ON THE ASSOCIATION OF LIDAR POINT CLOUDS AND TEXTURED MESHES FOR MULTI-MODAL SEMANTIC SEGMENTATION

Dominik Laupheimer*, Mohamad Hakam Shams Eddin, Norbert Haala

Institute for Photogrammetry, University of Stuttgart, Germany
(dominik.laupheimer, norbert.haala)@ifp.uni-stuttgart.de

Commission II, WG II/6

KEY WORDS: Urban Scene Understanding, Semantic Segmentation, Multi-Modality, Textured Mesh, Point Cloud

ABSTRACT:

The semantic segmentation of the huge amount of acquired 3D data has become an important task in recent years. We propose a novel association mechanism that enables information transfer between two 3D representations: point clouds and meshes. The association mechanism can be used in a two-fold manner: (i) feature transfer to stabilize semantic segmentation of one representation with features from the other representation and (ii) label transfer to achieve the semantic annotation of both representations. We claim that point clouds are an intermediate product whereas meshes are a final user product that jointly provides geometrical and textural information. For this reason, we opt for semantic mesh segmentation in the first place. We apply an off-the-shelf PointNet++ to a textured urban triangle mesh as generated from LiDAR and oblique imagery. For each face within a mesh, a feature vector is computed and optionally extended by inherent LiDAR features as provided by the sensor (e.g. *intensity*). The feature vector extension is accomplished with the proposed association mechanism. By these means, we leverage inherent features from both data representations for the semantic mesh segmentation (multi-modality). We achieve an overall accuracy of 86.40 % on the face-level on a dedicated test mesh. Neglecting LiDAR-inherent features in the per-face feature vectors decreases mean intersection over union by ~2 %. Leveraging our association mechanism, we transfer predicted mesh labels to the LiDAR point cloud at a stroke. To this end, we semantically segment the point cloud by implicit usage of geometric and textural mesh features. The semantic point cloud segmentation achieves an overall accuracy close to 84 % on the point-level for both feature vector compositions.

1. INTRODUCTION

The past decade has shown that 3D data acquisition and data processing has increasingly become feasible and important in the domain of photogrammetry and remote sensing. Common representations for 3D data are point clouds, volumetric representations, projected views (i.e. RGB-D images or renderings), and meshes. Amongst them, 3D point cloud processing may currently be the most popular topic – in particular concerning semantic segmentation (cf. section 2). However, textured meshes as generated from LiDAR point clouds and imagery have some favorable characteristics. Whereas point clouds are an unordered set of points, meshes are graphs consisting of vertices, edges, and faces that provide explicit adjacency information. Intrinsically, meshes facilitate data fusion by utilizing LiDAR points and Multi-View Stereo (MVS) points for the geometric reconstruction while leveraging high-resolution imagery for texturing (hybrid data storage). Therefore, meshes are realistic-looking 3D maps of our real world and are easily understandable – even for non-experts.

Meshes are less memory-consuming than point clouds since meshing algorithms try to minimize the number of entities. Before the meshing, point clouds will be filtered in such a way that only geometrically relevant points are kept. This embraces noise filtering and filtering of points that can be approximated by a face (e.g. points on planar surfaces). Furthermore, there will be geometric simplifications based on the desired level of detail and, as the case may be, due to 2.5D geometry. Georeferencing issues of LiDAR data and imagery will cause discrepancies between point clouds and meshes, too. Besides, meshes are surface descriptions that cannot handle multi-target capability like LiDAR point clouds. This inevitably leads to a drop in entities to be stored. Moreover, the high-resolution texture information is stored in texture atlases that avoid redundant image content. Therefore, textured meshes provide geometric and textural information in a lightweight fashion.

We are aware of the fact that proper meshing of our complex world is a hard task and still subject to research.

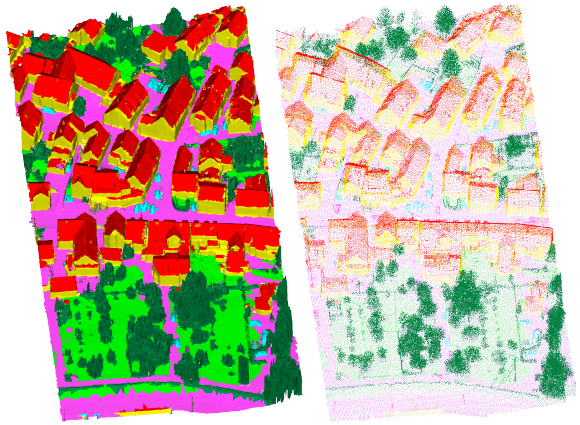

Figure 1. Predicted semantic mesh segmentation utilizing mesh-inherent and LiDAR-inherent features (*left*) and its transferred labels to the dense LiDAR point cloud (*right*; subsampled by factor 20 for visualization).
The following class color code is used throughout the paper: *building mass/facade* (yellow), *roof* (red), *impervious surface* (magenta), *green space* (light green), *mid and high vegetation* (dark green), *vehicle* (cyan), *chimney/antenna* (orange) and *clutter* (gray).

* Corresponding author.

Concerning the recent hybridization trend, from our point of view, enhancing the core 3D point clouds to textured meshes may replace unstructured point clouds as default representation for urban scenes in the future. Nowadays, joint photogrammetric and LiDAR acquisition (hybrid acquisition) is state of the art for airborne systems and starts to emerge even for Unmanned Airborne Vehicle (UAV)-based systems (Mandlburger et al., 2017; Cramer et al., 2018). Recently, Glira et al. (2019) proposed the hybrid orientation of Airborne Laser Scanning (ALS) point clouds and aerial imagery (hybrid adjustment). The increasing availability of simultaneously acquired airborne data with different acquisition methods calls for multi-modality. First software solutions already enable data fusion on the mesh-level. For instance, software SURE (version $\geq$ 3) by nFrames (Rothermel et al., 2012) produces more complete meshes based on LiDAR and MVS data. In our opinion, point clouds are not a final user product. Notwithstanding, textured meshes are a mostly overlooked topic in the domain of photogrammetry and remote sensing despite their advantageous characteristics. Very few works deal with semantic mesh segmentation in urban scenes (Rouhani et al., 2017; Tutzauer et al., 2019). One of the main differences between meshes and point clouds is the availability of high-resolution texture. Tutzauer et al. (2019) show that available color information is beneficial for semantic mesh segmentation. More precisely, Laupheimer et al. (2020) attest that color information per-face (i.e. texture) outperforms color information per-vertex (like e.g. a colored point cloud) by evaluation of several radiometric feature qualities. They achieved to double the performance gain by utilizing color information of the entire face instead of per-vertex only. However, they also show the inherent limitations of texture due to occlusions, absence of imagery, and the quality of the geometric reconstruction.

