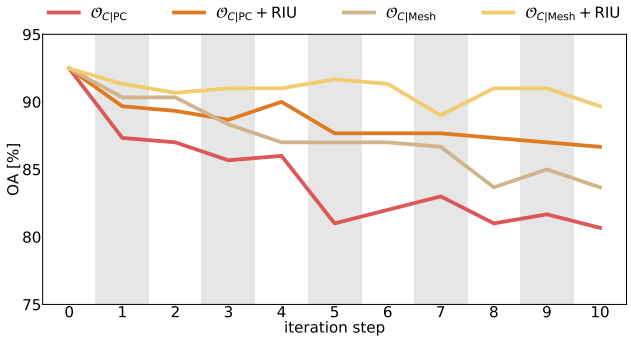


Figure 6.15: Iteration-specific crowd performance measured by *OA* for different oracles $\mathcal{O}_C$ (see Table 6.2). Please note the iteration-specific *OA* is evaluated on the selected points of each iteration only and not on the totality of points annotated up to that iteration.

a small subset of entities on a single modality (see Section 6.1.3). The published multi-modal benchmark data set H3D has been created by a combination of i) and ii) (Kölle et al., 2021a).

Manual Annotation. The most common and obvious way to generate fully annotated GT is manual labeling of the entire data. The manual annotation is tedious and time-consuming work. Conventionally, manual labeling is done by experts since the effectiveness of trained classifiers depends on GT quality. The employment of experts makes GT generation expensive labor.

Student assistants and Ph.D. students accomplished the manual labeling. We implemented a cross-checking and harmonization step to minimize label ambiguity and label noise. At the end of this work, the ALS PC of H3D, a small subset of the respective hybrid 3D mesh, and the 2.5D mesh of Hessigheim are manually annotated (see Chapter 4). The manually generated GT is used throughout the thesis for validating the proposed multi-modal entity linking, training classifiers, and evaluating multi-modal semantic segmentation.

Semi-Automatic Annotation Utilizing Multi-Modal Entity Linking. The semi-automatic annotation boosts multi-modal GT generation by limiting the manual labeling effort and the respective costs to a single modality. In particular, the label transfer allows relocating the manual annotation to 3D space for imagery and thus avoids redundant label work in overlapping images. Moreover, the label transfer extends the applicability of publicly available modality-wise GT to other modalities. Hence, modality-specific training and semantic segmentation on previously unlabeled modalities are possible. To give an example, we transferred labels from the ALS PC of V3D to the respective MVS mesh and nadir images. Likewise, the manually assigned annotations of the H3D PC have been propagated to the mesh and imagery (see Figure 1.1).

We qualitatively and quantitatively demonstrated the effectiveness of the explicit entity linking by analyzing the label transfer from the PCs to meshes and imagery for H3D and V3D. The quantitative analysis of the label transfer implicitly validates the feature transfer, too. The relationship of image space and 3D space is defined by collinearity equations and hence does not have to be validated. Therefore, we limited our quantitative analysis to PCMA. H3D and V3D feature significantly different characteristics (e.g., extension, point count, point density, face count, face size, asynchronous data acquisition, etc.).

The qualitative analysis of PCMA shows the effectiveness of the thresholding (see Figure 6.5), its robustness against meshing algorithms (software `SURE` vs. `Metashape`, see Figure 6.6), and underlying mesh dimensionalities (2.5D vs. 3D mesh). We showcased that the multi-modal entity linking is not confined to specific kinds of meshes or meshing algorithms. The linking method copes with complex and simple meshes. Compared to a NN-interpolated transfer, our method does not introduce new label noise during the transfer. Labels are only transferred where annotations are given. Figure 6.4 visualizes propagated labels side-by-side

achieved with the PCMA-steered transfer and the simple NN interpolation. The visualized examples show the natural limitations like label noise, data gaps, and mesh quality for an automated transfer.

We have proven the concept by transferring labels from the PC to the mesh and therefrom, back to the PC. The advantage of the consistency analysis on the PC is that manual annotation is not required on other modalities. The forward-backward passing of labels indicates that the majority of linked points (98.9%/99.6% for V3D/H3D) achieves consistent labels. Discrepancies in modalities and co-registration residuals cause inconsistencies and missed connections. Besides, the label consistency check helped to detect label noise in the manually annotated V3D PC (see Figure 6.9 and Figure 6.10).

We quantitatively validated the entity linking between PC and mesh through the manually annotated subset of the textured mesh of Hessigheim. In general, mismatches are inevitable due to the disjoint manual annotation processes on PC and mesh, which suffer from human perception, label noise, and label ambiguity.

96.71 % of PCMA-transferred labels are correct (with an $mF1$-score of 93.71 %). The NN interpolation provides 9.95 pp less correct labels. The respective $mF1$-score is 13.01 pp worse. On top of this, the NN interpolation introduces significant label noise for faces that are not linked by PCMA. 42.12 % of unlinked faces receive a wrong label through interpolation.

Focused Annotation Utilizing Active Learning and Crowdsourcing. The focused annotation boosts GT generation by limiting the manual annotation effort and the respective costs to a small subset of entities as selected during the AL loop. The costs are further reduced by employing non-experts for the annotation.

Within this work, we have proposed a human-in-the-loop AL pipeline (see Figure 6.12), which avoids the involvement of an expert in the tedious and costly labeling process of 3D geodata. The annotation process has been limited to 0.25 % of the subsampled H3D PC. The PC has been subsampled to point distance of 30 cm to speed up the AL loop (see Section 6.6). However, the PC has been visualized at full density during the annotation by the crowd. The annotation campaign cost 190 $ and took roughly 6 d (the net working time of crowdworkers is even lower). The multi-modal entity linking couples the mesh with the PC-driven AL pipeline. The coupling enables the transfer of assigned labels to the linked faces at a stroke. Hence, a single labeling effort is sufficient to annotate PC and mesh. The coupling guarantees consistent inter-modal annotations and minimizes the annotation costs and time. In summary, the manual annotation is limited to a small subset of the PC.

We investigated which form of data representation is best suited for presenting to crowdworkers and which modifications of the AL selection strategy help non-experts to label individual instances. Our results imply that RIU eases interpretability for non-experts: Labeling accuracy on the PC increased by 4 pp regardless of visualizing PC or mesh. We found that visualizing meshes instead of full-resolution PCs during point annotation improves the labeling accuracy by about 3 pp. The finding emphasizes the utility of the mesh for visualization purposes. We assume the mesh to outperform the PC as presentation modality by an even larger margin for less dense PCs, as the appearance of mesh and PC will differ more. In summary, the best crowd performance is achieved when RIU modifies the selection strategy, and the mesh is visualized.

6.2 Feature Calculation

ML methods use the implicit knowledge included in the data. Regardless of the modality, entities are equipped with measured features. An arbitrary number of handcrafted features may extend the inherent feature set aiming at a meaningful separation of considered classes. For instance, RGB tuples are inherent features measured by a color camera. On the contrary, the gradient magnitude is an engineered feature derived from measured quantities. In the scope of this work, we do not intend to address the semantic segmentation of imagery with all its particularities. We argue that handcrafting expressive features in 2D image space is more challenging than in 3D space since merely RGB tuples are given. In contrast, feature engineering in 3D space has access to radiometric properties and the underlying geometry. Therefore, we refrain from engineering features in the image space. However, the proposed feature transfer allows the transferring of features to image space that have been derived in 3D space (see Chapter 5). Rephrased,

the entity linking facilitates shifting the feature calculation to 3D space avoiding geometric defects of the image space. Detouring via 3D space, we still enable the semantic segmentation of imagery utilizing ML approaches, although we do not calculate any features directly in image space.

For each 3D entity, i.e., for each point and face, we generate a feature vector consisting of measured and engineered features. The scope is not to design new distinguishing features. Instead, we utilize standard features that are commonly applied (see Chapter 3, Table 6.3, Table 6.4). Generally, we focus on features that are rather intuitive and simple to compute. The handcrafted features are categorized according to their a) type (geometric, topologic and radiometric features), b) scale (per-entity and contextual features), and c) modality (derived from PC or mesh). Whereas per-entity features consider merely a single entity, contextual features gather information from adjacent instances. Thereby, contextual features implicitly act as spatial regularization and cause spatially smooth labeling (Schindler, 2012). In the following paragraphs, we detail the derived features of both modalities.

The majority of handcrafted features exploits the geometry (see Chapter 3). However, color is a fundamental attribute for human perception. Therefore, we additionally incorporate radiometric features (given that respective imagery is available) for both the PC and the mesh. We use the HSV color space to ensure lighting independent features. In RGB color space, a change in color brightness requires a non-proportional change of values in all three color channels.

By using PCMA, modality-specific features (listed in Tables 6.3 and 6.4) can be propagated to the other representations. In this way, LiDAR-inherent features can be transferred to the mesh and vice versa. In the case of PC $\mapsto$ Mesh, the transferred feature values will be aggregated by calculating the median. Figure 5.7 shows shared features on the ALS PC and the mesh of Hessigheim. The feature transfer enables the flexible and arbitrary concatenation of derived features ending up with multi-modal and multi-scale feature vectors for each entity (see Table 6.1). Feature vector compositions $\mathcal{FS}_a - \mathcal{FS}_d$ use features that have been derived on the mesh ("mesh-only"). $\mathcal{FS}_e - \mathcal{FS}_i$ deploy features that have been derived on the PC ("PC-only"). Feature vectors $\mathcal{FS}_j$ and $\mathcal{FS}_k$ combine "mesh-only" and "PC-only" features in a multi-modal descriptor.

The broad set of features from different modalities spans a wide spectrum of ranges and units. To achieve uniform relative importance across features and therefore comparability, we apply the z-transformation to each feature and all data splits (train, validation, and test set). The values are robustly standardized according to the central moments of the train set, i.e., mean and standard deviation. The standardized features are zero-centered dimensionless quantities whose standard deviations are normalized to one.

Point Cloud Features. For preparing the LiDAR data, we make use of well-established multi-scale features that proved to be well-performing for semantic segmentation of ALS PCs (see Chapter 3). Table 6.3 lists inherent and handcrafted features of LiDAR points.

We compute geometric features as proposed by Weinmann et al. (2015), Mallet et al. (2011), and Chehata et al. (2009) employing `CloudCompare` (Girardeau-Montaut, 2021). To describe the local point distribution, we estimate the covariance matrix of coordinates $\Sigma_{\mathcal{NH}_i}$ (also known as structure tensor) for the local neighborhood $\mathcal{NH}_i$ of each individual point $\boldsymbol{x}_i = [x_i, y_i, z_i]^T$ (see Equation 6.2). $\bar{\mathbf{x}}$ denotes the CoG of $\mathcal{NH}_i$ and N_i is the number of points inside $\mathcal{NH}_i$, i.e., $N_i = |\,\mathcal{NH}_i\,|$. We define multiple spherical neighborhoods $\mathcal{NH}_i^r$ with radius $r \in \{1.0\,\mathrm{m}, 2.0\,\mathrm{m}, 3.0\,\mathrm{m}, 5.0\,\mathrm{m}\}$ for each $\mathbf{x}_i$, i.e., $\mathcal{NH}_i^r = \{\mathbf{x}_j \mid ||\mathbf{x}_i - \mathbf{x}_j|| \leq r\}$, to establish a multi-scale approach analyzing features on multiple levels of abstraction. We extend the set of radii used in Blomley and Weinmann (2017); Kölle et al. (2019); Niemeyer et al. (2014) with small-scale radii $r \in \{0.125\,\mathrm{m}, 0.25\,\mathrm{m}, 0.5\,\mathrm{m}, 0.75\,\mathrm{m}\}$ for data set H3D to stay abreast on fine structures present in the dense PC. For simplicity and to reduce computational effort, we limit the neighborhood definition to spheres of fixed r.

$$\sum_{\substack{\mathcal{NH}_i \\ [3\times3]}} = \frac{1}{N_i} \sum_{j|\mathbf{x}_j \in \mathcal{NH}_i} (\mathbf{x}_j - \bar{\mathbf{x}})(\mathbf{x}_j - \bar{\mathbf{x}})^T \tag{6.2}$$

The PCA derives the eigenvalues λ_i from $\Sigma_{\mathcal{NH}_i}$ sorted in descending order, i.e., $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Eigenvalue-based features refine the eigenvalues to more comprehensive features: *linearity* L_λ, *planarity* P_λ, *anisotropy* A_λ,

Table 6.3: Set of inherent and handcrafted features for a LiDAR point $\mathbf{x} = (x, y, z)^\top$. The majority of handcrafted features are adopted from (Chehata et al., 2009; Demantké et al., 2012; Mallet et al., 2011; Weinmann et al., 2013).

Feature Group	Feature Name	Feature Formula
Eigenvalue-based with eigenvalues λ_i and normalized eigenvalues $\Lambda_i = \frac{\lambda_i}{\sum\lambda}$ where $\Lambda_1 + \Lambda_2 + \Lambda_3 = 1$	Linearity	$L_\lambda = \frac{\lambda_1 - \lambda_2}{\lambda_1}$
	Planarity	$P_\lambda = \frac{\lambda_2 - \lambda_3}{\lambda_1}$
	Anisotropy	$A_\lambda = \frac{\lambda_1 - \lambda_3}{\lambda_1}$
	Sphericity	$S_\lambda = \frac{\lambda_3}{\lambda_1} = 1 - A_\lambda$
	Change of Curvature	$C_\lambda = \frac{\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} = \Lambda_3$
	Omnivariance	$O_\lambda = (\Lambda_1 \cdot \Lambda_2 \cdot \Lambda_3)^{\frac{1}{3}}$
	Eigenentropy	$H_\lambda = -\sum \Lambda_i \cdot \ln(\Lambda_i)$
	Sum of Eigenvalues	$\Sigma_\lambda = \lambda_1 + \lambda_2 + \lambda_3$
Plane-based with normal $\mathbf{n}$ and distance to origin d	Verticality	$V_\mathbf{n} = 1 - \lvert\langle[0,0,1], \mathbf{n}\rangle\rvert$
	Roughness	$R_\mathbf{n} = \lvert\langle \mathbf{x}, \mathbf{n}\rangle - d\rvert$
	Inclination	$\theta = \arccos\left(\langle[0,0,1], \mathbf{n}\rangle\right)$
Densities with radius r and point count N	Surface Density	$D_{2\mathrm{D}} = \frac{N}{\pi \cdot r^2}$
	Volume Density	$D_{3\mathrm{D}} = \frac{N}{\frac{4}{3} \cdot \pi \cdot r^3}$
Curvature based on fitted quadric surface	Mean Curvature	see Har'el (1995) and Douros and Buxton (2002)
	Gaussian Curvature	
Height-based	Relative Height	$\delta h = \mathrm{nDSM} = \mathrm{DSM} - \mathrm{DTM}$
Echo-based	Echo Ratio	$\mathcal{Q}_\mathrm{echo} = \frac{\text{Return Number}}{\text{Number of Returns}}$
Radiometric	Reflectance	ρ
	Color	H, S, V

sphericity S_λ, *change of curvature* C_λ, *omnivariance* O_λ, *eigenentropy* H_λ, and *sum of eigenvalues* Σ_λ. We follow the convention of Mallet et al. (2011) utilizing normalized eigenvalues Λ_i for *omnivariance* and *eigenentropy*. Eigenvalues λ_i are normalized such that $\Lambda_1 + \Lambda_2 + \Lambda_3 = 1$, i.e., $\Lambda_i = \frac{\lambda_i}{\lambda_1 + \lambda_2 + \lambda_3}$.

Computing eigenvalues and eigenvectors eventually means fitting a local plane. The eigenvector corresponding to λ_3 represents the normal vector $\mathbf{n}$ of that local plane. We extend the feature set by plane-based features *verticality* $V_\mathbf{n}$, *roughness* $R_\mathbf{n}$, and *inclination* θ.

Furthermore, we extend the point descriptor with density characteristics of the local vicinity (Weinmann et al., 2013). The *volume density* $D_{3\mathrm{D}}$ describes the point density in the spherical neighborhood $\mathcal{NH}_i^r$. Its 2D counterpart, the *surface density* $D_{2\mathrm{D}}$, describes the point density of points projected to the cross-section area of a vertical cylinder with radius r and infinite extension.

The eigenvalue-based *change of curvature* estimates the local curvature based on the locally fitted plane. In comparison, the dedicated curvature features *mean curvature* and *Gaussian curvature* approximate curvature based on a quadric surface fitted onto the local vicinity (Douros and Buxton, 2002; Har'el, 1995).

Several works have revealed that the height above ground δh is the most expressive feature regardless of its calculation method (Chehata et al., 2009; Guo et al., 2011; Kölle et al., 2019). Chehata et al. (2009)

approximate δh by subtracting the minimum height of the defined neighborhood from the elevation z_i of $\mathbf{x}_i$. In this work, we robustly derive the *relative height* δh for each point by interpolating the terrain height from the respective DTM for V3D and H3D (see Chapter 4).

In addition to purely geometric features, LiDAR-inherent features such as *echo ratio* $\mathcal{Q}_{\text{echo}}$ and *reflectance* ρ of received echoes are used for the semantic segmentation, too. Moreover, we amplify the radiometric information by colorizing the PCs (see Chapter 4).

The per-entity features δh, $\mathcal{Q}_{\text{echo}}, \rho$, and H, S, V do not rely on surrounding points and can be computed unambiguously. However, to generate multi-scale instances of radiometric features (i.e., *reflectance*, *color*), we calculate the Gaussian weighted average within $\mathcal{NH}_i^r$. The Gaussian weights are determined by the Euclidean distance $d_j = ||\mathbf{x}_j - \mathbf{x}_i||$ and the radius r of the sphere defining the neighborhood (see Equation 6.3). Our reasoning is that contextual radiometric features may regularize radiometric intra-class anomalies such as signs on facades or solar panels on roofs.

$$w(\mathbf{x}_j) = \frac{1}{r\sqrt{2\pi}}\, e^{-\frac{||\mathbf{x}_j - \mathbf{x}_i||^2}{2r^2}} \qquad \text{with } ||\mathbf{x}_j - \mathbf{x}_i|| \leq r \tag{6.3}$$

Mesh Features. We implement a pipeline that parses wavefront OBJ files along with associated texture maps. The parsing derives geometric and radiometric features for each face. Similarly to preparing LiDAR data, we make use of standard features that are commonly used for (semantic) segmentation of triangulated models in the computer vision community (see Chapter 3). However, meshed models do not necessarily carry texture information. Commonly, small-scale meshes of the computer vision community are not textured, wherefore merely geometric features are calculated often. To emphasize the characteristics of photogrammetric meshes, i.e., textured meshes, we derive a broad spectrum of radiometric features. At the same time, we limit the set of geometric mesh features because the meshing of real-world data may reduce the geometric resolution. By these means, the mesh feature set accentuates the texturing, whereas PC features emphasize geometric characteristics. We compute geometric features by employing `PyMesh` (Zhou, 2018). Table 6.4 lists inherent and handcrafted mesh features deployed for V3D and H3D.

Table 6.4: Set of inherent and engineered features attached to each face f. Features are calculated for each face itself and its incident vertices v_1, v_2, and v_3. The majority of geometric and topologic features is calculated employing `PyMesh` (Zhou, 2018).

Feature Group		**Feature Name**	**Feature Formula**
Geometric and Topologic with 1-ring neighborhood $\mathcal{NH}_i^{①}$ of vertex v_i, normals $\mathbf{n}'_{\boldsymbol{e}_{ij}}$ and $\mathbf{n}''_{\boldsymbol{e}_{ij}}$ sharing edge $\boldsymbol{e}_{ij}$, triangle edges $\boldsymbol{e}_{v_i,v_j}$, and semi-perimeter $s = \frac{\|\boldsymbol{e}_{v_1,v_2}\|+\|\boldsymbol{e}_{v_2,v_3}\|+\|\boldsymbol{e}_{v_1,v_3}\|}{2}$		Valence	$\nu^{v_i} = \lvert\mathcal{NH}_i^{①}\rvert - 1$
		Dihedral Angle	$\Phi^{v_i} = \max\left(\{\arccos\langle \mathbf{n}'_{\boldsymbol{e}_{ij}}, \mathbf{n}''_{\boldsymbol{e}_{ij}}\rangle\}\, \forall \boldsymbol{e}_{ij\mid v_i \in \mathcal{NH}_i^{①}}\right)$
		Mean Curvature	see Zhou (2018)
		Gaussian Curvature	
		Face Normal	$\mathbf{n} = [n_x, n_y, n_z]^T = \frac{\boldsymbol{e}_{v_1,v_2} \times \boldsymbol{e}_{v_1,v_3}}{\lvert\boldsymbol{e}_{v_1,v_2} \times \boldsymbol{e}_{v_1,v_3}\rvert}$
		Face Area	$a = \sqrt{s(s - \|\boldsymbol{e}_{v_1,v_2}\|)(s - \|\boldsymbol{e}_{v_2,v_3}\|)(s - \|\boldsymbol{e}_{v_1,v_3}\|)}$
		Relative Height of Face	$\delta h = \text{nDSM} = \text{DSM} - \text{DTM}$
Radiometric with texture patch T_f	RGB	Face Color	$R, G, B = \text{median}(T_f)$
		Face Color Variance	$\sigma_R, \sigma_G, \sigma_B = \text{std}(T_f)$
		Face Color Signature	$histo_{RGB}^{8\text{bins}}(T_f)$
	HSV	Face Color	$H, S, V = \text{median}(T_f)$
		Face Color Variance	$\sigma_H, \sigma_S, \sigma_V = \text{std}(T_f)$
		Face Color Signature	$histo_{HSV}^{8\text{bins}}(T_f)$

Features may be calculated per face or vertex. We attach per-vertex features of the face-defining vertices v_1, v_2, and v_3 to the respective face. Per-vertex features account for 3 components in the per-face feature vector. Face features such as *face normal* $\mathbf{n}$, *face area* a, *relative height of face* δh, and radiometric features are straight-forward to calculate. We briefly introduce the calculated features in the following.

The most "native" geometric mesh feature is the *face normal* $\mathbf{n}$ that is unambiguously defined by the cross-product of the triangle edges. In contrast, the normal has to be estimated by an approximated local plane in case of PCs. *Face area* denotes the area of a face calculated with Heron's formula. To obtain terrain-specificity, we incorporate DTM information for the geolocated mesh. The *relative height of face* describes the height above ground of the CoG for each face. The required terrain height is bilinearly interpolated from the available DTM (see Chapter 4).

The geometric per-vertex features inherently consider contextual information by exploiting the 1-ring neighborhood $\mathcal{NH}_i^{①}$ of a vertex v_i. The 1-ring neighborhood $\mathcal{NH}_i^{①}$ comprises all adjacent vertices of v_i. Thereby, $\mathcal{NH}_i^{①}$ implicitly includes all incident edges and faces.

Valence, also known as *degree*, defines the number of edges incident to a vertex and hence describes its degree of connectivity. The *dihedral angle* of a vertex represents the maximum dihedral angle between its incident faces and is a measure for surface flatness. A dihedral angle of a vertex is the angle between two faces sharing a common edge that incidents at the considered vertex. It is calculated by the dot product of the *face normals*. Similarly, the curvature features describe the surface topography (Zhou, 2018).

To obtain radiometric features, we first extract a texture patch T_f for each face f from the lightweight texture atlas. Alternatively, we could derive multiple texture patches of varying size, resolution, and radiometric quality for each face from all associated images. Neglecting the reasoning that led to the texture atlas would increase processing time. The parsed OBJ files provide texture coordinates (u, v) in the range of $[0; 1]$ for each vertex. Equation 6.4 shows the transformation from texture coordinates (u, v) to pixel coordinates (r, c) accounting for different origin definitions and zero-indexing. Whereas the origin is located at the lower left in the texture atlas for texture coordinates, it is located at the upper left for pixel coordinates. The pixel coordinates fluctuate in the range of $[0; n_{\text{rows}} - 1]$ and $[0; n_{\text{cols}} - 1]$ respectively.

$$\begin{aligned} c &= \lfloor u \cdot (n_{\text{cols}} - 1) \rfloor \\ r &= \lfloor (n_{\text{rows}} - 1) - v \cdot (n_{\text{rows}} - 1) \rfloor \end{aligned} \tag{6.4}$$

The pixel positions of face-defining vertices form the corners of the triangular texture patch. Pixels inside the triangle are detected by a point-in-polygon test, which is applied to the pixel set spanned by the minimum bounding rectangle of the corner pixels.

The predominant *face color* is captured by the median RGB tuple of the texture patch. We use RGB color information as well as its transformed pendant in HSV color space to ensure lighting-independent features. To increase color expressivity, we calculate the *color variance* per face for both color spaces. Additionally, we calculate 8-binned histograms for each color channel to derive the *face color signature*. Binning of 8 was set heuristically to balance the number of empty bins and the required memory. For comparison, vanilla 256-binning would result in sparser and thus less meaningful histograms with an increased footprint by factor 2^5 per channel.

Mesh features intrinsically integrate contextual information. Per-face features gather information across the entire face. For instance, radiometric features integrate gray values over the texture patch. Per-vertex features exploit the mesh topology by incorporating $\mathcal{NH}_i^{①}$.

In analogy to PC feature calculation, we calculate multi-scale mesh features based on varying neighborhood sizes to establish multiple levels of abstraction. To efficiently derive multi-scale mesh features, we attach the derived per-vertex and per-face features to the CoG cloud and abuse the established PC pipeline. The CoG cloud allows handling the mesh like a PC while preserving mesh features that have been calculated on the mesh topology. For comparability, we use spherical neighborhoods $\mathcal{NH}_i^r$ with identical radii like for PCs to create smoothed features. Analogously to LiDAR data, we calculate the Gaussian weighted average within $\mathcal{NH}_i^r$ (see Equation 6.3).

Abusing the CoG cloud and the spherical neighborhood definitions is beneficial to circumvent the computationally expensive texture patch extraction and the subsequent radiometric feature calculation on multiple levels. The calculation of radiometric features takes significantly longer than the computation of geometric attributes due to the elaborate detection of texture patches (including coordinate transformation and point-in-polygon test) and the information integration across faces. Under this aspect, we argue that the simplified

feature smoothing not incorporating the mesh topology is justifiable. Recapitulating, while the feature calculation incorporates the mesh topology, the feature regularization neglects the mesh topology for the sake of simplicity and velocity. We limit the multi-scale feature calcultion to *valences*, *dihedral angles*, *face area*, and the radiometric features *face color* and *face color variance* in HSV color space to keep processing time within feasible bounds.

6.2.1 Summary

Feature-engineering plays an important role for feature-driven ML approaches. The 3D geometry allows comparably easy calculation of a plentitude of expressive features compared to image space, where perspective, occlusions, and geometric defects aggravate feature calculation. We draw on established handcrafted geometric and radiometric features for both 3D modalities. We calculate per-entity features and contextual features on multiple scales. For comparability and efficiency, multi-scale contextual features encode the local vicinity based on spheres of varying radii for both modalities and all features are standardized. Tables 6.3 and 6.4 list the derived features for the LiDAR PC and mesh respectively. We tried to encode the modality-specific characteristics into the derived features. We mainly derived geometric features for the PCs, whereas we mostly derived radiometric features for the textured meshes. We leveraged the proposed and validated multi-modal entity linking to generate multi-modal feature descriptors on the mesh. The resulting feature vectors fuse features from the fine-grained LiDAR data and the textured mesh. Their utility is analyzed for the semantic segmentation of meshes in Section 6.5.1. In general, feature sharing enables the transfer of modality-specific properties to other modalities. For instance, the feature *reflectance* is provided for the mesh (see Figure 5.7). On the other hand, PCs can be colorized with texture from the meshes (see Figure 4.3).

6.3 Evaluation Metrics

The semantic segmentation results are evaluated by means of the derived normalized confusion matrices. To obtain performance metrics for individual classes, we determine the number of True Positives TP_c, the number of False Positives FP_c, and the number of False Negatives FN_c for each class c. These quantities are used for deriving the per-class performance metrics Precision P_c (also known as User's Accuracy UA) and Recall R_c (also known as Producer's Accuracy PA) for each class c (see Equations 6.5 & 6.6). Additionally, we derive a per-class $F1$-score $F1_c$ as the harmonic mean of P_c and R_c (see Equation 6.7) (Goutte and Gaussier, 2005).

$$UA_c = P_c = \frac{TP_c}{TP_c + FP_c} \tag{6.5}$$

$$PA_c = R_c = \frac{TP_c}{TP_c + FN_c} \tag{6.6}$$

$$F1_c = \frac{2 \cdot P_c \cdot R_c}{P_c + R_c} \tag{6.7}$$

We derive surface-aware versions of the metrics given above to evaluate the performance on the meshes. The surface-aware evaluation bases on the covered area of correctly/incorrectly classified faces. Hence, the surface-aware counterparts interpret the quantities TP_c, FP_c, and FN_c as area instead of a simple count.

To describe the total performance of a classifier at a global scale, we combine individual class scores by computing i) the Overall Accuracy $OA = \sum_c TP_c/N$, with N being the total number/area of labeled instances and ii) the arithmetic mean of per-class $F1$-scores $mF1 = \sum_c F1_c/N_C$, with N_C being the total number of classes. These two metrics are also known as micro and macro $F1$-scores. Both metrics are depicted in our normalized confusion matrices in the rightmost column as second-last and last element (OA and $mF1$-score respectively).

The conjunction of both global metrics allows a meaningful analysis of imbalanced data sets. Whereas the OA represents the relative share of correct predictions, the $mF1$-score determines whether the classifier performs well across all classes.

6.4 Semantic Segmentation of Point Clouds

The automated interpretation of PCs has become an important task in recent years. In this section, we analyse the performances for data sets V3D and H3D of *directly* and *indirectly* achieved semantic segmentation results (see Section 6.4.1 and Section 6.4.2 respectively). The *direct* approach refers to the conventional semantic segmentation with classifier training and evaluation on the PC itself (see *left column* in Figure 6.1). In contrast, the *indirect* approach refers to taking a circuit to the mesh modality (via PCMA), where training and inference are done, and per-face predictions are backtransferred to the PC. Figure 6.1 mimics the indirect workflow by starting from the PC, deploying *Feature Sharing* and *GT Sharing*, classifier on the mesh (*center column*), and ending with an annotated PC by *Prediction Sharing* after the inference made on the mesh. The circuit to the mesh modality can be seen as an implicit face-based geometry-aware segmentation of the PC. Each face aggregates multiple points (see Figure 5.5) and thus reduces the number of entities the classifier has to handle. Eventually, the transfer of per-face predictions to the PC annotates multiple points at once and may reduce noisy individual per-point predictions. In our experiments, we utilize RF models as supervised ML classifiers. The RF classifiers deploy various feature vector compositions as defined in Table 6.1. We refrain from an in-depth analysis of the studied feature vector compositions in this section as they are discussed in detail in the context of semantic segmentation of meshes. Therefore, we kindly refer the reader to Section 6.5.1 for a more detailed analysis of the feature sets.

6.4.1 Direct Semantic Segmentation of Point Clouds

Table 6.5 shows the results of the *direct* semantic segmentation of PCs on data sets V3D and H3D achieved with RF classifiers deploying various feature vector compositions (see Table 6.1) that can be directly derived from the PCs. Please note, we do not derive "mesh-only" feature vector configurations on the PC although being technically possible as we put the mesh into the center of our multi-modal analyses. Besides, Section 6.5.1 shows that "PC-only" configurations perform better in terms of $mF1$-scores than "mesh-only" configurations wherefore we do not expect better performance on the PCs neither. Therefore, we transfer only texture from the mesh to colorize the LiDAR PCs. The trained RF models weight classes inversely proportional to the class-specific frequency to tackle the class imbalance.

We refrain from discussing per-class performances of all feature vector compositions in this chapter as the thesis focuses on semantic mesh segmentation. However, Table 6.5 gives class-specific $F1$-scores for the sake of completeness. The table shows that the investigated feature vector compositions behave and rank similarly like on the mesh (see Table 6.8). For both data sets, $\mathcal{FS}_g/\mathcal{FS}_{g'}$ performs worst while $\mathcal{FS}_h/\mathcal{FS}_{h'}$ and $\mathcal{FS}_i/\mathcal{FS}_{i'}$ are best performing "PC-only" configurations. The "colored" feature sets $\mathcal{FS}_k$ and $\mathcal{FS}_{k'}$ outperform the single-modal feature sets significantly for both data sets. In the following, we shortly discuss the color importance by comparing performances of $\mathcal{FS}_k/\mathcal{FS}_{k'}$ with its non-colored counterparts $\mathcal{FS}_i/\mathcal{FS}_{i'}$.

The analysis reveals that the availability of color information significantly improves the automated semantic segmentation of ALS PCs in terms of $mF1$-scores, OAs, and prediction confidence. We abuse the aggregated prediction probability as a simple measure for confidence. $\mathcal{FS}_k$ outperforms its non-colored counterparts $\mathcal{FS}_i$ by 2.99–5.76 pp for $mF1$-score and by 4.60–9.74 pp for OA on V3D and H3D. The primed versions achieve $\Delta mF1\text{-score} = 3.87$ pp and $\Delta OA = 6.17$ pp on H3D. Figure 6.16 gives a visual comparison of $\mathcal{FS}_{i'}$ and $\mathcal{FS}_{k'}$ on H3D. The statistics consolidate that the classifier does not only achieve more correct predictions, it is also more confident in the made predictions. The median prediction probability increases by approximately 13 pp on H3D when color information is considered (comparing $\mathcal{FS}_{i'}$ with $\mathcal{FS}_{k'}$). More precisely, the median prediction probability for correct predictions improves by approximately 10 pp. The additional color information improves the median prediction probability of correct predictions specifically for ground classes *Low Vegetation*, *Impervious Surface*, and *Gravel/Soil* by roughly 18–19 pp, 6 pp, and 13–15 pp respectively (see Figure 6.16). The ground classes feature similar geometric properties such as similarly oriented normals and smooth surfaces while bearing different radiometric signatures. Likewise, the per-class $F1$-scores increase by 8.86 pp, 6.36 pp, and 17.84 pp respectively when color is incorporated. The improvement is mainly caused by the improved inter-class separation. For instance, 21 % of *Impervious Surface* is

Table 6.5: Point-level per-class $F1$-scores [%], $mF1$-scores, and OAs of *direct* semantic segmentation of PCs achieved with RF models deploying various feature vector compositions $\mathcal{FS}$ (see Table 6.1) for test sites of V3D (*top*) and H3D (*bottom*). Configurations on H3D that consider PC features based on small-scale radii are marked with a prime and highlighted in *gray*. Best performing metrics are marked in *bold*; worst are *underlined*. The extreme performance metrics are marked for both the default and the primed configurations in the case of H3D. The *black bars* above class names reflect the class distributions of the test sets.

Vaihingen 3D

Data	$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Car*	*Fence/ Hedge*	*Roof*	*Facade*	*Shrub*	*Tree*	$mF1$ [%]	*OA* [%]
V3D (PC)	e	55.32	70.79	43.31	16.14	75.03	54.10	28.56	65.06	51.04	62.83
	f	66.22	72.58	44.97	19.77	86.41	56.22	41.08	69.12	57.05	70.30
	g	46.04	74.29	17.26	17.27	37.67	6.95	16.80	30.03	30.79	44.55
	h	80.64	90.82	**67.22**	**21.27**	86.81	54.92	42.18	68.67	64.07	78.68
	i	80.65	90.83	66.91	19.34	86.76	54.89	42.46	68.78	63.83	78.67
	k	**81.29**	**91.35**	66.76	14.97	**94.80**	**60.78**	**43.56**	**81.07**	**66.82**	**83.27**

Hessigheim 3D

Data	$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	*OA* [%]
H3D (PC)	e	74.82	68.45	33.65	32.02	86.50	70.64	53.81	92.31	16.66	37.11	61.55	57.05	71.60
	e'	78.86	78.75	44.64	39.38	88.01	75.55	59.51	94.14	18.71	51.31	66.56	63.22	76.40
	f	77.55	71.16	38.78	35.24	94.26	74.44	56.13	92.74	27.70	52.40	73.39	63.07	76.11
	f'	81.50	82.20	50.16	42.31	**95.85**	77.29	61.24	94.37	26.19	**57.30**	78.44	67.90	80.58
	g	44.53	29.02	11.08	10.11	45.17	15.77	7.95	69.48	25.28	1.59	18.64	25.33	37.20
	g'	50.27	29.12	18.44	15.37	46.14	21.43	11.36	72.53	25.76	2.54	21.82	28.62	41.05
	h	77.87	72.02	54.10	38.48	93.94	74.81	58.16	92.90	34.07	50.72	75.18	65.66	76.88
	h'	81.07	82.04	61.37	44.78	95.71	76.76	64.11	94.66	27.78	54.29	80.27	69.35	80.61
	i	77.53	71.54	54.58	39.55	93.62	74.51	58.61	93.42	32.28	44.90	**75.42**	65.09	76.51
	i'	81.00	81.66	60.74	45.31	95.60	77.00	64.42	94.87	29.19	54.98	**80.30**	69.55	80.59
	k	**89.99**	**87.48**	**55.79**	**45.90**	**95.25**	**76.33**	**62.24**	**94.46**	**45.76**	**53.82**	72.32	**70.85**	**86.25**
	k'	**89.86**	**88.02**	**63.43**	**49.04**	95.57	**77.71**	**66.31**	**95.42**	**47.03**	55.80	79.39	**73.42**	**86.76**

confused with *Low Vegetation* for $\mathcal{FS}_{i'}$. The confusion rate is roughly halved when color is incorporated. In the same way, the confusion of *Gravel/Soil* and *Low Vegetation* improves significantly. However, the performance for class *Gravel/Soil* is rather poor for all configurations since there is often confusion with other ground classes. Some instances of *Gravel/Soil* feature similar radiometric properties like *Low Vegetation* (similar to bare soil) and *Impervious Surface* (similar to debris and gravel). The confusion and uncertainty are further amplified by the acquisition during the leaf-off season, in which vegetation is often dominated by a non-green color. Similarly, class *Urban Furniture* performs badly due to the great variety of objects belonging to this class since essentially it serves as quasi-class *Other*. For instance, power lines are consistently predicted as cars for all configurations (see Figure 6.16). Correctly predicting them as *Urban Furniture* is demanding since the train set does not comprise any power lines. The analysis indicates that fine-grained texture is not that important for vegetational classes (*Tree* and *Shrub*) in the case of H3D. On V3D, we see a performance gain of 12.29 pp for $F1$-score of class *Tree*. We assume that H3D data benefits from higher point density and better multi-echo ability of the employed sensor. Besides, vegetation in H3D does not necessarily show the typical green color as it was flown in early spring.