For these reasons, we focus our work on meshes and investigate the semantic segmentation of textured meshes in urban areas as generated from LiDAR data and oblique imagery. The additional usage of LiDAR features may overcome the issues of textural features. This study utilizes data described in (Cramer et al., 2018). We briefly present the data acquisition and the derived labeled mesh in subsection 3.1. We establish a feature-based pipeline for semantic mesh segmentation. As a result, we can utilize any arbitrary feature-based approach that allows a simple extension of the feature vector. Previous work compared different classifiers and proved PointNet++ to work best (Laupheimer et al., 2020). Hence, we utilize a PointNet++ classifier in this study to accomplish the semantic segmentation.

However, our main contribution is the geometric linking of (LiDAR) point clouds and meshes. In this work, we propose a novel association mechanism that allows transferring information between (LiDAR) point clouds and meshes. Naturally, the point density of the LiDAR cloud is higher than the face density. That is why the association assigns many LiDAR points to each face (many-to-one relationship) and injects great flexibility and versatility due to the simple transfer of labels and features between the two representations. Thereby, we aim to jointly leverage information from both data representations for their semantic segmentation. To the best of our knowledge, no other approach leverages both representations at the same time. Similarly to squashing point clouds into grid-like representations, we abuse the mesh as a proxy to semantically segment point clouds. In subsection 3.3 and subsection 3.4, we describe in detail the association of LiDAR points and mesh faces along with the respective particular challenges due to the mentioned discrepancies between the mesh and the LiDAR point cloud.

The label transfer from one representation to the other may reduce the manual labeling effort. Since the dawn of the Deep Learning (DL) era, ground truth generation has become a very important task. However, manual ground truth generation is tedious and time-consuming work, specifically for 3D point clouds consisting of millions of points. We claim that manual mesh labeling is easier and faster than point cloud labeling due to fewer entities, visibility checks, and realistic-looking textured faces. The proposed label transfer offers to boost the generation of labeled data sets with various representations in a semi-automatic manner.

Utilizing our mechanism, we enhance per-face feature vectors with available LiDAR features. Due to the many-to-one relationship, we calculate the per-face median for LiDAR features. Subsection 3.2 lists the considered mesh and point cloud features. After the semantic mesh segmentation, the predicted mesh labels will be transferred to the LiDAR point cloud. Hence, we end up with a labeled mesh and a labeled point cloud by using available information from both data representations. Figure 1 shows the prediction result for the semantic mesh segmentation supported by LiDAR features (*left*) and its transferred prediction to the LiDAR point cloud (*right*).

In section 4, we report the best performing parameters for the association mechanism with respect to the used data. Furthermore, we analyze the benefit of the additional LiDAR features by comparing the classifier performance with/without LiDAR support on the face-level and the point-level.

The presented pipeline could be reversed entirely. However, we argue that semantic mesh segmentation is faster than the semantic point cloud segmentation because the number of points exceeds the number of faces. Moreover, the one-to-many relationship between face and LiDAR points enables fast label transfer to the point cloud.

2. RELATED WORK

The semantic segmentation of 3D data has become a standard task in the domain of photogrammetry and remote sensing. A large part of the community deals with semantic segmentation of 3D LiDAR point clouds. In contrast, only a few works deal with the semantic segmentation of meshes (cf. subsection 2.1). Regardless of the data representation, DL methods rely on a huge amount of ground truth data. We briefly review available ground truth in subsection 2.2.

2.1 Semantic Segmentation of 3D Data

DL methods, particularly Convolutional Neural Networks (CNNs), are state of the art for semantic segmentation in image space. Therefore, it seems reasonable to apply well-established DL methods of the image space to point clouds. However, the unstructured nature of 3D point clouds prevents to apply CNNs directly to point clouds. To overcome the non-Euclidean structure, point clouds are commonly structured into grid-like 3D or 2D representations by voxelization or multi-view rendering respectively. Several works voxelize the point cloud and train a supervised classifier. The predicted labels for the voxels will be transferred to all contained points (Hackel et al., 2016; Huang and You, 2016). Detouring via image space, multi-view approaches leverage well-performing semantic image segmentation methods. The per-pixel predictions are back-projected to 3D space (Boulch et al., 2017; 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 (which comes along with loss of geometric information).

The rise of PointNet and its hierarchical successor PointNet++ (Qi et al., 2017a,b) constitutes a milestone in semantic point

cloud segmentation since they operate directly on unstructured 3D point clouds. The gist of PointNet is to use a symmetric function that is independent of set permutation. Its extension PointNet++ hierarchically applies PointNets to the iteratively subsampled point cloud and, hence, operates on several scales. This procedure mimics hierarchical feature learning with increased contextual information similar to CNNs in image space. Likewise, Boulch (2019) introduces continuous convolutional kernels that can be applied directly to point clouds. Griffiths and Boehm (2019) review the current state-of-the-art DL architectures for processing unstructured point clouds.

The emerging field of geometric DL extends basic DL operations to non-Euclidean domains such as graphs and manifolds in order to use topological information (Bronstein et al., 2016). Point clouds do not provide topological information per se. Therefore, Landrieu and Simonovsky (2017) organize point clouds in Superpoint Graphs (SPGs). 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. To summarize, the adaption of (geometric) DL methods contributed to substantial progress in the field of semantic point cloud segmentation in the last decade.

On the contrary, mesh interpretation has hardly been explored by the community. In comparison, meshes are a default data representation in the domain of computer vision. However, that community typically deals with small-scale (indoor) data sets (George et al., 2017). By analogy to semantic point cloud segmentation, common approaches for semantic mesh segmentation make a circuit to 2D image space to take advantage of image-based DL methods. Those approaches render 2D views of the 3D scene, learn the segmentation for different views and finally, back-project the segmented 2D images onto the 3D surface (Su et al., 2015). Wu et al. (2015) voxelize the mesh and apply a convolutional deep belief network. Qiao et al. (2019) propose a geometric DL approach that encodes the mesh connectivity using Laplacian spectral analysis and aggregates global information via mesh pooling blocks.

Semantic segmentation of real-world large-scale meshes is a mostly overlooked topic. Rouhani et al. (2017) gather faces of a 3D textured mesh as generated from MVS imagery into so-called superfacets and train a Random Forest (RF) using geometric and photometric features. Tutzauer et al. (2019) utilize a DL approach by training a multi-branch 1D CNN with contextual features and compare the achieved results to a RF.

2.2 Ground Truth Availability

The computer vision community provides annotated mesh data for indoor scenes and single objects (Armeni et al., 2017; Shilane et al., 2004). However, to the best of our knowledge, there are no labeled data sets that cover urban scenes. In contrast, there are many available labeled urban data sets for 3D point clouds provided by the community of photogrammetry and remote sensing (Zolanvari et al., 2019; Wichmann et al., 2018; Niemeyer et al., 2014). Generally, there is a lack of ground truth data sets that provide point clouds and oriented imagery (and in this way textured meshes). Therefore, available annotations are limited to a single representation.

Ramirez et al. (2019) present a virtual reality tool that gamifies the manual labeling of meshes and point clouds. Our proposed transfer tool could derive labeled meshes from publicly available annotated point cloud data and vice versa (provided that the necessary data is available). Hence, it has the potential to accelerate ground truth generation for several representations.

3. METHODOLOGY

The core of our work(flow) is the linking of point clouds and meshes as described in subsection 3.3. For the semantic mesh segmentation of the used data (cf. subsection 3.1), we represent each face by a feature vector (cf. subsection 3.2). We transfer LiDAR features to the mesh and extend the mesh-inherent feature vectors with LiDAR features by utilizing our proposed association mechanism. We adopt an off-the-shelf PointNet++ to achieve the semantic mesh segmentation with mesh-inherent and LiDAR-inherent features. Finally, we transfer predicted mesh labels to the LiDAR point cloud to semantically segment the given LiDAR point cloud, too. Non-associated points will be labeled by majority voting of associated adjacent LiDAR points. The challenges and limitations of the association mechanism are discussed in subsection 3.4.

3.1 Data Preparation

The simultaneous acquisition and georeferencing of the high-resolution LiDAR and image data is described in (Cramer et al., 2018). The data is captured in Hessigheim, Germany, and covers an area of 800 m × 300 m. The ALS data consists of up to 800 points/m^2 for the entire area. The Ground Sampling Distance (GSD) of the oblique aerial images is ~2.5 cm. Tutzauer et al. (2019) derived a textured 2.5D mesh by fusing the simultaneously acquired ALS data and oblique imagery with software SURE 2 from nFrames. A GSD of 5 cm is used for meshing. Due to the high density of the LiDAR point cloud and the relatively low resolution and overlap of oblique imagery, Tutzauer et al. (2019) refrained from an integrated geometric reconstruction from LiDAR and MVS. To this end, the mesh geometry purely relies on the LiDAR point cloud. The oblique images have been used for texturing.

For the time being, we choose the 2.5D mesh for this work due to its relatively simple and fast generation which comes along with fewer faces compared to a 3D mesh. This fact is beneficial for fast semantic segmentation of the mesh. Moreover, compared to 3D meshing, 2.5D meshing is rather simple, unambiguous, and independent from the used meshing algorithm. Currently, a lot of research is done for proper 3D meshing. The 2.5D nature calls for a robust association of LiDAR point clouds and meshes like discussed in subsection 3.4 and serves as adequate test data.

Another aspect that must not be neglected is the availability of ground truth annotations for the textured 2.5D mesh. Manual labeling is described in (Tutzauer et al., 2019). The considered classes are (relative class frequencies for the considered data are given in parentheses): *building mass/facade* (9.28 %), *roof* (6.34 %), *impervious surface* (5.67 %), *green space* (5.97 %), *mid and high vegetation* (63.38 %), *vehicle* (0.83 %), *chimney/antenna* (0.31 %) and *clutter* (8.22 %). The rejection class *clutter* gathers all faces that do not match the other classes. After the manual labeling, the data set is split into mutually exclusive sets dedicated to training, validation, and testing. Throughout this paper, we report results only for testing tile A which consists of 270k labeled faces (Tutzauer et al., 2019). The tile refers to 40M LiDAR points. In order to process data with the PointNet++ architecture, we partition the data into spatially overlapping tiles. The tiles cover an area of 50 m × 50 m and use an 80 % overlap. A detailed description of the network-specific data preparation is given in (Laupheimer et al., 2020).

3.2 Per-Face Feature Vector Composition

We represent each face by its Center of Gravity (COG) attached with a 1D feature vector. We refer to this as "COG cloud" that

can be treated like a common point cloud. Hence, we can apply classifiers that have been designed for point clouds originally. However, the COG cloud is not a common point cloud since it still benefits from inherent mesh properties like the availability of high-resolution texture and adjacency knowledge. One strength of the feature-based representation is its flexibility. The per-face feature vector can be composed arbitrarily and extended with LiDAR features. By these means, multi-modality is achieved easily. To do so, we first have to establish the one-to-many relationship between faces and LiDAR points (cf. subsection 3.3).