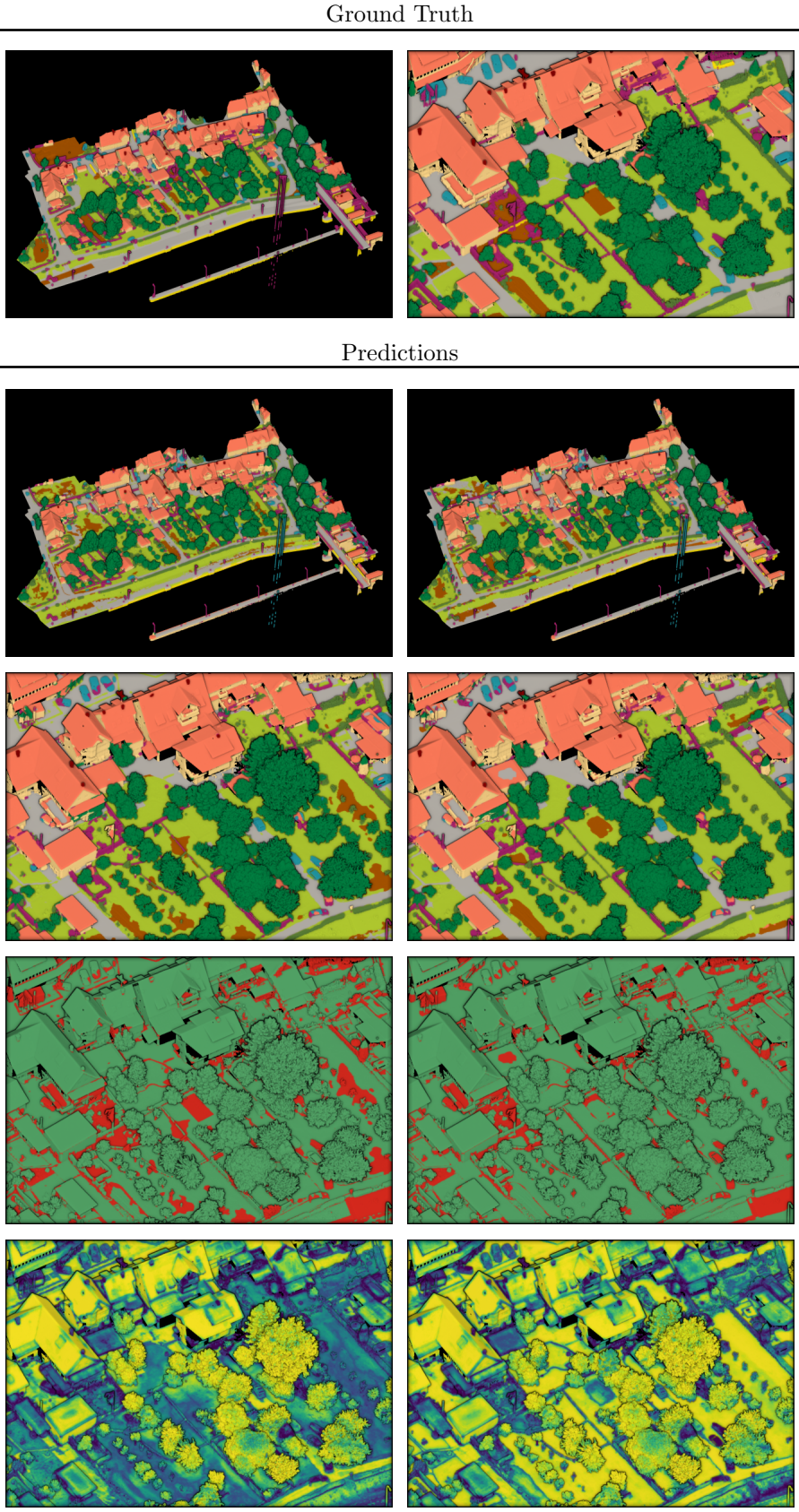


Figure 6.16: Ground truth (*top row*) and predictions for the H3D PC based on the feature vector compositions $\mathcal{FS}_{i'}$ (*left*) and $\mathcal{FS}_{k'}$ (*right*). Table 6.1 defines the feature vector sets. The prediction section shows the predictions on the test set as overview (*second row*) and close-up (*third row*), difference plots between the predictions and the GT (*forth row*), and the prediction probabilities (*last row*). The difference plots show correct predictions in *green* and false predictions in *red*. The prediction probabilities are colored in *blue* (low) and *yellow* (high).

Table 6.5 and the respective analysis does not consider the PCMA-derived linking information as filter condition for the *direct* semantic segmentation. Incorporating the linking information generates two subsets for both the train data $\mathcal{T}^{\mathrm{PC}}$ and test data $\mathcal{E}^{\mathrm{PC}}$. Subsets $\mathcal{T}_a^{\mathrm{PC}}$ and $\mathcal{E}_a^{\mathrm{PC}}$ contain only points associated with the mesh. The respective complementary subsets $\mathcal{T}_n^{\mathrm{PC}}$ and $\mathcal{E}_n^{\mathrm{PC}}$ consist of the remaining unlinked points. The linking information allows training and evaluation classifiers on arbitrary pairs of train and test sets. Tables 7.1-7.4 list the performances for all possible combinations of train sets and test sets for the studied feature vector compositions on both data sets for the sake of completeness. The evaluation on $\mathcal{E}^{\mathrm{PC}}$ shows best or close-to-best $mF1$-scores for the vast majority of feature sets regardless of the utilized train set. Likewise, the classifiers achieve the best $mF1$-scores for the vast majority of feature sets when trained on the entire train set $\mathcal{T}^{\mathrm{PC}}$. Hence, filtering points by their linking state does not improve the classifier training. In summary, the tuple $(\mathcal{T}^{\mathrm{PC}}, \mathcal{E}^{\mathrm{PC}})$ achieves best $mF1$-scores on both data sets. We want to emphasize that unlinked subsets should not be confused with the subset of linked points that carry an inconsistent label after the forward-backward passing (discussed in Section 6.1.2). We assume that the filtering of these linked points will slightly improve the classifier's training and its performance. However, such an analysis is out of the scope of this work.

As we are interested in the classifier's performance on the entire cloud, $\mathcal{E}^{\mathrm{PC}}$ is the most reasonable choice as the test set. However, we refer to tuple $(\mathcal{T}_a^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{PC}})$ for fair a comparison with results from the *indirect* semantic segmentation (see Section 6.4.2). The associated subsets guarantee to train and evaluate on the same receptive fields on both modalities. Therefore, the excerpt of this particular tuple is compactly listed in Table 6.6 for V3D and H3D. The comparison is done in Section 6.4.2.

Table 6.6: Point-level per-class $F1$-scores [%], $mF1$-scores, and OAs of *direct* semantic segmentation of PCs achieved with RF models deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data sets V3D (*top*) and H3D (*bottom*). The RF models are trained and evaluated on the subset of points that have been linked to the mesh via PCMA, i.e., on $(\mathcal{T}_a^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{PC}})$. The performance metrics can be compared in a fair way with performances of the *indirect* semantic segmentation of PCs (see Table 6.7). Configurations on H3D that consider PC features based on small-scale radii are marked with a prime and highlighted in *gray*.

Vaihingen 3D $(\mathcal{T}_a^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{PC}})$

$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Car*	*Fence/ Hedge*	*Roof*	*Facade*	*Shrub*	*Tree*	$mF1$ [%]	*OA* [%]
e	52.00	72.54	29.10	17.65	78.10	53.43	29.87	64.74	49.68	65.00
f	60.49	74.74	29.44	20.35	88.31	55.11	40.44	66.98	54.48	71.38
g	47.10	76.65	11.68	18.26	41.58	8.03	15.20	27.90	30.80	47.84
h	81.13	93.21	58.20	21.75	88.73	54.17	42.24	67.38	63.35	81.38
i	81.15	93.32	57.02	21.49	88.60	54.18	42.18	67.26	63.15	81.34
k	81.93	93.81	56.59	13.87	95.72	62.63	41.63	80.79	65.87	85.92

Hessigheim 3D $(\mathcal{T}_a^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{PC}})$

$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	*OA* [%]
e	75.28	69.36	33.70	31.67	86.67	72.80	55.89	90.23	14.74	42.55	60.82	57.61	71.44
e'	79.67	79.30	45.56	38.67	88.01	77.16	61.88	92.60	15.75	49.86	67.35	63.26	76.40
f	77.56	71.50	40.70	35.35	94.27	76.57	57.67	90.88	23.67	58.17	73.36	63.61	75.69
f'	82.01	82.31	50.96	41.29	95.75	79.08	63.03	93.06	23.24	57.75	77.33	67.80	80.44
g	47.27	29.34	13.20	9.25	44.18	15.79	5.88	62.23	25.04	1.64	19.67	24.86	36.69
g'	51.95	29.27	21.69	14.16	45.04	21.73	7.78	65.32	25.77	2.67	24.21	28.14	40.07
h	77.66	72.29	55.69	37.11	94.08	76.37	59.57	91.02	30.35	52.82	72.04	65.36	76.21
h'	82.01	82.20	62.81	43.94	95.82	78.51	65.66	93.35	27.79	54.80	78.08	69.54	80.88
i	77.85	72.61	55.68	38.40	94.20	76.39	60.28	91.55	31.62	49.99	72.97	65.59	76.48
i'	81.60	82.18	62.32	44.20	95.72	78.89	65.48	93.46	26.55	57.52	80.52	69.86	80.62
k	90.79	88.56	58.04	44.92	95.28	78.34	62.38	93.33	47.96	59.32	71.26	71.83	86.94
k'	90.43	89.18	64.61	49.35	95.77	79.84	66.64	94.48	48.97	60.92	77.05	74.30	87.36

6.4.2 Indirect Semantic Segmentation of Point Clouds

Owing to the PCMA, we achieve to semantically segment PCs indirectly by transferring the semantic segmentation results from the meshes (see Section 6.5.1, Table 6.8) to the correspondingly linked points. Table 6.7 compactly lists the performance metrics of the *indirect* semantic segmentation of PCs achieved with various RF models deploying different feature vector compositions (see Table 6.1) for data sets V3D and H3D. The classifiers are trained on $\mathcal{T}_a^{\text{Mesh}}$ which consists of faces that have been associated with the PC carrying a valid label and multi-modal information. The performance metrics are evaluated on the points of the test set that have been linked to the mesh, i.e., on $\mathcal{E}_a^{\text{PC}}$. Hence, Table 6.7 reflects the propagated results of the *direct* semantic segmentation of the respective meshes as given in Table 6.8. Generally, the metrics evaluated on the PC (see Table 6.7) differ slightly from those evaluated on the mesh (see Table 6.8) due to the one-to-many relationship between faces and points. However, the performance ranking of the feature vector compositions behaves similarly on both modalities. Best-performing RF classifiers deploy the multi-modal feature vectors for V3D and H3D. Feature vectors $\mathcal{FS}_g$ and $\mathcal{FS}_{g'}$ respectively perform the worst for both data sets. Section 6.5.1 gives a more detailed analysis of the deployed feature sets.

In the following, we discuss the results of the indirect semantic segmentation of the PCs by comparing the transferred predictions to the corresponding directly achieved predictions. We evaluate both approaches on subset $\mathcal{E}_a^{\text{PC}}$ to enable a fair comparison as the transferred predictions are limited to the linked subset. To assure comparable training setups, we train the modality-specific classifiers with the associated subsets $\mathcal{T}_a^{\text{PC}}$ and $\mathcal{T}_a^{\text{Mesh}}$ respectively. $|\mathcal{T}_a^{\text{PC}}|$ and $|\mathcal{T}_a^{\text{Mesh}}|$ differ due to the one-to-many relationship between faces and points. Nevertheless, the counterparts access the same receptive field on both modalities. Table 6.6 lists the performance metrics for the direct semantic segmentation of PCs achieved with various RF classifiers deploying different feature vectors for that specific tuple $(\mathcal{T}_a^{\text{PC}}, \mathcal{E}_a^{\text{PC}})$ on V3D and H3D.

As stated in Section 6.4.1, we limit the direct semantic segmentation of PCs to "PC-only" configurations and the sparse version of the multi-modal feature vector (see Table 6.1). Therefore, the comparison of direct and indirect results is constrained to these dedicated feature vector compositions.

The comparison does not show a clear superior approach for the investigated data sets and feature vector compositions. For instance, the geometric "PC-only" configurations ($\mathcal{FS}_e$ and $\mathcal{FS}_f$) show superiority for the indirect approach in terms of OA for V3D while achieving similar $mF1$ scores. $\mathcal{FS}_e$ is 0.75 pp better in OA when applying the indirect approach on V3D. On the contrary, $\mathcal{FS}_e$ and $\mathcal{FS}_{e'}$ show better performance for the direct approach on H3D. The direct approach outperforms the indirect approach by 0.81 pp and 0.96 pp for $mF1$-score and OA respectively deploying $\mathcal{FS}_e$. The primed version is 1.06 pp and 0.73 pp better for $mF1$-score and OA respectively when training and inference is done directly on the PC. The pure radiometric feature set $\mathcal{FS}_g$ is 0.92 pp better in $mF1$ for the indirect approach on V3D. On H3D, $\mathcal{FS}_g$ and $\mathcal{FS}'_g$ show on par performance for the two approaches.

To make a grouped analysis of the "PC-only" configurations, we calculate the median of performance differences as a robust indicator to assess whether the direct or indirect approach performs better in an overall view. For completeness, the mean μ is denoted in brackets. The median difference across all "PC-only" feature sets $\mathcal{FS}$ indicates that the direct and indirect approaches are roughly on par for V3D and H3D in terms of $mF1$-scores ($|\operatorname{med}(\Delta mF1_{\mathcal{FS}})| < 0.26$ pp, $|\mu(\Delta mF1_{\mathcal{FS}})| < 0.34$ pp). The same holds for OAs on H3D ($|\operatorname{med}(\Delta OA_{\mathcal{FS}})| < 0.19$ pp, $|\mu(\Delta mF1_{\mathcal{FS}})| < 0.06$ pp). On V3D, the median difference between direct and indirect approach across "PC-only" configurations is 0.66 pp better in the indirect than the direct approach ($\mu(\Delta OA_{\mathcal{FS}}) = +0.45$ pp).

In Section 6.5.1, we show that the multi-modal feature sets outperform the "PC-only" configurations. Therefore, the comparisons of the direct and indirect approach for $\mathcal{FS}_k$ and $\mathcal{FS}_{k'}$ respectively are the most relevant to us. The multi-modal descriptor $\mathcal{FS}_k$ is 1.39 pp better in $mF1$ for the indirect approach on V3D. The evaluation on H3D shows the reversed behavior. The direct approach outperforms the indirect method significantly in terms of $mF1$ by 3.15 pp and 1.02 pp for $\mathcal{FS}_k$ and $\mathcal{FS}_{k'}$ respectively. For $\mathcal{FS}_k$, the direct approach achieves also a better OA by 1.08 pp. The direct approach has the advantage to access and process the high-resolution texture for each point, whereas per-face descriptors of the indirect approach aggregate the texture information per face. We assume that the high point density of the H3D ALS data (800 pts/m^2)

Table 6.7: Point-level per-class $F1$-scores [%], $mF1$-scores, and OAs of *indirect* semantic segmentation of PCs achieved with RF models deploying various feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data sets V3D (*top*) and H3D (*bottom*). The RF models are trained on faces that have been associated with the PC, i.e., on $\mathcal{T}_a^{\text{Mesh}}$. The per-face predictions are transferred to the corresponding points via PCMA. The performance metrics are evaluated on the subset of the test split $\mathcal{E}_a^{\text{PC}}$ where points are linked to the mesh.
Configurations on H3D that consider PC features based on small-scale radii are marked with a prime and highlighted in *gray*. Best performing metrics are marked in *bold*; worst are *underlined*. The extreme performance metrics are marked for both the default and the primed configurations in the case of H3D.
The *black bars* above the class names reflect the class distributions of the respective $\mathcal{E}_a^{\text{PC}}$.

Vaihingen 3D ($\mathcal{T}_a^{\text{Mesh}}, \mathcal{E}_a^{\text{PC}}$)

Data	$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Car*	*Fence/ Hedge*	*Roof*	*Facade*	*Shrub*	*Tree*	$mF1$ [%]	*OA* [%]
V3D	*a*	30.98	59.99	4.97	2.79	66.60	31.39	7.82	50.38	31.86	50.42
	b	53.32	66.85	14.10	6.36	86.14	30.59	35.04	61.29	44.21	65.20
	c	58.62	81.34	20.63	8.98	82.06	20.05	26.92	56.40	44.37	68.35
	d	79.80	91.70	27.57	14.64	94.33	44.44	39.75	76.94	58.65	83.81
	e	51.93	75.11	28.50	19.92	77.63	52.53	29.11	64.78	49.94	65.75
	f	61.15	76.99	27.84	21.66	88.16	53.87	41.32	66.95	54.74	72.25
	g	46.24	75.19	12.75	19.57	44.91	8.72	16.26	30.12	31.72	48.50
	h	80.96	93.32	57.08	22.01	88.34	52.96	42.40	67.21	63.04	81.24
	i	80.94	93.21	**57.51**	**23.60**	88.77	53.31	42.70	67.37	63.43	81.43
	j	80.52	93.60	52.70	18.33	95.12	**63.57**	41.28	80.75	65.73	85.54
	k	**81.78**	**93.75**	55.97	22.18	**95.54**	**63.57**	**43.63**	**81.63**	**67.26**	**86.08**

Hessigheim 3D ($\mathcal{T}_a^{\text{Mesh}}, \mathcal{E}_a^{\text{PC}}$)

Data	$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	*OA* [%]
H3D	*a*	61.01	55.98	6.60	15.98	65.14	53.15	19.05	70.96	6.84	41.65	47.04	40.31	56.80
	b	71.69	60.55	18.91	20.35	91.11	60.81	26.08	80.68	8.96	58.51	58.34	50.54	67.99
	c	76.76	78.06	16.34	23.11	77.31	32.64	16.38	45.07	26.21	32.83	6.29	39.18	66.66
	d	87.38	87.78	22.84	41.23	91.52	58.30	44.41	79.75	26.50	56.75	53.47	59.09	82.15
	e	74.56	68.34	33.71	30.74	85.83	72.62	54.79	90.55	14.50	39.59	59.58	56.80	70.48
	e′	79.28	79.23	43.95	36.61	87.09	76.59	58.38	92.26	15.76	49.76	65.31	62.20	75.67
	f	77.60	71.70	40.56	33.74	94.79	76.72	56.62	91.23	24.07	58.14	71.72	63.35	75.59
	f′	82.46	82.61	49.58	39.65	**95.79**	78.88	60.32	92.73	25.60	60.23	79.11	67.90	80.66
	g	50.16	27.65	10.21	9.76	50.02	16.05	5.43	62.46	24.05	1.97	16.01	24.89	37.50
	g′	52.21	28.52	16.50	14.02	53.03	21.88	7.90	65.24	24.60	3.07	20.30	27.93	40.33
	h	78.27	72.19	54.92	36.64	94.21	76.13	58.13	91.30	34.63	52.77	**76.03**	65.93	76.61
	h′	82.35	82.42	60.72	41.66	95.61	78.43	**62.95**	93.08	30.48	57.64	80.52	69.62	80.95
	i	78.20	72.43	**55.81**	37.36	94.24	76.16	58.31	91.86	33.56	52.37	74.77	65.92	76.62
	i′	82.14	82.60	**62.01**	42.16	95.56	77.85	62.73	93.35	29.84	53.97	79.61	69.26	80.81
	j	**89.82**	**88.32**	51.27	**45.07**	93.95	**79.09**	**58.81**	92.22	32.28	**58.41**	73.05	**69.30**	85.68
	j′	89.80	**89.14**	60.21	**48.26**	94.17	79.41	62.08	93.74	34.86	59.95	**81.52**	72.10	86.26
	k	89.79	87.54	49.76	41.12	**95.35**	78.09	58.66	**93.58**	**36.56**	55.80	69.21	68.68	**85.86**
	k′	**91.05**	88.99	59.11	48.25	94.48	**80.09**	62.51	**94.71**	**42.00**	**66.71**	78.12	**73.28**	**87.16**

is the reason why the direct approach is superior on H3D but not on V3D (4–8 points/m^2). Thereby, the H3D PC manages to represent textural details that cannot be featured on the colorized V3D PC.

One systemic weakness of the indirect approach is that only linked entities can be classified. In contrast, the direct approach classifies all points regardless of their association state. Moreover, direct classifiers have access to the entire training pool $\mathcal{T}^{\mathrm{PC}}$ and are not bound to the linked subset $\mathcal{T}_a^{\mathrm{PC}}$. The direct approach that trains classifiers on $\mathcal{T}^{\mathrm{PC}}$ achieves on par $mF1$-scores with its counterpart training on $\mathcal{T}_a^{\mathrm{PC}}$ for the "PC-only" feature sets $\mathcal{FS}_g$, $\mathcal{FS}_h$, $\mathcal{FS}_i$ on V3D. The additional training samples improve the $mF1$-scores of $\mathcal{FS}_e$, $\mathcal{FS}_f$, and $\mathcal{FS}_k$ significantly by 0.98 pp, 1.50 pp, and 0.53 pp respectively. The achieved OAs are on par except for feature sets $\mathcal{FS}_f$ and $\mathcal{FS}_g$. $\mathcal{FS}_f$ is 0.48 pp better when training on the entire training set. On the other hand, $\mathcal{FS}_g$ is 0.41 pp better when training on $\mathcal{T}_a^{\mathrm{PC}}$ only. Similarly, the $mF1$-scores are on par for the unprimed features sets on H3D except for $\mathcal{FS}_g$ and $\mathcal{FS}_h$. The additional training samples result in 0.66 pp worse and 0.61 pp better $mF1$-scores for $\mathcal{FS}_g$ and $\mathcal{FS}_h$ respectively. Feature sets $\mathcal{FS}_e$, $\mathcal{FS}_g$, $\mathcal{FS}_i$ achieve better OAs when training on $\mathcal{T}_a^{\mathrm{PC}}$ only (+0.50 pp, +1.10 pp, and +0.58 pp). The remaining feature sets achieve similar OA for both variants of the direct approach. The equivalents of the "PC-only" feature sets incorporating contextual features with the small-scale radii show consistently better $mF1$-scores when trained on $\mathcal{T}^{\mathrm{PC}}$. $\mathcal{FS}_{g'}$ is an exception that performs 0.62 pp better in terms of $mF1$-score when training on the linked subset. For instance, $\mathcal{FS}_{e'}$ and $\mathcal{FS}_{f'}$ show a significant improvement of 1.01 pp and 0.97 pp in $mF1$ when training on the entire training set. The OAs are roughly on par except for $\mathcal{FS}_{g'}$ and $\mathcal{FS}_{h'}$ which are 0.59 pp and 0.51 pp better when training on the linked subset. The multi-modal feature sets achieve similar performance metrics for both training sets.

In summary, the additional training samples do not consistently improve the performance of direct classifiers for all feature vectors on both data sets. However, the majority tends to better $mF1$-scores at the cost of slightly decreased OA. In other words, the additional training points make the direct classifiers better regarding the total class range.

The direct approach trained on entire $\mathcal{T}^{\mathrm{PC}}$ impacts logically also the comparison with the indirect method trained on $\mathcal{T}_a^{\mathrm{Mesh}}$. Training on the entire training pool $\mathcal{T}^{\mathrm{PC}}$ increases the performance gap between the direct semantic segmentation and the indirect approach for some feature configurations. For example, $\mathcal{FS}_e$ and $\mathcal{FS}_f$ perform numerically slightly worse in $mF1$ (0.26 pp each) for the direct approach (trained on $\mathcal{T}_a^{\mathrm{PC}}$) than the indirect approach for V3D. The additional train points improve the direct approach by 0.98 pp and 1.50 pp and therefore exceed the indirect approach by 0.72 pp and 1.24 pp respectively. On H3D, the extended train set achieves 0.69 pp and 1.91 pp better $mF1$-score and OA than the indirect approach for $\mathcal{FS}_g$ increasing the performance gap between the direct and indirect approach by 0.66 pp and 1.10 pp. Likewise, $\mathcal{FS}_{e'}$ roughly doubles the performance gap in $mF1$ (from 1.06 pp to 2.07 pp) between the direct and indirect approach when replacing $\mathcal{T}_a^{\mathrm{PC}}$ with $\mathcal{T}^{\mathrm{PC}}$. For other feature sets, the additional training samples have no significant effect on the achieved performance metrics. For instance, $\mathcal{FS}_i$ on V3D and $\mathcal{FS}_f$ on H3D show similar performance for both direct approaches and therefore, similar performance gaps in comparison with the indirect approach. The same holds for the multi-modal feature sets $\mathcal{FS}_k$ and $\mathcal{FS}_{k'}$ on H3D which keep the performance gap rather constant (3.15 pp/2.95 pp and 1.02 pp/1.11 pp for $mF1$ and OA respectively) as their direct variants achieve similar performance.

6.4.3 Summary

Direct Semantic Segmentation of Point Clouds. Section 6.4.1 displays the conventional feature-based semantic segmentation of PCs (direct approach) by example of dedicated feature vector compositions. In particular, we discuss the importance of color as derived from textured meshes. The color incorporation causes more correct and more confident predictions. The additional color information improves the inter-class separation for classes with similar geometric properties such as *Low Vegetation*, *Impervious Surface*, and *Gravel/Soil*. By these means, color is not only essential for the human spectator but also for machines. The comparison between data sets V3D and H3D shows that the colorization quality increases with increased point density. As a result, the colorized feature set $\mathcal{FS}_k$ outperforms the colorless $\mathcal{FS}_i$ more significantly on H3D than on V3D. The color-induced performance gains differ roughly by factor 2 across the data sets.

Additionally, we analyze the impact of the linking information (derived via PCMA) on the semantic segmentation of the direct approach. Therefore, we train and evaluate the classifiers on all possible tuples of training and test sets with/without incorporating the linking information. We observe that the best performance is achieved for the full tuple $(\mathcal{T}^{\mathrm{PC}}, \mathcal{E}^{\mathrm{PC}})$.

Indirect Semantic Segmentation of Point Clouds. In Section 6.4.2, we utilize the linking information to semantically segment PCs indirectly by transferring predictions from the meshes to the PCs. The comparison between the direct and indirect results does not show a clear superior approach for the investigated data sets and feature vector compositions. For instance, the indirect approach is 1.39 pp better in $mF1$ for the multi-modal feature vector $\mathcal{FS}_k$ on V3D. On H3D, the direct approach outperforms the indirect method significantly in terms of $mF1$ by 3.15 pp. The high point density of H3D may be the reason why the per-face descriptors cannot compete with the granularity of the per-point feature vectors for that feature vector composition. Neglecting the multi-modal feature sets, the discrepancies between the direct and indirect approach are moderate in general. The majority of differences is less than 1 pp for both data sets. We assume that the contextual features smooth the direct predictions such that we do not note a significant and consistently positive impact of the face-induced implicit segmentation. The visualization of direct results on the PC undergirds our assumption of smooth predictions (e.g., see Figure 6.16). Besides, the indirect approach is limited by the reconstruction quality. For the time being, the meshing suffers from reconstruction artifacts and non-class-aware reconstructed faces. Consequently, faces cover points of multiple classes and cause mispredictions on the PC for the indirect approach.

The operating range of the indirect approach is limited to the associated entities in both the training and the evaluation. Accessing the entire train set does not consistently improve the performance of the direct approach for all feature vectors on both data sets. However, the majority of feature sets tends to better $mF1$-scores at cost of slightly decreased OA when training on $\mathcal{T}^{\mathrm{PC}}$ instead of $\mathcal{T}_a^{\mathrm{PC}}$.

Conclusion. To summarize, the decision of whether the indirect or direct approach is better depends on the deployed feature set and investigated data set. However, discrepancies are quite moderate for most cases. The main drawback of the indirect approach is its limitation to linked entities. For these reasons, the conventional and direct semantic segmentation of PCs is a good choice in general. The direct approach makes the meshing and the circuit to the mesh obsolete – in particular when deploying "PC-only" feature sets. On the other hand, the indirect approach achieves comparable results on linked points while superseding an individual PC-specific classifier that has to be trained on significantly more training samples.

6.5 Semantic Segmentation of Meshes

In this section, we deep-dive the semantic segmentation of 3D textured meshes. We analyse the performances for V3D and H3D of *directly* and *indirectly* achieved semantic segmentation results (see Section 6.5.1 and Section 6.5.2 respectively). The *direct* approach refers to the conventional straightforward semantic segmentation with classifier training and evaluation on the mesh itself (see Figure 6.1, *center column*). In contrast, the *indirect* approach abuses the results of the direct semantic segmentation of PCs (see Section 6.4.1) by transferring the per-point predictions to the mesh. Figure 6.1 mimics the indirect workflow by starting from the manually annotated PC (potentially enhanced with shared mesh features), following the direct semantic segmentation of PC (*left column*), and ending with an annotated mesh by prediction sharing after the inference on the PC. Like for the semantic segmentation of PCs, we utilize RF models as supervised ML classifiers in our experiments. Unlike with the evaluation on the PCs, we evaluate performance on the meshes with surface-aware metrics (see Section 6.3). The RF classifiers deploy various feature vector compositions as defined in Table 6.1. We perform an in-depth analysis of the studied feature sets in Section 6.5.1.

6.5.1 Direct Semantic Segmentation of Meshes

We train several RF models on the meshes of V3D and H3D with the proposed feature-driven pipeline for semantic mesh segmentation. As manually assigned GT is not available for the meshes, we derive the required annotation from the respective manually annotated LiDAR PCs by transferring the given labels via PCMA. Likewise, we transfer PC-specific features to the meshes. By these means, we can arbitrarily fuse radiometric, geometric, per-instance, and contextual features computed on the mesh or the LiDAR PC in a face-specific descriptor. The calculated features derived from the PC or mesh are listed in Tables 6.3 and 6.4 respectively. Table 6.1 gives an overview of the studied feature sets $\mathcal{FS}$.

For comparability, all trained models deploy the same weighting strategy, which considers class-dependent and sample-dependent surface-aware weights. In particular, we set face areas as sample weights to account for the non-uniformity of the mesh modality. To tackle the class imbalance, we weight classes inversely proportional to the class-specific covered area. Eventually, the product of sample weight and class weight delivers the face-specific weight for each face.

The evaluation is done with surface-aware metrics (see Section 6.3). For both data sets, the trained RF models are evaluated on the dedicated test sets with semi-automatically generated labels. For H3D, we additionally analyzed performance metrics on the manually annotated subset of the test set, too. Inevitably, the recorded metrics differ numerically from their counterparts on the semi-automatically annotated GT due to disjoint manual labeling processes on the PC and mesh. However, the selected GT variant of the test set is of minor importance to compare different feature vector configurations (see paragraph *Analysis of Different Feature Vector Compositions*). The analysis has revealed that the pile of feature sets is ordered similarly on both GT variants when being ranked by global performance metrics OA or $mF1$-score. For compactness, we report and discuss only results evaluated against the semi-automatically transferred GT. The semi-automatically transferred GT features a larger extension giving a more representative result compared to the manual GT (for H3D). This decision is also in the spirit of consistency since only semi-automatically GT is given in the case of V3D.

In paragraph *Analysis of Different Inter-Modal Propagation Methods on Semantic Segmentation of Meshes*, we compare the effects of a simple NN interpolation and our proposed propagation method on classifier training and semantic segmentation. Figure 6.4 visualizes the bare differences of the propagation methods.

Analysis of Different Feature Vector Compositions. We discuss the versatile feature vector compositions $\mathcal{FS}$ listed in Table 6.1 by the example of the achieved performance metrics of RF models weighted with face areas as sample weights and the class area distribution as class weights. Feature vector compositions $\mathcal{FS}_a - \mathcal{FS}_d$ use features that have been derived on the mesh for each face ("mesh-only"). $\mathcal{FS}_e - \mathcal{FS}_i$ deploy features for each face that have been derived on the PC and propagated to the mesh ("PC-only"). To analyze the impact of modality-specific features, we refrained from transferring per-point color to the mesh. We ar-

gue that propagating colors from the PC to the mesh prevents a clear separation of modalities – particularly as PC colors have been interpolated from the mesh texture in the first place. Besides, curvature features which have shown little relevance on semantic segmentation of PCs have not been transferred from the PC to avoid redundancy of modality-intrinsic attributes in the multi-modal feature vector compositions. Feature vector $\mathcal{FS}_j$ combines the entire sets of "mesh-only" and "PC-only" features in a multi-modal descriptor for each face on the mesh. Its sparse variant $\mathcal{FS}_k$ combines "PC-only" features with texture from the mesh.

Table 6.8 lists per-class $F1$-scores, $mF1$-scores and OAs for the deployed feature vector compositions $\mathcal{FS}$ on both data sets. The class area distributions are sketched for the test splits of both data sets. For a more detailed analysis, Figures 6.17 and 6.18 show the surface-aware confusion matrices of a selection of $\mathcal{FS}$ for V3D and H3D respectively. The respective feature relevance plots (generated with python package `scikit-learn`) are depicted in Figures 7.1 and 7.2. Figure 6.19, Figure 6.20, and Figure 6.21 visualize the predictions for a close-up of the test set of H3D. The entire test area is shown in Figure 7.4 and Figure 7.5. The respective counterparts for V3D are shown in Figure 7.3.

"Mesh-Only" Feature Vector Compositions. The feature relevance plots of "mesh-only" configurations $\mathcal{FS}_a - \mathcal{FS}_d$ show that contextual features are more important than the per-face features for both radiometric and geometric features (see Figures 7.1 and 7.2 for V3D and H3D respectively). Exceptions are the *relative height* δh which, in total, is the most important feature, and the *face normal* (more precisely, its third component n_z), which is the most relevant mesh-intrinsic feature. Another distinctive feature is the *face color* expressed by median R, G, B and median H, S, V. We note that the importance gap between contextual and per-face features is larger for H3D. Generally, faces in the fine-grained H3D mesh are smaller than in the MVS mesh of V3D. The median face in V3D covers an area that is roughly 14 times larger than in H3D. Therefore, per-face features integrate more "per-face context" and contextual features are derived from fewer faces in the case of V3D. By these means, a single face of V3D impacts more the contextual features than a face of the H3D mesh. In other words, per-face features differ more from contextual features in the case of H3D.

Inspecting $\mathcal{FS}_d$, we observe that contextual color features are of similar importance as contextual geometric features for H3D. In contrast, contextual color features are slightly more important than contextual geometric features for V3D. We claim that the increased importance of contextual texture features for V3D is due to the comparably low geometric quality of the MVS mesh generated from nadir imagery (GSD = 8 cm).

Comparing confusion matrices of $\mathcal{FS}_a$ and $\mathcal{FS}_c$, we notice radiometric features perform better than mesh-intrinsic geometric features for both data sets. For V3D, $\mathcal{FS}_c$ significantly outperforms $\mathcal{FS}_a$ by more than 10 pp for both global performance metrics. Likewise, $\mathcal{FS}_c$ outperforms $\mathcal{FS}_a$ by 3.88 pp in terms of *OA* for H3D but both configurations achieve similar $mF1$-scores. The more fine-grained class catalog of H3D prevents $\mathcal{FS}_c$ from outperforming $\mathcal{FS}_a$ in $mF1$-score, too. The smaller performance gap between $\mathcal{FS}_a$ and $\mathcal{FS}_c$ on H3D is further explained by the discrepancy in acquisition times. H3D imagery has been acquired under leaf-off conditions in March when grassland appears brownish and looks similar to soil. On the other hand, V3D has been acquired in September. The purely radiometric feature set is insufficient to reliably resolve classes *Chimney* and *Roof* (confusion rate: 91.85 %) while outperforming $\mathcal{FS}_a$ for almost all other classes (see Figure 6.18). These findings indicate the strengths and weaknesses of the texture regarding semantic segmentation. Textural information helps considerably to cope with geometric reconstructions of lower quality like V3D meshes. On the other hand, the pure textural information is insufficient for separating classes reliably. From these findings, we deduce that textural features gain relative importance with decreasing geometric quality of meshes.