In this work, we limit ourselves to representation-inherent features in order to reduce computation overhead. Additionally, we rely on per-face features only. We do not use contextual features or other sophisticated handcrafted features since deeper feature analysis is not the focus of this work. Notwithstanding, our approach can be extended easily by other handcrafted features. In our investigations we consider the mesh-inherent per-face features *normal vector* and *median HSV*. The *normal vector* is the cross product of face edges. The *median HSV* is derived from associated pixels in the texture atlas. We choose HSV space to be independent of illumination conditions. Furthermore, we incorporate LiDAR features by leveraging our proposed association mechanism. We use LiDAR-inherent features that are directly provided by the sensor: *intensity* and *return number* (not to be confused with *number of returns*, which is the total number of returns for a given pulse). The latter may attract your attention since the mesh is a surface description where the *return number* should be constant for all faces. However, reality shows that vegetational areas may not only be associated with first pulses (cf. Figure 2, top), since our association mechanism may assign non-canopy points to the faces due to thresholding. However, such points could be filtered easily by their *return number*. Without filtering, *return number* may improve the prediction of vegetational classes.

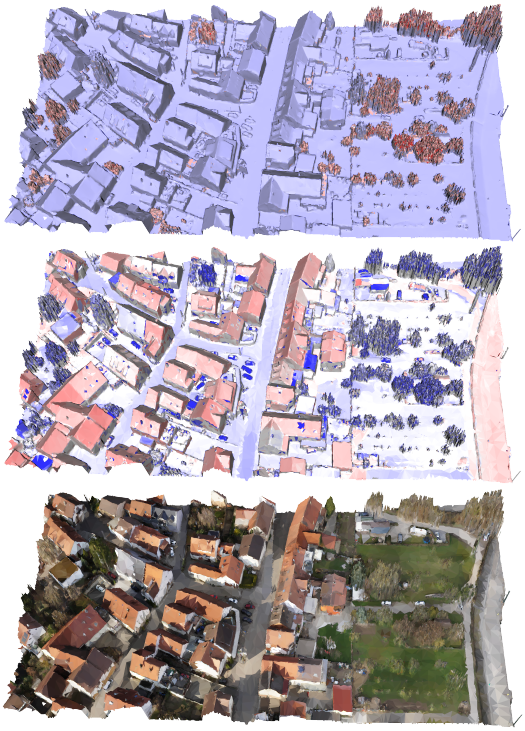

Figure 2. Per-face features (top: *median return number*, center: *median intensity*, bottom: *median RGB*). Blue indicates low values; red high values.

Due to the many-to-one relationship between LiDAR points and mesh face, we have to aggregate the associated LiDAR points per face. For each face, we calculate the median values for the features of the associated LiDAR points since it retains real measurements and is robust against outliers.

To summarize, for each face, we construct a multi-modal 1D feature vector consisting of mesh-inherent features (geometry and radiometry) and additional LiDAR-inherent features. All features are normalized according to statistics of the training set. Figure 2 depicts the test mesh with its faces colored by associated LiDAR features and *median RGB*.

3.3 Associating LiDAR Point Cloud and Mesh

Each face (represented by its COG) is assigned with LiDAR points that represent the same surface. The association mechanism operates in three steps per COG: (i) clipping of the LiDAR point cloud to a spherical vicinity, (ii) filtering of *out-of-face* LiDAR points, and (iii) filtering of *off-the-face* LiDAR points (Figure 3). *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. Such points exist due to the simplification during the meshing, the 2.5D mesh geometry, and the representation type differences as discussed in section 1. These two groups are not mutually exclusive. We filter these two groups with the help of barycentric coordinates, which enable a fast execution based on vector algebra. For each face, we parameterize the respective 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 $\mathbf{p} = b_1\mathbf{v}_1 + b_2\mathbf{v}_2 + b_3\mathbf{v}_3$.

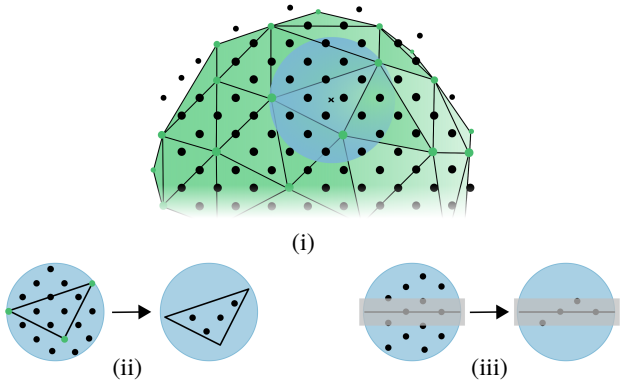

Figure 3. Steps (i) - (iii) of the association mechanism.
(i): Clipping of the LiDAR point cloud (black dots) to the vicinity (blue sphere) of the considered face. Its COG is marked with a black cross. The mesh surface and its vertices are depicted in green.
(ii): Filtering of *out-of-face points* based on the clipping result (orthogonal view concerning the face surface).
(iii): Filtering of *off-the-face points* (side view with respect to the 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 COG in order to accelerate the association. To this end, we build a kD tree for the LiDAR points and query the built tree with COGs. For each COG, we find all LiDAR points within distance r of the respective COG (ball query). The query parameter r is set to the minimum value that guarantees to enclose the entire face

and the manually set threshold(s) to be effective. The ball query delivers a subset of the LiDAR point cloud, which may contain *off-the-face points* and *out-of-face points*.

Second, we filter *out-of-face points*. 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 refer the interested reader to (Ericson, 2005) for a detailed discussion of barycentric coordinates. By replacing operator $\leq$ with $<$ we can easily neglect points on the edges (i.e. vertices as well). Figure 4 shows a face parameterized in barycentric coordinates. The green marked area shows the inside of the face that fulfills the above condition (with $<$ operator). We exclude points on the edges since such points may inject ambiguous feature information into per-face feature vectors. Technically, edge points belong to two adjacent faces A and B. It is hard to decide whether to assign them to face A or its adjacent face B. Having in mind the high density of the considered data set, neglecting those points is of minor importance for the aggregation of LiDAR points for the feature vector composition. Since the filter conditions work only on points in the plane, we orthogonally project all LiDAR points of the subset onto the plane before filtering. The orthogonal projection is only relevant to the association mechanism. The original LiDAR point cloud remains unchanged.

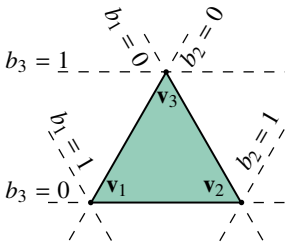

Figure 4. Barycentric coordinates b_i of the face $\triangle \mathbf{v}_1\mathbf{v}_2\mathbf{v}_3$.

Visually, the result of this filtering is the intersection of a triangular prism and the sphere of the ball query. Finally, we filter the remaining *off-the-face points*. Eventually, a manually set threshold decides whether a LiDAR point is associated with a face or not. We calculate the distance for each remaining point and its orthogonal projection on the face. If the distance exceeds the chosen threshold the LiDAR point is not associated with the face. Since we deal with a challenging 2.5D mesh, we use a more sophisticated adaptive thresholding that associates faces with LiDAR points where 2.5D and 3D geometry differ significantly (cf. Figure 5 and subsection 3.4 for a detailed discussion). The association information is stored as a per-point attribute. For each associated LiDAR point, the respective face index is attached to its attributes. Non-associated LiDAR points are marked with -1. We utilize the stored association information to transfer LiDAR features to the mesh and, reversely, to transfer (predicted) labels from the mesh to the LiDAR cloud. By these means, the transferred point cloud labeling implicitly uses mesh-inherent features. This approach is similar to voxel-based semantic segmentation approaches. However, the meshing process is not data-agnostic like voxelization. Hence, the mesh may be a more reasonable proxy. Nevertheless, the voxelization process is significantly faster than the meshing process.

3.4 Association Challenges and Limitations

The association of LiDAR point clouds and (2.5D) meshes comes along with particular challenges due to discrepancies between both representations (cf. section 1). As a general rule, the better the mesh represents the real 3D structure of the world, the better works the proposed association mechanism and the subsequent information transfer.

2.5D meshing facilitates the mesh generation in the first place. However, the 2.5D geometry makes the association of faces and 3D points harder. The most obvious challenge is the association of LiDAR points that differ from the reconstructed 2.5D mesh geometry (e.g. facades or tree stems, cf. Figure 5). Such points appear to be subsurface points but in reality, they are canopy points. We refer to them as *apparent non-canopy points*.

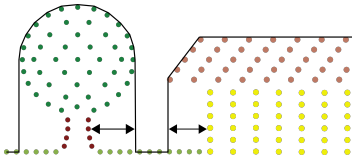

Figure 5. The black arrows show the discrepancy between the 2.5D mesh (*black line*) and the colored 3D point cloud for a tree (*left*) and a building (*right*).

Our semantic mesh segmentation relies on 1D feature vectors. Therefore, we have to provide a feature vector of the same length consisting of the same features for each face. For this reason, we want to establish as many point-face-connections as possible. The additional association of *apparent non-canopy points* is reasonable since they are likely to belong to the same class as the respective face.

For these reasons, we use adaptive thresholding with 3 filter levels for the filtering of *off-the-face points* (step (iii) in subsection 3.3). The threshold values increase with increasing level. Each level l consists of two independent thresholds θ_l^+ and θ_l^-. Thresholds θ_l^+ and θ_l^- limit the orthogonal distance to the face plane in the normal direction or the opposite direction respectively. The two-fold thresholding per level enables non-symmetric filtering improving flexibility and adaptiveness. The adaptive thresholding enables a proper association of high-quality 3D meshes. Level 2 and 3 can be seen as a fall-back for the association of 2.5D meshes. Therefore, our association approach is agnostic to the geometric structure of mesh geometry: 2.5D or 3D meshes can be processed. Moreover, the adaptive thresholding dampens georeferencing issues. However, the used 2.5D mesh does not suffer from georeferencing discrepancies as it is generated from the LiDAR point cloud.

It may happen that even the last level does not associate any LiDAR points (cf. Figure 6). As a consequence, non-associated faces lack of LiDAR features. We propose three options to bridge the missing LiDAR features in feature vectors. The most obvious approach is to use vertices, too. This option is applicable only if all vertices coincide with LiDAR points. However, the face vertices do not necessarily coincide with LiDAR points, when the mesh is generated from LiDAR and MVS points. Furthermore, in our association mechanism, we neglect points on the edges/vertices since their association is ambiguous. Another option is to lever out the association mechanism and incorporate LiDAR features from close-by LiDAR points that have been filtered previously, i.e. *out-of-face points* or *off-the-face points*. Such LiDAR points may belong to another class and their features may not be representative. Precisely for the same reason, we neglected points on the edges/vertices in the first place. Consequently, our method of choice is to set all LiDAR features to a constant value (e.g. zero). The zeroed LiDAR features introduce constant feature noise regardless of the corresponding class. We implemented the latter option to be independent on the meshing algorithm and with a view to the future usage of 3D meshes as generated from LiDAR and MVS points.