Adding δh to the mesh-intrinsic geometric feature set (i.e., $\mathcal{FS}_a \mapsto \mathcal{FS}_b$) outperforms the purely radiometric $\mathcal{FS}_c$ by 1.53 pp for $mF1$-score on the V3D mesh. The performance is on par concerning *OA* since $\mathcal{FS}_c$ achieves a significantly better per-class recall for class *Impervious Surface* which covers roughly 26 % of the surface area in the test set. For almost all remaining classes, $\mathcal{FS}_c$ performs worse than $\mathcal{FS}_b$. However, when δh is added to the mesh-intrinsic geometric feature set for the H3D mesh (i.e., $\mathcal{FS}_a \mapsto \mathcal{FS}_b$), geometric features outperform radiometry by 7.91 pp and 4.89 pp for $mF1$-score and *OA* respectively. We claim that

Table 6.8: Surface-aware per-class $F1$-scores [%], $mF1$-scores, and OAs of *direct* semantic segmentation of meshes achieved with RF models deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data sets V3D (*top*) and H3D (*bottom*). Configurations on H3D that consider PC features based on small-scale radii are marked with a prime and highlighted in *gray*. Best performing metrics are marked in *bold*; worst are *underlined*. The extreme performance metrics are marked for both the default and the primed configurations in the case of H3D. The *black bars* above class names reflect the class area distributions of the test sets.

Vaihingen 3D

Data	$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Car*	*Fence/ Hedge*	*Roof*	*Facade*	*Shrub*	*Tree*	$mF1$ [%]	*OA* [%]
V3D Mesh	*a*	29.53	61.73	6.94	1.33	66.26	38.87	5.92	53.30	32.98	51.49
	b	53.09	70.17	14.34	3.96	85.23	38.48	33.62	62.52	45.18	65.97
	c	52.89	78.70	16.92	6.62	81.51	28.77	25.68	58.11	43.65	66.34
	d	75.90	91.19	28.87	10.07	93.16	53.18	37.86	77.89	58.51	82.17
	e	50.47	75.46	32.27	16.03	78.83	61.39	28.14	66.15	51.09	65.99
	f	60.31	77.42	33.94	17.58	88.20	61.93	40.02	67.96	55.92	72.25
	g	39.10	74.23	15.56	16.29	40.03	13.11	15.38	31.82	30.69	44.45
	h	77.34	93.17	63.85	16.78	87.93	61.55	41.03	67.88	63.69	79.65
	i	77.51	93.09	**63.95**	**18.37**	88.22	61.98	41.19	68.00	64.04	79.84
	j	77.39	93.42	55.38	13.57	94.73	71.70	39.88	80.98	65.88	84.55
	k	**78.52**	**93.58**	58.57	16.07	**95.22**	**72.44**	**41.80**	**81.58**	**67.22**	**84.99**

Hessigheim 3D

Data	$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	*OA* [%]
H3D Mesh	*a*	58.59	54.86	5.08	25.87	61.27	63.54	19.62	85.05	11.97	56.95	43.52	44.21	59.90
	b	69.99	59.67	11.27	29.50	85.60	67.78	25.75	88.26	11.44	65.85	56.79	51.99	68.67
	c	70.97	71.15	12.25	29.32	73.39	52.86	23.66	69.23	26.12	48.70	7.21	44.08	63.78
	d	86.32	85.98	15.51	46.32	86.54	68.78	45.26	90.84	27.06	67.04	49.81	60.86	80.51
	e	73.52	69.21	34.58	37.53	83.21	76.09	57.86	92.94	14.70	43.43	56.03	58.10	72.70
	e′	78.59	78.36	41.03	42.15	85.46	79.40	61.42	94.08	17.03	57.34	61.88	63.34	77.10
	f	76.40	71.47	38.89	41.21	90.70	78.84	59.06	93.35	19.93	54.08	69.72	63.06	76.35
	f′	81.60	81.47	44.77	45.42	**92.72**	80.46	62.35	94.25	22.98	57.80	77.31	67.37	80.49
	g	46.62	25.70	12.35	19.13	47.39	29.91	10.99	76.91	17.23	5.50	17.36	28.10	40.87
	g′	48.24	26.24	19.33	22.94	50.54	36.68	15.49	79.54	17.62	7.45	20.68	31.34	43.76
	h	76.96	71.94	51.45	42.64	90.13	78.47	60.54	93.53	29.42	48.59	**71.97**	65.06	77.03
	h′	81.33	81.57	55.22	47.17	92.54	80.07	64.50	94.46	27.74	53.55	77.82	68.72	80.78
	i	76.78	72.24	**52.26**	43.61	90.32	78.75	60.73	93.95	29.12	49.17	71.62	65.32	77.16
	i′	81.23	81.92	**56.07**	47.33	92.71	79.72	**64.55**	94.67	27.96	51.65	77.46	68.66	80.79
	j	**88.52**	**88.25**	45.94	**51.89**	90.91	**82.84**	**61.04**	94.85	31.52	**67.38**	67.81	**70.09**	**84.84**
	j′	88.55	89.31	54.49	**55.61**	91.30	**82.55**	63.82	95.45	34.41	**68.88**	**78.61**	73.00	85.54
	k	88.34	87.79	44.40	47.38	**92.34**	80.40	60.67	**95.40**	**37.00**	55.64	63.99	68.48	84.47
	k′	**89.62**	**89.35**	53.55	54.18	91.71	81.98	64.32	**95.92**	**42.86**	66.07	74.71	**73.11**	**85.97**

Vaihingen 3D

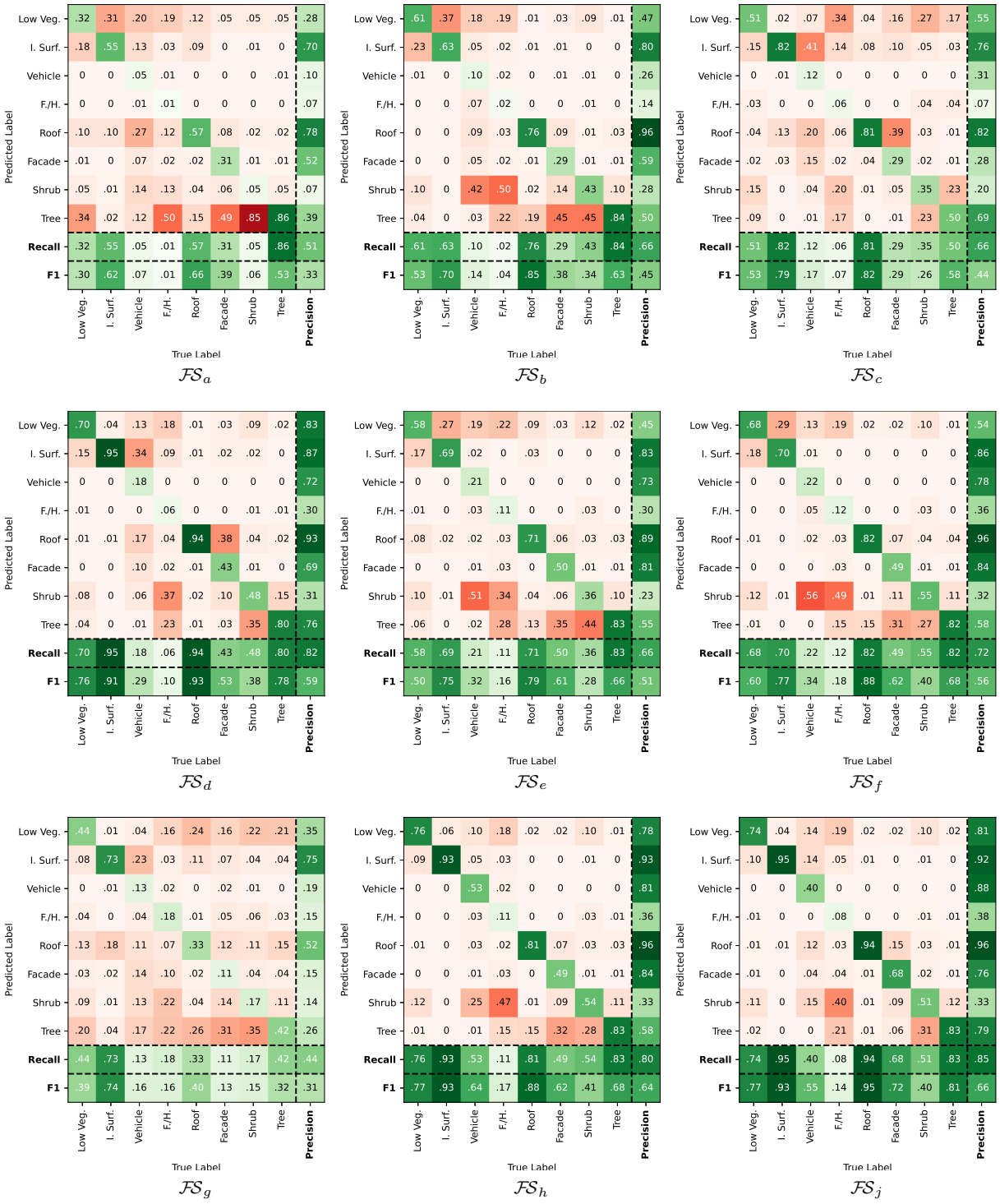


Figure 6.17: Normalized surface-aware confusion matrices of *direct* semantic segmentation of meshes achieved with RF models deploying different feature sets $\mathcal{FS}$ (see Table 6.1) for data set V3D.

Hessigheim 3D

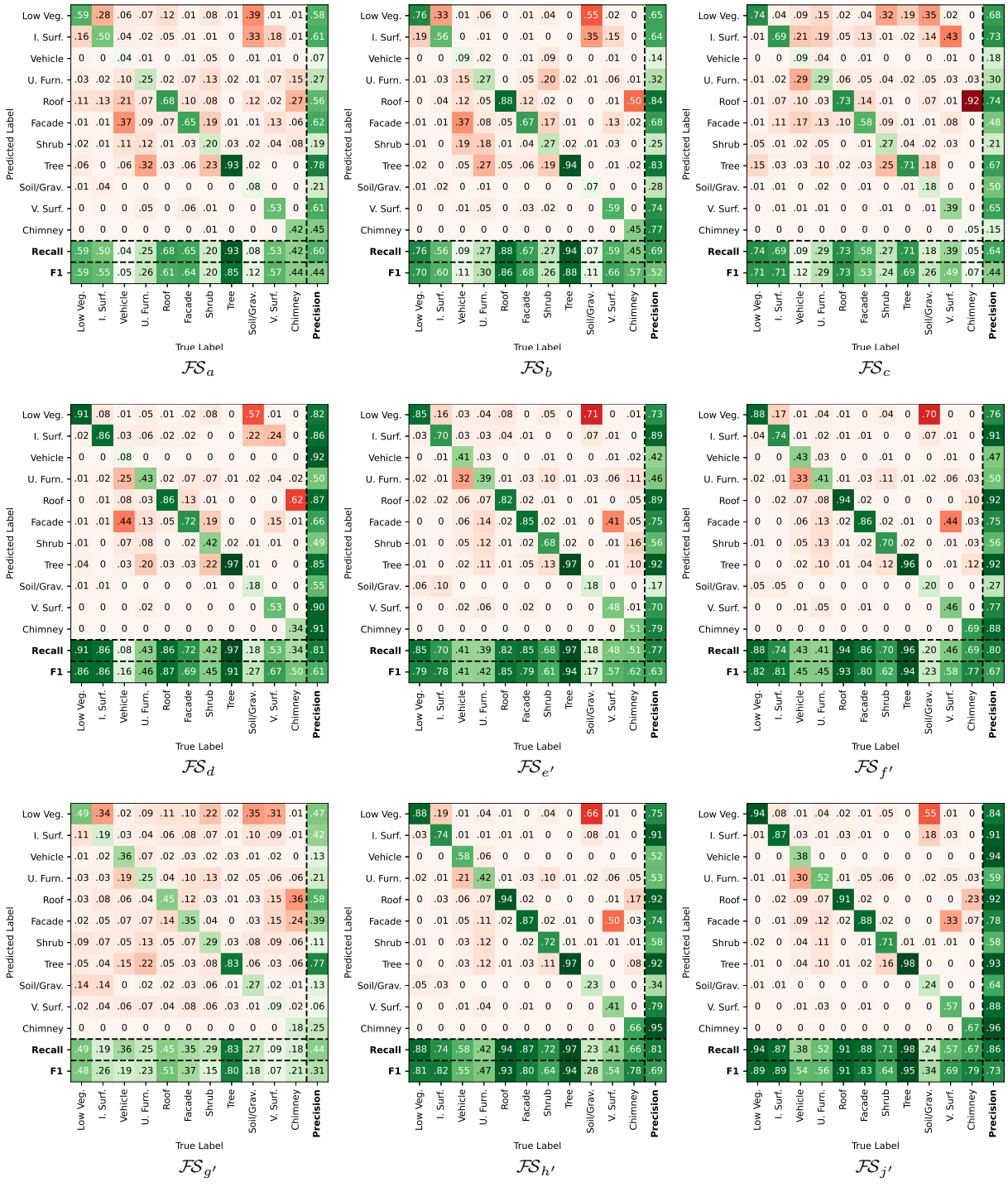


Figure 6.18: Normalized surface-aware confusion matrices of *direct* semantic segmentation of meshes achieved with RF models deploying different feature sets $\mathcal{FS}$ (see Table 6.1) for data set H3D.

the comparably low geometric quality of the nadir MVS mesh is the reason why $\mathcal{FS}_c$ achieves almost similar results like $\mathcal{FS}_b$ on V3D.

$\mathcal{FS}_d$ achieves the best performance for the "mesh-only" configurations on both data sets. The median prediction probability of correct predictions for $\mathcal{FS}_d$ is 68.23 % and 83.17 % for V3D and H3D respectively outperforming other "mesh-only" configurations by 2.48–15.33 pp and 6.55–25.21 pp respectively. The combination of geometric and radiometric features improves performance with respect to $\mathcal{FS}_b$ by 13.33 pp and 16.20 pp for $mF1$-score and OA respectively for V3D. For H3D, the improvement registers 8.87 pp and 11.84 pp for $mF1$-score and OA respectively.

Enhancing purely geometric feature sets with radiometric features undergoes a larger performance boost on V3D. The $mF1$-score is 6.81 pp higher on H3D for $\mathcal{FS}_b$ but only 2.35 pp for $\mathcal{FS}_d$.

These observations further underline the increased relative importance of texture when the reconstruction quality is comparably low. In other words, the performance metrics map the geometric superiority of H3D compared to V3D data.

"PC-Only" Feature Vector Compositions. We first analyze configurations listed in Table 6.1 for both data sets. Subsequently, we analyze the primed configurations on H3D.

Considering "PC-only" configurations $\mathcal{FS}_e - \mathcal{FS}_g$, we note that geometric features perform better than radiometric features for both data sets. The superiority of $\mathcal{FS}_e$ and $\mathcal{FS}_f$ over $\mathcal{FS}_g$ indicates the strong geometric information encoded in the PC features. Table 6.8 reveals that $\mathcal{FS}_g$ achieves worst or close-to-worst $F1$-scores for almost all classes for V3D and H3D. $\mathcal{FS}_g$ is the worst "PC-only" feature composition.

Adding δh to the PC-intrinsic geometric feature set (i.e., $\mathcal{FS}_e \mapsto \mathcal{FS}_f$) further increases the superiority of geometric features regarding radiometric features. $\mathcal{FS}_f$ outperforms $\mathcal{FS}_g$ by 25.23 pp and 27.80 pp for $mF1$-score and OA respectively on V3D. For H3D, $\mathcal{FS}_f$ outperforms $\mathcal{FS}_g$ by 34.96 pp and 35.48 pp for $mF1$-score and OA respectively. The larger performance gain on H3D underlines the geometric superiority of the H3D PC compared to the V3D PC.

The combination of geometric and radiometric features (i.e., $\mathcal{FS}_h$) outperforms configurations that use only a single feature type on both data sets. $\mathcal{FS}_h$ outperforms configurations $\mathcal{FS}_e$, $\mathcal{FS}_f$, and $\mathcal{FS}_g$ by 7.77–33.00 pp and 7.40–35.20 pp for $mF1$-score and OA respectively for V3D. The performance gain spans 2.00–36.96 pp and 0.68–36.16 pp for $mF1$-score and OA respectively on H3D. The performance gain between $\mathcal{FS}_f$ and $\mathcal{FS}_h$ is larger on V3D than on H3D. The smaller performance gain on H3D is another indicator that the H3D PC features better geometry. Adding echo characteristics to $\mathcal{FS}_h$ (i.e., $\mathcal{FS}_h \mapsto \mathcal{FS}_i$) does not have a significant effect for both data sets. $\mathcal{FS}_i$ achieves only marginally better performance metrics than $\mathcal{FS}_h$ ($\leq +0.35$ pp for both global metrics). $\mathcal{FS}_i$ achieves a median prediction probability of correct predictions of 82.85 % on V3D and 83.14 % on H3D outperforming other "PC-only" configurations significantly (neglecting $\mathcal{FS}_h$). For comparison, $\mathcal{FS}_g$ achieves median prediction probabilities for correct predictions of 48.98 % and 53.70 % for V3D and H3D respectively.

The relevance plots of "PC-only" configurations show that contextual features remain constant or become slightly more important with increasing radii (see Figures 7.1 and 7.2 for V3D and H3D). For all radii, *verticality* $V_{\mathbf{n}}$ and *inclination* Θ exhibit high importance values for geometric contextual features. These features play a similar role like *face normal* on the mesh and can be seen as its pendants on the PC. For both data sets, the most relevant feature is δh.

For the H3D PC, we also derive contextual geometric features of small-scale radii. The extended feature vector configurations $\mathcal{FS}_{e'}$ - $\mathcal{FS}_{i'}$ perform better than their unprimed counterparts (that incorporate only contextual features based on radii ≥ 1 m) for almost all classes. The superiority of primed versions over default feature vector compositions condenses at 3–5 pp better performance metrics on the global scale. On average, the primed configurations outperform the respective configurations that do not incorporate small-scale radii by 3.96 pp and 3.76 pp for $mF1$-score and OA respectively.

Multi-Modal Feature Vector Compositions. The full feature set $\mathcal{FS}_j$ fuses features computed on the LiDAR cloud and the mesh. $\mathcal{FS}_j$ achieves a $mF1$-score of 65.88 %/70.09 % and an OA of 84.55 %/84.84 % for V3D/H3D. It outperforms the remaining "mesh-only" and "PC-only" feature vector compositions.

The feature relevance plots of $\mathcal{FS}_j$ on both data sets show that the majority of per-face features are negligible. The *face normal*, mainly the third component n_z, the *face color*, and the relative height δh are an exception. The most relevant feature is δh. Contextual mesh features show a slightly increasing importance with increasing radius. For V3D, contextual PC features are similarly important like contextual mesh features. For H3D, many of them are less important than contextual mesh features. This suggests that mesh features gain in relative importance with increased geometric quality of the reconstructed mesh. For both data sets, important PC features are *verticality* $V_{\mathbf{n}}$ and *inclination* Θ as for "PC-only" mode.

$\mathcal{FS}_{j'}$ outperforms $\mathcal{FS}_j$ by 2.91 pp and 0.70 pp for $mF1$-score and OA respectively. $\mathcal{FS}_{j'}$ achieves best $F1$-scores for almost all classes – when compared against "mesh-only" and "PC-only". The median prediction probability for correct predictions increases by 1.18 pp compared to its counterpart that uses only large-scale radii for contextual feature computation. The median prediction probability of correct predictions is 75.50 % for V3D and 88.81 % for H3D. The multi-modal feature vectors $\mathcal{FS}_j$ and $\mathcal{FS}_{j'}$ achieve both the highest performance metrics and prediction probabilities. Hence, the multi-modal integration significantly improves the semantic mesh segmentation.

The feature importance analysis of $\mathcal{FS}_j$ motivated to analyze a sparse version of $\mathcal{FS}_j$ that combines "PC-only" features with texture from the mesh (i.e., $\mathcal{FS}_k$). Table 6.8 reveals that the multi-modal feature vectors achieve best $F1$-scores for almost all classes. For V3D, $\mathcal{FS}_k$ performs significantly better in terms of $mF1$ than $\mathcal{FS}_j$ by 1.34 pp. For H3D, $\mathcal{FS}_j$ achieves better $mF1$-score (+1.61 pp) as $F1$-scores of several classes differ. In particular, many small classes perform significantly better for the full feature set. The OA is on par for both feature sets. Likewise, the primed equivalents achieve similar results.

The superiority of multi-modal features demonstrates the utility of entity linking and combining information from both modalities. The feature importance analysis and the partial superiority of $\mathcal{FS}_k$ over $\mathcal{FS}_j$ indicate that best classifier results are achieved when strengths of both modalities are combined: geometry from PC and texture from mesh.

Grouped Comparison of Feature Vector Compositions. The analysis shows that the most distinctive feature is *relative height* δh in all three $\mathcal{FS}$ groups: "mesh-only", "PC-only", and multi-modal. Adding δh to modality-intrinsic geometric features improves the performance significantly for "mesh-only" and "PC-only" configurations (see Table 6.9).

Table 6.9: Performance gain expressed by $\Delta mF1$-scores and ΔOA values between $\mathcal{FS}_a$ & $\mathcal{FS}_b$ ("mesh-only") and $\mathcal{FS}_e$ & $\mathcal{FS}_f$ ("PC-only") achieved by *relative height* δh. $\mathcal{FS}_b$ and $\mathcal{FS}_f$ enhance the feature sets of $\mathcal{FS}_a$ and $\mathcal{FS}_e$ respectively by δh.

	V3D		H3D	
	$\Delta mF1$ [pp]	ΔOA [pp]	$\Delta mF1$ [pp]	ΔOA [pp]
"mesh-only" ($\mathcal{FS}_b - \mathcal{FS}_a$)	+12.20	+14.48	+7.78	+8.77
"PC-only" ($\mathcal{FS}_f - \mathcal{FS}_e$)	+ 4.83	+ 6.26	+4.96	+3.65

The analysis of δh-induced performance gains across $\mathcal{FS}$ groups and data sets shows that the impact of δh is significantly larger a) in "mesh-only" than in "PC-only" mode (factor 2.3–2.5 for V3D and factor 1.6–2.4 for H3D) and b) on V3D than on H3D (up to factor 1.7). The first finding hints at the superiority of geometric features derived on the PC. Geometric PC features seem to be more expressive than geometric mesh features regardless of the underlying data set. The second finding indicates that geometric properties of the H3D data outperform geometry of the older V3D data set regardless of the considered modality (as already deduced in previous paragraphs). We are aware that the second deduction has to be taken with a grain of salt as data distributions differ between the data sets. A bullet-proof deduction requires to compare the performance on data acquired with different sensors from the same urban area.

The geometric "PC-only" configurations $\mathcal{FS}_e$ and $\mathcal{FS}_f$ significantly outperform their geometric "mesh-only" equivalents $\mathcal{FS}_a$ and $\mathcal{FS}_b$. The comparison of radiometric feature sets $\mathcal{FS}_c$ and $\mathcal{FS}_g$ shows the opposite behaviour: $\mathcal{FS}_c$ outperforms $\mathcal{FS}_g$ significantly for both data sets. We are aware of the fact that these inter-

modal comparisons are not entirely fair due to differing feature counts. The plentitude of geometric PC features uses 68 more features than the geometric feature sets of "mesh-only" compositions. Likewise, $\mathcal{FS}_c$ entails several textural features, whereas $\mathcal{FS}_g$ encompasses merely a handful of reflectance features. However, the features are derived straightforwardly on each modality and hence, implicitly encode and reflect the strengths of both modalities. The feature design is steered inevitably by the modalities' properties and maps the balance of power between the two modalities. The comparisons show that geometric PC features contribute more to the classifier performance than geometric mesh features. In contrast, radiometric mesh features are more expressive than radiometric PC features. This means radiometry in form of texture is superior to radiometry expressed by reflectance values. The findings validate our initial assumption: PCs are characterized by high-quality geometry, whereas meshes provide high-quality textural information. Rephrased, the mesh is superior in texture, while the PC is superior in geometry. The finding is in accordance with the results of the previous paragraph. Comparing $\mathcal{FS}_h$ with $\mathcal{FS}_d$, we see that $\mathcal{FS}_h$ outperforms $\mathcal{FS}_d$ in terms of $mF1$-score for both data sets. The same holds for $\mathcal{FS}_i$. The full "PC-only" feature set is thus better for semantic mesh segmentation concerning the whole class pallette. The result documents the utility of the proposed multi-modal entity linking and the subsequent information transfer for semantic mesh segmentation. However, $\mathcal{FS}_d$ outperforms $\mathcal{FS}_h$ and $\mathcal{FS}_i$ regarding OA on both data sets as it achieves good recall values on large classes. On V3D, $\mathcal{FS}_d$ achieves worse per-class recall values than $\mathcal{FS}_h$ for almost all classes. However, $\mathcal{FS}_d$ outperforms $\mathcal{FS}_h$ by 12.82 pp in terms of recall for class *Roof* which covers approximately 33 % of the surface area in the test set. Similarly, $\mathcal{FS}_d$ achieves significantly higher recall values for the two largest classes *Low Vegetation* and *Impervious Surface* on H3D.

So far, we compared the corresponding equivalents in "mesh-only" and "PC-only" mode. In the following, we arbitrarily compare feature sets across both modes. Neglecting $\mathcal{FS}_g$, "mesh-only" configurations perform worse than "PC-only" configurations. The best "mesh-only" configuration (i.e., $\mathcal{FS}_d$) is an exception that can compete with few "PC only" compositions in terms of $mF1$-score. $\mathcal{FS}_d$ performs better than configurations that use only PC-intrinsic features of one type (i.e., $\mathcal{FS}_e$ and $\mathcal{FS}_g$) on both data sets. Nonetheless, $\mathcal{FS}_d$ is significantly outperformed by its PC-equivalent $\mathcal{FS}_h$ in terms of $mF1$-score ($+\sim$5 pp). Regarding OA, $\mathcal{FS}_d$ outperforms all "PC-only" configurations due to good recall values on large classes as previously discussed. $\mathcal{FS}_d$ achieves even on par OA with primed "PC-only" configurations $\mathcal{FS}_{f'}$, $\mathcal{FS}_{h'}$, $\mathcal{FS}_{i'}$ but loses regarding $mF1$-scores (-2 pp to -8 pp). To emphasize, $\mathcal{FS}_{f'}$, $\mathcal{FS}_{h'}$, and $\mathcal{FS}_{i'}$ perform significantly better than $\mathcal{FS}_d$ although they do not consider any feature derived on the mesh representation itself. Their superiority further demonstrates the utility of the proposed method for semantic mesh segmentation.

Comparing equivalent feature sets across data sets V3D and H3D, we note that each feature vector composition performs better in terms of $mF1$-score on H3D than its counterpart on V3D (except $\mathcal{FS}_g$). The better performance on H3D is expected as H3D has been acquired with better sensors. The superiority reflects the technical developments in the last decade (see Chapter 4). The non-matching class catalogs make a comparison across data sets regarding OA difficult. As good recall values on large classes cause good OA values, a fair comparison regarding OA is only possible for identic class catalogs and class area distributions. Still, H3D shows tendentially better results in terms of OA.

In the following, we discuss in detail the semantic segmentation results for various feature vector compositions on V3D by analyzing the respective confusion matrices (see Figure 6.17). Geometric "PC-only" configurations struggle the most in separating *Vehicle* from *Shrub*. Both classes feature similar geometry in the PC. The confusion rate is 55.83 % for $\mathcal{FS}_f$. Deploying only reflectance features (i.e., radiometric feature set $\mathcal{FS}_g$) drops the confusion to 13.42 %. However, $\mathcal{FS}_g$ performs poorly at the global scale. The combination of radiometric and geometric features (i.e., $\mathcal{FS}_h$) achieves a confusion rate of 25.22 %.

$\mathcal{FS}_h$ registers the biggest confusion for classes *Fence/Hedge* and *Shrub* with 47.11 % (for actual *Fence/Hedge*). The confusion between these classes is 33.61 % in $\mathcal{FS}_e$ and 49.15 % in $\mathcal{FS}_f$. This shows that δh aggravates their separation as both classes feature similar heights (1.36 m and 1.49 m on average). In contrast, their separation based on reflectance features is only confused by 21.54 %. The geometric configuration $\mathcal{FS}_e$ struggles also considerably in separating *Shrub* from *Tree*. $\mathcal{FS}_e$ mispredicts class *Tree* for 44.43 % of *Shrub* instances. The added *relative height* δh reduces confusion to 27.00 % which reveals that δh is a crucial feature to separate these two classes.

The "mesh-only" equivalents show similar behavior for separating these two classes. Specifically, the biggest confusion is observed for classes *Shrub* and *Tree* in $\mathcal{FS}_a$. The confusion rates are 85.11 % and 45.45 % for $\mathcal{FS}_a$ and $\mathcal{FS}_b$ respectively. We observe that the confusion drop is larger in the "mesh-only" mode than in the "PC-only" mode. Nevertheless, the confusion remains at a higher level for features derived directly from the mesh. This observation further emphasizes the geometric superiority of the PC over the mesh. $\mathcal{FS}_c$ shows a decreased confusion of 22.66 % indicating that textural signature differs significantly for these two classes on V3D. The mesh inspection gives the impression that instances of *Shrub* appear in brighter green than instances of *Tree*. The fused feature set $\mathcal{FS}_d$ confuses 34.51 % of *Shrub* instances with *Tree* instances.

The "mesh-only" configurations confuse geometrically similar classes like *Impervious Surface* and *Low Vegetation*. The confusion rate is 30.53 % for $\mathcal{FS}_a$ and 36.59 % for $\mathcal{FS}_b$ (for actual *Impervious Surface* instances). In this particular case, δh impairs the classifier's confusion as it is a non-decisive feature for the separation of these two classes. Considering the texture information only (i.e., $\mathcal{FS}_c$) drops confusion to 1.74 %. $\mathcal{FS}_d$ confuses 3.51 % of *Impervious Surface* instances with *Low Vegetation* instances.

$\mathcal{FS}_j$ fuses both modalities and superimposes positive and negative effects. The fusion dampens the confusion rates of the discussed class pairs concerning their maximum confusion rates.

In the following, we deep-dive semantic segmentation results for primed feature vector compositions on H3D by analyzing the respective confusion matrices (see Figure 6.18). Geometric "PC-only" configurations struggle the most in separating *Gravel/Soil* and *Low Vegetation*. Both classes feature similar geometric properties. The confusion rate is 70.78 % and 70.45 % for $\mathcal{FS}_{e'}$ and $\mathcal{FS}_{f'}$ respectively (for actual *Gravel/Soil*). Adding δh does not further impair the already high confusion for the geometrically similar ground classes. Deploying only reflectance features (i.e., radiometric feature set $\mathcal{FS}_{g'}$) drops the confusion to 35.15 %. However, this configuration performs poorly in general. The fusion of geometric and radiometric features (i.e., $\mathcal{FS}_{h'}$) confuses 65.71 % of *Gravel/Soil* instances with *Low Vegetation*. The separation of classes *Vertical Surface* from *Facade* shows similar behaviour across "PC-only" configurations.

These two classes show good separation for all "mesh-only" configurations, which also struggle in separating *Gravel/Soil* from *Low Vegetation*. $\mathcal{FS}_a$ confuses 38.92 % of *Gravel/Soil* instances with *Low Vegetation*. Adding δh to the feature set increases the confusion by 16.39 pp. The results suggest that separating these two classes is simpler with geometric features derived from the mesh than from the PC. Most probably, the reconstruction with fewer large faces of class *Impervious Surface* is the reason for the better separability, i.e., the lower confusion rates. The median face of *Impervious Surface* is approximately 8 % bigger than the median face of *Low Vegetation*. Adding δh impairs the separation on "mesh-only" configurations whereas it does not for "PC-only" compositions. We claim that the already high confusion rate in $\mathcal{FS}_{e'}$ prevents an increased confusion rate on $\mathcal{FS}_{f'}$ although both classes have similar δh values. The median δh of *Low Vegetation* and *Impervious Surface* differ by less than 7 mm. Considering the pure texture information (i.e., $\mathcal{FS}_c$) drops confusion to 35.44 %. The confusion rate of the multi-type feature set $\mathcal{FS}_d$ is 57.08 %. The separation of class *Chimney* from *Roof* shows a similar trend across "mesh-only" configurations. The highest confusion is registered for $\mathcal{FS}_c$ with 91.85 % showing that textural features do not allow to differentiate between these two classes. On the other hand, deploying only intrinsic geometric features performs comparatively well with a confusion of 27.11 % only.

$\mathcal{FS}_{j'}$ fuses "mesh-only" and "PC-only" features encompassing both radiometric and geometric features. As a consequence, $\mathcal{FS}_{j'}$ fuses the strengths and weaknesses of both modalities and feature types. The fused feature vector mimics the previously discussed confusions. However, the superposition dampens the confusion rates regarding their maximum confusion rates: 55.41 % for separating *Soil/Gravel* from *Impervious Surface*, 22.78 % for separating *Vertical Surface* from *Facade*, and 32.69 % for separting *Chimney* from *Roof*.

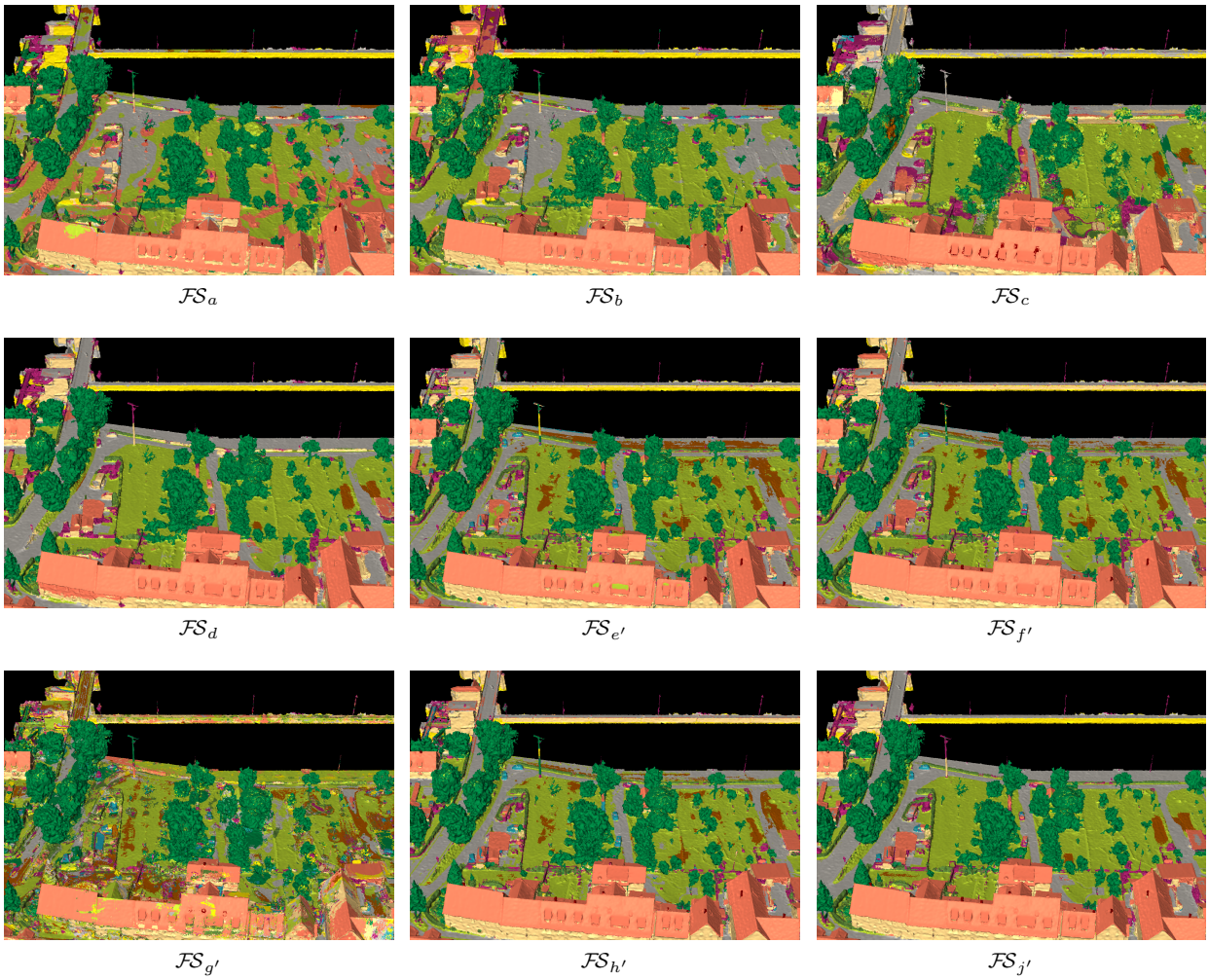


Figure 6.19: Per-face predictions of *direct* semantic segmentation achieved with RF models deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1). The snapshots show a close-up of the lock area of the H3D mesh. Figure 7.4 depicts predictions on the entire test set.

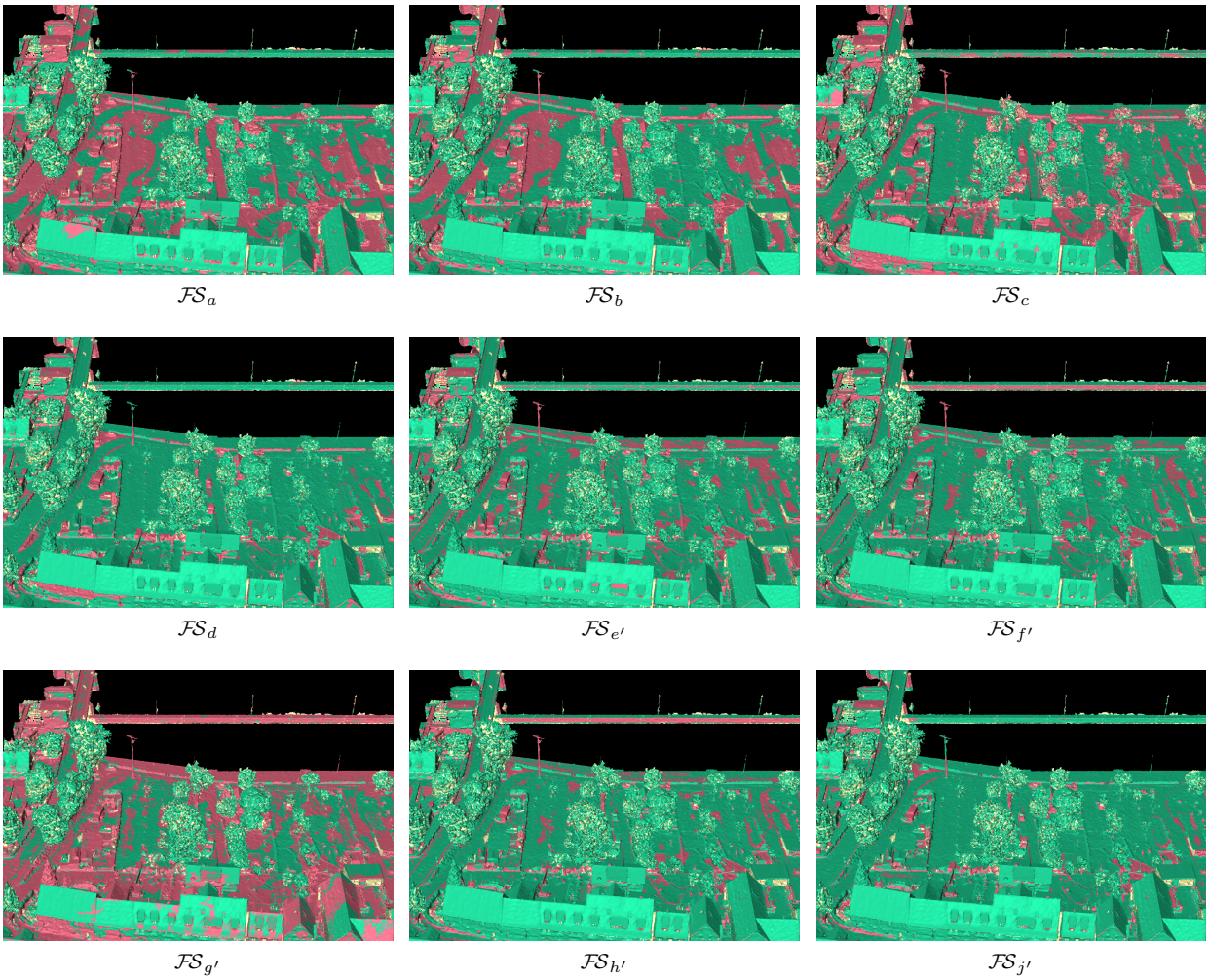


Figure 6.20: Difference plots of the lock area of the H3D mesh for the predictions as shown in Figure 6.19. The predictions of the *direct* semantic segmentation are made with RF models deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1). Correct predictions are shown in *green*; false predictions in *red*. Faces with unknown ground truth are marked in *yellow*. Figure 7.5 depicts the difference plots for the entire test set.

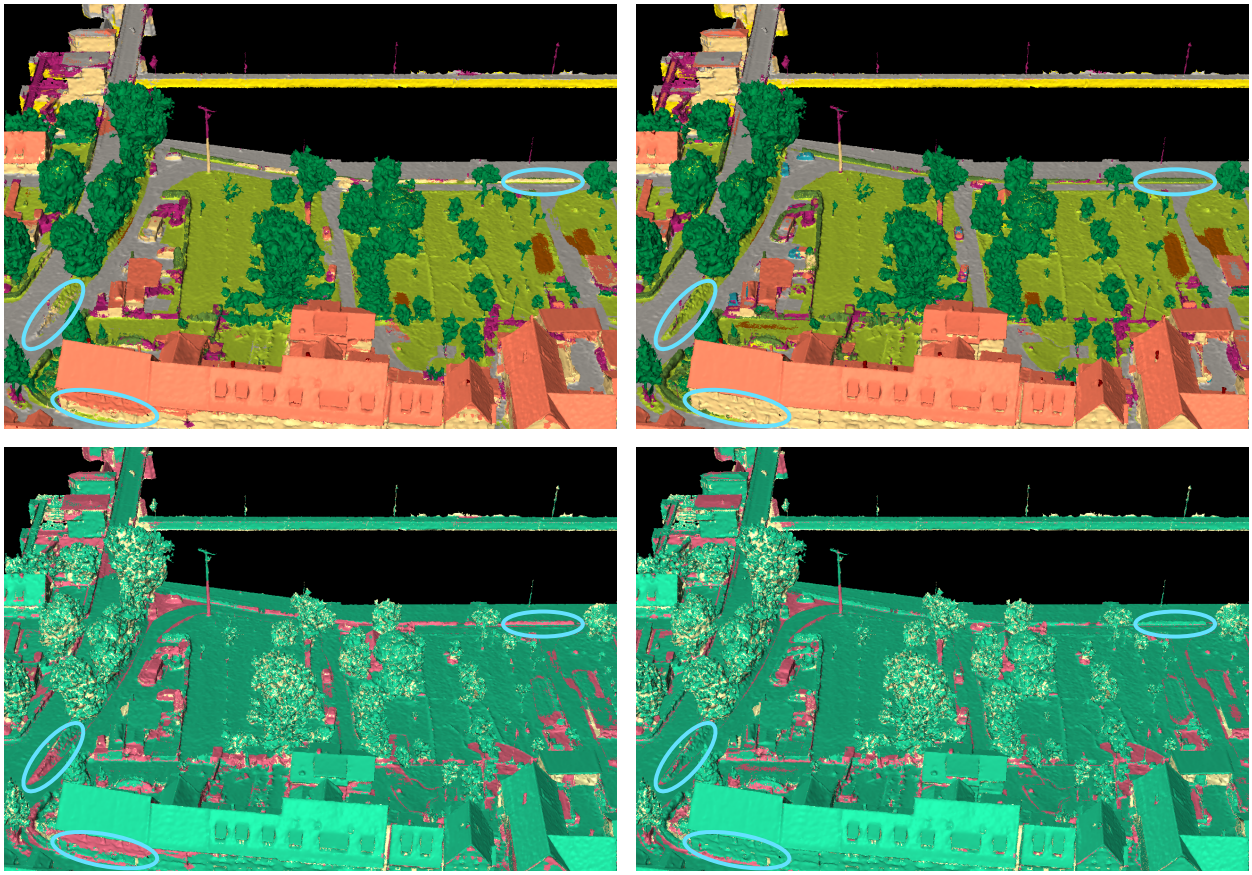


Figure 6.21: Figures 6.19 and 6.20 show the results of the *direct* semantic segmentation achieved with RF models deploying various feature vector compositions (see Table 6.1) for H3D. Here, the results of feature sets $\mathcal{FS}_d$ (*left*, mesh features only) and $\mathcal{FS}_{j'}$ (*right*, multi-modal features) are highlighted in a side-by-side comparison. The *top* shows the per-face predictions, whereas the *bottom* shows the respective red-green plots indicating correct (*green*) and false (*red*) predictions. Faces with unknown ground truth are marked in *yellow*. The most obvious differences between the predictions are circled in *sky-blue*.

Summary. The excessive ablation studies on data sets V3D and H3D along with their detailed analyses reveal the impact and synergy of radiometric and geometric features derived from the 3D modalities mesh and PC for semantic mesh segmentation.

The feature relevance plots show that multi-scale contextual features are more important than per-face features for both radiometric and geometric features (see Figures 7.1 and 7.2 for V3D and H3D respectively). The most important mesh-intrinsic features are *face normal*, more precisely its third component n_z, and the *face color*, underlining the strengths and key properties of the meshed representation. *Verticality* $V_{\mathbf{n}}$ and *inclination* Θ showed to be important features derived from the PC. These features can be seen as the PC-pendants of the *face normal* on the mesh. *Relative height* δh is the most crucial feature that steers the classifier's performance regardless of whether it has been derived on the mesh or the PC. For instance, mesh-intrinsic features are up to 3 times less important than δh. The dominance of δh explains why confusion rates of feature vectors that enhance geometric features with radiometric features behave similarly to geometry-only feature vectors. Categorically, the incorporation of δh improves the global classifiers' performance by decreasing the confusion between the majority of class pairs, e.g., *Shrub* vs. *Tree*. However, it increases the

confusion for few class pairs where δh is not a distinctive feature like for separating *Hedge/Fence* from *Shrub* on V3D or *Impervious Surface* from *Low Vegetation* on H3D (see Figures 6.17 and 6.18).

We note that the integration of δh causes a huger performance gain for "mesh-only" configurations than for "PC-only" compositions. We conclude that geometric PC features are more expressive than geometric mesh features regardless of the underlying data set. The comparison of equivalent feature sets across the two significantly different data sets reveals the superiority of the H3D mesh over the V3D mesh. The geometric deficit of the V3D mesh increases the relative importance of radiometric features for the semantic segmentation of meshes. The influence of radiometry seems to increase with decreasing geometric quality.

The discussed examples show that radiometry and geometry are complementary feature types. Geometric features are util for the separation of radiometrically similar classes. On the other hand, radiometric features help to separate geometrically similar classes such as flat greened areas and impervious surfaces. Therefore, feature vectors that combine both feature types perform better at the global scale than configurations that incorporate only either geometry or radiometry for "mesh-only" and "PC-only" mode. The performance metrics of $\mathcal{FS}_c$ and $\mathcal{FS}_g$ show that texture information from the mesh is superior to the natively available radiometric information on the PC (i.e., reflectance) which has been transferred to the mesh. In contrast, the comparison of geometric "PC-only" configurations $\mathcal{FS}_e$ and $\mathcal{FS}_f$ with their "mesh-only" equivalents $\mathcal{FS}_a$ and $\mathcal{FS}_b$ shows that geometric information derived from the PC outperforms geometric information derived on the mesh. Taken together, the two modalities are complementary representations: while PC features high-quality geometry, the mesh features high-quality texture.

For these reasons, the combination of multi-scale, multi-type, and multi-modal features outperforms other feature vector compositions in terms of $mF1$-scores, OA values, and prediction confidence. In particular, the superposition of complementary multi-modal features allows compensating for deficiencies of a single modality (i.e., "mesh-only" and "PC-only" compositions). $\mathcal{FS}_j$ integrates both feature types and both modalities for semantic segmentation. Compared to best "mesh-only" configuration (i.e., $\mathcal{FS}_d$), the multi-modal semantic segmentation achieves +7.37 pp and +2.38 pp for $mF1$ and OA on V3D. Similarly, $\mathcal{FS}_j$ outperforms $\mathcal{FS}_d$ by +9.23 pp and +4.33 pp for $mF1$ and OA on H3D. The sparse version $\mathcal{FS}_k$ integrates textural features from the mesh with geometric features from the dense PC. The prediction results of $\mathcal{FS}_k$ are better than these from $\mathcal{FS}_j$ on V3D. For H3D, $\mathcal{FS}_{j'}$ and $\mathcal{FS}_{k'}$ perform similarly. The improved performance metrics demonstrate the utility of the proposed multi-modal entity linking and the subsequent information transfer for semantic mesh segmentation.

The comparison of primed and unprimed configurations on H3D indicates that small-scale PC features improve semantic segmentation of meshes by 3–5 pp.

Analysis of Different Inter-Modal Propagation Methods on Semantic Segmentation. We concluded in the previous paragraph that multi-type feature sets incorporating geometric and radiometric attributes outperform feature sets that use only a single type for "mesh-only" and "PC-only" configurations. The modal counterparts $\mathcal{FS}_d$ and $\mathcal{FS}_h$ outperform the remaining "mesh-only" and "PC-only" configurations respectively (neglecting $\mathcal{FS}_i$ that enhances $\mathcal{FS}_h$ with echo characteristics). The multi-modal $\mathcal{FS}_j$ fuses features derived from both modalities and is the best performing feature set for V3D and H3D (neglecting $\mathcal{FS}_k$ and primed configurations for H3D). Therefore, we utilize feature sets $\mathcal{FS}_d$, $\mathcal{FS}_h$, and $\mathcal{FS}_j$ as representatives of the three $\mathcal{FS}$ groups for experiments in this paragraph.

The goal of this paragraph is to analyze the impact of the deployed information propagation methods on automated semantic mesh segmentation. The propagation method defines how information is propagated across different modalities. In Section 6.1.2, we already documented differences of the a) simple nearest-neighbor (NN) interpolation between faces & points and b) the proposed information transfer via PCMA (see Figure 6.4). The PCMA-induced transfer consists of the explicit entity linking between mesh and PC and the subsequent information transfer. The information transfer is accomplished as aggregation due to the one-to-many relationship between faces and points. In this paragraph, we analyze how both propagation methods affect the trained ML classifiers and thus semantic segmentation on data sets V3D and H3D. Precisely, we compare the performance of different RF models deploying feature sets $\mathcal{FS}_d$, $\mathcal{FS}_h$, and $\mathcal{FS}_j$ where labels and PC features have been propagated from the PC to the mesh with either NN interpolation

or PCMA-driven information transfer. To this end, the propagation methods lead to different realizations of the label vector $\mathbf{y}$ and the feature matrix $\mathbf{X}$ (for each $\mathcal{FS}$). This, in turn, results in a bunch of train sets which are denoted as tuples of the different realizations of $\mathbf{X}$ and $\mathbf{y}$ for each $\mathcal{FS}$: $(\mathbf{X}, \mathbf{y})_{\mathcal{FS}}$.

The feature matrix $\mathbf{X}$ depends on both the propagation method and the deployed feature set. The "mesh-only" feature set $\mathcal{FS}_d$ uses merely features derived from the mesh. Hence, its composition is independent of the chosen propagation method. The related RF variants differ only by the used annotation of the train set. Therefore, the recorded performance difference reflects the impact of the differently derived GT on the mesh. Feature sets $\mathcal{FS}_h$ and $\mathcal{FS}_j$ deploy features derived from the PC. Thus, their composition depends on the propagation method as the propagated labels.

Training Setup. The propagation methods NN and PCMA generate two differently sized train sets $(\mathbf{X}^{\mathrm{NN}}_{\mathrm{train}}, \mathbf{y}^{\mathrm{NN}}_{\mathrm{train}})$ and $(\mathbf{X}^{\mathrm{PCMA}}_{\mathrm{train}}, \mathbf{y}^{\mathrm{PCMA}}_{\mathrm{train}})$ for each feature vector composition. Whereas the proposed label transfer via PCMA propagates labels only to faces that have been linked to the PC, the interpolation method annotates all surface elements. $(\mathbf{X}^{\mathrm{NN}}_{\mathrm{train}}, \mathbf{y}^{\mathrm{NN}}_{\mathrm{train}})$ consists of 1.730 M/11.814 M faces for V3D/H3D. In contrast, $(\mathbf{X}^{\mathrm{PCMA}}_{\mathrm{train}}, \mathbf{y}^{\mathrm{PCMA}}_{\mathrm{train}})$ contains 247 k/6.6 M faces for V3D/H3D. The entire interpolated train sets exceed the transferred train sets by factors 7.0 and 1.8 in the face count for V3D and H3D respectively. These unequal train sets mimic the real circumstances and hence allow the analysis of the real effective impact of the propagation methods. We mark their extension as E_0. Please note we did not apply any shape filtering to the meshes before the interpolation to emphasize the discrepancy between the propagation methods.

To enable a fair training setup and comparison, we generate equally sized train sets by selecting faces, where both propagation methods provide a valid label, i.e., where PCMA links faces to the PC (i.e., transferred labels are unequal to -1). The PCMA-steered filtering is indicated with a bar over the variables. Thus, the PCMA-filtered train sets are denoted as $(\overline{\mathbf{X}}^{\mathrm{NN}}_{\mathrm{train}}, \overline{\mathbf{y}}^{\mathrm{NN}}_{\mathrm{train}})$ and $(\overline{\mathbf{X}}^{\mathrm{PCMA}}_{\mathrm{train}}, \overline{\mathbf{y}}^{\mathrm{PCMA}}_{\mathrm{train}})$. Their respective extension are marked as $E_{\overline{0}}$. Please note that it holds: $(\overline{\mathbf{X}}^{\mathrm{PCMA}}_{\mathrm{train}}, \overline{\mathbf{y}}^{\mathrm{PCMA}}_{\mathrm{train}}) = (\mathbf{X}^{\mathrm{PCMA}}_{\mathrm{train}}, \mathbf{y}^{\mathrm{PCMA}}_{\mathrm{train}})$. The filter condition excludes 86 %/44 % of faces for V3D/H3D.

For a more detailed analysis, we split the equally sized train sets into equally sized subfractions depending on the relationship of the propagated labels. Subfraction $E_{\overline{0},A}$ considers only faces where the propagated labels differ. Its complement $E_{\overline{0},B}$ deploys only faces were the propagated labels are equal. $E_{\overline{0},B}$ entails 96.35 %/97.66 % of faces belonging to $E_{\overline{0}}$ for V3D/H3D. $E_{\overline{0},A}$ encloses the remaining 3.65 %/2.34 % for V3D/H3D. RF models deploying $E_{\overline{0},A}$ train on faces f where transferred labels exist (i.e., $\ell_f^{\mathrm{PCMA}} \neq -1$) & the interpolated labels ℓ_f^{NN} and transferred labels ℓ_f^{PCMA} differ, i.e., $\overline{\mathbf{y}}^{\mathrm{NN}|\ell_f^{\mathrm{PCMA}} \neq \ell_f^{\mathrm{NN}}}_{\mathrm{train}} \neq \overline{\mathbf{y}}^{\mathrm{PCMA}|\ell_f^{\mathrm{PCMA}} \neq \ell_f^{\mathrm{NN}}}_{\mathrm{train}}$. The resulting tuples are $(\overline{\mathbf{X}}^{\mathrm{NN}|\ell_f^{\mathrm{PCMA}} \neq \ell_f^{\mathrm{NN}}}_{\mathrm{train}}, \overline{\mathbf{y}}^{\mathrm{NN}|\ell_f^{\mathrm{PCMA}} \neq \ell_f^{\mathrm{NN}}}_{\mathrm{train}})$ and $(\overline{\mathbf{X}}^{\mathrm{PCMA}|\ell_f^{\mathrm{PCMA}} \neq \ell_f^{\mathrm{NN}}}_{\mathrm{train}}, \overline{\mathbf{y}}^{\mathrm{PCMA}|\ell_f^{\mathrm{PCMA}} \neq \ell_f^{\mathrm{NN}}}_{\mathrm{train}})$. $E_{\overline{0},B}$ contains faces f where transferred labels exist (i.e., $\ell_f^{\mathrm{PCMA}} \neq -1$) & interpolated labels ℓ_f^{NN} and transferred labels ℓ_f^{PCMA} match, i.e., $\overline{\mathbf{y}}^{\mathrm{NN}|\ell_f^{\mathrm{PCMA}} = \ell_f^{\mathrm{NN}}}_{\mathrm{train}} = \overline{\mathbf{y}}^{\mathrm{PCMA}|\ell_f^{\mathrm{PCMA}} = \ell_f^{\mathrm{NN}}}_{\mathrm{train}}$. The resulting tuples are $(\overline{\mathbf{X}}^{\mathrm{NN}|\ell_f^{\mathrm{PCMA}} = \ell_f^{\mathrm{NN}}}_{\mathrm{train}}, \overline{\mathbf{y}}^{\mathrm{NN}|\ell_f^{\mathrm{PCMA}} = \ell_f^{\mathrm{NN}}}_{\mathrm{train}})$ and $(\overline{\mathbf{X}}^{\mathrm{PCMA}|\ell_f^{\mathrm{PCMA}} = \ell_f^{\mathrm{NN}}}_{\mathrm{train}}, \overline{\mathbf{y}}^{\mathrm{PCMA}|\ell_f^{\mathrm{PCMA}} = \ell_f^{\mathrm{NN}}}_{\mathrm{train}})$.

The previously described train sets use either NN interpolation or PCMA for constructing both the feature matrices and the label vectors. By these means, we train a RF model on the interpolated GT deploying interpolated PC features (if part of the $\mathcal{FS}$) for each considered feature vector composition. Concurrently, we train a RF model on the transferred GT deploying transferred PC features (if part of the $\mathcal{FS}$) for each considered feature set. That setup enables the comparison of the propagation methods in general. However, it cannot analyze the impact of the propagated features or propagated labels individually (except for $\mathcal{FS}_d$). Therefore, we train RF models on a combination of NN interpolated features and PCMA-transferred labels. We opted for PCMA-transferred labels as NN are likely to introduce label errors (see Figure 6.4). The "intertwined" setup allows analyzing the impact of the feature quality by comparing its performance to the respective fully PCMA-transferred counterpart. The comparison of the fully NN-interpolated setups and the "intertwined" configurations highlights the impact of the label quality.

For all train sets of all feature sets, features are standardized according to the first and second moment of the respective feature matrices. The achieved performance metrics of the versatile train sets and extensions

deploying the two propagation methods are listed in Table 6.10 and Table 6.11. The details of the evaluation strategy are given in the next paragraph.

Please note the extension E refers to the size of the train set and the relationship of propagated labels, too. The equally sized train sets of extensions $E_{\overline{0}}$ or its subfractions $E_{\overline{0},A}$ and $E_{\overline{0},B}$ are limited to faces with a valid PCMA-transferred label regardless of whether NN or PCMA has been used as propagation method. Therefore the NN-interpolated information implicitly benefits from the PCMA as the filtering excludes faces that are likely to introduce interpolation artifacts (i.e., label noise and feature noise) as seen in Figure 6.4. However, the PCMA-steered filtering does not avoid interpolation artifacts in general.

Evaluation Setup. To enable a fair analysis independent from tedious area-wide manual annotation, we evaluate the performance of the various RF models on faces of the test set where the propagated labels of both propagation methods match, i.e., $\overline{\mathbf{y}}_{\text{test}} = \overline{\mathbf{y}}_{\text{test}}^{\text{NN}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}} = \overline{\mathbf{y}}_{\text{test}}^{\text{PCMA}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}$. The condition implicitly restricts the test set to faces where PCMA-transferred labels exist (indicated by the bar over the variables). We argue that the labels are likely to be correct when the two propagation methods agree. We verified this assumption by comparing the subset of matching labels to the manually labeled subset of H3D. The constrained test set matches the manual GT by 95.32 %/97.95 % ($mF1$-score/OA) and outperforms a pure PCMA-filtered test set (see Subfigure a) in Figure 6.11). This proxy analysis enables a fair impact comparison of the propagation methods on any data set as it does not favor any propagation method and provides an equally sized test set deploying either interpolated features $\overline{\mathbf{X}}_{\text{test}}^{\text{NN}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}$ or transferred features $\overline{\mathbf{X}}_{\text{test}}^{\text{PCMA}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}$. We perform the proxy analysis on data sets V3D and H3D. As for the training setups, the proxy analysis is likely to positively bias the measured performance of configurations that deploy NN interpolation as PCMA-steered filtering is conducted implicitly reducing NN interpolation artifacts.

Independently acquired GT can solve this structural weakness. Therefore, we also evaluate the trained RF models on the manually annotated part of the H3D test set (see Figure 4.5). However, evaluating on faces without a PCMA-detected link to the PC means to incorporate faces with zeroed PC features in case of information transfer with PCMA. Such faces are out of the area of validity for PCMA-based propagation.

To enable a fair comparison for feature sets incorporating transferred PC features, we limit the evaluation on faces where PCMA detects a link between faces and the PC. For this reason, the performance of configurations that deploy NN interpolation benefit from the implicit PCMA-steered filtering in case of evaluating against manual GT, too. We denote the two fashions of the PCMA-filtered manually labeled test set as $(\overline{\mathbf{X}}_{\text{test}}^{\text{PCMA}}, \overline{\mathbf{y}}_{\text{test}}^{\text{manual}})$ and $(\overline{\mathbf{X}}_{\text{test}}^{\text{NN}}, \overline{\mathbf{y}}_{\text{test}}^{\text{manual}})$. For the sake of completeness, we compare the performance metrics of each RF model as achieved when evaluating on the entire manually annotated test set $(\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}}^{\text{manual}})$ and the PCMA-filtered manually labeled test set $(\overline{\mathbf{X}}_{\text{test}}, \overline{\mathbf{y}}_{\text{test}}^{\text{manual}})$.

Table 6.10 lists the results of the proxy analysis for V3D and H3D. The achieved global surface-aware performance metrics of the various feature vectors are given for the versatile train set compositions. The metrics are evaluated on the faces of the test sets where interpolated and transferred labels match. Table 6.11 lists the analogous results evaluated on the manually annotated subset of H3D where PCMA-transferred labels exist. Both tables report the $mF1$-score and OA of the different RF models trained on the equally sized train sets (with extensions $E_{\overline{0}}, E_{\overline{0},A}, E_{\overline{0},B}$) and the total available train sets (with extension E_0) deploying either interpolation or PCMA-based transfer.

Table 6.10: Global surface-aware performance metrics of RF models trained on versatile train set compositions for feature vectors $\mathcal{FS}_d$, $\mathcal{FS}_h$, and $\mathcal{FS}_j$ as achieved on the subset of test set where propagated labels match, i.e., with labels $\overline{\mathbf{y}}_{\text{test}} = \overline{\mathbf{y}}_{\text{test}}^{\text{NN}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}} = \overline{\mathbf{y}}_{\text{test}}^{\text{PCMA}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}$ (proxy analysis). The extensions E of the train sets are given along with the number of faces n_f (for data sets V3D/H3D). The bar indicates PCMA-induced filtering has been applied (i.e., only faces with valid PCMA-transferred labels). Index A refers to faces where PCMA-transferred and interpolated labels do not match. Index B refers to faces where PCMA-transferred and interpolated labels match. The train sets are denoted as tuples of feature matrix $\mathbf{X}$ and label vector $\mathbf{y}$. Please note that feature matrices of $\mathcal{FS}_d$ are independent of the propagation method as only mesh features are considered. Besides, for $E_{\overline{0},B}$ it holds: $\overline{\mathbf{y}}_{\text{train}}^{\text{NN}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}} = \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}$.
Above the line: Feature matrix and label vector have been generated with the same propagation method ("fully NN-interpolated" and "fully PCMA-transferred" setup respectively).
Below the line: Combination of different propagation methods, i.e., interpolated features and PCMA-transferred labels ("intertwined" setup).

Evaluation on Matching GT (Proxy Analysis)

Train Set			$\mathcal{FS}_d$				$\mathcal{FS}_h$				$\mathcal{FS}_j$			
			V3D		H3D		V3D		H3D		V3D		H3D	
E	Tuple	n_f	$mF1$ [%]	OA [%]	$mF1$ [%]	OA [%]	$mF1$ [%]	OA [%]	$mF1$ [%]	OA [%]	$mF1$ [%]	OA [%]	$mF1$ [%]	OA [%]
$E_{\overline{0},A}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}})$	9 k/155 k	26.53	37.90	30.06	41.78	40.26	49.00	30.39	42.50	32.39	42.78	35.00	50.35
$E_{\overline{0},A}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}})$	9 k/155 k	32.33	51.65	31.34	60.73	43.55	53.39	40.45	60.46	40.42	49.50	40.33	69.72
$E_{\overline{0},B}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}})$	237.5 k/6.466 M	59.68	83.00	61.47	81.34	64.03	80.49	64.97	77.04	66.09	85.23	70.84	85.35
$E_{\overline{0},B}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}})$	237.5 k/6.466 M	59.68	83.00	61.47	81.34	63.97	80.41	65.77	77.45	66.15	85.38	70.97	85.42
$E_{\overline{0}}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}}, \overline{\mathbf{y}}_{\text{train}}^{\text{NN}})$	246.5 k/6.621 M	58.68	82.32	61.39	80.73	64.14	80.38	63.79	76.46	66.28	85.35	69.85	84.59
$E_{\overline{0}/0}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{PCMA}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}}) = (\mathbf{X}_{\text{train}}^{\text{PCMA}}, \mathbf{y}_{\text{train}}^{\text{PCMA}})$	246.5 k/6.621 M	59.71	83.12	60.71	80.96	64.22	80.45	65.97	77.73	66.59	85.53	70.62	85.52
E_0	$(\mathbf{X}_{\text{train}}^{\text{NN}}, \mathbf{y}_{\text{train}}^{\text{NN}})$	1.730 M/11.814 M	50.39	74.31	57.92	78.36	62.66	79.56	63.19	74.51	66.53	85.68	69.32	83.63
$E_{\overline{0},A}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}})$	9 k/155 k	32.33	51.65	31.34	60.73	20.94	35.74	32.14	54.36	35.00	47.57	36.00	66.44
$E_{\overline{0},B}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}})$	237.5 k/6.466 M	59.68	83.00	61.47	81.34	64.03	80.49	64.97	77.04	66.09	85.23	70.84	85.35
$E_{\overline{0}}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}})$	246.5 k/6.621 M	59.71	83.12	60.71	80.96	63.46	80.47	63.95	76.87	66.26	85.39	70.36	85.31

Table 6.11: Global surface-aware performance metrics of RF models trained on versatile train set compositions for feature vectors $\mathcal{FS}_d$, $\mathcal{FS}_h$, and $\mathcal{FS}_j$ as achieved on the subset of the manually annotated test set of H3D where transferred labels exist, i.e., with labels $\overline{\mathbf{y}}_{\text{test}} = \overline{\mathbf{y}}_{\text{test}}^{\text{manual}}$. The extensions E of the train sets are given along with the number of faces n_f. The bar indicates PCMA-induced filtering has been applied (i.e., only faces with valid PCMA-transferred labels). Index A refers to faces where PCMA-transferred and interpolated labels do not match. Index B refers to faces where PCMA-transferred and interpolated labels match. The train sets are denoted as tuples of feature matrix $\mathbf{X}$ and label vector $\mathbf{y}$. Please note that feature matrices of $\mathcal{FS}_d$ are independent of the propagation method as only mesh features are considered. Besides, for $E_{\overline{0},B}$ it holds: $\overline{\mathbf{y}}_{\text{train}}^{\text{NN}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}} = \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}$.
Above the line: Feature matrix and label vector have been generated with the same propagation method ("fully NN-interpolated" and "fully PCMA-transferred" setup respectively).
Below the line: Combination of different propagation methods, i.e., interpolated features and PCMA-transferred labels ("intertwined" setup).

Evaluation on Manually Annotated Subset of H3D (where PCMA detects a connection to the PC)

Train Set			$\mathcal{FS}_d$		$\mathcal{FS}_h$		$\mathcal{FS}_j$	
E	Tuple	n_f	$mF1$ [%]	OA [%]	$mF1$ [%]	OA [%]	$mF1$ [%]	OA [%]
$E_{\overline{0},A}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}})$	155 k	26.27	37.58	26.68	42.08	30.24	48.62
$E_{\overline{0},A}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}})$	155 k	31.13	59.47	40.21	67.99	41.33	72.13
$E_{\overline{0},B}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}})$	6.466 M	51.71	81.51	57.30	80.50	59.39	85.33
$E_{\overline{0},B}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}})$	6.466 M	51.71	81.51	58.22	80.94	59.16	85.43
$E_{\overline{0}}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}}, \overline{\mathbf{y}}_{\text{train}}^{\text{NN}})$	6.621 M	51.76	80.93	56.69	80.09	58.92	84.85
$E_{\overline{0}/0}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{PCMA}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}}) = (\mathbf{X}_{\text{train}}^{\text{PCMA}}, \mathbf{y}_{\text{train}}^{\text{PCMA}})$	6.621 M	51.02	81.09	57.68	80.86	59.20	85.56
E_0	$(\mathbf{X}_{\text{train}}^{\text{NN}}, \mathbf{y}_{\text{train}}^{\text{NN}})$	11.814 M	49.43	78.62	56.41	78.20	59.50	84.50
$E_{\overline{0},A}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}\neq\ell_f^{\text{NN}}})$	155 k	31.13	59.47	32.86	61.29	37.09	67.98
$E_{\overline{0},B}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}\|\ell_f^{\text{PCMA}}=\ell_f^{\text{NN}}})$	6.466 M	51.71	81.51	57.30	80.50	59.39	85.33
$E_{\overline{0}}$	$(\overline{\mathbf{X}}_{\text{train}}^{\text{NN}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}})$	6.621 M	51.02	81.09	57.11	80.35	59.40	85.48

Results – Discussion. The complexity of the training and evaluation setup allows us to analyze the effect of the propagation method on semantic segmentation – both for fair and actual circumstances. We unwrap the main findings from Tables 6.10 and 6.11. As a first step, we compare the performance metrics of the RF models trained on the fully NN-interpolated or fully PCMA-transferred train set for each extension (setups *above the line* in the tables). The comparison of the two respective RF models trained on the equally sized train sets of extensions $E_{\overline{0}}$, $E_{\overline{0},A}$, and $E_{\overline{0},B}$ shows the superiority of the proposed PCMA over a simple NN interpolation for both the proxy analysis and the analysis against manual GT.

$E_{\overline{0},A}$ shows the biggest discrepancies in performance metrics for all considered feature sets and both evaluation scenarios. The maximum discrepancy is explained by the fact that the train sets differ considerably in propagated features and labels as only faces have been used for the training where propagated labels do not match. In other equally-sized train sets, faces with matching labels and hence – most probably – with similar feature values have been considered. Therefore, the comparison demonstrates the impact of different variants of the propagated features and labels most clearly. One has to keep in mind that interpolation artifacts will be marginal as only faces are considered that have a valid PCMA-transferred label. However, the filtering does not avoid interpolation artifacts in general, as highlighted by the differences of the models trained on $E_{\overline{0},A}$. For the proxy analysis, the RF trained on the fully PCMA-transferred data on $E_{\overline{0},A}$ outperforms the NN-interpolated counterpart up to 8.03 pp/13.75 pp for V3D and 10.06 pp/19.37 pp for H3D in $mF1$-score/OA. The analysis against the manual GT shows the largest difference for $\mathcal{FS}_h$ with $\Delta mF1\text{-score} = 13.53$ pp and $\Delta\, OA = 25.91$ pp respectively.

On the other hand, the comparison of achieved performance metrics on $E_{\overline{0},B}$ shows mostly on par performances for both propagation methods and both evaluation methods. The similar performance can be explained by the similarity of propagated features for faces with matching propagated labels in the train set. Only $\mathcal{FS}_h$ records a slight improvement of the PCMA-depending train set over its NN counterpart for data set H3D considering only faces in the train set with matching propagated labels. $\mathcal{FS}_h$ shows the most significant differences in the feature sets as its composition purely depends on PC features and hence, the propagation method. The differences in the feature matrix proof to be effective for H3D only whose train set of extension $E_{\overline{0},B}$ is 27 times larger than on V3D. The PCMA-steered version outperforms the NN-interpolated version by 0.80–0.92 pp/0.41–0.44 pp ($\Delta mF1$-score/$\Delta\, OA$) for the proxy analysis and analysis against the manual GT.

$E_{\overline{0}}$ is the union of $E_{\overline{0},A}$ and $E_{\overline{0},B}$, incorporating matching and differing propagated labels. Therefore, RF models trained on $E_{\overline{0}}$ do not allow that crisp conclusions like models trained on its subfractions which mimic laboratory conditions. The trained RF models show a similar behavior like those trained on $E_{\overline{0},B}$, as $E_{\overline{0},B}$ entails 96.35 %/97.66 % of faces of $E_{\overline{0}}$. For $\mathcal{FS}_h$ on H3D, the RF trained on the fully PCMA-transferred train set outperforms the NN counterpart by $\Delta mF1\text{-score} = 2.18$ pp and $\Delta\, OA = 1.27$ pp in the proxy analysis. The evaluation against the manual GT shows a performance gain of 0.99 pp and 0.77 pp for $mF1$-score and OA respectively.