Non-association is not merely unfavorable for feature vector composition but also for label transfer. A high connection rate is beneficial for a comprising point cloud labeling because non-

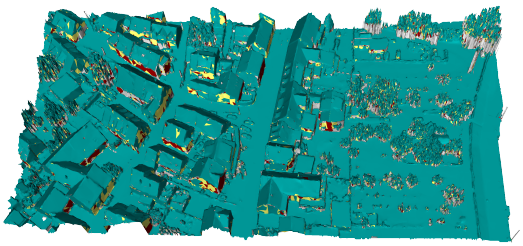

Figure 6. Color-coded threshold levels per face (level 1: *green*, level 2: *yellow*, level 3: *red*). Non-associated faces are marked in gray. Best viewed digitally.

associated LiDAR points cannot be labeled directly by the label transfer. To end up with the entire LiDAR point cloud labeled, we aggregate labels of the k nearest associated neighbors via majority voting for each unassociated LiDAR point.
Besides the non-association problem, information transfer suffers from a general issue. The one-to-many relationship between face and LiDAR points is beneficial provided that the mesh is a good reconstruction of the 3D world. At a stroke, we can attach mesh labels to many LiDAR points (cf. Figure 7).

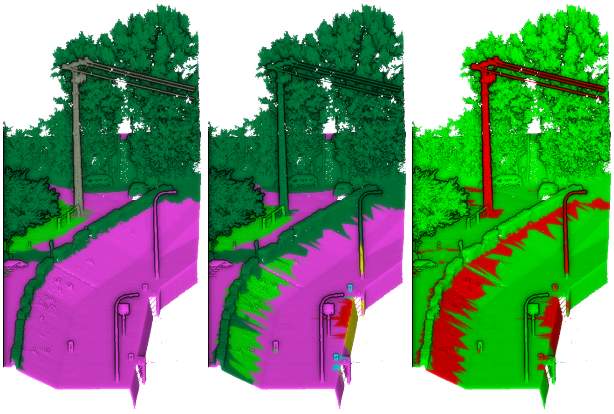

Figure 7. Ground truth and prediction as transferred from the mesh to the point cloud (*left* and *center*). The correct/false predictions are marked in green/red (*right*).
Please note, transferred ground truth errors cannot be detected (e.g. light poles labeled as *impervious surface*).

However, if the mesh geometry and the LiDAR geometry do not coincide, the transfer may cause mislabeling. Furthermore, meshing algorithms do not incorporate semantic borders yet, wherefore transferred mislabeling may happen. Consider the transition of a planar *impervious surface* to *green space* in the real world. The mesh representation may simplify this scenario to one large face. In such a scenario, the label transfer causes mislabeling in the LiDAR point cloud. The reverted process (label transfer from LiDAR point cloud to mesh) will suffer from the same issue since you have to opt for one label per face.
Ideally, the transferred labels could be compared to a manually generated LiDAR ground truth. Yet, such ground truth does not exist. For this reason, ground truth and predicted annotations are generated in the same way by transferring the mesh labels to the point cloud. We are aware of the fact that such generated ground truth data cannot reveal mislabeling within faces like depicted in Figure 7 (*left*). These discrepancies can only be detected with a manually labeled point cloud. The missing manual ground truth for the LiDAR point cloud is a limiting factor. Nevertheless, the comparison facilitates the proof of concept – even operating with the challenging 2.5D mesh geometry.

4. RESULTS

The pipeline is tested on a machine with an NVIDIA GeForce GTX 1080 Ti GPU, 64 GB RAM, and a 12-core CPU. PointNet++ is implemented with the DL framework TensorFlow 1.9. Figure 1 shows the prediction result for the semantic mesh segmentation (*left*) and its transferred prediction to the LiDAR point cloud (after majority voting of labels for unassociated points, *right*). 88.95 % of the surface area is predicted correctly when incorporating LiDAR features.

4.1 Analysis of the Association Mechanism

For the used data set, we found in an empirical process the association to perform best with thresholds $\theta^+ = \{5\,\text{cm}, 10\,\text{cm}, 15\,\text{cm}\}$ and $\theta^- = \{20\,\text{cm}, 40\,\text{cm}, 80\,\text{cm}\}$ for the respective levels 1, 2 and 3. Threshold θ_1^+ is defined with respect to the precision of the LiDAR point cloud: $\theta_1^+ = 5\,\text{cm} \approx 2 \cdot \sigma_{LiDAR}$. In particular, thresholds θ_l^- are fine-tuned for associating as many *apparent non-canopy points* with the 2.5D mesh (cf. subsection 3.4). Therefore, these hyperparameters highly depend on the underlying data set. Compared to a static threshold, adaptive thresholding increases the association rate by 36k faces (+13 %) and 720k points (+2 %). 18.4 % of faces and 31.1 % of points remain without association. Still, 88.3 % of the surface area is associated with LiDAR points due to varying face areas. The analysis of non-associated faces shows that their majority covers an area smaller than 0.5 m^2 (for classes *roof*, *green space*, *impervious surface* and *vehicle*) and/or has a vertical extension (like classes *building mass/facade*, *mid and high vegetation*, *chimney/antenna* and *clutter*). Generally, classes with vertical elements register worse association rates. For instance, roughly 16 % and 28 % of faces of *building mass/facade* and *mid and high vegetation* respectively are not associated with any LiDAR points. This is illustrated in Figure 5 and Figure 6. On the contrary, predominantly planar classes such as *roof*, *impervious surface*, and *green space* have good association rates ($\geq$ 93 %). To conclude, the association mechanism works but its performance is constrained by the coincidence of mesh and LiDAR geometry (including proper georeference). However, the adaptive thresholding loosens this dependency to some extent. The association mechanism lasts 8 min (of these, 5.5 min for the tree generation with LiDAR points) processing 270k faces and 40M LiDAR points. Therefore, the tree generation is the bottleneck of our association mechanism. The final label transfer to the associated LiDAR points is very fast since we store the face index per associated LiDAR point. We transfer mesh labels to 27M associated LiDAR points in less than 0.2 s.