The performance metrics of the train sets of extensions $E_{\overline{0}}$, $E_{\overline{0},A}$, and $E_{\overline{0},B}$ deploying NN interpolation are positively biased as PCMA-steered filtering is implicitly used – both during training and evaluation. Hence, faces are not considered where the interpolation is likely to introduce interpolation artifacts (i.e., label noise and feature noise, see Figure 6.4). The comparison of performance metrics achieved with RF models trained on E_0 and $E_{\overline{0}}$ reveals the positive impact of the PCMA-steered filtering during training, i.e., the filtering of potential interpolation artifacts. The RF models trained on the PCMA-filtered but NN-interpolated train set $(\overline{\mathbf{X}}^{\mathrm{NN}}_{\mathrm{train}}, \overline{\mathbf{y}}^{\mathrm{NN}}_{\mathrm{train}})$ outperform models trained on the entire interpolated train set $(\mathbf{X}^{\mathrm{NN}}_{\mathrm{train}}, \mathbf{y}^{\mathrm{NN}}_{\mathrm{train}})$ for almost all feature sets on both data sets. For instance, $\mathcal{FS}_d$ shows a performance gain of 8.29 pp/8.01 pp on V3D and 3.47 pp/2.37 pp on H3D for $mF1$-score/OA in the proxy analysis. The performance differences are larger for V3D as E_0 is 7 times larger than $E_{\overline{0}}$, whereas for H3D the ratio is 1.8 : 1. The analysis on the manual GT shows a performance gain of 2.33 pp/2.31 pp ($\Delta mF1$-score/$\Delta\, OA$). For the comparison of E_0 and $E_{\overline{0}}$, the proxy analysis gives a more crisp image as the independent GT. We assume that the label noise and label ambiguity between the two manual annotation processes on the two modalities are the reason for the minor performance gaps. Please note that the additional faces in the entire interpolated train set do not necessarily introduce feature noise and label noise. If surrounding faces carry the correct information, so will

the interpolated ones. Therefore, it is unlikely that the additionally labeled faces carry only false labels and features noise. Such "controlled interpolation" is more common for H3D due to the higher point density of the LiDAR PC, fewer data gaps, and the cohesive structure of the data set. V3D consists of two disjointed parts, which inevitably cause interpolation artifacts. Hence, the discrepancy is more obvious for V3D. We note that feature sets with larger feature counts reduce the performance gap, which hints at the bigger importance of the feature vector composition than the propagation method. The multi-modal $\mathcal{FS}_j$ copes best with the entire NN-interpolated train set.

The comparison of $(\mathbf{X}_{\text{train}}^{\text{NN}}, \mathbf{y}_{\text{train}}^{\text{NN}})$ and $(\overline{\mathbf{X}}_{\text{train}}^{\text{PCMA}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}})$ shows the true effect of the propagation methods considering the two differently sized train sets. We register a similar behavior as for the comparison of the interpolated versions. Tendentially, models trained on $(\overline{\mathbf{X}}_{\text{train}}^{\text{PCMA}}, \overline{\mathbf{y}}_{\text{train}}^{\text{PCMA}})$ perform better than models trained on $(\mathbf{X}_{\text{train}}^{\text{NN}}, \mathbf{y}_{\text{train}}^{\text{NN}})$.

The comparison of results evaluated on the entire manually annotated test set $(\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}}^{\text{manual}})$ and the PCMA-filtered manually labeled test set $(\overline{\mathbf{X}}_{\text{test}}, \overline{\mathbf{y}}_{\text{test}}^{\text{manual}})$ shows better performance metrics on the PCMA-filtered manually labeled test set for the "fully NN-interpolated" setups. The $\Delta mF1\text{-score}/\Delta OA$ is in the range of 1.18–1.99 pp/0.61–2.76 pp deploying $\mathcal{FS}_d$, $\mathcal{FS}_h$, or $\mathcal{FS}_j$. Hence, the performance of RF models deploying NN interpolation as propagation method is positively biased when evaluated on the PCMA-filtered manual GT – regardless of the deployed feature vector composition.

To analyze the impact of the two label versions used during training, we first compare the "intertwined" setups with the respective "fully NN-interpolated" setups for each extension. Secondly, we analyze the two RF models deploying $\mathcal{FS}_d$ for each extension, as its composition purely depends on mesh features and therefore only differs in labels when comparing on the same extension. The comparison with intertwined setups shows the superiority of the PCMA, particularly for $E_{\overline{0},A}$. For instance, $\mathcal{FS}_j$ registers a performance gap of 1.00 pp/16.09 pp ($\Delta mF1\text{-score}/\Delta OA$) on H3D in the proxy analysis. The evaluation on the manual GT gives 6.85 pp/19.36 pp ($\Delta mF1\text{-score}/\Delta OA$).

The analysis of $\mathcal{FS}_d$ emphasizes the previously made conclusions. To give an example, the RF model trained with PCMA-transferred labels outperforms its counterpart using NN-interpolated labels by 5.80 pp and 13.75 pp for $mF1$-score and OA respectively in the proxy analysis (for $E_{\overline{0},A}$ on V3D). Similarly, the analysis with the independent manual GT shows a superiority by 4.86 pp/21.89 pp for $mF1\text{-score}/OA$.

Comparing the performances of "intertwined" setups to "fully PCMA transferred" setups maps the influence of feature quality. The PCMA-transferred features outperform the NN-interpolated features following already performed analyses. The biggest discrepancy is noted for $\mathcal{FS}_h$ on subset of $E_{\overline{0},A}$ as $\mathcal{FS}_h$ contains only propagated features. $\mathcal{FS}_h$ shows a performance gap of 22.61 pp/17.65 pp ($\Delta mF1\text{-score}/\Delta OA$) on V3D and 8.31 pp/6.10 pp ($\Delta mF1\text{-score}/\Delta OA$) on H3D in the proxy analysis. The analysis on the manual GT shows differences of 7.35 pp/6.70 pp ($\Delta mF1\text{-score}/\Delta OA$).

Summary. The bare superiority of PCMA over NN interpolation has already been documented qualitatively and quantitatively in Section 6.1.2 (see Figure 6.4). In this paragraph, we have shown that the discrepancy of PCMA-transferred and NN-interpolated information affects the semantic segmentation measurably. We have empirically demonstrated the positive impact of PCMA-prepared data on semantic segmentation as accomplished with a RF classifier. We verified the utility and effectiveness of PCMA by comparison to NN interpolation for classifier training and semantic segmentation of meshes under real-world conditions on data sets V3D and H3D. The verification is done for a) the proxy analysis that operates on the subset of faces with congruent labels propagated by the two investigated propagation methods and b) the independent analysis leveraging manual annotations of the H3D mesh. The detected performance differences depend fully on the deployed propagation method as propagated features and labels are derived from the same data source and evaluated against identical GT for both evaluation strategies.

For these reasons, it is recommendable to leverage PCMA for the preparation of multi-modal feature vectors and the transfer of (manual) annotations (see *Feature Sharing*, *GT Sharing*, and *Prediction Sharing* in Figure 6.1). The benefit of PCMA is two fold:

I) Utilizing the PCMA-established linking information as filter condition:
PCMA links points and faces and thereby, gives control of the propagated information. The output of PCMA can be used as a filter constraint regardless of whether the information is propagated by a simple NN interpolation or majority vote. The performances of RF models trained on pure NN-interpolated train sets or PCMA-filtered NN-interpolated train sets show that PCMA-filtering improves the train set quality by reducing the interpolation artifacts. In other words: NN-interpolation benefits from the PCMA-based filtering.

II) Utilizing the PCMA-established linking information for the propagation method:
The implementation of PCMA does not only provide the linking information but immediately aggregates multiple data points for each linked face. The aggregation causes more robust feature values and labels respectively than a simple NN interpolation. As a consequence, PCMA transfer improves the classifier performance.

One has to keep in mind that the achieved evaluation metrics refer to test sets consisting of faces with a valid PCMA-transferred label. That said, classifier results deploying interpolated data (during training or testing) are inevitably positively biased. In general, classifier results deploying PCMA-transferred data still outperform the NN counterparts. The positive bias is shown by comparison of performance metrics evaluated on the entire manually annotated test set $(\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}}^{\text{manual}})$ and the PCMA-filtered manually labeled test set $(\overline{\mathbf{X}}_{\text{test}}, \overline{\mathbf{y}}_{\text{test}}^{\text{manual}})$. The bias amounts to 2–3 pp for the global performance metrics of "fully NN-interpolated" setups.

6.5.2 Indirect Semantic Segmentation of Meshes

Like the indirect semantic segmentation of PCs, we achieve to semantically segment meshes indirectly by transferring the semantic segmentation results from the PCs (see Section 6.4.1) to the correspondingly linked faces using PCMA. Table 6.12 compactly lists the performance metrics of the *indirect* semantic segmentation of meshes achieved with various RF models deploying different feature vector compositions (see Table 6.1) for data sets V3D and H3D. The respective classifiers are trained on $\mathcal{T}_a^{\text{PC}}$ which consists of points that have been associated with the mesh. By these means, the train sets of the direct and indirect approach, i.e., $\mathcal{T}_a^{\text{Mesh}}$ and $\mathcal{T}_a^{\text{PC}}$, operate on the same receptive field and allow for a fair comparison. However, $|\mathcal{T}_a^{\text{PC}}|$ differs from $|\mathcal{T}_a^{\text{Mesh}}|$ due to the one-to-many relationship between faces and points. For both variants, the performance metrics are evaluated on the faces of the test set that have been linked to the PC carrying a valid label, i.e., on $\mathcal{E}_a^{\text{Mesh}}$. Hence, Table 6.12 reflects the propagated results of the *direct* semantic segmentation of the respective PCs as given in Table 6.6. Generally, the metrics evaluated on the mesh (see Table 6.12) differ slightly from those evaluated on the PC (see Table 6.6) due to the area-weighted evaluation on the mesh and the one-to-many relationship between faces and points. However, the performance ranking of the feature vector compositions behaves similarly on both modalities. Best-performing indirect RF classifiers deploy the sparse multi-modal feature vectors $\mathcal{FS}_k$ and $\mathcal{FS}_{k'}$ respectively for V3D and H3D. Feature vectors $\mathcal{FS}_g$ and $\mathcal{FS}_{g'}$ respectively perform the worst for both data sets. Section 6.5.1 gives a more detailed analysis of the deployed feature sets. Table 6.8 lists the performance metrics for the direct semantic segmentation of meshes evaluated on $\mathcal{E}_a^{\text{Mesh}}$ and achieved with various RF models deploying different feature sets on V3D and H3D.

As stated in Section 6.4.1, we limit the direct semantic segmentation of PCs to "PC-only" configurations and the sparse version of the multi-modal feature vector (see Table 6.1). Therefore, comparing the direct and indirect results is constrained to these dedicated descriptors.

The comparison does not show a clear superior approach for the investigated data sets and feature vector compositions. For instance, the indirect approach performs slightly better for $mF1$-score on both data sets deploying $\mathcal{FS}_e$ (0.36–0.51 pp). On H3D, the indirect approach shows also better OAs for $\mathcal{FS}_e$ and $\mathcal{FS}_{e'}$ by 0.74 pp and 0.69 pp respectively. On the contrary, the direct approach performs 0.59 pp better in OA on V3D. The direct approach is 0.87 pp better in terms of OA for $\mathcal{FS}_f$ while achieving a similar $mF1$-score on V3D. Likewise, $\mathcal{FS}_f$ achieves similar $mF1$-scores on H3D with both the direct and indirect approach. The primed counterpart $\mathcal{FS}_{f'}$ is 0.52 pp better in $mF1$-score when applying the direct approach. The pure

Table 6.12: Surface-aware per-class $F1$-scores [%], $mF1$-scores, and OAs of *indirect* semantic segmentation of meshes achieved with RF models deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data sets V3D (*top*) and H3D (bottom). Configurations on H3D that consider PC features based on small-scale radii are marked with a prime and highlighted in *gray*. The RF models are trained and evaluated on the subsets of entities that have been linked via PCMA, i.e., on $(\mathcal{T}_a^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{Mesh}})$. The performance metrics can be compared in a fair way with performances of the *direct* semantic segmentation of meshes (see Table 6.8). The *black bars* above class names reflect the class distributions of the test sets.

Vaihingen 3D $(\mathcal{T}_a^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{Mesh}})$

$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Car*	*Fence/ Hedge*	*Roof*	*Facade*	*Shrub*	*Tree*	$mF1$ [%]	*OA* [%]
e	50.57	71.42	37.13	14.51	80.35	63.12	28.59	66.76	51.56	65.40
f	59.29	73.61	37.42	17.15	88.86	64.23	39.72	68.90	56.15	71.38
g	40.57	75.43	13.17	17.54	38.59	12.50	14.28	29.68	30.22	44.70
h	77.90	93.12	64.45	17.76	88.77	63.13	41.59	68.73	64.43	80.22
i	77.73	93.12	63.27	17.75	88.59	63.33	41.63	68.70	64.27	80.13
k	79.10	93.68	63.54	10.13	95.66	72.48	40.54	81.99	67.14	85.26

Hessigheim 3D $(\mathcal{T}_a^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{Mesh}})$

$\mathcal{FS}$	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	*OA* [%]
e	74.36	70.04	33.89	37.83	84.15	76.13	58.48	92.80	14.47	45.66	55.28	58.46	73.44
e'	79.13	78.59	42.46	43.07	86.35	79.62	63.60	94.19	16.90	56.66	61.83	63.85	77.79
f	76.55	70.94	38.68	42.24	89.84	78.71	59.71	93.27	19.41	54.22	68.27	62.89	76.44
f'	81.44	81.34	46.18	46.15	92.75	80.47	64.13	94.44	21.33	54.44	72.69	66.85	80.55
g	44.79	26.61	16.11	18.08	43.28	30.14	10.18	77.90	19.14	4.09	24.51	28.62	41.24
g'	47.92	26.61	24.06	23.31	44.04	36.84	14.12	80.24	18.71	5.92	21.43	31.20	43.86
h	76.42	71.91	52.15	42.11	89.85	78.63	61.21	93.40	26.50	48.76	64.23	64.11	76.80
h'	81.19	81.78	57.08	47.95	93.05	80.12	66.12	94.56	26.41	51.86	72.71	68.44	81.01
i	76.63	72.17	52.80	44.05	89.93	78.63	61.80	93.76	27.00	47.16	65.87	64.53	77.05
i'	80.83	81.71	56.99	48.19	92.85	80.47	66.23	94.65	25.50	54.03	75.31	68.79	80.92
k	89.18	88.66	54.24	48.74	92.04	80.13	64.37	95.16	47.27	55.84	64.14	70.89	85.14
k'	88.95	89.09	58.48	54.16	93.14	81.17	67.35	95.74	47.95	57.65	70.93	73.15	85.85

radiometric feature set $\mathcal{FS}_g$ achieves a better $mF1$-score for the direct approach on V3D (+0.47 pp), but a worse on H3D (−0.52 pp). $\mathcal{FS}'_g$ shows on par performance for the two approaches in both global metrics.

In Section 6.5.1, we show that the multi-modal feature set outperforms the "PC-only" configurations. Therefore, the comparison of the direct and indirect approach deploying $\mathcal{FS}_k$ and $\mathcal{FS}_{k'}$ respectively is the most relevant to us. The comparison between the direct and indirect approach for $\mathcal{FS}_k$ shows on par performance for V3D. The indirect approach outperforms the direct semantic segmentation result by 2.41 pp and 0.67 pp for $mF1$-score and OA respectively for $\mathcal{FS}_k$ on H3D. The primed equivalent shows on par performance for both approaches. This means the transferred contextual features with small-scale radii close the performance gap between the direct and indirect approach on H3D. The similar performance for $\mathcal{FS}_{k'}$ indicates that the transferred feature granularity is beneficial for direct semantic segmentation as the per-face aggregated feature vectors compete with per-point descriptors while outperforming the unprimed counterpart $\mathcal{FS}_k$ by 2.26 pp and 0.71 pp for $mF1$ and OA respectively.

Naturally, indirect classifiers can access the entire training pool $\mathcal{T}^{\mathrm{PC}}$ and are not bound to the linked

subset $\mathcal{T}_a^{\mathrm{PC}}$. In Section 6.4.2 we already discussed the effect of the extended train set on the semantic segmentation of PCs. We concluded that additional training samples do not consistently impact the performance for all feature sets on both data sets. However, the extended train set tends to improve $mF1$-scores while slightly decreasing OAs for most feature sets. The prediction sharing propagates this behavior directly to the mesh but evaluation metrics differ due to the one-to-many relationship between faces and points. In the following, we focus on the comparison of the direct and the *extended* indirect approach (trained on the entire train set $\mathcal{T}^{\mathrm{PC}}$) and highlight how additional training samples affect the discrepancy to the direct approach for selected feature sets. The performance metrics of the extended indirect approach can be taken from Table 7.5.

Training on $\mathcal{T}^{\mathrm{PC}}$ improves the performance of the indirect semantic segmentation for some feature configurations. For example, the extended indirect approach performs 0.94 pp better than the indirect approach for $\mathcal{FS}_e$ on V3D in terms of $mF1$-score. Consequently, the performance gain compared to the direct approach increases from +0.47 pp to +1.41 pp. Similarly, the extended indirect approach performs 1.55 pp and 0.69 pp better for $mF1$-score and OA respectively than the indirect approach for $\mathcal{FS}_f$ on V3D. The superiority increases the performance gain compared to the direct approach from +0.23 pp to +1.78 pp for $mF1$-score. At the same time, the extended indirect approach achieves similar OA like the direct approach, which outperformed the plain indirect approach by 0.87 pp. For $\mathcal{FS}_{f'}$, the direct and extended indirect approaches are on par, whereas the direct approach achieves 0.52 pp better $mF1$ than the plain indirect approach. The comparison of the direct and extended indirect approach for the multi-modal feature vector $\mathcal{FS}_k$ shows a better performance for the latter on V3D ($\Delta mF1 = 0.50$ pp, $\Delta\, OA = 0.40$ pp). The performance gap is larger for the extended indirect approach as it outperforms the plain indirect version trained on $\mathcal{T}_a^{\mathrm{PC}}$. However, training on $\mathcal{T}^{\mathrm{PC}}$ may also worsen the performance of the indirect way for a few feature sets. On H3D, the extended indirect approach is 1.00 pp and 1.21 pp worse for $mF1$-score and OA respectively deploying $\mathcal{FS}_g$ than the indirect approach. The extended indirect approach is 0.48 pp and 0.84 pp worse than the direct approach for $mF1$-score and OA respectively. Utilizing the indirect approach trained on linked train set $\mathcal{T}_a^{\mathrm{PC}}$ shows the reversed scenario: The indirect approach is 0.52 pp and 0.37 pp better than the direct approach for $mF1$-score and OA respectively. Besides, the extended indirect approach performs worse than the indirect semantic segmentation trained on $\mathcal{T}_a^{\mathrm{PC}}$ for $\mathcal{FS}_f$ on H3D. The direct and indirect approaches perform on par, whereas the direct semantic segmentation achieves 0.56 pp better $mF1$-score than the extended indirect approach. The primed counterpart shows similar behavior. The direct approach outperforms the extended indirect significantly by 0.95 pp and 0.73 pp for $mF1$-score and OA respectively. For other feature sets, the additional training samples have no significant effect on the achieved performance metrics. For example, the multi-modal feature sets $\mathcal{FS}_k$ and $\mathcal{FS}_{k'}$ achieve very similar results for both variants of the indirect approach on H3D. The performance gaps between the direct and the indirect variants remain constant.

6.5.3 Summary

Direct Semantic Segmentation of Meshes. In Section 6.5.1, we deep-dived the semantic segmentation of textured meshes by evaluating the performance of various feature vector compositions (see paragraph *Analysis of Different Feature Vector Compositions*). Besides, we analyzed the impact of the inter-modal information propagation on semantic segmentation by comparing the effects of PCMA-steered transfer and simple NN interpolation (see paragraph *Analysis of Different Inter-Modal Propagation Methods on Semantic Segmentation*. Detailed summaries are given at the end of these paragraphs in Section 6.5.1. In the following, we condense the main findings.

Analysis of Different Feature Vector Compositions. The excessive ablation study revealed the importance of the deployed features. We noted that contextual features are more important than per-face features. Amongst these, normal vector (more precisely, the third component n_z), per-face color, verticality and inclination (both transferred from the PC), and relative height δh are the most crucial.

The normal vector is the most relevant mesh-intrinsic geometric feature. Its inter-class separability is quasi non-existent for different ground classes and the vertically oriented classes. Besides, the meshing suffers

from reconstruction artifacts for the time being. For these reasons, the geometric information derived from the mesh is limited. Therefore, we argue that it is reasonable to incorporate features derived from the PC in per-face descriptors.

We observed that transferring PC features to the mesh effectively supports the semantic mesh segmentation. In particular, the full "PC-only" feature sets $\mathcal{FS}_i$ and $\mathcal{FS}_{i'}$ outperform the full "mesh-only" feature set $\mathcal{FS}_d$ in terms of $mF1$. The primed feature sets with small-scale contextual radii outperform their unprimed counterparts by 3–5 pp for H3D. We conclude that the geometric features derived from the PC are very informative that even compensate for the comparably low radiometric information (*reflectance* only).

The combination of radiometric and geometric features performs better than descriptors deploying features of a single feature type – regardless of the calculation modality. On both data sets, $\mathcal{FS}_d$ outperforms $\mathcal{FS}_b$ significantly ("mesh-only" configurations). The "PC-only" equivalents $\mathcal{FS}_h$ and $\mathcal{FS}_f$ behave similarly.

The multi-modal feature sets combining radiometric, geometric, individual, and contextual features from both modalities perform the best. For instance, $\mathcal{FS}_j$ outperforms $\mathcal{FS}_d$ by 8.87 pp and 11.84 pp for $mF1$-score and OA respectively on H3D.

The sparse variants $\mathcal{FS}_k$ and $\mathcal{FS}_{k'}$ achieve comparable results to their full equivalents $\mathcal{FS}_j$ and $\mathcal{FS}_{j'}$. For V3D, $\mathcal{FS}_k$ even outperforms its non-sparse counterpart $\mathcal{FS}_j$. In other words, the integration of geometric features from the PC with textural features from the mesh performs better than a holistic integration with textural and geometric features from the mesh. This finding underlines the strengths of both modalities: The PC features strong geometric information, whereas the mesh provides high-resolution texture.

The ablation study demonstrated that multi-modal entity linking already pays off for semantic segmentation of meshes when PC features are transferred to the mesh and utilized solely in the face descriptors. In particular, enhancing "mesh-only" compositions to multi-modal feature vectors significantly improves the performance resulting in the best configuration.

Analysis of Different Inter-Modal Propagation Methods on Semantic Segmentation. Apart from the extensive ablation studies regarding the deployed features, we also investigated the impact of the propagation method on classifier training and semantic segmentation. Specifically, we investigated whether a simple NN interpolation is sufficient or whether PCMA-steered transfer achieves significantly better performance metrics. To this end, we compared the achievable performance of classifiers trained with features and labels that have been propagated from the PC to the mesh via a) PCMA-steered aggregation or b) a simple nearest-neighbor interpolation. Figure 6.4 visualizes the bare differences of the propagation methods which have also be quantified relative to the manually annotated subset of the H3D mesh in Section 6.1.2.

We showed that utilizing the established linking information as filtering condition before the training improves classifiers' performance significantly when NN interpolation has been used to transfer information from PC to mesh. The PCMA-induced filtering improves the classifier performance by 8.29 pp and 8.01 pp for $mF1$-score and OA respectively for $\mathcal{FS}_d$ on V3D. On H3D, the performance gain amounts to 3.47 pp/2.37 pp for $mF1$-score/OA.

Moreover, we demonstrated that classifiers trained on data prepared with the PCMA-steered aggregation are superior to classifiers trained on NN-interpolated data. For instance, the RF trained on the fully PCMA-transferred data outperforms the NN-interpolated counterpart up to 8.03 pp/13.75 pp for V3D and 10.06 pp/19.37 pp for H3D in $mF1$-score/OA.

To conclude, incorporating the PCMA-established linking information as a filter condition or input to the transfer mechanism significantly improves the classifier performance and should be preferred over a simple NN interpolation.

Indirect Semantic Segmentation of Meshes. Section 6.5.2 discusses the indirect semantic segmentation of meshes and compares its performance to the direct approach (see Section 6.5.1). The indirect method transfers predictions from the PCs to the meshes utilizing the linking information between faces and points. The comparison between the direct and indirect results does not show a clear superior approach for the investigated data sets and feature vector compositions. For instance, both approaches achieve on par results for $\mathcal{FS}_k$ on V3D and for $\mathcal{FS}_{k'}$ on H3D. However, the indirect approach outperforms the direct

semantic segmentation significantly by 2.41 pp and 0.67 pp for $mF1$ and OA respectively deploying $\mathcal{FS}_k$ on H3D. Neglecting the multi-modal feature sets, the discrepancies between the direct and indirect approach are less than 1 pp on both data sets for all remaining feature sets.

For a fair comparison, we limited the training pool of the indirect approach to $\mathcal{T}_a^{\mathrm{PC}}$. We also investigated the impact of the extended training pool $\mathcal{T}^{PC}$ on the indirect semantic segmentation of meshes. Training on $\mathcal{T}^{\mathrm{PC}}$ improves the performance of the indirect approach for some feature configurations but also worsens the performance for others. Few feature descriptors achieve similar performance metrics for both variants of the indirect approach. The achieved performance depends on the deployed features and their separability on the utilized train and test sets. The inconsistent behavior across feature sets indicates that the feature information is more important than the number of training samples.

Conclusion. In summary, choosing the direct or (extended) indirect approach depends on the deployed feature vector composition and investigated data set. The multi-modal feature vectors achieve the best results regardless of the approach. The direct semantic segmentation of meshes is a good choice in general, although it might be outperformed by the indirect approach for a few feature vector compositions such as $\mathcal{FS}_k$. The major advantage of the direct approach is that classifiers have to deal with significantly fewer entities reducing training and inference time. For instance, the indirect approach deals with roughly 10 times more linked entities in the training pool than the direct approach for H3D.

6.6 Entangled Semantic Segmentation Using Active Learning

Whereas the previous sections investigate the semantic segmentation of PCs and meshes using PL, this chapter analyzes the performance of AL classifiers on these modalities. The classifiers are trained on sparse, iteratively refined training pools deploying the proposed coupled pipeline (see Figure 6.12). The training pools are generated by a variety of AL configurations deploying omniscient and crowd-based oracles (see Table 6.2). We evaluate their performance on both 3D modalities. Simulated AL loops deploying an omniscient oracle $\mathcal{O}_O$ serve as baselines for the AL configurations deploying a crowd C of non-experts $\mathcal{O}_C$.

As stated in Section 6.1.3, we simplify the H3D class catalog to query non-experts for annotation. Besides, the PC is colorized without incorporating the textured mesh to delineate better the visual appearance of mesh and PC. Therefore, the achieved results cannot be compared with the results of previous sections where H3D data is used like described in Section 4.2. We apply PL to both modalities of the modified H3D data for comparative purposes.

Table 6.13 lists the performance metrics of the RF models employing AL and PL. The performance is evaluated on both modalities. We discuss the results for the PC and mesh considering the visualized modalities and used selection strategies in Sections 6.6.1 and 6.6.2 respectively. During labeling, the crowd is shown the entity's vicinity by the colorized PC or textured mesh. In Section 6.1.3, we evaluate the influence of the visualized modality on the crowd's annotation quality. Please note that we do not evaluate the RF model deploying $\mathcal{O}_{C|\mathrm{PC}}$ on the mesh. From our point of view, visualizing a discrete 3D PC is not reasonable when we want to analyze the performance on the mesh.

The RF models trained and evaluated on the PC use a colored version of $\mathcal{FS}_{i'}$. Color has been interpolated from the respective photogrammetric cloud (see Section 6.1.3). The models trained and evaluated on the mesh deploy the multi-modal $\mathcal{FS}_{j'}$ (see Table 6.1). As stated in Section 6.1.3, we spatially subsample the dense H3D PC with 30 cm point distance for efficiency reasons.

The coupling of the mesh to the PC allows for parallel RF training on the mesh. Hence, the coupling reduces processing time and costs. We stress that the PC classifiers determine the requested entities during the coupled AL loops (see Section 6.1.3), which means that selected and annotated faces are not optimized for training ML models on the mesh. We try to compensate for this design defect by enhancing the feature vector for the mesh with per-face aggregated LiDAR features as transferred from the PC. The coupled semantic segmentation of the 3D mesh performs achieves good results (see Table 6.13). The results suggest that the coupled approach is a good, cheap, and fast approximation of a truly mesh-steered AL solution for semantic mesh segmentation.

Table 6.13: Semantic segmentation results for PL and AL on the test site of the modified H3D (see Section 6.1.3) both for the ALS PC and the mesh. Different oracles utilizing different representations for visualization (colorized PC or textured mesh) and different selection strategies (with/without RIU) provide annotated training data for the respective AL configurations (see Table 6.2).

			*F*1-score [%]								
Method	*Modality*	*Oracle* $\mathcal{O}$	*U. Furn.*	*Low Veg.*	*I. Surf.*	*Vehicle*	*Roof*	*Facade*	*Veg.*	*mF*1 [%]	*OA* [%]
PL	PC	–	41.14	90.68	85.26	51.63	92.96	83.77	93.05	76.92	88.16
	Mesh	–	48.57	91.38	88.15	49.81	91.86	85.34	91.88	78.14	88.00
AL	PC	$\mathcal{O}_O$	33.93	**90.31**	82.70	56.34	88.33	**79.73**	**92.66**	74.86	**86.65**
		$\mathcal{O}_O$ + RIU	**36.97**	89.91	**83.84**	55.42	**90.05**	79.61	91.91	**75.39**	86.60
		$\mathcal{O}_{C\|PC}$	32.32	89.92	76.26	53.95	88.88	75.43	91.70	72.64	83.38
		$\mathcal{O}_{C\|PC}$ + RIU	31.00	88.54	79.69	53.04	86.82	76.43	90.67	72.31	83.55
		$\mathcal{O}_{C\|Mesh}$	33.37	88.34	78.14	**57.40**	88.89	79.83	92.07	74.01	85.15
		$\mathcal{O}_{C\|Mesh}$ + RIU	34.56	89.59	81.81	56.20	89.18	79.35	92.56	74.75	86.26
	Mesh	$\mathcal{O}_O$	43.32	90.00	84.00	54.65	86.65	81.95	90.95	75.93	85.32
		$\mathcal{O}_O$ + RIU	**46.57**	**90.14**	**85.95**	57.37	**88.88**	81.91	90.99	**77.40**	**85.95**
		$\mathcal{O}_{C\|Mesh}$	45.93	88.71	82.83	**57.51**	87.45	**82.08**	90.58	76.44	84.99
		$\mathcal{O}_{C\|Mesh}$ + RIU	44.91	90.02	85.39	53.36	86.84	81.40	**91.15**	76.15	85.54

Figure 6.22 shows the gradually increasing *mF1*-scores of the deployed AL configurations for semantic segmentation of the test set considering the increasing annotated training pools. The trained RF models are evaluated on the PC (*top*) and the mesh (*bottom*). For both modalities, the AL configurations anneal to the respective PL performance which uses significantly more annotated entities. For instance, the fully annotated PL train set of the subsampled cloud provides roughly 400 times more annotated points. PL performs up to 4.61 pp and 4.78 pp better in $mF1$ and OA respectively on the PC. On the mesh, PL outperforms AL configurations up to 2.21 pp and 3.01 pp ($mF1/OA$).

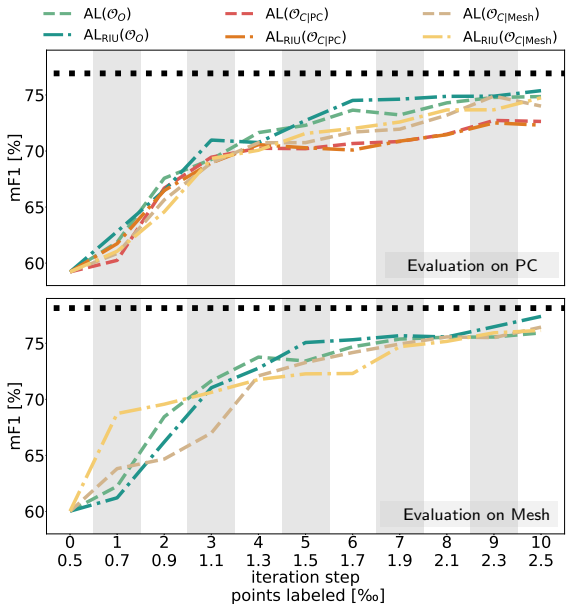


Figure 6.22: Iteration-wise performances of various RF models regarding the PC (*top*) and the mesh (*bottom*) for semantic segmentation of the modified H3D data. The RF models are trained on different iteratively increasing training pools annotated by oracles $\mathcal{O}$. *Dotted black* lines represent the respective PL result.

6.6.1 Entangled Semantic Segmentation of the Point Cloud

In Figure 6.22 (*top*), we observe that all AL configurations perform similarly well within the first 4 iteration steps. From $i = 5$, the approaches utilizing the mesh as presentation modality ($\mathrm{AL}(\mathcal{O}_{C|\mathrm{Mesh}})$ & $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$) start diverging from the ones visualizing the PC ($\mathrm{AL}(\mathcal{O}_{C|\mathrm{PC}})$ & $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{PC}})$) and anneal to the baseline solutions, which utilize correct labels only ($\mathrm{AL}(\mathcal{O}_O)$ & $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_O)$). Table 6.13 reveals that the classifiers on the PC perform up to 2.44 pp and 2.71 pp better in $mF1$ and OA respectively after 10 iterations when the mesh is presented to the crowd during labeling. $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$ is the best performing AL configuration. It performs 2.17 pp and 1.90 pp worse in $mF1$ and OA respectively than the PL results. The baseline solution using the omniscient oracle $\mathcal{O}_O$ is 0.64 pp and 0.34 pp better for $mF1$ and OA respectively than $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$. Section 6.1.3 discloses that visualizing the mesh improves crowd performance by 3–4 pp for both performance metrics. The results emphasize the impact of the visualized modality on the annotation quality and the classifier performance.

RIU improves $mF1$-score and OA significantly by 0.74 pp and 1.11 pp respectively when the mesh is visualized. In contrast, RIU neither seems to improve nor to diminish the performance when the PC is visualized during the annotation, although RIU reduces systematic confusions resulting in more correct crowd labels (see Section 6.1.3). $\mathrm{AL}(\mathcal{O}_{C|\mathrm{PC}})$ and $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{PC}})$ perform significantly worse than $\mathrm{AL}(\mathcal{O}_{C|\mathrm{Mesh}})$ ($\Delta mF1$: upto -1.70 pp, ΔOA: upto -1.77 pp). This might seem counterintuitive for $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{PC}})$ which features a similar OA for the provided crowd labels like $\mathrm{AL}(\mathcal{O}_{C|\mathrm{Mesh}})$ (roughly 87–88 %, see Figure 6.14 b and c). The analysis of the number of correctly labeled instances of class *Urban Furniture* hints at a possible explanation. Approaches utilizing the mesh as representation modality achieve up to 1.7 times more correct manual annotations than their counterparts visualizing the neighborhood as PC. The more correct samples for this particular class are available in the training set, the better is the OA of the RF. This corresponds well to the performance of the trained RF models. However, one would assume that AL is capable of detecting required points and requesting corresponding labels by design. Since this seems not to be the case, we conclude that the model is rather confident about this class – or at least, it is more unconfident in other classes. Figure 6.23 supports the made conclusion. Until the fourth iteration step, a similar cumulative amount of *Urban Furniture* points is queried in each AL run. From then on, almost no points of this class are requested for the runs relying on $\mathcal{O}_{C|\mathrm{PC}}$ (*upper two rows*), unlike AL loops using $\mathcal{O}_{C|\mathrm{Mesh}}$ (*lower two rows*). Probably, the crowd's confusion of *Urban Furniture* with other classes in early iteration steps causes uncertainty in other classes (and low precision). As a consequence, the models, therefore, mainly request labels of the confused classes.

Regardless of the AL configuration, class *Vegetation* achieves high $F1$-scores. However, we see a small drop in performance for $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{PC}})$ compared to $\mathrm{AL}(\mathcal{O}_{C|\mathrm{PC}})$. The performance loss might be explained by the color quality in vegetational areas which confuses the crowd and the trained classifier (see Section 6.1.3).

6.6.2 Entangled Semantic Segmentation of the Mesh

The tracked $mF1$-scores of the mesh classifiers diverge more than their equivalents on the PC (Figure 6.22, *bottom*). We assume that this is because AL loops run on the PC and hence, selected points are optimized for semantic segmentation of the PC. In the first iteration step, $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$ improves its $mF1$-score significantly suggesting that RIU is beneficial for fast learning. Afterwards, the $mF1$-score increases almost linearly while being clearly ahead of $\mathrm{AL}(\mathcal{O}_{C|\mathrm{Mesh}})$ until $i = 4$. From that iteration step on, $\mathrm{AL}(\mathcal{O}_{C|\mathrm{Mesh}})$ increases linearily, too. Both reach approximately the same $mF1$-score at the end of the last iteration. The two baseline AL runs employing the omniscient oracle $\mathcal{O}_O$ yield top AL results in most iteration steps (see Table 6.13).

AL configurations deploying RIU achieve marginally better OAs than their plain counterparts. We assume that the impact of RIU is minor than on the PC, since applying RIU means to select points that are likely to be further away from class borders but still may be linked to the same face. This is an immediate consequence of coupling the semantic segmentation of the mesh to the AL loop running in the PC branch. In

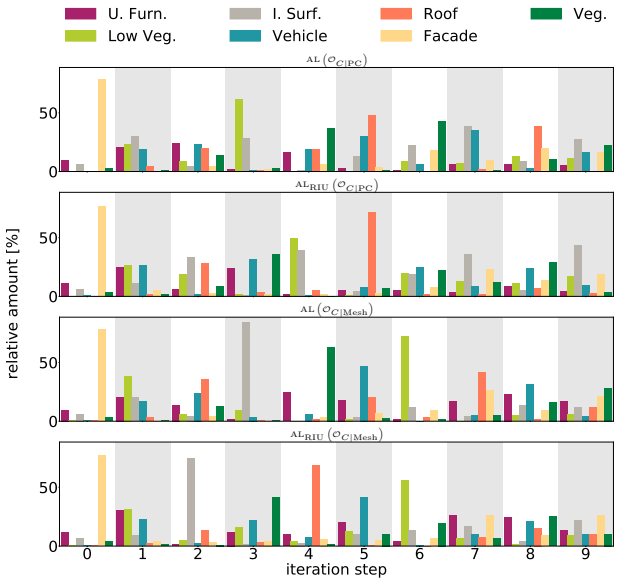


Figure 6.23: Histograms of true class affiliation of selected most informative ALS points for each iteration step (considering only points that are selected in the same iteration). The simplified class catalog of H3D has been used.

general, the selected faces are not optimized for the classifier training. $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$ reaches top $mF1/OA$ on the mesh differing by 1.99 pp/2.46 pp from the PL solution.

Class *Urban Furniture* reaches a significantly higher $F1$-score on the mesh than on the PC (9.60–15.57 pp). This can be explained by the fact that we evaluate accuracies merely for faces of the test set that can be matched with LiDAR data. PCMA links faces with points and automatically derives the GT for the mesh from the manually annotated PC. Faces that do not reconstruct the underlying geometry properly are likely to remain unassociated and unlabeled. For instance, a sharp class border of the underlying geometry may be reconstructed as a smoothed border by a few large faces, which will not be linked to the PC. Unlabeled faces do not form part of the evaluation. Such unlinked faces are often hard to interpret – for humans and classifiers. In contrast, GT is given everywhere on the PC wherefore performance can be analyzed at difficult locations, too. Besides, meshing algorithms struggle to reconstruct complex and thin structures (see Figure 5.13). In particular, *Urban Furniture* has a high intra-class variance featuring fine-grained and complex structures. The simplified reconstruction implicitly reduces the intra-class variance by filtering the complexity of data. Eventually, the meshing and the way GT is derived for the mesh cause an improved classifier performance compared to the PC classifier by implicitly increasing the relative amount of simple-to-interpret entities.

6.6.3 Summary

AL intertwines GT generation and classifier training. Section 6.1.3 discusses the GT generation deploying various AL loops with crowd-based and omniscient oracles. In this section, we analyzed the performance of the correspondingly trained RF classifiers on the mesh and subsampled H3D PC. The PC has been subsampled with point distance of 30 cm to speed up the AL loop. However, the PC has been visualized at full density during the annotation by the crowd.

Section 6.1.3 showed the best crowd performance when the mesh is visualized to the crowdworkers, and RIU modifies the entropy-based selection strategy. The superiority in labeling accuracy is handed down straightly to the performance of the respective PC classifier. The trained classifier of $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$ achieves the best performance on the PC for non-omniscient oracles with a $mF1$-score of 74.75 % and an OA of 86.26 %. In comparison, the respective classifier deploying an omniscient oracle achieved 0.64 pp and

0.34 pp better $mF1$-score and OA respectively. The trained classifier of $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{PC}})$, i.e., when the PC is visualized, achieved 2.44 pp and 2.71 pp worse $mF1$-score and OA respectively on the PC (see Table 6.13).

The coupling of mesh and PC via PCMA enables parallel classifier training on both modalities. More precisely, the mesh classifier is coupled to the AL loop operating on the PC branch (see Figure 6.12). The coupled semantic segmentation of the 3D mesh achieves good results for $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$ (76.15 %/85.54 % for $mF1/OA$) although the AL loop is optimized for the PC: The selected entities for the labeling are chosen such that the PC classifier performance improves as much as possible per iteration. However, the achieved results on the mesh suggest that the coupled approach is a good, cheap, and fast approximation of a truly mesh-steered AL solution for semantic segmentation of meshes. The mesh classifier deploying $\mathrm{AL}_{\mathrm{RIU}}(\mathcal{O}_{C|\mathrm{Mesh}})$ achieves 1.25 pp/0.41 pp worse $mF1/OA$ than the respective classifier harnessing the omniscient oracle.

We demonstrated for both modalities that labels received solely by the crowd can power a ML system. However, they cannot compete with the performance of PL classifiers that train on the entire labeled training pool (400× more annotated points). The best performing AL configuration is 2.17 pp/1.90 pp worse in $mF1/OA$ for the semantic segmentation of the H3D PC compared to the PL result. The best performing AL configuration evaluated on the mesh achieves 1.99 pp/2.46 pp worse $mF1/OA$ than PL.

Chapter 7

Conclusion and Outlook

In this thesis, we presented a holistic association between imagery, PCs, and meshes that explicitly links their entities pixels, points, and faces (see Chapter 5). The inter-modal association allows information sharing across modalities in an arbitrary direction. Measured and engineered features can be propagated to other modalities enhancing instance-level descriptors to multi-modal feature vectors. Besides, predictions and manually assigned labels can be shared across modalities. Hence, the proposed association mechanism enables flexible *Juggling with Representations* and serves as a powerful integrative backbone boosting GT generation, multi-modal learning, and joint semantic segmentation of imagery, PCs, and meshes.

The sharing of manually assigned labels facilitates the semi-automatic annotation of unlabeled modalities without another manual interaction and enables classifier training on them. Likewise, the prediction sharing enables indirect semantic segmentation profiting from well-performing classifiers on other modalities. In particular, we can semantically segment a modality by training a classifier on the same modality (*direct* approach) or by transferring predictions from other modalities (*indirect* approach). Hence, any established well-performing modality-specific classifier can be used for semantic segmentation of these modalities – regardless of whether they follow an end-to-end learning or feature-driven scheme. Figure 6.1 summarizes the versality of the semantic segmentation utilizing the entity linking as the backbone.

The mesh is the core modality of the inter-modal association coupling the three bilateral subprocesses PCMA, ImgMA, and PCImgA (see Sections 5.1, 5.2, and 5.3). The geometry-driven association is a generic approach that is not bound to airborne imagery or ALS data. In addition, the approach is robust against meshing algorithms of different software and can handle 2.5D and 3D meshes. The implementation follows a tile-wise strategy to handle data sets of arbitrary size (scalability). Concurrently, the implementation enables parallel, distributed processing reducing the processing time. In the light of good scientific practice, we discussed the preconditions and limitations of the association process in Section 5.4. Mainly, inter-modal discrepancies, co-registration residuals, and the reconstruction quality of the mesh affect the association.

We demonstrated the effectiveness of the association mechanism for GT generation and semantic segmentation by deploying the ISPRS benchmark data sets V3D and H3D featuring different resolutions and scales (see Chapter 4). Throughout the thesis, we focused on linking the 3D modalities and their semantic segmentation as the relationship between 3D space and imagery is well-defined by collinearity equations. We verified qualitatively and quantitatively that the proposed label propagation outperforms a simple inter-modal NN interpolation (see Section 6.1.2). Furthermore, classifiers perform better when multi-modal information has been prepared with the proposed entity linking and respective data fusion instead with NN interpolation.

Since the dawn of the DL era, efficient GT generation has become a compelling task (see Section 6.1). The explicit entity linking and subsequent label transfer boost the GT generation by limiting the manual annotation to a single modality. For instance, labeled 3D data can be projected into image space to annotate multiple images at once, avoiding labor-intensive pixel-wise manual annotation. Besides, the semi-automatic labeling avoids disjoint manual annotation processes and thus achieves consistent annotation across the modalities. As a product of this thesis, we published multiple epochs of manually annotated PCs and semi-

automatically labeled meshes of the Hessigheim data (Kölle et al., 2021a). The community is encouraged to test any developed classifier on the benchmark. By these means, we implicitly foster a sustainable ablation study of classifiers that is not limited to the period of this thesis and keeps up with the time.

Combining the proposed inter-modal linking with AL further reduces the manual annotation effort to a few entities on a single modality. We utilized a human-in-the-loop AL pipeline that couples the iterative refinement of a mesh classifier to the PC-driven AL loop and its PC classifier (see Figure 6.12). The iterative annotation was done by a group of paid non-experts. The application of paid crowdsourcing avoids recruiting expensive experts for the annotation process (see Section 6.1.3). Specifically, annotation time and costs dropped from several months and thousands of dollars to a few days and hundreds of dollars when comparing the full annotation by experts with sparse AL-steered labeling done by a crowd of non-experts. We demonstrated for both modalities that labels received solely by the crowd can power a ML system. However, the AL-driven semantic segmentation of the H3D data performs up to 3 pp worse than the compared PL results which utilized 400 times more training points. Additionally, we analyzed the impact of the visualized modality on the annotation quality as achieved by the non-experts. We observed that visualizing the mesh improves the crowd's labeling accuracy by up to 3 pp compared to visualizing the PC during the annotation process. Likewise, the classifier performance is 2.44 pp and 2.71 pp for $mF1$ and OA respectively better when being trained with labels where the mesh has been visualized instead of the PC. To sum up, visualizing the mesh improves both the crowd's labeling accuracy and the resulting semantic segmentation.

For the conventional PL scheme, we performed an excessive ablation study on the 3D modalities deploying fast off-the-shelf RF classifiers. Modality-specific feature vectors revealed that the combination of geometric and radiometric features outperforms single-type descriptors. Geometric features are util for the separation of radiometrically similar classes. On the other hand, radiometric features help to separate geometrically similar classes. The most relevant feature is relative height. These findings are in accordance with existing literature. In Section 6.4.1, we have seen that enhancing LiDAR PCs with colors from the respective textured mesh improves semantic segmentation of PCs up to 5.76 pp and 9.74 pp for $mF1$ and OA respectively. Likewise, Section 6.5.1 shows significantly better semantic segmentation of meshes when features derived from the PC extend mesh features. The improvement is up to 12.25 pp and 5.46 pp for $mF1$ and OA respectively. Besides, feature vectors deploying transferred PC features only outperform "mesh-only" feature vectors on the mesh, indicating that propagated PC features compensate for the loss of geometric details during the meshing process. The multi-modal feature combination achieves the best results. The excessive ablation study revealed the strengths of both modalities. Whereas PCs provide high-quality geometry, the meshes provide high-quality textural information.

The analysis of indirect approaches shows comparable performances with respective direct results (see Sections 6.4.1 – 6.5.2). The experiments did not detect a clear advantage of the direct over the indirect approach and vice versa for the investigated feature compositions and data sets on both 3D modalities. However, the proposed linking method provides us the flexibility to choose the best-performing approach individually.

We briefly presented the main results of this work utilizing the multi-modal entity linking as the backbone. We have proven its concept, scalability, and versatile application possibilities focusing on the airborne configuration. Generally, the proposed methodology captivates with its simplicity and genericity. The method benefits from the recent hybridization trend and advances in data acquisition, adjustment, automatic surface reconstruction, and classifier design, making it very powerful for future developments. In the following, we venture an outlook on possible future trends.

In the future, we expect simultaneous data acquisition and their joint adjustment to become the default in practical applications on any scale. The joint acquisition reduces the capture time and avoids discrepancies caused by asynchronous data acquisition. The joint adjustment explicitly co-registers ALS data, imagery, and their derived products. Hence, under ideal conditions, inter-modal discrepancies reduce to sensor noise and still existing imperfections of automatically reconstructed meshes.

However, mesh quality is expected to improve gradually in the future. In our opinion, semantics is not restricted to the bare interpretation and enhancement of acquired data. For instance, the results of the semantic segmentation of PCs might be utilized to come up with a class-aware meshing of the PCs. The

integration of semantics might result in sharper edges for the reconstruction of hand-made objects like buildings, better reconstruction of vegetational classes like trees, and class-aware borders between geometrically similar classes like a street and a lawn. On the other hand, already existing meshes might be remeshed in the postprocessing by incorporating semantic segmentation results of the direct approach. Besides, inconsistencies between forward and backward passed labels between PC and mesh could also be used to improve the automatically generated meshes via remeshing while avoiding the training and application of a classifier. From our point of view, the integration of semantics on a grand scale into the geometric reconstruction pipeline is the next logical step to improve the reconstruction quality.

In turn, the improved geometric reconstruction is likely to improve the semantic segmentation of meshes. In this work, we utilized fast RF classifiers deploying conventional handcrafted features to evaluate the added value of multi-modality on the semantic segmentation. In the future, the RF models could be replaced by more sophisticated end-to-end learning classifiers or other classifier designs. Specifically, we expect evolvement for graph-based DL methods that operate smoothly on large-scale geospatial meshes without the need of downsizing input data. Alternatively, we could extend the current feature-driven approach by incorporating feature learners on 2D and 3D modalities. By these means, well-performing handcrafted features could be replaced or extended with learned descriptors. In particular, the application of feature learners in the original image space could enhance the expressivity of textural features. Moreover, the learned features could enrich the compact texture atlas as additional channels. Similarly, the linked LiDAR information could extend existing mesh formats to all-embracing formats that explicitly encode and bundle all modalities.

In summary, we expect future developments to improve the automated surface reconstruction and the confluence of imagery, PCs, and meshes. In our opinion, the anticipated developments will further increase the utility and the acceptance of the mesh in photogrammetry and remote sensing.

Appendix

Vaihingen 3D

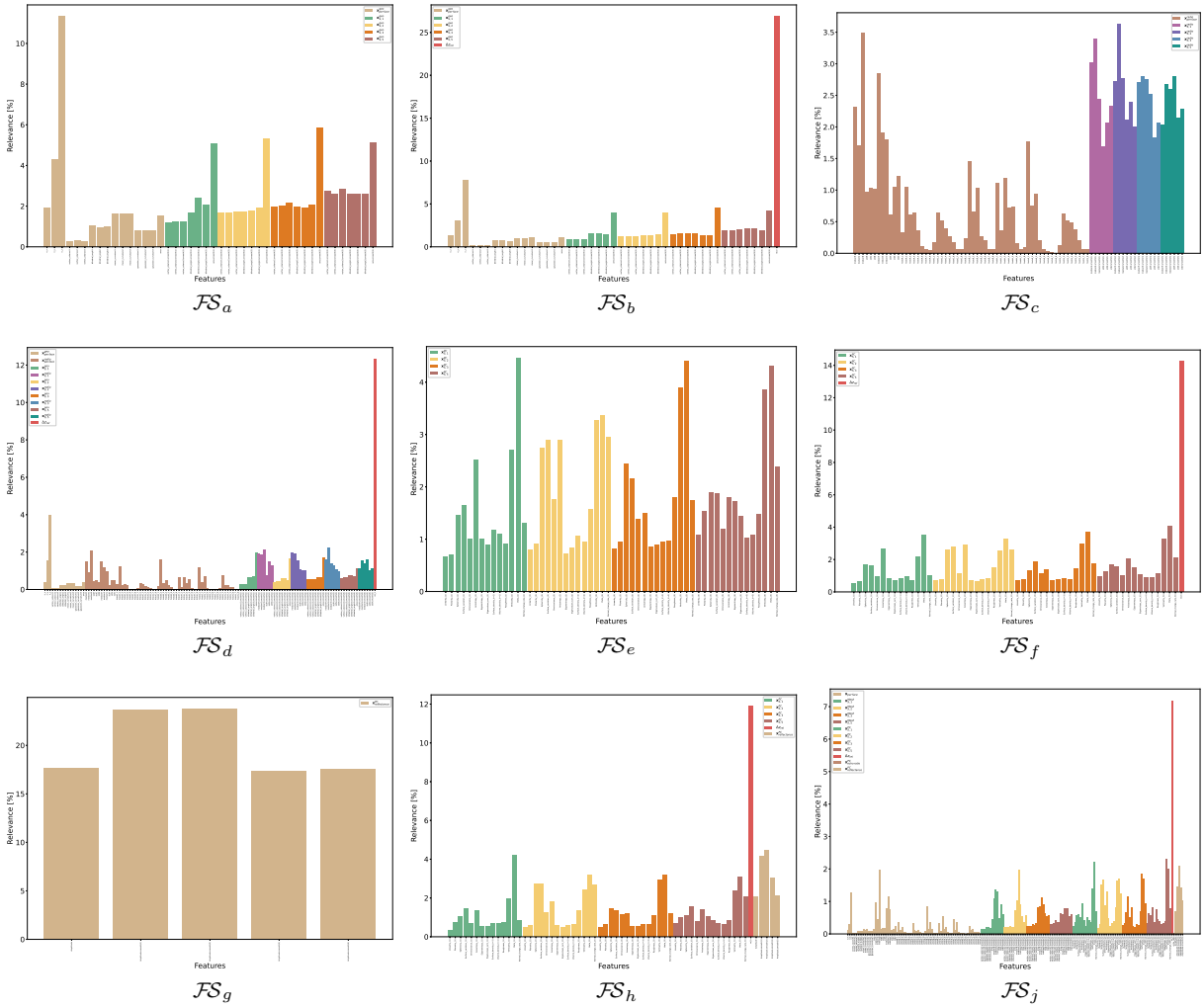


Figure 7.1: Feature relevance plots of RF models deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data set V3D (best viewed digitally). Please note the different scales on the y-axes.

Hessigheim 3D

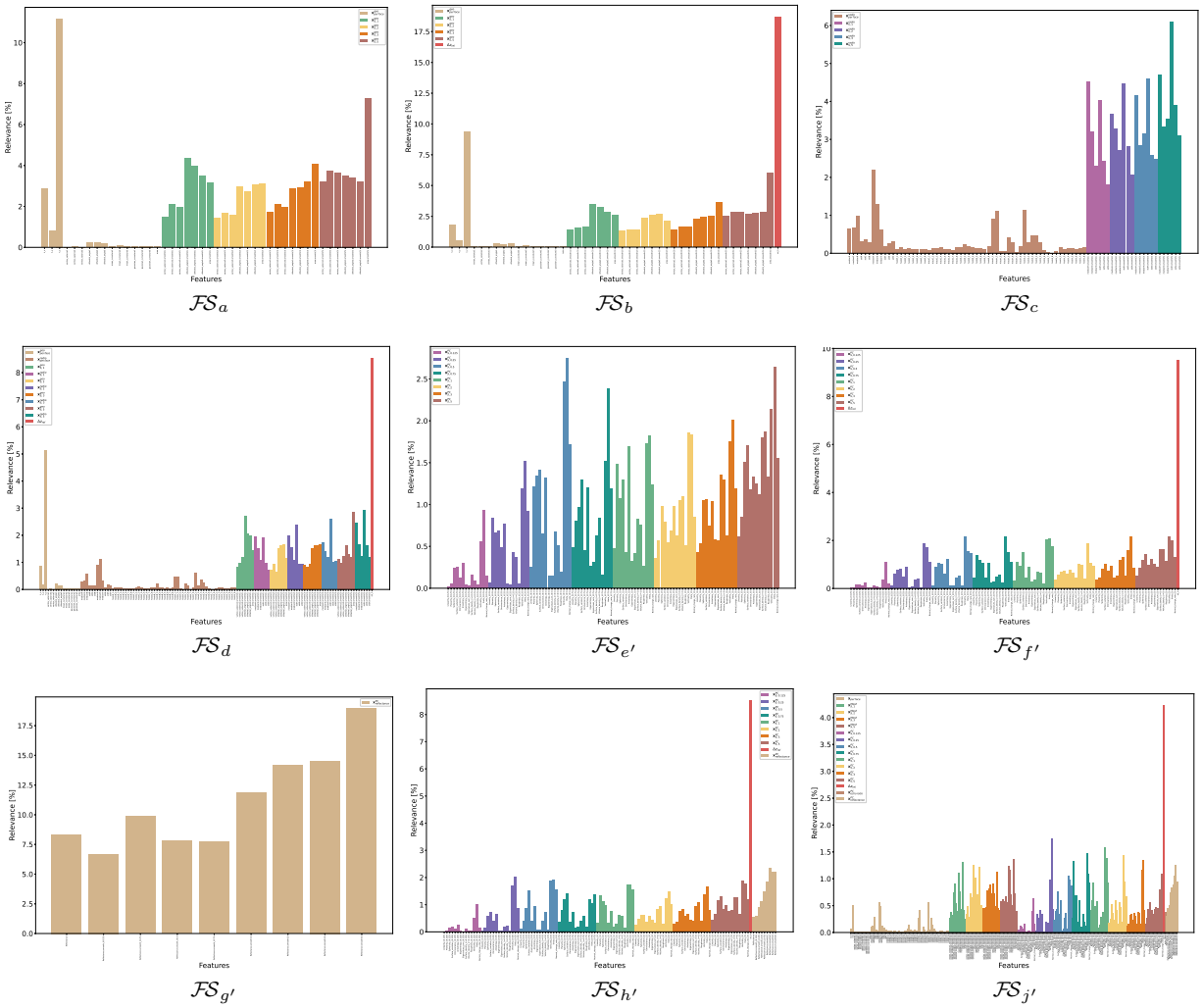


Figure 7.2: Feature relevance plots of RF models deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data set H3D (best viewed digitally). Please note the different scales on the y-axes.

Table 7.1: Point-level per-class $F1$-scores [%], $mF1$-scores, and OAs of *direct* semantic PC segmentation achieved with RF deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data set V3D. The RF models are trained and evaluated on different subsets of the train set and test set respectively. The classifiers are trained on a) all points of train set $\mathcal{T}^{\mathrm{PC}}$, b) subset $\mathcal{T}_a^{\mathrm{PC}}$ containing all points that have been linked to the mesh via PCMA, and c) subset $\mathcal{T}_n^{\mathrm{PC}}$ containing all points that are not linked to the mesh. The trained RF classifiers are evaluated on A) all points of test set $\mathcal{E}^{\mathrm{PC}}$, B) subset $\mathcal{E}_a^{\mathrm{PC}}$ containing all points that have been linked to the mesh via PCMA, and C) subset $\mathcal{E}_n^{\mathrm{PC}}$ containing all points that are not linked to the mesh.

Vaihingen 3D

$\mathcal{FS}$	*Eval* (PC)	*Low Veg.*	*I. Surf.*	*Car*	*Fence/ Hedge*	*Roof*	*Facade*	*Shrub*	*Tree*	$mF1$ [%]	OA [%]
$\mathcal{T}^{\mathrm{PC}}$: **Trained on all Points**											
e	$\mathcal{E}$	55.32	70.79	43.31	16.14	75.03	54.10	28.56	65.06	51.04	62.83
	$\mathcal{E}_a$	51.54	73.90	36.15	18.39	77.69	53.87	29.33	64.44	50.66	65.21
	$\mathcal{E}_n$	61.85	50.24	51.94	11.57	57.32	54.45	27.42	66.08	47.61	56.08
f	$\mathcal{E}$	66.22	72.58	44.97	19.77	86.41	56.22	41.08	69.12	57.05	70.30
	$\mathcal{E}_a$	60.78	75.88	37.93	21.77	88.11	55.36	41.04	66.99	55.98	71.86
	$\mathcal{E}_n$	74.63	51.76	53.57	15.14	73.73	57.36	41.16	72.90	55.03	65.90
g	$\mathcal{E}$	46.04	74.29	17.26	17.27	37.67	6.95	16.80	30.03	30.79	44.55
	$\mathcal{E}_a$	47.28	76.78	13.44	19.05	40.74	7.26	15.74	27.17	30.93	47.43
	$\mathcal{E}_n$	43.51	61.60	22.84	13.55	22.39	6.46	18.40	35.43	28.02	36.39
h	$\mathcal{E}$	80.64	90.82	67.22	21.27	86.81	54.92	42.18	68.67	64.07	78.68
	$\mathcal{E}_a$	81.56	93.27	60.02	22.04	88.44	54.15	42.71	66.74	63.61	81.35
	$\mathcal{E}_n$	79.23	77.20	75.60	19.42	74.72	55.97	41.34	72.05	61.94	71.14
i	$\mathcal{E}$	80.65	90.83	66.91	19.34	86.76	54.89	42.46	68.78	63.83	78.67
	$\mathcal{E}_a$	81.49	93.28	58.24	20.34	88.40	54.00	42.61	66.73	63.14	81.28
	$\mathcal{E}_n$	79.36	77.09	76.90	16.85	74.56	56.12	42.23	72.41	61.94	71.30
k	$\mathcal{E}$	81.29	91.35	66.76	14.97	94.80	60.78	43.56	81.07	66.82	83.27
	$\mathcal{E}_a$	82.15	93.74	58.59	14.86	95.81	62.70	42.14	81.24	66.40	86.04
	$\mathcal{E}_n$	79.98	78.31	76.29	15.22	87.63	58.12	45.81	80.79	65.27	75.45
$\mathcal{T}_a^{\mathrm{PC}}$: **Trained on Associated Points**											
e	$\mathcal{E}$	53.98	69.47	37.14	15.14	74.75	53.22	28.86	65.13	49.71	62.07
	$\mathcal{E}_a$	52.00	72.54	29.10	17.65	78.10	53.43	29.87	64.74	49.68	65.00
	$\mathcal{E}_n$	57.63	48.89	46.67	10.32	54.29	52.96	27.38	65.79	45.49	53.76
f	$\mathcal{E}$	65.10	71.33	36.84	18.71	86.60	55.88	39.91	68.96	55.42	69.52
	$\mathcal{E}_a$	60.49	74.74	29.44	20.35	88.31	55.11	40.44	66.98	54.48	71.38
	$\mathcal{E}_n$	72.52	49.64	45.67	15.01	73.89	56.90	39.14	72.50	53.16	64.28
g	$\mathcal{E}$	45.76	74.03	14.90	16.32	38.42	7.69	16.49	30.51	30.52	44.83
	$\mathcal{E}_a$	47.10	76.65	11.68	18.26	41.58	8.03	15.20	27.90	30.80	47.84
	$\mathcal{E}_n$	43.05	60.84	19.34	12.31	23.09	7.16	18.44	35.47	27.46	36.29
h	$\mathcal{E}$	79.49	90.51	64.99	20.53	87.08	55.26	41.55	69.15	63.57	78.37
	$\mathcal{E}_a$	81.13	93.21	58.20	21.75	88.73	54.17	42.24	67.38	63.35	81.38
	$\mathcal{E}_n$	76.91	75.89	72.90	17.61	74.94	56.73	40.50	72.25	60.97	69.87
i	$\mathcal{E}$	79.44	90.71	64.27	20.59	86.92	55.06	41.50	69.15	63.46	78.34
	$\mathcal{E}_a$	81.15	93.32	57.02	21.49	88.60	54.18	42.18	67.26	63.15	81.34
	$\mathcal{E}_n$	76.73	76.49	72.67	18.45	74.44	56.25	40.48	72.48	61.00	69.84
k	$\mathcal{E}$	80.49	91.42	65.48	14.26	94.67	60.77	42.38	80.57	66.25	82.91
	$\mathcal{E}_a$	81.93	93.81	56.59	13.87	95.72	62.63	41.63	80.79	65.87	85.92
	$\mathcal{E}_n$	78.24	78.41	75.81	15.16	87.31	58.24	43.53	80.20	64.61	74.38
$\mathcal{T}_n^{\mathrm{PC}}$: **Trained on Non-Associated Points**											
e	$\mathcal{E}$	56.17	73.91	36.47	14.76	74.75	53.39	25.38	64.53	49.92	64.15
	$\mathcal{E}_a$	49.97	76.79	28.74	16.66	77.00	53.73	25.45	63.67	49.00	65.91
	$\mathcal{E}_n$	66.09	55.50	45.65	10.40	59.20	52.88	25.27	65.93	47.62	59.17
f	$\mathcal{E}$	65.97	75.72	40.01	17.54	85.38	54.58	39.62	67.98	55.85	70.77
	$\mathcal{E}_a$	59.30	79.01	32.19	19.45	87.13	53.27	39.81	65.52	54.46	72.29
	$\mathcal{E}_n$	75.44	55.25	49.53	13.03	72.35	56.38	39.31	72.36	54.21	66.46
g	$\mathcal{E}$	44.95	77.18	21.56	16.24	27.51	5.74	14.37	28.88	29.55	42.87
	$\mathcal{E}_a$	44.83	80.12	18.14	18.06	29.29	5.41	14.22	25.55	29.45	44.91
	$\mathcal{E}_n$	45.19	61.95	26.83	12.33	17.86	6.29	14.59	35.68	27.59	37.09
h	$\mathcal{E}$	81.65	90.73	64.81	20.40	85.54	51.97	41.41	67.33	62.98	78.51
	$\mathcal{E}_a$	81.79	93.20	57.37	21.15	87.20	51.71	41.44	64.98	62.36	80.79
	$\mathcal{E}_n$	81.44	76.92	73.34	18.50	73.15	52.35	41.34	71.50	61.07	72.05
i	$\mathcal{E}$	81.35	90.52	65.98	19.74	85.65	52.29	40.46	67.45	62.93	78.41
	$\mathcal{E}_a$	81.59	93.06	58.66	21.06	87.31	51.75	40.48	65.22	62.39	80.76
	$\mathcal{E}_n$	80.98	76.39	74.50	16.50	73.35	53.05	40.43	71.38	60.82	71.75
k	$\mathcal{E}$	81.61	91.15	64.45	14.43	94.36	60.77	41.96	80.02	66.09	83.13
	$\mathcal{E}_a$	82.02	93.55	56.88	14.67	95.32	62.24	40.34	80.01	65.63	85.69
	$\mathcal{E}_n$	80.98	78.13	73.22	13.85	87.56	58.69	44.66	80.00	64.64	75.88

Table 7.2: Point-level per-class $F1$-scores [%], $mF1$-scores, and OAs of *direct* semantic PC segmentation achieved with RF deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data set H3D. The RF models are trained on all points of train set $\mathcal{T}^{\mathrm{PC}}$ and evaluated on different subsets of the test set. The trained RF classifiers are evaluated on A) all points of test set $\mathcal{E}^{\mathrm{PC}}$, B) subset $\mathcal{E}_a$ containing all points that have been linked to the mesh via PCMA, and C) subset $\mathcal{E}_n^{\mathrm{PC}}$ containing all points that are not linked to the mesh. It holds: $\mathcal{E}^{\mathrm{PC}} = \mathcal{E}_a^{\mathrm{PC}} \cup \mathcal{E}_n^{\mathrm{PC}}$.

Hessigheim 3D

$\mathcal{FS}$	*Eval* (PC)	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	OA [%]
$\mathcal{T}^{\mathrm{PC}}$: **Trained on all Points**														
	$\mathcal{E}$	74.82	68.45	33.65	32.02	86.50	70.64	53.81	92.31	16.66	37.11	61.55	57.05	71.60
e	$\mathcal{E}_a$	74.75	69.19	33.99	31.64	86.45	72.48	56.05	90.94	16.11	38.40	64.41	57.67	70.94
	$\mathcal{E}_n$	75.50	55.78	32.77	33.13	86.80	62.35	48.10	94.06	22.47	26.97	50.66	53.51	75.65
	$\mathcal{E}$	78.86	78.75	44.64	39.38	88.01	75.55	59.51	94.14	18.71	51.31	66.56	63.22	76.40
e'	$\mathcal{E}_a$	78.97	79.36	46.26	39.26	87.96	77.71	61.91	93.22	18.29	53.98	70.09	64.27	76.04
	$\mathcal{E}_n$	77.95	68.34	40.46	39.76	88.29	65.97	53.39	95.29	23.19	29.00	53.40	57.73	78.66
	$\mathcal{E}$	77.55	71.16	38.78	35.24	94.26	74.44	56.13	92.74	27.70	52.40	73.39	63.07	76.11
f	$\mathcal{E}_a$	77.38	71.84	39.67	34.22	94.27	76.46	57.45	91.12	27.47	54.32	74.35	63.51	75.58
	$\mathcal{E}_n$	79.01	59.55	36.25	38.45	94.24	65.67	52.52	94.80	29.78	35.92	70.08	59.66	79.36
	$\mathcal{E}$	81.50	82.20	50.16	42.31	95.85	77.29	61.24	94.37	26.19	57.30	78.44	67.90	80.58
f'	$\mathcal{E}_a$	81.59	82.82	51.75	41.61	95.95	79.45	63.09	93.36	26.11	59.90	80.82	68.77	80.41
	$\mathcal{E}_n$	80.80	71.49	45.85	44.39	95.15	67.84	56.33	95.64	26.97	34.93	69.86	62.66	81.63
	$\mathcal{E}$	44.53	29.02	11.08	10.11	45.17	15.77	7.95	69.48	25.28	1.59	18.64	25.33	37.20
g	$\mathcal{E}_a$	45.04	29.39	10.26	9.04	43.94	15.54	5.59	63.38	24.75	1.55	17.73	24.20	35.59
	$\mathcal{E}_n$	39.61	24.06	13.68	14.80	53.75	17.02	18.56	78.38	30.23	1.91	21.94	28.54	47.05
	$\mathcal{E}$	50.27	29.12	18.44	15.37	46.14	21.43	11.36	72.53	25.76	2.54	21.82	28.62	41.05
g'	$\mathcal{E}_a$	50.71	29.37	18.01	13.62	44.91	21.53	7.90	66.64	25.59	2.66	21.79	27.52	39.48
	$\mathcal{E}_n$	46.04	25.83	19.70	21.45	54.98	20.91	24.00	80.96	27.50	1.84	21.91	31.37	50.60
	$\mathcal{E}$	77.87	72.02	54.10	38.48	93.94	74.81	58.16	92.90	34.07	50.72	75.18	65.66	76.88
h	$\mathcal{E}_a$	77.67	72.69	55.82	37.38	93.96	76.65	59.12	91.31	33.98	52.18	74.94	65.97	76.34
	$\mathcal{E}_n$	79.59	60.97	49.37	41.76	93.81	66.87	55.51	94.94	34.96	38.14	75.98	62.90	80.16
	$\mathcal{E}$	81.07	82.04	61.37	44.78	95.71	76.76	64.11	94.66	27.78	54.29	80.27	69.35	80.61
h'	$\mathcal{E}_a$	81.07	82.66	63.62	43.86	95.82	78.75	65.65	93.74	27.72	56.12	81.73	70.07	80.37
	$\mathcal{E}_n$	81.08	71.35	55.22	47.48	94.96	68.02	60.00	95.83	28.39	37.70	75.18	65.02	82.12
	$\mathcal{E}$	77.53	71.54	54.58	39.55	93.62	74.51	58.61	93.42	32.28	44.90	75.42	65.09	76.51
i	$\mathcal{E}_a$	77.30	72.18	55.52	38.70	93.62	76.27	59.73	91.93	32.13	45.96	75.46	65.34	75.90
	$\mathcal{E}_n$	79.53	60.97	52.05	42.21	93.64	66.91	55.56	95.32	33.69	36.21	75.28	62.85	80.20
	$\mathcal{E}$	81.00	81.66	60.74	45.31	95.60	77.00	64.42	94.87	29.19	54.98	80.30	69.55	80.59
i'	$\mathcal{E}_a$	81.00	82.27	62.60	44.41	95.66	78.95	65.81	93.92	29.23	56.98	82.24	70.28	80.32
	$\mathcal{E}_n$	81.06	71.22	55.68	48.02	95.13	68.50	60.69	96.08	28.88	36.78	73.43	65.04	82.27
	$\mathcal{E}$	89.99	87.48	55.79	45.90	95.25	76.33	62.24	94.46	45.76	53.82	72.32	70.85	86.25
k	$\mathcal{E}_a$	90.58	88.48	56.63	46.04	95.38	78.11	63.63	93.44	47.47	55.52	72.62	71.63	86.78
	$\mathcal{E}_n$	85.22	70.66	53.58	45.41	94.33	68.75	58.46	95.75	30.50	38.34	71.32	64.76	83.05
	$\mathcal{E}$	89.86	88.02	63.43	49.04	95.57	77.71	66.31	95.42	47.03	55.80	79.39	73.42	86.76
k'	$\mathcal{E}_a$	90.48	88.85	65.38	48.68	95.62	79.57	67.92	94.80	48.62	57.66	80.77	74.39	87.21
	$\mathcal{E}_n$	84.95	73.74	58.09	50.12	95.24	69.57	61.97	96.21	32.60	38.76	74.58	66.90	84.03

Table 7.3: Point-level per-class $F1$-scores [%], $mF1$-scores, and OAs of *direct* semantic PC segmentation achieved with RF deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data set H3D. The RF models are trained on the subset of linked points $\mathcal{T}_a^{\mathrm{PC}}$ and evaluated on different subsets of the test set. The trained RF classifiers are evaluated on A) all points of test set $\mathcal{E}^{\mathrm{PC}}$, B) subset $\mathcal{E}_a^{\mathrm{PC}}$ containing all points that have been linked to the mesh via PCMA, and C) subset $\mathcal{E}_n^{\mathrm{PC}}$ containing all points that are not linked to the mesh. It holds: $\mathcal{E}^{\mathrm{PC}} = \mathcal{E}_a^{\mathrm{PC}} \cup \mathcal{E}_n^{\mathrm{PC}}$.