4.2 Semantic Mesh Segmentation

In this section, we compare the performance of two different feature vector compositions: with and without additional LiDAR-inherent features. We evaluate the per-face predictions with the manually attached ground truth. The semantic mesh segmentation is done by a PointNet++ classifier with multi-scale grouping trained on the provided training set. PointNet++ achieves an Overall Accuracy (OA) of 86.40 % and an Mean Intersection over Union (mIoU) of 51.29 % while utilizing additionally attached LiDAR features. The prediction lasts 2.3 minutes. The use of additional LiDAR features improves OA by 0.72 % compared to a feature vector that does not consider additional LiDAR features. Due to the one-to-many-relationship between face and points, the improvement might be beneficial for the label transfer to point clouds since it comes along with an increased correctly

predicted area of 225 m^2. The adaption of point cloud metrics to meshes and the high class imbalance of the used data cause the discrepancy in OA and mIoU. The point-based metric mIoU achieves low values since each face has the same impact on the evaluation metric regardless of its area. In particular, mIoU is dampened by bad performance of class *clutter* (f1 = 18.23 % with LiDAR features). On the contrary, OA suffers from the accuracy paradox concerning the highly imbalanced data set. Regardless of the feature vector composition, the best performing class is the most prominent class *mid and high vegetation* which achieves a constant recall close to 99 % for both scenarios. Therefore, the improvement of OA indicates improved per-class recall values for other classes. In fact, mIoU increases by 2.11 % when LiDAR features are additionally used. The per-class recall values for *building mass/facade* and *impervious surface* improve by roughly 5 %. Class *vehicle* registers a significant recall gain of 19 %. On the contrary, recall of *chimney/antenna* decreases by 6 % and is often mispredicted as *roof*. *Clutter* has high intra-class variance per definition and performs poorly in both scenarios. It is often predicted as *building mass/facade* or *mid and high vegetation*.

The analysis of non-associated faces with zeroed LiDAR features shows significantly worse performance than for the associated faces. mIoU differs by approximately 6 %. However, the majority of non-associated faces belongs to classes *building mass/facade* and *mid and high vegetation* which still show good performance. In comparison, f1 score for *green space* is rather low (26.80 %) but there are only 755 non-associated faces covering 46.01 m^2 (0.8 ‰ of entire surface area). It is mainly confused with *mid and high vegetation*.

The best performance is achieved with both radiometric and LiDAR features. Performance drops by 2.27 % (OA) and 10.45 % (mIoU) when LiDAR features and radiometric features are ignored. We refer the interested reader to (Laupheimer et al., 2020) for a detailed discussion of radiometric feature quality.

4.3 Semantic Point Cloud Segmentation

We argue that point clouds are not a final consumer product. However, semantic point cloud segmentation is a standard task and we can easily transfer the (predicted) mesh labels to the point cloud leveraging our association mechanism. By this, we achieve to label the LiDAR point cloud via the mesh as a proxy. Therefore, the semantic point cloud segmentation implicitly uses mesh-inherent features. Figure 8 depicts the face indices for a subset of the LiDAR point cloud showing a facade and a street. Non-associated LiDAR points are marked with black-framed grayish polygons. Typically, they represent measurements that penetrate the surface (e.g. through windows) or measurements of constructional elements that are shifted with respect to the reconstructed (2.5D) mesh surface (e.g. a door).

After the label transfer, we perform a majority voting for each non-associated LiDAR point in order to expand the transferred labels to them. The majority vote considers labels of the 80 nearest associated LiDAR points. This hyperparameter has been chosen heuristically concerning the LiDAR density (800 points/m^2) of the given data. The label expansion is time-consuming due to the needed neighborhood query. The processing time depends on the considered nearest neighbors k and the number of non-associated points. In our case, it takes 12.2 min (including 2.5 min for kD tree generation).

On the LiDAR point cloud, we evaluate the transferred predictions with the transferred ground truth from the mesh on the point-level. The label transfer achieves an OA of 86 % for both feature vector compositions. However, the mIoU is slightly better for the LiDAR-supported feature vector (53.09 % and 52.66 % respectively). Compared to the semantic mesh segmentation, recall values increase for almost all classes or are on par. The achieved f1 scores are consistently bigger than 70 % except for *chimney/antenna* and *clutter* (38.38 % and 10.73 % respectively). As we have seen in subsection 4.2, *clutter* and *chimney/antenna* have many mispredictions which are transferred to the point cloud. On the contrary, for class *green space*, we notice the impact of face areas. Whereas its recall value on the mesh is only 44.07 %, the recall value on the point cloud is 76.35 %. Hence, the correctly labeled faces cover a large area consisting of many points.

After the label expansion, the achieved OA is 84 % for all LiDAR points (both with and without LiDAR features). The mIoU is 55.74 % when LiDAR features are used and 54.70 % without LiDAR features.

We are aware of the circular performance analysis since ground truth and prediction on the point cloud are generated in the same way. However, since manually attached ground truth is not available, there is no other possibility to crosscheck the performance quantitatively on the point cloud. Nevertheless, the functionality is demonstrated and visually, the achieved results seem to be reasonable. All in all, the semantic segmentation of 40M points takes 22.7 min. Of these, approximately 90 % is for tree generation and subsequent neighborhood queries that mark the bottleneck of our pipeline.

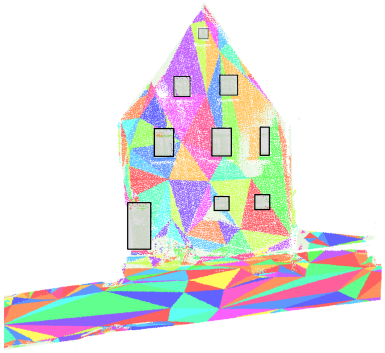

Figure 8. LiDAR points randomly colored by the associated face index. Non-associated points are marked by the black-framed grayish polygons.

5. CONCLUSIONS AND OUTLOOK

In this paper, we presented an association mechanism that facilitates information exchange between (LiDAR) point clouds and meshes. We embedded the association mechanism in our semantic segmentation pipeline (i) to attach LiDAR features to per-face feature vectors and (ii) to transfer mesh labels to the LiDAR point cloud. Hence, the semantic mesh segmentation uses inherent features from both representations (multi-modality). 88.95 % of the surface area is predicted correctly when incorporating LiDAR features. Additionally, as a side product, we achieved semantic point cloud segmentation with an OA of 84 %. Incorporating LiDAR features stabilized the semantic mesh segmentation and improved performance by 2.11 % (mIoU). Particularly, class *vehicle* improved significantly by a recall gain of 19 %.