Hessigheim 3D

$\mathcal{FS}$	*Eval* (PC)	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	OA [%]
$\mathcal{T}_a^{\mathrm{PC}}$: **Trained on Associated Points**														
	$\mathcal{E}$	75.06	68.58	32.37	32.10	86.60	70.68	53.09	91.17	15.27	40.73	57.79	56.68	71.82
e	$\mathcal{E}_a$	75.28	69.36	33.70	31.67	86.67	72.80	55.89	90.23	14.74	42.55	60.82	57.61	71.44
	$\mathcal{E}_n$	73.05	54.95	28.63	33.27	86.16	61.09	45.95	92.37	20.67	26.46	46.05	51.70	74.16
	$\mathcal{E}$	79.31	78.66	43.12	38.43	87.95	74.81	58.67	93.19	16.14	47.44	63.43	61.92	76.51
e'	$\mathcal{E}_a$	79.67	79.30	45.56	38.67	88.01	77.16	61.88	92.60	15.75	49.86	67.35	63.26	76.40
	$\mathcal{E}_n$	76.16	67.42	36.30	37.80	87.52	64.40	50.41	93.92	20.24	28.08	48.62	55.53	77.23
	$\mathcal{E}$	77.59	70.79	38.76	36.26	94.23	74.36	55.87	92.21	23.74	55.99	70.66	62.77	76.07
f	$\mathcal{E}_a$	77.56	71.50	40.70	35.35	94.27	76.57	57.67	90.88	23.67	58.17	73.36	63.61	75.69
	$\mathcal{E}_n$	77.84	58.73	33.25	38.98	93.97	64.78	50.88	93.92	24.38	37.22	60.51	57.68	78.43
	$\mathcal{E}$	81.80	81.66	48.22	41.69	95.60	76.63	60.45	93.89	23.28	55.30	73.81	66.57	80.47
f'	$\mathcal{E}_a$	82.01	82.31	50.96	41.29	95.75	79.08	63.03	93.06	23.24	57.75	77.33	67.80	80.44
	$\mathcal{E}_n$	80.06	70.58	40.44	42.75	94.58	65.86	53.57	94.92	23.57	34.74	60.52	60.15	80.65
	$\mathcal{E}$	46.58	28.95	13.81	10.30	45.44	15.99	7.30	68.54	25.50	1.67	19.14	25.75	38.16
g	$\mathcal{E}_a$	47.27	29.34	13.20	9.25	44.18	15.79	5.88	62.23	25.04	1.64	19.67	24.86	36.69
	$\mathcal{E}_n$	39.77	23.77	15.58	14.86	54.29	17.16	14.18	78.00	29.85	1.90	17.07	27.86	47.13
	$\mathcal{E}$	51.33	29.01	21.64	15.68	46.26	21.60	8.97	71.41	25.83	2.54	23.43	28.88	41.46
g'	$\mathcal{E}_a$	51.95	29.27	21.69	14.16	45.04	21.73	7.78	65.32	25.77	2.67	24.21	28.14	40.07
	$\mathcal{E}_n$	45.17	25.49	21.53	20.73	55.00	20.96	14.26	80.36	26.45	1.74	20.95	30.24	49.94
	$\mathcal{E}$	77.78	71.63	52.63	38.18	94.04	74.40	58.26	92.41	30.49	51.29	68.54	64.51	76.66
h	$\mathcal{E}_a$	77.66	72.29	55.69	37.11	94.08	76.37	59.57	91.02	30.35	52.82	72.04	65.36	76.21
	$\mathcal{E}_n$	78.78	60.59	43.95	41.30	93.69	65.83	54.51	94.20	31.74	38.13	55.24	59.82	79.45
	$\mathcal{E}$	81.86	81.56	59.15	44.32	95.67	76.35	63.31	94.10	27.65	53.00	73.88	68.26	80.94
h'	$\mathcal{E}_a$	82.01	82.20	62.81	43.94	95.82	78.51	65.66	93.35	27.79	54.80	78.08	69.54	80.88
	$\mathcal{E}_n$	80.61	70.49	48.77	45.35	94.64	66.84	56.92	95.06	26.45	37.07	57.63	61.80	81.30
	$\mathcal{E}$	77.98	71.94	53.33	39.27	94.12	74.49	58.78	92.91	31.77	48.77	69.59	64.81	76.94
i	$\mathcal{E}_a$	77.85	72.61	55.68	38.40	94.20	76.39	60.28	91.55	31.62	49.99	72.97	65.59	76.48
	$\mathcal{E}_n$	79.06	60.88	46.55	41.81	93.64	66.20	54.59	94.64	33.09	38.52	56.84	60.53	79.74
	$\mathcal{E}$	81.48	81.55	59.16	44.68	95.58	76.66	63.22	94.23	26.52	55.50	76.32	68.63	80.72
i'	$\mathcal{E}_a$	81.60	82.18	62.32	44.20	95.72	78.89	65.48	93.46	26.55	57.52	80.52	69.86	80.62
	$\mathcal{E}_n$	80.50	70.80	49.97	46.00	94.66	66.87	57.08	95.20	26.22	37.48	60.17	62.27	81.36
	$\mathcal{E}$	90.06	87.59	55.43	44.51	95.12	76.30	60.78	94.01	45.97	57.45	67.37	70.42	86.27
k	$\mathcal{E}_a$	90.79	88.56	58.04	44.92	95.28	78.34	62.38	93.33	47.96	59.32	71.26	71.83	86.94
	$\mathcal{E}_n$	84.03	70.88	47.86	43.24	93.99	67.63	56.45	94.88	28.65	40.40	52.53	61.87	82.14
	$\mathcal{E}$	89.71	88.33	61.40	49.50	95.65	77.62	64.51	94.88	47.16	58.89	73.14	72.80	86.75
k'	$\mathcal{E}_a$	90.43	89.18	64.61	49.35	95.77	79.84	66.64	94.48	48.97	60.92	77.05	74.30	87.36
	$\mathcal{E}_n$	83.82	73.50	52.05	49.93	94.79	67.93	58.86	95.38	30.88	40.19	58.25	64.14	83.05

Table 7.4: Point-level per-class $F1$-scores [%], $mF1$-scores, and OAs of *direct* semantic PC segmentation achieved with RF deploying different feature vector compositions $\mathcal{FS}$ (see Table 6.1) for data set H3D. The RF models are trained on the subset of unlinked points $\mathcal{T}_n^{\mathrm{PC}}$ and evaluated on different subsets of the test set. The trained RF classifiers are evaluated on A) all points of test set $\mathcal{E}^{\mathrm{PC}}$, B) subset $\mathcal{E}_a^{\mathrm{PC}}$ containing all points that have been linked to the mesh via PCMA, and C) subset $\mathcal{E}_n^{\mathrm{PC}}$ containing all points that are not linked to the mesh. It holds: $\mathcal{E}^{\mathrm{PC}} = \mathcal{E}_a^{\mathrm{PC}} \cup \mathcal{E}_n^{\mathrm{PC}}$.

Hessigheim 3D

$\mathcal{FS}$	*Eval* (PC)	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	OA [%]
$\mathcal{T}_n^{\mathrm{PC}}$: **Trained on Non-Associated Points**														
	$\mathcal{E}$	69.91	68.15	27.09	29.11	81.33	69.61	53.66	91.02	22.85	48.57	44.91	55.11	67.95
e	$\mathcal{E}_a$	69.33	68.76	25.89	27.51	80.86	71.57	54.38	88.94	22.35	50.56	46.00	55.10	66.75
	$\mathcal{E}_n$	74.50	58.34	30.51	33.90	84.43	60.80	51.75	93.70	28.82	30.79	41.15	53.52	75.27
	$\mathcal{E}$	73.33	78.34	24.43	32.90	83.39	74.63	56.08	93.41	26.26	51.24	23.49	56.14	72.86
e'	$\mathcal{E}_a$	72.99	78.95	20.31	29.90	82.88	76.48	56.10	92.08	25.81	53.50	22.64	55.60	71.98
	$\mathcal{E}_n$	5.95	68.61	34.26	41.35	86.78	66.31	56.03	95.10	31.60	31.30	26.26	55.78	78.26
	$\mathcal{E}$	73.81	71.83	33.42	32.26	92.70	74.36	56.13	91.03	29.04	48.42	60.77	60.34	73.50
f	$\mathcal{E}_a$	73.31	72.39	33.41	30.82	92.69	76.38	56.85	88.65	28.79	49.78	61.82	60.44	72.64
	$\mathcal{E}_n$	77.68	62.58	33.61	36.92	92.84	65.49	54.14	94.15	31.68	36.37	57.18	58.42	78.72
	$\mathcal{E}$	76.47	81.99	39.06	35.69	95.17	76.14	61.08	93.86	32.28	49.65	41.91	62.12	77.68
f'	$\mathcal{E}_a$	76.17	82.58	38.02	33.20	95.26	78.01	62.06	92.58	32.07	51.31	42.05	62.12	77.12
	$\mathcal{E}_n$	78.75	72.15	41.96	42.87	94.45	67.81	58.45	95.49	34.55	34.89	41.45	60.26	81.12
	$\mathcal{E}$	34.45	22.39	5.23	8.67	39.47	12.90	6.93	68.38	25.09	0.91	16.77	21.93	31.22
g	$\mathcal{E}_a$	34.40	22.69	4.32	7.47	38.33	12.48	4.34	61.82	24.68	0.83	14.89	20.57	29.04
	$\mathcal{E}_n$	34.94	18.63	8.91	14.28	47.36	15.44	19.65	78.43	29.27	1.61	23.04	26.51	44.54
	$\mathcal{E}$	36.73	24.42	12.16	12.74	40.82	17.06	10.76	71.49	24.35	0.97	24.44	25.09	33.93
g'	$\mathcal{E}_a$	36.32	24.71	11.24	10.70	39.65	16.79	5.92	65.13	23.76	0.92	21.18	23.30	31.61
	$\mathcal{E}_n$	40.32	21.00	15.22	20.12	48.90	18.63	26.94	80.98	30.73	1.29	34.11	30.75	48.07
	$\mathcal{E}$	73.61	71.73	46.81	33.56	92.73	74.31	57.44	91.68	35.42	38.55	67.40	62.11	73.98
h	$\mathcal{E}_a$	72.97	72.22	48.12	32.06	92.68	76.21	57.92	89.55	35.06	39.12	66.99	62.08	73.03
	$\mathcal{E}_n$	78.65	63.91	43.17	38.14	93.02	66.01	56.06	94.45	39.40	33.71	68.70	61.38	79.75
	$\mathcal{E}$	74.53	81.73	52.35	36.60	95.08	76.35	61.72	94.20	32.86	49.50	65.17	65.46	77.04
h'	$\mathcal{E}_a$	73.96	82.31	53.19	33.85	95.14	78.10	62.26	93.03	32.62	50.92	65.02	65.49	76.32
	$\mathcal{E}_n$	78.85	72.12	50.19	44.44	94.61	68.57	60.26	95.68	35.68	36.51	65.64	63.87	81.47
	$\mathcal{E}$	72.98	71.80	49.77	36.10	93.01	74.46	57.94	92.57	35.91	30.28	66.25	61.92	73.85
i	$\mathcal{E}_a$	72.29	72.30	50.12	34.86	92.98	76.23	58.45	90.62	35.55	29.98	65.51	61.72	72.84
	$\mathcal{E}_n$	78.43	63.52	48.96	39.93	93.24	66.80	56.53	95.09	39.87	32.69	68.64	62.15	79.97
	$\mathcal{E}$	73.73	81.72	54.71	37.89	94.63	76.70	62.56	94.42	33.49	54.21	59.44	65.77	76.59
i'	$\mathcal{E}_a$	73.11	82.32	55.39	35.09	94.71	78.45	63.16	93.29	33.20	55.97	58.47	65.74	75.81
	$\mathcal{E}_n$	78.42	71.91	53.02	45.78	94.00	68.95	60.93	95.84	36.99	37.60	62.55	64.18	81.34
	$\mathcal{E}$	89.52	85.45	47.35	41.49	93.99	75.00	61.40	93.68	45.86	35.53	61.76	66.46	84.96
k	$\mathcal{E}_a$	90.04	86.44	46.32	41.36	94.06	76.68	62.16	92.28	47.28	35.74	61.75	66.74	85.32
	$\mathcal{E}_n$	85.37	69.26	50.63	41.85	93.56	67.75	59.26	95.50	32.95	33.80	61.79	62.88	82.78
	$\mathcal{E}$	89.11	87.36	55.60	41.59	94.72	76.30	64.49	94.98	47.60	45.73	59.04	68.77	85.92
k'	$\mathcal{E}_a$	89.67	88.26	56.23	39.78	94.73	77.80	65.29	94.10	48.96	46.88	58.48	69.11	86.27
	$\mathcal{E}_n$	84.71	72.40	54.09	46.88	94.61	69.64	62.35	96.09	35.25	35.38	60.86	64.75	83.80

Table 7.5: Surface-aware per-class $F1$-scores [%], $mF1$-scores, and OAs of *indirect* semantic segmentation of meshes achieved with RF models deploying different feature sets $\mathcal{FS}$ (see Table 6.1) for data sets V3D (*top*) and H3D (bottom). Configurations on H3D that consider PC features based on small-scale radii are marked with a prime and highlighted in *gray*. The RF models are trained on different subsets of the train set (PC). The classifiers are trained on a) all points of train set $\mathcal{T}^{\mathrm{PC}}$, b) subset $\mathcal{T}_a^{\mathrm{PC}}$ containing all points that have been linked to the mesh via PCMA, and c) subset $\mathcal{T}_n^{\mathrm{PC}}$ containing all points that are not linked to the mesh. The trained RF classifiers are evaluated on faces with a valid GT label, i.e., on subset $\mathcal{E}_a^{\mathrm{Mesh}}$ containing all faces that have been linked to the PC via PCMA.

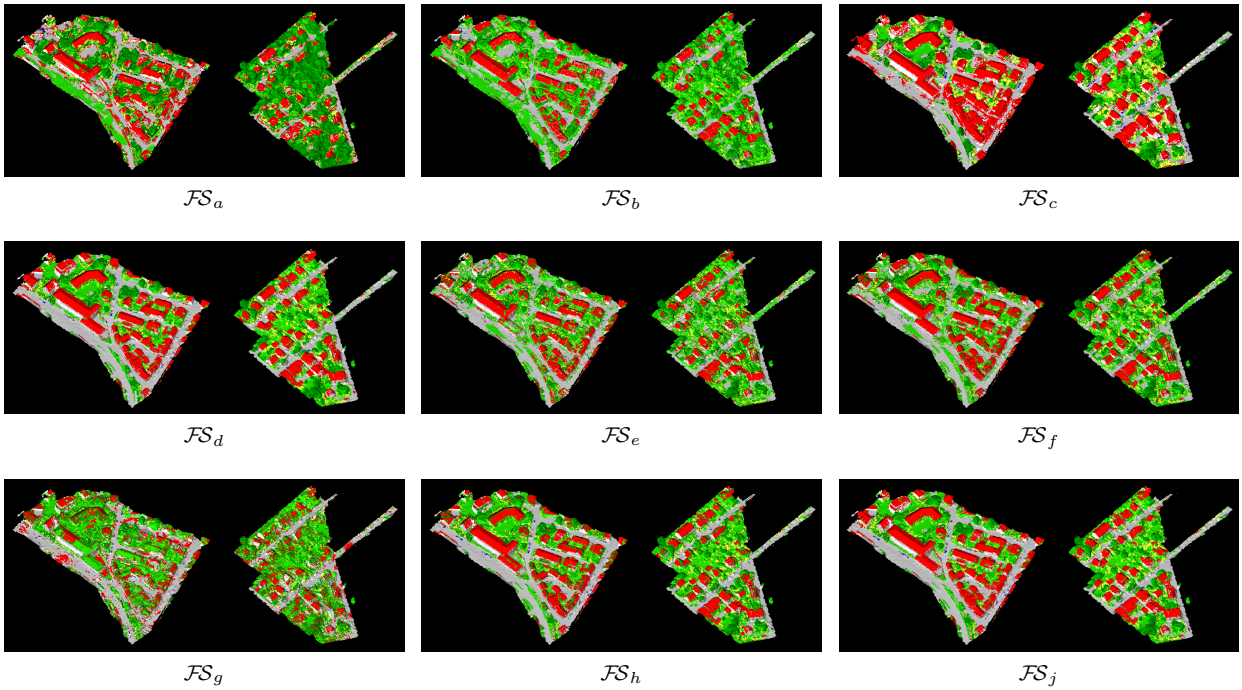
Vaihingen 3D ($\mathcal{T}_X^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{Mesh}}$)

$\mathcal{FS}$	*Train* (PC)	*Low Veg.*	*I. Surf.*	*Car*	*Fence/ Hedge*	*Roof*	*Facade*	*Shrub*	*Tree*	$mF1$ [%]	OA [%]
e	$\mathcal{T}$	50.70	73.51	43.43	14.94	79.47	63.46	28.02	66.46	52.50	65.64
	$\mathcal{T}_a$	50.57	71.42	37.13	14.51	80.35	63.12	28.59	66.76	51.56	65.40
	$\mathcal{T}_n$	49.88	77.88	33.90	14.08	77.76	62.80	24.95	65.58	50.85	66.45
f	$\mathcal{T}$	60.20	75.59	45.86	18.23	88.69	64.45	39.85	68.72	57.70	72.07
	$\mathcal{T}_a$	59.29	73.61	37.42	17.15	88.86	64.23	39.72	68.90	56.15	71.38
	$\mathcal{T}_n$	60.48	79.75	39.22	17.56	87.44	62.09	39.22	67.22	56.62	72.84
g	$\mathcal{T}$	40.55	75.82	16.48	17.80	37.93	11.20	14.67	29.01	30.43	44.38
	$\mathcal{T}_a$	40.57	75.43	13.17	17.54	38.59	12.50	14.28	29.68	30.22	44.70
	$\mathcal{T}_n$	38.54	78.57	19.67	17.12	26.89	7.94	13.13	27.64	28.69	41.84
h	$\mathcal{T}$	78.30	93.12	65.62	18.65	88.55	63.23	41.91	68.40	64.72	80.19
	$\mathcal{T}_a$	77.90	93.12	64.45	17.76	88.77	63.13	41.59	68.73	64.43	80.22
	$\mathcal{T}_n$	78.38	92.92	63.59	17.86	87.12	60.63	40.73	66.42	63.46	79.42
i	$\mathcal{T}$	78.22	93.13	64.17	17.42	88.51	63.56	41.75	68.29	64.38	80.10
	$\mathcal{T}_a$	77.73	93.12	63.27	17.75	88.59	63.33	41.63	68.70	64.27	80.13
	$\mathcal{T}_n$	78.33	92.87	64.40	18.62	87.12	61.62	39.69	66.60	63.66	79.42
k	$\mathcal{T}$	79.54	93.69	65.37	11.57	95.74	72.76	41.07	82.05	67.72	85.39
	$\mathcal{T}_a$	79.10	93.68	63.54	10.13	95.66	72.48	40.54	81.99	67.14	85.26
	$\mathcal{T}_n$	79.10	93.46	62.57	12.03	95.31	72.80	39.30	81.19	66.97	85.06

Hessigheim 3D ($\mathcal{T}_X^{\mathrm{PC}}, \mathcal{E}_a^{\mathrm{Mesh}}$)

$\mathcal{FS}$	*Train* (PC)	*Low Veg.*	*I. Surf.*	*Vehicle*	*U. Furn.*	*Roof*	*Facade*	*Shrub*	*Tree*	*Gravel/ Soil*	*V. Surf.*	*Chimney*	$mF1$ [%]	OA [%]
e	$\mathcal{T}$	73.71	69.86	33.85	37.28	84.20	76.06	58.10	92.96	15.63	42.20	61.09	58.63	72.99
	$\mathcal{T}_a$	74.36	70.04	33.89	37.83	84.15	76.13	58.48	92.80	14.47	45.66	55.28	58.46	73.44
	$\mathcal{T}_n$	67.07	69.28	25.24	32.26	79.63	76.92	55.38	91.80	20.89	51.02	48.00	56.14	69.89
e'	$\mathcal{T}$	78.48	78.69	41.98	43.51	86.56	79.95	62.78	94.34	19.05	57.98	65.08	64.40	77.60
	$\mathcal{T}_a$	79.13	78.59	42.46	43.07	86.35	79.62	63.60	94.19	16.90	56.66	61.83	63.85	77.79
	$\mathcal{T}_n$	71.50	78.14	23.84	37.52	82.29	79.07	58.15	93.47	24.69	56.83	25.45	57.36	74.29
f	$\mathcal{T}$	76.17	71.25	37.13	40.51	89.94	78.40	58.58	93.25	22.30	49.48	70.48	62.50	76.07
	$\mathcal{T}_a$	76.55	70.94	38.68	42.24	89.84	78.71	59.71	93.27	19.41	54.22	68.27	62.89	76.44
	$\mathcal{T}_n$	71.42	71.34	28.95	35.41	88.60	78.54	57.59	92.14	24.82	48.97	62.30	60.01	73.71
f'	$\mathcal{T}$	80.87	81.91	46.10	45.94	93.07	80.54	63.52	94.46	23.54	55.79	76.84	67.51	80.48
	$\mathcal{T}_a$	81.44	81.34	46.18	46.15	92.75	80.47	64.13	94.44	21.33	54.44	72.69	66.85	80.55
	$\mathcal{T}_n$	74.94	81.87	34.60	39.47	92.85	79.52	61.79	93.90	28.67	50.40	41.18	61.75	77.87
g	$\mathcal{T}$	42.46	26.80	12.82	16.90	43.38	29.98	8.96	77.86	18.72	3.85	22.12	27.62	40.03
	$\mathcal{T}_a$	44.79	26.61	16.11	18.08	43.28	30.14	10.18	77.90	19.14	4.09	24.51	28.62	41.24
	$\mathcal{T}_n$	30.03	22.13	7.71	14.08	38.66	26.97	6.99	77.29	17.97	2.32	14.55	23.52	34.99
g'	$\mathcal{T}$	46.50	26.81	21.11	21.61	43.91	36.90	12.85	80.37	18.40	5.83	20.04	30.39	43.03
	$\mathcal{T}_a$	47.92	26.61	24.06	23.31	44.04	36.84	14.12	80.24	18.71	5.92	21.43	31.20	43.86
	$\mathcal{T}_n$	30.52	23.59	16.25	17.22	39.58	32.64	8.39	79.07	16.76	2.17	17.60	25.80	36.68
h	$\mathcal{T}$	76.27	72.20	52.27	42.33	89.78	79.05	59.49	93.46	28.41	48.47	68.31	64.55	76.75
	$\mathcal{T}_a$	76.42	71.91	52.15	42.11	89.85	78.63	61.21	93.40	26.50	48.76	64.23	64.11	76.80
	$\mathcal{T}_n$	69.34	71.07	45.44	36.94	88.63	78.98	57.55	92.68	27.09	40.87	63.60	61.11	73.37
h'	$\mathcal{T}$	80.27	82.19	57.86	48.01	93.13	80.22	65.53	94.65	26.13	52.09	76.90	68.82	80.69
	$\mathcal{T}_a$	81.19	81.78	57.08	47.95	93.05	80.12	66.12	94.56	26.41	51.86	72.71	68.44	81.01
	$\mathcal{T}_n$	71.21	82.07	49.07	40.78	93.17	79.87	61.96	94.12	27.76	50.67	64.91	65.05	77.14
i	$\mathcal{T}$	75.83	71.35	52.42	43.48	89.03	78.68	60.71	93.89	27.26	43.65	68.89	64.11	76.35
	$\mathcal{T}_a$	76.63	72.17	52.80	44.05	89.93	78.63	61.80	93.76	27.00	47.16	65.87	64.53	77.05
	$\mathcal{T}_n$	68.96	70.96	46.41	39.78	89.08	78.67	58.23	93.35	27.66	33.15	62.61	60.81	73.37
i'	$\mathcal{T}$	80.11	81.63	57.30	48.26	92.87	80.20	66.04	94.84	26.93	51.96	77.42	68.87	80.57
	$\mathcal{T}_a$	80.83	81.71	56.99	48.19	92.85	80.47	66.23	94.65	25.50	54.03	75.31	68.79	80.92
	$\mathcal{T}_n$	70.53	82.03	51.37	41.98	92.69	79.75	62.35	94.32	27.69	52.97	59.26	64.99	76.89
k	$\mathcal{T}$	89.10	88.53	53.12	49.92	92.29	79.95	64.49	95.13	47.22	51.85	65.58	70.65	85.03
	$\mathcal{T}_a$	89.18	88.66	54.24	48.74	92.04	80.13	64.37	95.16	47.27	55.84	64.14	70.89	85.14
	$\mathcal{T}_n$	88.04	86.36	43.05	44.84	91.34	79.16	61.89	94.52	46.63	38.37	57.86	66.55	83.44
k'	$\mathcal{T}$	89.00	88.71	58.92	52.81	93.02	80.80	67.21	95.75	46.88	55.23	76.17	73.14	85.59
	$\mathcal{T}_a$	88.95	89.09	58.48	54.16	93.14	81.17	67.35	95.74	47.95	57.65	70.93	73.15	85.85
	$\mathcal{T}_n$	87.76	88.50	50.26	45.50	92.82	79.34	64.44	95.07	47.54	46.96	56.93	68.65	84.55

Vaihingen 3D – Predictions

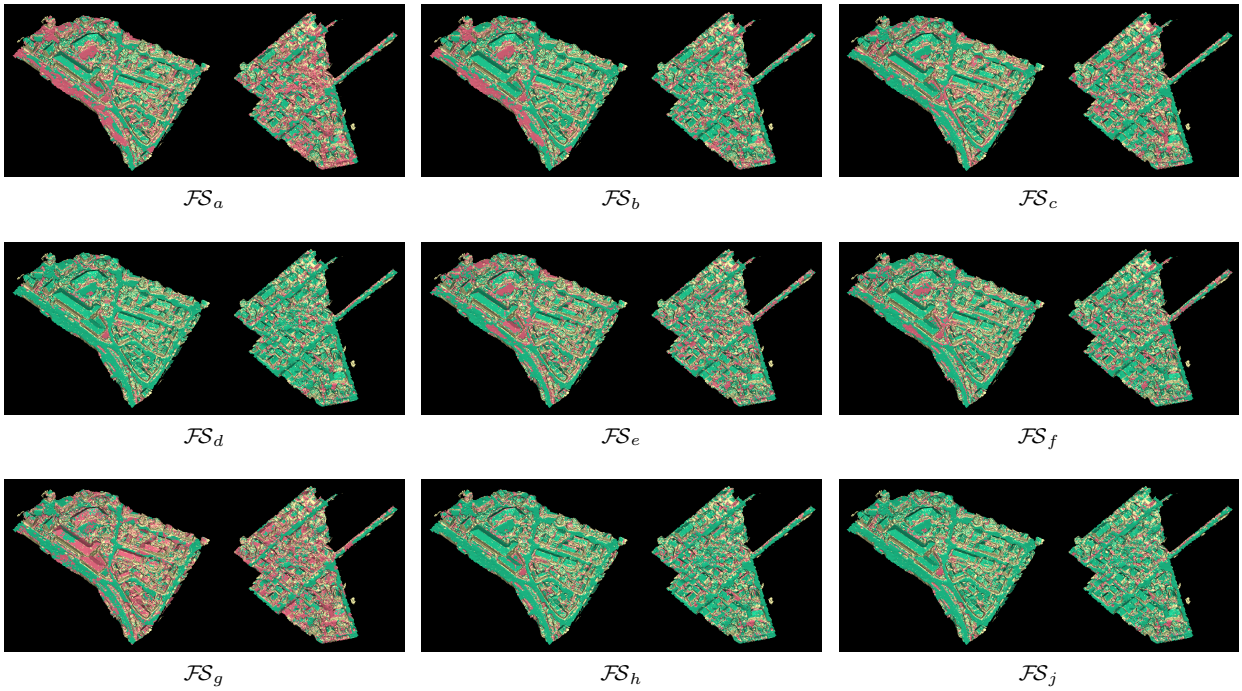


$\mathcal{FS}_a$ $\mathcal{FS}_b$ $\mathcal{FS}_c$

$\mathcal{FS}_d$ $\mathcal{FS}_e$ $\mathcal{FS}_f$

$\mathcal{FS}_g$ $\mathcal{FS}_h$ $\mathcal{FS}_j$

Vaihingen 3D – Difference Plots



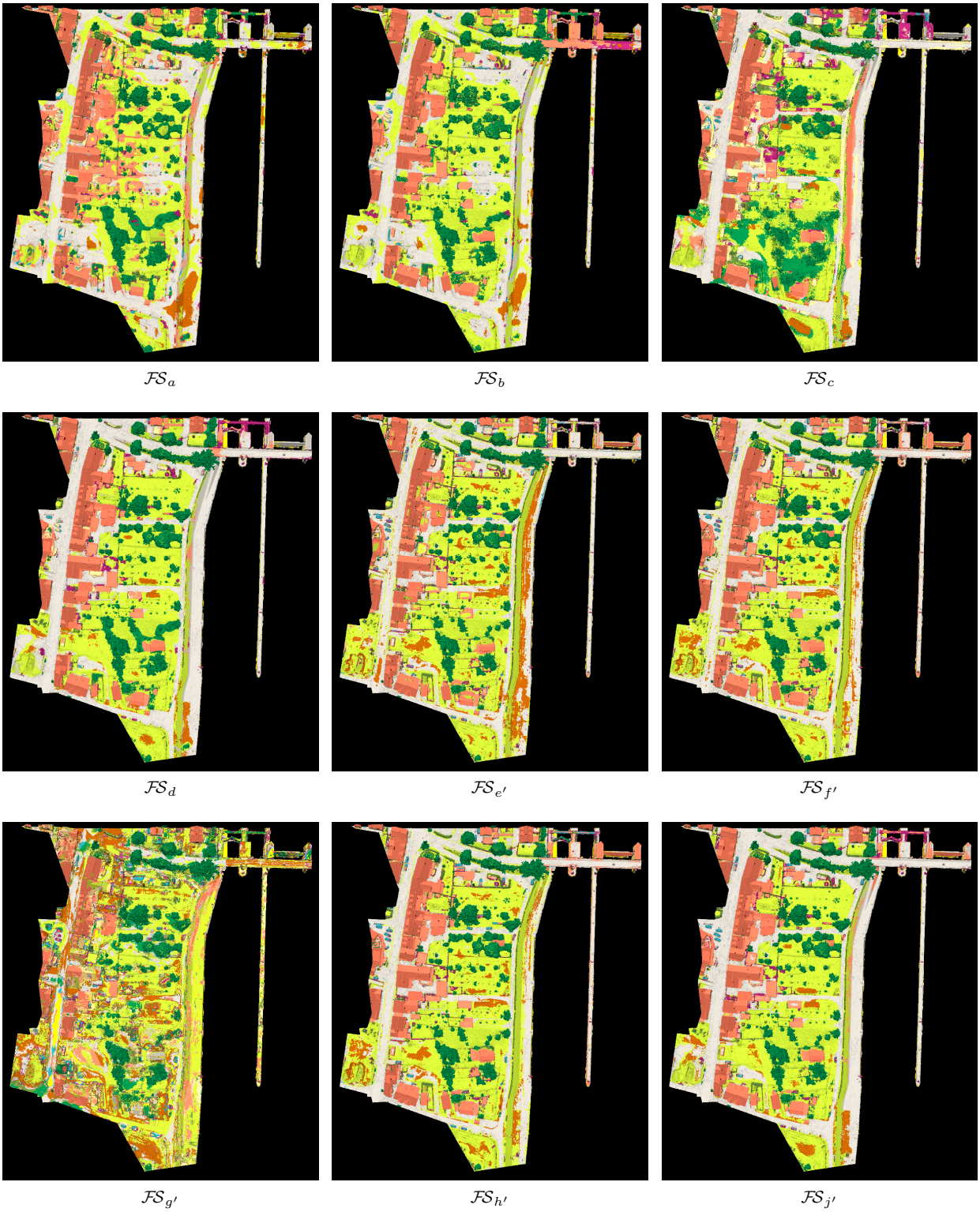
$\mathcal{FS}_a$ $\mathcal{FS}_b$ $\mathcal{FS}_c$

$\mathcal{FS}_d$ $\mathcal{FS}_e$ $\mathcal{FS}_f$

$\mathcal{FS}_g$ $\mathcal{FS}_h$ $\mathcal{FS}_j$

Figure 7.3: Predictions (*top*) and respective difference plots (*bottom*) of trained RF models deploying different feature vector compositions (see Table 6.1) on the test set of the V3D mesh. The difference plots show correct predictions in *green*; false predictions are shown in *red*. Faces with unknown ground truth are marked in *yellow*.

Hessigheim 3D – Predictions



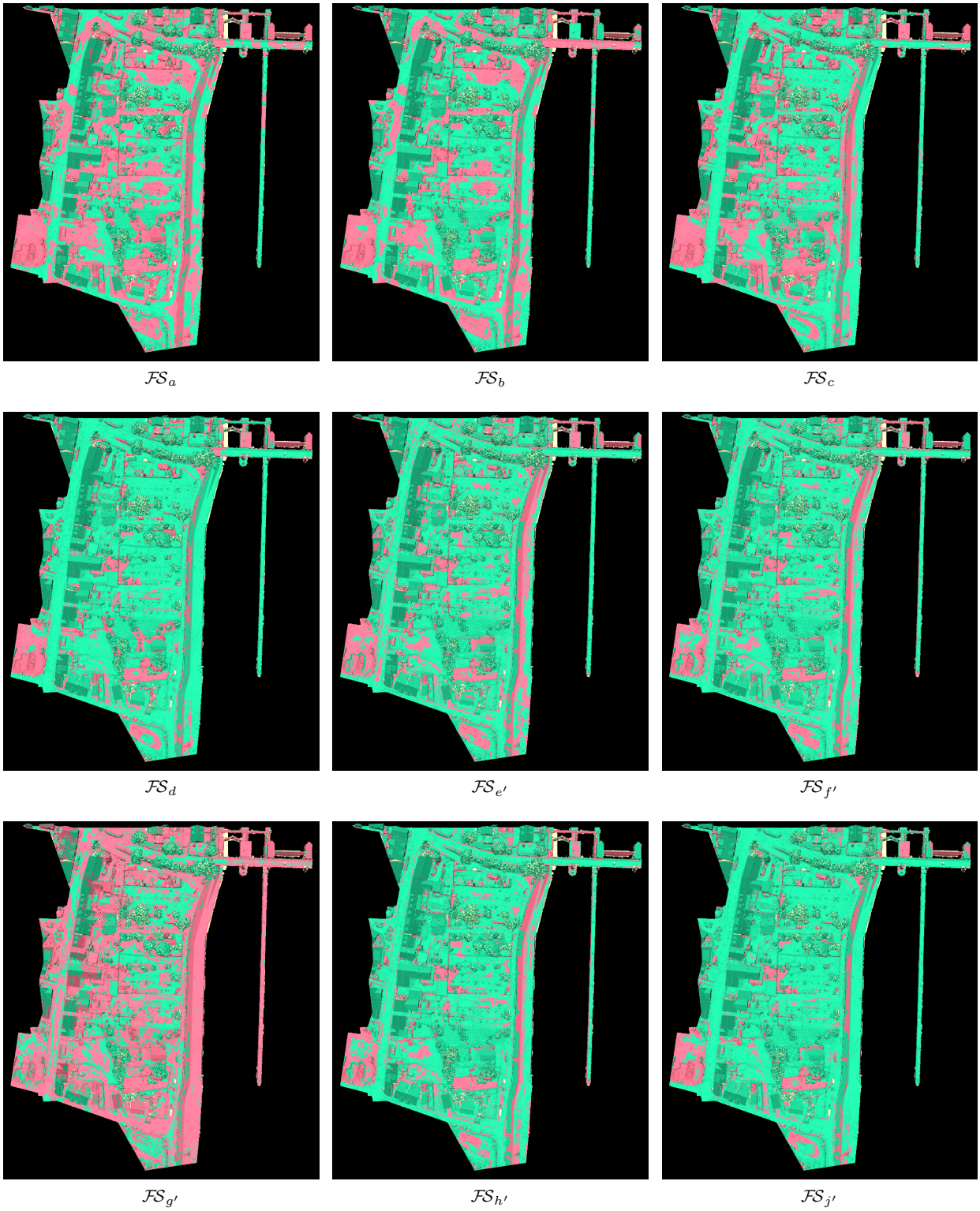
$\mathcal{FS}_a$ $\mathcal{FS}_b$ $\mathcal{FS}_c$

$\mathcal{FS}_d$ $\mathcal{FS}_{e'}$ $\mathcal{FS}_{f'}$

$\mathcal{FS}_{g'}$ $\mathcal{FS}_{h'}$ $\mathcal{FS}_{j'}$

Figure 7.4: Predictions of trained RF models deploying different feature vector compositions (see Table 6.1) on the test set of the H3D mesh.