Taken together, we proved the efficacy and limitations of the association method utilizing a 2.5D mesh. We showed that the association rate suffers from the discrepancy between the 3D point cloud and the 2.5D mesh. After the fast label transfer, non-associated LiDAR points call for a time-consuming

label expansion. Therefore, we would like to leverage a 3D mesh as generated from jointly oriented MVS and LiDAR data in the future. We expect the 3D mesh geometry to improve the association rate and therefore, decrease the expansion time. Furthermore, we want to extend the association mechanism to the image space. The linking of mesh and imagery will link LiDAR point cloud and imagery by transduction, too. Thereby, we do not have to squash the 2D texture information into a 1D feature vector and can use texture information to a full extent. We are aware of the fact that the investigations at hand are limited to one data set of a rather simple urban scene. We would like to extend investigations to more complex urban data captured with different sensors and flight configurations under different conditions (e.g. different seasons). However, such annotated reference data do not exist so far. For this reason, we plan to leverage the proposed association mechanism in combination with crowdsourcing to accelerate ground truth generation.

ACKNOWLEDGEMENTS

The mesh is a result of a research project in collaboration with the German Federal Institute of Hydrology (BfG) in Koblenz.

REFERENCES

Armeni, I., Sax, S., Zamir, A. R., Savarese, S., 2017. Joint 2D-3D-Semantic Data for Indoor Scene Understanding. *CoRR*, abs/1702.01105.

Boulch, A., 2019. Generalizing Discrete Convolutions for Unstructured Point Clouds. *Eurographics Workshop on 3D Object Retrieval*, The Eurographics Association.

Boulch, A., Le Saux, B., Audebert, N., 2017. Unstructured Point Cloud Semantic Labeling Using Deep Segmentation Networks. *Eurographics Workshop on 3D Object Retrieval*, The Eurographics Association.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P., 2016. Geometric Deep Learning: Going Beyond Euclidean Data. *CoRR*, abs/1611.08097.

Chang, J., Gu, J., Wang, L., Meng, G., Xiang, S., Pan, C., 2018. Structure-Aware Convolutional Neural Networks. *Advances in Neural Information Processing Systems 31*, Curran Associates, Inc., 11–20.

Cramer, M., Haala, N., Laupheimer, D., Mandlburger, G., 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.

Ericson, C., 2005. *Real-Time Collision Detection*. Morgan Kaufmann Series in Interactive 3D Technology, Elsevier.

George, D., Xie, X., Tam, G. K. L., 2017. 3D Mesh Segmentation via Multi-branch 1D Convolutional Neural Networks. *CoRR*, abs/1705.11050.

Glira, P., Pfeifer, N., 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.

Griffiths, D., Boehm, J., 2019. A Review on Deep Learning Techniques for 3D Sensed Data Classification. *Remote Sensing*, 11(12).

Hackel, T., Wegner, J. D., Schindler, K., 2016. Fast Semantic Segmentation of 3D Point Clouds With Strongly Varying Density. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, III-3, 177 – 184.

Huang, J., You, S., 2016. Point Cloud Labeling Using 3D Convolutional Neural Network. *23rd International Conference on Pattern Recognition (ICPR)*, 2670–2675.

Landrieu, L., Simonovsky, M., 2017. Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs. *CoRR*, abs/1711.09869.

Laupheimer, D., Shams Eddin, M. H., Haala, N., 2020. The Importance of Radiometric Feature Quality for Semantic Mesh Segmentation. *40. Wissenschaftlich-Technische Jahrestagung der DGPF in Stuttgart*, 29, 205–218.

Lawin, F. J., Danelljan, M., Tosteberg, P., Bhat, G., Khan, F. S., Felsberg, M., 2017. Deep Projective 3D Semantic Segmentation. *CoRR*, abs/1705.03428.

Mandlburger, G., Wenzel, K., Spitzer, A., Haala, N., Glira, P., 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.

Niemeyer, J., Rottensteiner, F., 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.

Qi, C. R., Su, H., Mo, K., Guibas, L. J., 2017a. Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 77–85.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017b. Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *Advances in Neural Information Processing Systems*, 5105–5114.

Qiao, Y.-L., Gao, L., Yang, J., Rosin, P. L., Lai, Y.-K., Chen, X., 2019. LaplacianNet: Learning on 3D Meshes with Laplacian Encoding and Pooling. abs/1910.14063.

Ramirez, P. Z., Paternesi, C., De Gregorio, D., Di Stefano, L., 2019. Shooting Labels: 3D Semantic Labeling by Virtual Reality. *arXiv preprint*, abs/1910.05021.

Rothermel, M., Wenzel, K., Fritsch, D., Haala, N., 2012. SURE: Photogrammetric Surface Reconstruction from Imagery. *Proceedings LC3D Workshop, Berlin*, 8, 2.

Rouhani, M., Lafarge, F., Alliez, P., 2017. Semantic Segmentation of 3D Textured Meshes for Urban Scene Analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 123, 124–139.

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

Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E., 2015. Multi-View Convolutional Neural Networks for 3D Shape Recognition. *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, 945–953.

Tutzauer, P., Laupheimer, D., 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.

Wichmann, A., Agoub, A., Kada, M., 2018. ROOFN3D: Deep Learning Training Data For 3D Building Reconstruction. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2, 1191–1198.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2015. 3D ShapeNets: A Deep Representation for Volumetric Shapes. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1912–1920.

Zolanvari, S. M. I., Ruano, S., Rana, A., Cummins, A., da Silva, R. E., Rahbar, M., Smolic, A., 2019. DublinCity: Annotated LiDAR Point Cloud and its Applications. BMVC, 30th British Machine Vision Conference.