Hessigheim 3D – Difference Plots



$\mathcal{FS}_a$ $\mathcal{FS}_b$ $\mathcal{FS}_c$

$\mathcal{FS}_d$ $\mathcal{FS}_{e'}$ $\mathcal{FS}_{f'}$

$\mathcal{FS}_{g'}$ $\mathcal{FS}_{h'}$ $\mathcal{FS}_{j'}$

Figure 7.5: Difference plots of the test set of the H3D mesh. The predictions are made with RF models deploying different feature vector compositions (see Table 6.1). Correct predictions are shown in *green*; false predictions in *red*. Faces with unknown ground truth are marked in *yellow*.

Bibliography

Actueel Hoogtebestand Nederland (AHN) (2021). AHN quality description [www document]. `https://www.ahn.nl/kwaliteitsbeschrijving` (accessed 17.08.2021).

Agisoft Metashape (2018). Agisoft Metashape Professional (Version 1.4.2) (Software). Retrieved from `http://www.agisoft.com/downloads/installer/`.

Ahmed, E., Saint, A., Shabayek, A. E. R., Cherenkova, K., Das, R., Gusev, G., Aouada, D., and Ottersten, B. E. (2018). Deep learning advances on different 3D data representations: A survey. *CoRR*, abs/1808.01462.

Ali Khan, S., Shi, Y., Shahzad, M., and Xiang Zhu, X. (2020). FGCN: Deep feature-based graph convolutional network for semantic segmentation of urban 3D point clouds. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 778–787.

Armeni, I., Sax, A., Zamir, A. R., and Savarese, S. (2017). Joint 2D-3D-semantic data for indoor scene understanding. *ArXiv e-prints*.

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., and Savarese, S. (2016). 3D semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*.

Autodesk (2019). Autodesk 3ds Max - 3D modeling and rendering software for design visualization, games, and animation. Retrieved from: `https://www.autodesk.com/products/3ds-max/overview` (accessed 24th April 2019).

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495.

Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., and Gall, J. (2019). SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*.

Bello, S. A., Yu, S., Wang, C., Adam, J. M., and Li, J. (2020). Review: Deep learning on 3D point clouds. *Remote Sensing*, 12(11).

Bentley (2017). ContextCapture (Software). Retrieved from `http://www.bentley.com/products/brands/contextcapture`.

Biljecki, F., Ledoux, H., and Stoter, J. (2016). Generation of multi-LOD 3D city models in CityGML with the procedural modelling engine Random3Dcity. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, pages 51–59.

Blomley, R. and Weinmann, M. (2017). Using multi-scale features for the 3D semantic labeling of airborne laser scanning data. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W4:43–50.

Bláha, M., Rothermel, M., Oswald, M. R., Sattler, T., Richard, A., Wegner, J. D., Pollefeys, M., and Schindler, K. (2017). Semantically informed multiview surface refinement. *CoRR*, abs/1706.08336.

Boulch, A. (2019). Generalizing discrete convolutions for unstructured point clouds. *CoRR*, abs/1904.02375.

Boulch, A., Saux, B. L., and Audebert, N. (2017). Unstructured point cloud semantic labeling using deep segmentation networks. In Pratikakis, I., Dupont, F., and Ovsjanikov, M., editors, *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association.

Boussaha, M., Vallet, B., and Rives, P. (2018). Large scale textured mesh reconstruction from mobile mapping images and LiDAR scans. In *ISPRS 2018 - International Society for Photogrammetry and Remote Sensing*, pages 49–56.

Breiman, L. (2001). Random forests. *Mach. Learn.*, 45(1):5–32.

Brodu, N. and Lague, D. (2012). 3D terrestrial LiDAR data classification of complex natural scenes using a multi-scale dimensionality criterion: Applications in geomorphology. *ISPRS Journal of Photogrammetry and Remote Sensing*, 68:121–134.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2016). Geometric deep learning: Going beyond Euclidean data. *CoRR*, abs/1611.08097.

Buhrmester, M., Kwang, T., and Gosling, S. D. (2011). Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 6(1):3–5.

Cabezas, R., Straub, J., and Fisher, J. W. (2015). Semantically-aware aerial reconstruction from multi-modal data. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2156–2164.

Can, G., Mantegazza, D., Abbate, G., Chappuis, S., and Giusti, A. (2021). Semantic segmentation on Swiss3DCities: A benchmark study on aerial photogrammetric 3D pointcloud dataset. *Pattern Recognition Letters*, 150:108–114.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Nießner, M., Savva, M., Song, S., Zeng, A., and Zhang, Y. (2017). Matterport3D: Learning from RGB-D data in indoor environments. arxiv:1709.06158.

Chang, A. X., Funkhouser, T. A., Guibas, L. J., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., and Yu, F. (2015). ShapeNet: An information-rich 3D model repository. *CoRR*, abs/1512.03012.

Chang, J., Gu, J., Wang, L., Meng, G., Xiang, S., and Pan, C. (2018). Structure-aware convolutional neural networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 11–20. Curran Associates, Inc.

Chehata, N., Guo, L., and Mallet, C. (2009). Airborne LIDAR feature selection for urban classification using random forests. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38.

Chiang, H., Lin, Y., Liu, Y., and Hsu, W. H. (2019). A unified point-based framework for 3D segmentation. *CoRR*, abs/1908.00478.

Cramer, M. (2010). The DGPF-test on digital airborne camera evaluation overview and test design. *PFG Photogrammetrie, Fernerkundung, Geoinformation*, 2010(2):73–82.

Cramer, M., Haala, N., Laupheimer, D., Mandlburger, G., and Havel, P. (2018). Ultra-high precision UAV-based LiDAR and dense image matching. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-1:115–120.

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T. A., and Nießner, M. (2017). ScanNet: Richly-annotated 3D reconstructions of indoor scenes. *CoRR*, abs/1702.04405.

Dai, A. and Nießner, M. (2018). 3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation. *CoRR*, abs/1803.10409.

Demantké, J., Mallet, C., David, N., and Vallet, B. (2011). Dimensionality based scale selection in 3D LIDAR point clouds. In *Laserscanning*, Calgary, Canada.

Demantké, J., Vallet, B., and Paparoditis, N. (2012). Streamed vertical rectangle detection in terrestrial laser scans for facade database production. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, I-3:99–104.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Li, F. F. (2009). ImageNet: A large-scale hierarchical image database. In *CVPR 2009*, pages 248–255.

Douros, I. and Buxton, B. (2002). Three-dimensional surface curvature estimation using quadric surface patches. In *Scanning 2002 Proceedings*.

Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern Classification*. Wiley, New York, 2nd edition.

Dyn, N., Hormann, K., Kim, S.-J., and Levin, D. (2001). *Optimizing 3D Triangulations Using Discrete Curvature Analysis*, page 135–146. Vanderbilt University, USA, 1st edition.

Eitel, J., Höfle, B., Vierling, L., Abellán, A., Asner, G., Deems, J., Glennie, C., Joerg, P., LeWinter, A., Magney, T., Mandlburger, G., Morton, D., Müller, J., and Vierling, K. (2016). Beyond 3-D: The new spectrum of LiDAR applications for earth and ecological sciences. *Remote Sensing of Environment*, 186:372–392.

Epic Games (2019). Unreal engine. `https://www.unrealengine.com`.

Ericson, C. (2005). *Real-Time Collision Detection*. Elsevier, 1st edition.

Ertekin, S., Huang, J., Bottou, L., and Giles, L. (2007). Learning on the border: Active learning in imbalanced data

classification. In *CIKM 2007*, pages 127–136, New York, NY, USA. ACM.

ESRI (2021). ArcGIS CityEngine (Software). Retrieved from `https://www.esri.com/en-us/arcgis/products/arcgis-cityengine/overview`.

Fedorova, S., Tono, A., Nigam, M. S., Zhang, J., Ahmadnia, A., Bolognesi, C., and Michels, D. (2021). Synthetic data generation pipeline for geometric deep learning in architecture. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B2-2021:337–344.

Feng, Y., Feng, Y., You, H., Zhao, X., and Gao, Y. (2018). MeshNet: Mesh neural network for 3D shape representation. *AAAI 2019*.

Filin, S. and Pfeifer, N. (2005). Neighborhood systems for airborne laser data. *Photogrammetric Engineering and Remote Sensing*, 71(6):743–755.

Gadiraju, U., Kawase, R., Siehndel, P., and Fetahu, B. (2015). Breaking bad: Understanding behavior of crowd workers in categorization microtasks. In *HT 2015*, pages 33–38. ACM.

Gao, W., Nan, L., Boom, B., and Ledoux, H. (2021). SUM: A benchmark dataset of semantic urban meshes. *ISPRS Journal of Photogrammetry and Remote Sensing*, 179:108–120.

Gao, W., Nan, L., Boom, B., and Ledoux, H. (2022). PSSNet: Planarity-sensible semantic segmentation of large-scale urban meshes. *CoRR*, abs/2202.03209.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., and Rodríguez, J. G. (2017). A review on deep learning techniques applied to semantic segmentation. *CoRR*, abs/1704.06857.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361.

George, D., Xie, X., and Tam, G. K. L. (2017). 3D mesh segmentation via multi-branch 1D convolutional neural networks. *CoRR*, abs/1705.11050.

Girardeau-Montaut, D. (2021). CloudCompare - 3D point cloud and mesh processing software open source project. Version 2.11.0. Retrieved from: `https://www.danielgm.net/cc/` (accessed 2nd February 2021).

Glira, P., Pfeifer, N., and Mandlburger, G. (2019). Hybrid orientation of airborne LiDAR point clouds and aerial images. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W5:567–574.

Google (2001). 3D surface in Google Earth. `https://earth.google.com/web/` (last accessed: 30th March 2022).

Goutte, C. and Gaussier, E. (2005). A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In *Lecture Notes in Computer Science*, pages 345–359. Springer Berlin Heidelberg.

Graham, B., Engelcke, M., and van der Maaten, L. (2017). 3D semantic segmentation with submanifold sparse convolutional networks. *CoRR*, abs/1711.10275.

Griffiths, D. and Boehm, J. (2019). A review on deep learning techniques for 3D sensed data classification. *Remote Sensing*, 11(12).

Gröger, G., Kolbe, T., Nagel, C., and Häfele, K. (2012). OGC city geography markup language (CityGML) encoding standard, version 2.0, OGC doc no. 12-019. *Open Geospatial Consortium*.

Gross, H. and Thoennessen, U. (2006). Extraction of lines from laser point clouds.

Günther, M. (1989). M. Berger, B. Gostiaux. Differential geometry: Manifolds, curves and surfaces. Graduate Texts in Mathematics. Springer-Verlag New York-Berlin-Heidelberg-London-Paris-Tokyo 1988. 474 p., 249 ill., ISBN 0-387-96 626-9, ISBN 3-540-96 626-9. *Crystal Research and Technology*, 24(2):234–234.

Guo, L., Chehata, N., Mallet, C., and Boukir, S. (2011). Relevance of airborne LiDAR and multispectral image data for urban scene classification using random forests. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(1):56–66.

Haala, N., Kölle, M., Cramer, M., Laupheimer, D., Mandlburger, G., and Glira, P. (2020). Hybrid georeferencing, enhancement and classification of ultra-high resolution UAV LiDAR and image point clouds for monitoring applications. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-2-2020:727–734.

Haala, N., Kölle, M., Cramer, M., Laupheimer, D., and Zimmermann, F. (2022). Hybrid georeferencing of images and LiDAR data for UAV-based point cloud collection at millimetre accuracy. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 4:100014.

Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K., and Pollefeys, M. (2017). semantic3D.net: A new large-scale point cloud classification benchmark. In *ISPRS Annals of the Photogrammetry, Remote Sensing and*

Spatial Information Sciences, volume IV-1-W1, pages 91–98.

Hackel, T., Wegner, J. D., and Schindler, K. (2016). Fast semantic segmentation of 3D point clouds with strongly varying density. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, 3(3).

Han, X., Laga, H., and Bennamoun, M. (2019). Image-based 3D object reconstruction: State-of-the-art and trends in the deep learning era. *CoRR*, abs/1906.06543.

Handa, A., Patraucean, V., Badrinarayanan, V., Stent, S., and Cipolla, R. (2015). SceneNet: Understanding real world indoor scenes with synthetic data. *CoRR*, abs/1511.07041.

Hanocka, R., Hertz, A., Fish, N., Giryes, R., Fleishman, S., and Cohen-Or, D. (2019). MeshCNN: A network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):90.

Har'el, Z. (1995). Curvature of curves and surfaces – a parabolic approach. `https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.24.5154&rep=rep1&type=pdf`.

He, H. and Upcroft, B. (2013). Nonparametric semantic segmentation for 3D street scenes. In *IROS2013: IEEE/RSJ International Conference on Intelligent Robots and Systems: New Horizon*, Tokyo Big Sight, Tokyo, Japan.

Herfort, B., Höfle, B., and Klonner, C. (2018). 3D micro-mapping: Towards assessing the quality of crowdsourcing to support 3D point cloud analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 137:73–83.

Hirth, M., Hoßfeld, T., and Tran-Gia, P. (2011). Anatomy of a crowdsourcing platform - using the example of microworkers.com. In *IMIS 2011*, pages 322–329, Washington, DC, USA. IEEE Computer Society.

Hoppe, H. (1997). View-dependent refinement of progressive meshes. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, page 189–198, USA. ACM Press/Addison-Wesley Publishing Co.

Hu, Q., Yang, B., Khalid, S., Xiao, W., Trigoni, A., and Markham, A. (2021a). Towards semantic segmentation of urban-scale 3D point clouds: A dataset, benchmarks and challenges. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4975–4985.

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., and Markham, A. (2020). RandLA-Net: Efficient semantic segmentation of large-scale point clouds. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Hu, S., Liu, Z., Guo, M., Cai, J., Huang, J., Mu, T., and Martin, R. R. (2021b). Subdivision-based mesh convolution networks. *CoRR*, abs/2106.02285.

Hu, W., Zhao, H., Jiang, L., Jia, J., and Wong, T. (2021c). Bidirectional projection network for cross dimension scene understanding. *CoRR*, abs/2103.14326.

Hua, B., Pham, Q., Nguyen, D. T., Tran, M., Yu, L., and Yeung, S. (2016). SceneNN: A scene meshes dataset with aNNotations. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 92–101.

Huang, J. and You, S. (2016). Point cloud labeling using 3D convolutional neural network. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2670–2675.

Huang, J., Zhang, H., Yi, L., Funkhouser, T., Niessner, M., and Guibas, L. J. (2019). TextureNet: Consistent local parametrizations for learning from high-resolution signals on meshes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jaritz, M., Gu, J., and Su, H. (2019). Multi-view PointNet for 3D scene understanding. *CoRR*, abs/1909.13603.

Jutzi, B. and Gross, H. (2009). Nearest neighbour classification on laser point clouds to gain object structures from buildings. In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. ISPRS Hannover Workshop. High-Resolution Earth Imaging for Geospatial Information. Online proceedings*, volume XXXVIII-1-4-7.

Kalogerakis, E., Averkiou, M., Maji, S., and Chaudhuri, S. (2017). 3D shape segmentation with projective convolutional networks. *CoRR*, abs/1612.02808.

Kalogerakis, E., Hertzmann, A., and Singh, K. (2010). Learning 3D mesh segmentation and labeling. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH 2010, pages 102:1–102:12, New York, NY, USA. ACM.

Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP 2006, pages 61–70, Aire-la-Ville, Switzerland. Eurographics Association.

Kim, S., Chi, H., Hu, X., Huang, Q., and Ramani, K. (2020). A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks. In Vedaldi, A., Bischof, H., Brox,

T., and Frahm, J.-M., editors, *Computer Vision – ECCV 2020 - 16th European Conference, 2020, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 175–191, Germany. Springer Science and Business Media Deutschland GmbH.

Klokov, R. and Lempitsky, V. S. (2017). Escape from cells: Deep Kd-Networks for the recognition of 3D point cloud models. *CoRR*, abs/1704.01222.

Knott, M. and Groenendijk, R. (2021). Towards mesh-based deep learning for semantic segmentation in photogrammetry. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Koch, S., Matveev, A., Jiang, Z., Williams, F., Artemov, A., Burnaev, E., Alexa, M., Zorin, D., and Panozzo, D. (2019). ABC: A big CAD model dataset for geometric deep learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kolbe, T. H., Gröger, G., and Plümer, L. (2005). CityGML – interoperable access to 3D city models. In *Proceedings of the first International Symposium on Geo-Information for Disaster Management*, pages 21–23. Springer Verlag.

Kölle, M., Laupheimer, D., and Haala, N. (2019). Klassifikation hochaufgelöster LiDAR- und MVS-Punktwolken zu Monitoringzwecken. In *39. Wissenschaftlich-Technische Jahrestagung der OVG, DGPF und SGPF in Wien*, volume 28, pages 692–701. Deutsche Gesellschaft für Photogrammetrie, Fernerkundung und Geoinformation e.V.

Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J. D., and Ledoux, H. (2021a). 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 Journal of Photogrammetry and Remote Sensing*, 1:11.

Kölle, M., Laupheimer, D., Walter, V., Haala, N., and Soergel, U. (2021b). Which 3D data representation does the crowd like best? Crowd-based active learning for coupled semantic segmentation of point clouds and textured meshes. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-2-2021:93–100.

Kölle, M., Walter, V., Schmohl, S., and Soergel, U. (2020). Hybrid acquisition of high quality training data for semantic segmentation of 3D point clouds using crowd-based active learning. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-2-2020:501–508.

Kölle, M., Walter, V., Schmohl, S., and Soergel, U. (2021c). Remembering both the machine and the crowd when sampling points: Active learning for semantic segmentation of ALS point clouds. In Del Bimbo, A., Cucchiara, R., Sclaroff, S., Farinella, G. M., Mei, T., Bertini, M., Escalante, H. J., and Vezzani, R., editors, *Pattern Recognition. ICPR International Workshops and Challenges*, pages 505–520, Cham. Springer International Publishing.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Labatut, P., Pons, J., and Keriven, R. (2007). Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8.

Landrieu, L., Raguet, H. R., Vallet, B., Mallet, C., and Weinmann, M. (2017). A structured regularization framework for spatially smoothing semantic labelings of 3D point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 132:102–118.

Landrieu, L. and Simonovsky, M. (2017). Large-scale point cloud semantic segmentation with superpoint graphs. *CoRR*, abs/1711.09869.

Laupheimer, D. and Haala, N. (2021). Juggling with representations: On the information transfer between imagery, point clouds, and meshes for multi-modal semantics. *ISPRS Journal of Photogrammetry and Remote Sensing*, 176:55–68.

Laupheimer, D., Shams Eddin, M. H., and Haala, N. (2020a). The importance of radiometric feature quality for semantic mesh segmentation. In *40. Wissenschaftlich-Technische Jahrestagung der DGPF in Stuttgart*, volume 29, pages 205–218. Deutsche Gesellschaft für Photogrammetrie, Fernerkundung und Geoinformation (DGPF) e.V.

Laupheimer, D., Shams Eddin, M. H., and Haala, N. (2020b). On the association of LiDAR point clouds and textured meshes for multi-modal semantic segmentation. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-2-2020:509–516.

Lawin, F. J., Danelljan, M., Tosteberg, P., Bhat, G., Khan, F. S., and Felsberg, M. (2017). Deep projective 3D semantic segmentation. *CoRR*, abs/1705.03428.

Li, N. and Pfeifer, N. (2019). Active learning to extend training data for large area airborne LIDAR classification. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2/W13:1033–

1037.

Li, W., Saeedi, S., McCormac, J., Clark, R., Tzoumanikas, D., Ye, Q., Huang, Y., Tang, R., and Leutenegger, S. (2018). InteriorNet: Mega-scale multi-sensor photo-realistic indoor scenes dataset. In *British Machine Vision Conference (BMVC)*.

Li, X., Li, C., Tong, Z., Lim, A., Yuan, J., Wu, Y., Tang, J., and Huang, R. (2020). Campus3D: A photogrammetry point cloud benchmark for hierarchical understanding of outdoor scene. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM 2020, New York, NY, USA. Association for Computing Machinery.

Lin, Y., Vosselman, G., Cao, Y., and Yang, M. Y. (2020). Efficient training of semantic point cloud segmentation via active learning. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-2-2020:243–250.

Linsen, L. and Prautzsch, H. (2001). Local Versus Global Triangulations. In *Eurographics 2001 - Short Presentations*. Eurographics Association.

Liu, W., Sun, J., Li, W., Hu, T., and Wang, P. (2019). Deep learning on point clouds and its application: A survey. *Sensors*, 19(19).

Liu, Y., Xue, F., and Huang, H. (2021). UrbanScene3D: A large scale urban scene dataset and simulator.

Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–137.

Long, J., Shelhamer, E., and Darrell, T. (2014). Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038.

Maas, H.-G. and Vosselman, G. (1999). Two algorithms for extracting building models from raw laser altimetry data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54(2):153–163.

Mallet, C., Bretar, F., Roux, M., Soergel, U., and Heipke, C. (2011). Relevance assessment of full-waveform LiDAR data for urban area classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(6, Supplement):S71–S84. Advances in LIDAR Data Processing and Applications.

Mandlburger, G., Wenzel, K., Spitzer, A., Haala, N., Glira, P., and Pfeifer, N. (2017). Improved topographic models via concurrent airborne LiDAR and dense image matching. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W4:259–266.

Masci, J., Boscaini, D., Bronstein, M. M., and Vandergheynst, P. (2015). Geodesic convolutional neural networks on riemannian manifolds. *CoRR*, abs/1501.06297.

Matrone, F., Lingua, A., Pierdicca, R., Malinverni, E. S., Paolanti, M., Grilli, E., Remondino, F., Murtiyoso, A., and Landes, T. (2020). A benchmark for large-scale heritage point cloud semantic segmentation. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B2-2020:1419–1426.

Maturana, D. and Scherer, S. (2015). VoxNet: A 3D convolutional neural network for real-time object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, page 922 – 928.

Metashape, A. (2021). AgiSoft Metashape Professional (Version 1.7.5) (Software). Retrieved from `http://www.agisoft.com/downloads/installer/`.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*.

Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., and Terzopoulos, D. (2020). Image segmentation using deep learning: A survey. *CoRR*, abs/2001.05566.

Mo, K., Zhu, S., Chang, A. X., Yi, L., Tripathi, S., Guibas, L. J., and Su, H. (2019). PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Munoz, D., Bagnell, J. A. D., Vandapel, N., and Hebert, M. (2009). Contextual classification with functional max-margin markov networks. In *Proceedings of (CVPR) Computer Vision and Pattern Recognition*, pages 975 – 982.

Niemeyer, J., Rottensteiner, F., and Soergel, U. (2014). Contextual classification of LiDAR data and building object detection in urban areas. *ISPRS Journal of Photogrammetry and Remote Sensing*, 87:152 – 165.

OpenStreetMap contributors (2017). Planet dump retrieved from https://planet.osm.org. `https://www.openstreetmap.org`.

Özdemir, E., Remondino, F., and Golkar, A. (2021). An efficient and general framework for aerial point cloud classification in urban scenarios. *Remote Sensing*, 13(10).

Pauly, M., Keiser, R., and Gross, M. (2003). Multi-scale feature extraction on point-sampled surfaces. Blackwell Publishers, Inc and the Eurographics Association.

Peters, T. and Brenner, C. (2019). Automatic generation of large point cloud training datasets using label transfer. *Tagungsband der 39. Wissenschaftlich-Technischen Jahrestagung der DGPF*.

Pfeifer, N., Stadler, P., and Briese, C. (2001). Derivation of digital terrain models in the SCOP++ environment. In *OEEPE Workshop on Airborne Laserscanning and Interferometric SAR for Digital Elevation Models*.

Qi, C. R., Su, H., Mo, K., and 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)*.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). PointNet++: Deep hierarchical feature learning on point sets in a metric space. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 5105–5114. Curran Associates, Inc.

Qiao, Y.-L., Gao, L., Yang, J., Rosin, P. L., Lai, Y.-K., and Chen, X. (2019). LaplacianNet: Learning on 3D meshes with laplacian encoding and pooling. *CoRR*, abs/1910.14063.

Ramirez, P. Z., Paternesi, C., Gregorio, D. D., and Stefano, L. D. (2019). Shooting labels: 3D semantic labeling by virtual reality. *arXiv preprint*, abs/1910.05021.

Riegler, G., Ulusoy, A. O., and Geiger, A. (2017). OctNet: Learning deep 3D representations at high resolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6620–6629.

Riemenschneider, H. (2014). ETHZ CVL RueMonge dataset. Zurich. ETH Zurich.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham. Springer International Publishing.

Rothermel, M., Wenzel, K., Fritsch, D., and Haala, N. (2012). SURE: Photogrammetric surface reconstruction from imagery. In *Proceedings LC3D Workshop*, volume 8, Berlin.

Rouhani, M., Lafarge, F., and Alliez, P. (2017). Semantic segmentation of 3D textured meshes for urban scene analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 123:124 – 139.

Roynard, X., Deschaud, J.-E., and Goulette, F. (2018). Paris-Lille-3D: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification. *The International Journal of Robotics Research*, 37(6):545–557.

Schindler, K. (2012). An overview and comparison of smooth labeling methods for land-cover classification. *IEEE Transactions on Geoscience and Remote Sensing*, 50:4534–4545.

Schmohl, S. and Sörgel, U. (2019). Submanifold sparse convolutional networks for semantic segmentation of large-scale ALS point clouds. In *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume IV-2/W5, pages 77–84.

Schult, J., Engelmann, F., Kontogianni, T., and Leibe, B. (2020). DualConvMesh-Net: Joint geodesic and Euclidean convolutions on 3D meshes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Selvaraju, P., Nabail, M., Loizou, M., Maslioukova, M., Averkiou, M., Andreou, A., Chaudhuri, S., and Kalogerakis, E. (2021). BuildingNet: Learning to label 3D buildings. In *IEEE/CVF International Conference on Computer Vision (ICCV)*.

Serna, A., Marcotegui, B., Goulette, F., and Deschaud, J.-E. (2014). Paris-rue-Madame database: a 3D mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods. In *4th International Conference on Pattern Recognition, Applications and Methods ICPRAM 2014*, Angers, France.

Settles, B. (2009). Active Learning Literature Survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Shah, S., Dey, D., Lovett, C., and Kapoor, A. (2017). AirSim: High-fidelity visual and physical simulation for autonomous vehicles. *CoRR*, abs/1705.05065.

Shilane, P., Min, P., Kazhdan, M., and Funkhouser, T. (2004). The Princeton Shape Benchmark. In *Shape modeling applications, 2004. Proceedings*, pages 167–178. IEEE.

Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3D shape recognition. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV 2015, pages 945–953, Washington, DC, USA. IEEE Computer Society.

Taime, A., Saaidi, A., and Satori, K. (2018). A new semantic segmentation approach of 3D mesh using the stereoscopic

image colors. *Multimedia Tools and Applications*, 77(20):27143–27162.

Tan, W., Qin, N., Ma, L., Li, Y., Du, J., Cai, G., Yang, K., and Li, J. (2020). Toronto-3D: A large-scale mobile LiDAR dataset for semantic segmentation of urban roadways. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 202–203.

Tatarchenko, M., Dosovitskiy, A., and Brox, T. (2017). Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs. *CoRR*, abs/1703.09438.

Tatarchenko, M., Park, J., Koltun, V., and Zhou, Q. (2018). Tangent convolutions for dense prediction in 3D. *CoRR*, abs/1807.02443.

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

Tutzauer, P., Laupheimer, D., and Haala, N. (2019). Semantic urban mesh enhancement utilizing a hybrid model. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W7:175–182.

Uggla, G. and Horemuz, M. (2021). Towards synthesized point clouds as training data for parsing and interpreting the built environment. QC 20210527.

Uy, M. A., Pham, Q.-H., Hua, B.-S., Nguyen, D. T., and Yeung, S.-K. (2019). Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *International Conference on Computer Vision (ICCV)*.

Valentin, J. P. C., Sengupta, S., Warrell, J., Shahrokni, A., and Torr, P. H. S. (2013). Mesh based semantic modelling for indoor and outdoor scenes. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2067–2074. Exported from https://app.dimensions.ai on 2019/02/21.

Vallet, B., Brédif, M., Serna, A., Marcotegui, B., and Paparoditis, N. (2015). TerraMobilita/iQmulus urban point cloud analysis benchmark. *Computers & Graphics*, 49:126–133.

Varney, N., Asari, V. K., and Graehling, Q. (2020). DALES: A large-scale aerial LiDAR data set for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Verdie, Y., Lafarge, F., and Alliez, P. (2015). LOD generation for urban scenes. *ACM Trans. Graph.*, 34(3):30:1–30:14.

Verma, N., Boyer, E., and Verbeek, J. (2018). FeaStNet: Feature-steered graph convolutions for 3D shape analysis. In *CVPR - IEEE Conference on Computer Vision and Pattern Recognition*, pages 2598–2606, Salt Lake City, United States. IEEE.

Vosselman, G., Coenen, M., and Rottensteiner, F. (2017). Contextual segment-based classification of airborne laser scanner data. *ISPRS journal of photogrammetry and remote sensing*, 128:354–371.

Vu, H.-H., Keriven, R., Labatut, P., and Pons, J.-P. (2009). Towards high-resolution large-scale multi-view stereo. In *CVPR*, pages 1430–1437, Miami, United States.

Walter, V. and Soergel, U. (2018). Implementation, results, and problems of paid crowd-based geospatial data collection. *PFG*, 86:187–197.

Wang, P.-S., Liu, Y., Guo, Y.-X., Sun, C.-Y., and Tong, X. (2017). O-CNN. *ACM Transactions on Graphics*, 36(4):1–11.

Weinmann, M., Jutzi, B., Hinz, S., and Mallet, C. (2015). Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. *ISPRS Journal of Photogrammetry and Remote Sensing*, 105:286 – 304.

Weinmann, M., Jutzi, B., and Mallet, C. (2013). Feature relevance assessment for the semantic interpretation of 3D point cloud data. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-5/W2:313–318.

Weinmann, M., Jutzi, B., and Mallet, C. (2014). Semantic 3D scene interpretation: A framework combining optimal neighborhood size selection with relevant features. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-3:181–188.

West, K. F., Webb, B. N., Lersch, J. R., Pothier, S., Triscari, J. M., and Iverson, A. E. (2004). Context-driven automated target detection in 3D data. In Sadjadi, F. A., editor, *Automatic Target Recognition XIV*, volume 5426, pages 133 – 143. International Society for Optics and Photonics, SPIE.

Wichmann, A., Agoub, A., and Kada, M. (2018). ROOFN3D: Deep learning training data for 3D building re-

construction. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2:1191–1198.

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

Winiwarter, L., Mandlburger, G., Schmohl, S., and Pfeifer, N. (2019). Classification of ALS point clouds using end-to-end deep learning. *Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 87(3):75–90.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3D ShapeNets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920. IEEE Computer Society.

Xiang, Y., Kim, W., Chen, W., Ji, J., Choy, C., Su, H., Mottaghi, R., Guibas, L., and Savarese, S. (2016). ObjectNet3D: A large scale database for 3D object recognition. In *European Conference Computer Vision (ECCV)*.

Xie, Y., Tian, J., and Zhu, X. X. (2020). Linking points with labels in 3D: A review of point cloud semantic segmentation. *IEEE Geoscience and Remote Sensing Magazine*, 8(4):38–59.

Yao, Z., Nagel, C., Kunde, F., Hudra, G., Willkomm, P., Donaubauer, A., Adolphi, T., and Kolbe, T. H. (2018). 3DCityDB - a 3D geodatabase solution for the management, analysis, and visualization of semantic 3D city models based on CityGML. *Open Geospatial Data, Software and Standards*, 3:1–26.

Ye, Z., Xu, Y., Huang, R., Tong, X., Li, X., Liu, X., Luan, K., Hoegner, L., and Stilla, U. (2020). LASDU: A large-scale aerial LiDAR dataset for semantic labeling in dense urban areas. *ISPRS International Journal of Geo-Information*, 9(7).

Zhang, J., Zhao, X., Chen, Z., and Lu, Z. (2019). A review of deep learning-based semantic segmentation for point cloud. *IEEE Access*, 7:179118–179133.

Zhdanov, F. (2019). Diverse mini-batch active learning. *CoRR*, abs/1901.05954.

Zheng, J., Zhang, J., Li, J., Tang, R., Gao, S., and Zhou, Z. (2020). Structured3D: A large photo-realistic dataset for structured 3D modeling. In *Proceedings of The European Conference on Computer Vision (ECCV)*.

Zhou, Q. (2018). Pymesh. `https://github.com/PyMesh/PyMesh`.

Zhou, Y., Huang, J., Dai, X., Luo, L., Chen, Z., and Ma, Y. (2020). HoliCity: A city-scale data platform for learning holistic 3D structures.

Zolanvari, S. I., Ruano, S., Rana, A., Cummins, A., da Silva, R. E., Rahbar, M., and Smolic, A. (2019). DublinCity: Annotated LiDAR point cloud and its applications. BMVC, 30th British Machine Vision Conference.

Acknowledgements

My gratitude goes to all the people who have contributed to creating this thesis in any conceivable way – through advice, discussions, caring, well-measured distractions at the right time, et al.

First of all, I would like to thank apl. Prof. Dr.-Ing. Norbert Haala and Prof. Dr.-Ing. Uwe Sörgel for allowing me to work at the Institute for Photogrammetry (ifp) and for creating the required conditions to write this dissertation. In particular, I thank apl. Prof. Dr.-Ing. Norbert Haala for his valuable input, continuous supervision, support, and encouragement throughout the years. Besides, I want to thank Prof. Dr. Ir. George Vosselman for his time and review as co-referee.

My appreciation goes to all (former) colleagues at the ifp. I enjoyed fruitful and insightful discussions, had successful collaborations, and experienced a pleasant working atmosphere where all necessary tools were provided at any time. Special thanks go to Markus Englich for outstanding technical support. Particular mention and big thanks go to Dr.-Ing. Patrick Tutzauer, Stefan Schmohl, and Michael Kölle. Thank you for *slack*ly and deep discussions – that sometimes may have missed the point but never failed to make me smile. Shout-out to you! Additionally, I want to thank Fangwen Shu, Mohamad Hakam Shams Eddin, Xiao Tan, and Vishal Pani, who assisted with the research during their theses or interns. Also, I would like to thank Philipp Schneider for lending a hand and sharing his experiences with practical crafting. The chocolate Mini-Me looks awesome! Sustainable thanks go to the climate team of ifp.

A crucial prerequisite for research is the availability of data. Therefore, I appreciate having been part of a research project in collaboration with the German Federal Institute of Hydrology (BfG) in Koblenz, which had been the basis for our Hessigheim 3D benchmark. Vaihingen 3D was provided by the German Society for Photogrammetry, Remote Sensing and Geoinformation (DGPF). Furthermore, I thank the whole nFrames team for their support regarding the mesh generation.

Last but not least, I want to thank all my relatives and friends for always being there and helping to overcome deep valleys of doubts. I appreciate your constant support and encouragement to finalize this thesis. Never underestimate the importance of adjacency – both in semantics and real life. In this context, thanks to my Ph.D. friends from other disciplines and/or universities who shared this journey and their leveling experiences. Thank you, Julia Aichinger, Tobias Gruber, Fabian Kempter, Hannes Riebl, and Lukas Winiwarter! I owe a huge gratitude to Lutzenhausen, CLS & JCS: Thank you for your exceptional support – particularly during the crunch time! Finally, without more words: Thank you, Caren!

```
if someone_forgotten == True:
    print("Sorry and thank YOU in particular!")
```

Curriculum Vitae

Personal

Name	Dominik Laupheimer
Date of Birth	02.11.1991
Place of Birth	Laupheim, Germany

Education

2014 – 2017	Master of Science, Geodesy and Geoinformatics, University of Stuttgart, Germany
2011 – 2014	Bachelor of Science, Geodesy and Geoinformatics, University of Stuttgart, Germany
2002 – 2011	Abitur, Carl-Laemmle-Gymnasium Laupheim, Germany

Experience

Nov 2017 – current	Research Associate, Institute for Photogrammetry, University of Stuttgart, Germany
Oct 2016 – Dec 2016	Research Intern, Robert Bosch GmbH, Renningen, Germany
Oct 2015 – Feb 2016	Research Assistant, Institute for Photogrammetry, University of Stuttgart, Germany
Jan 2015 – Aug 2015	Software Engineer, Roomplan TGU, Stuttgart, Germany
Oct 2014 – Apr 2015	Research Assistant, Institute for Photogrammetry, University of Stuttgart, Germany
May 2013 – Jul 2014	Research Assistant, Institute of Engineering Geodesy, University of Stuttgart, Germany