Object identification in 3D urban environments

OLIVIER MARION

Master in Computer Science
Date: December 20, 2018
Supervisor: Pawel Herman
Examiner: Erik Fransén
Swedish title: Identifiera objekt i 3D-stadsmiljöer
School of Electrical Engineering and Computer Science

Abstract

This degree project investigates the problem of object identification in 3D urban models represented by meshes. More specifically, the objective is to detect defects or poorly rendered objects, called artefacts, in order to remove them later on. The literature on related urban analyses is marginal for meshes while abundant for 3D point clouds. The contribution of this project is then twofold: studying whether objects can be identified in meshes and how semantic mesh segmentation methods can be extended to lower-resolution meshes.
This project suggests an unsupervised pipeline algorithm commonly used for object classification in 3D point clouds and investigates alternative solutions for different steps. First, a ground model is generated either from an elevation image-based approach or from direct clustering of the triangles. The latter corresponds to a mesh segmentation problem and was investigated using either k-means or a Markov Random Field formulation. The clustering approach divides the input mesh in different meshes with the following classes: ground, façade, roof and optionally vegetation. The project investigates two new features that can help identify vegetation in lower-resolution meshes. Then, objects are segmented from the ground model using a watershed approach with local maxima as markers and additional propagation constraints based on textures.
As accurate ground-truths were not available for this project, the project results are inspected through visual inspection. Artefact identification for mesh quality improvement is a solvable problem and a feature based on density holds potential for such problems.

Sammanfattning

Detta examensprojekt undersöker problemet med objektidentifikation i 3D-stadsmodeller som representeras av polygonytor. Mer specifikt är målet att upptäcka defekter eller dåligt renderade objekt, artefakter, för att kunna ta bort dem i ett senare skede. Litteraturen om relaterade stadsmodellsanalyser är marginell för polygonytor medan den är riklig för 3D-punktmoln. Projektets bidrag är då dubbelt: Det studerar hur objekt kan identifieras i polygonytor samt hur semantiska polygonytesegmenteringsmetoder kan utvidgas till polygonytor med lägre upplösning.
Detta projekt föreslår en oövervakad pipelinealgoritm som vanligtvis används för objektklassificering i 3D-punktmoln och undersöker alternativa lösningar för de olika stegen. Först genereras en markmodell, antingen från ett höjdbildsbaserat tillvägagångssätt eller från direkt klustring av trianglarna. Det sistnämnda motsvarar en polygonytessegmentering problem och undersöktes med antingen k-means-klustring eller en Markov Random Field-modell. Klustringsmetoden delar polygonytan i separata polygonytor med följande klasser: mark, fasad, tak och eventuellt vegetation. Projektet undersöker två nya egenskaper för representationsvektor som kan hjälpa till att identifiera vegetation i polygonytor med lägre upplösning. Därefter segmenteras objekt från markmodellen med hjälp av ett watershed-tillvägagångssätt med lokala maxima som markörer och ytterligare utbredningsbegränsningar baserat på texturingen.
Eftersom det inte fanns något facit för klassificeringen i detta projekt kontrollerades projektresultaten genom visuell inspektion. Artefaktidentifiering för förbättring av polygonytekvaliteten är ett lösbart problem och en egenskap för representationsvektor baserad på densitet har potential för sådana problem.

Contents

Chapter 1

Introduction

The availability of massive airborne data sets at the scale of entire cities has led to a proliferation of methods for the generation of three-dimensional (3D) city models. Different solutions exist in order to recreate detailed urban environments, varying in resolution and level of details ([37]). 3D city models can be useful for different applications: urban planning, emergency response simulation, virtual tourism and cultural heritage documentation, itinerary planning, accessibility analysis for different types of mobility, to name a few ones. Some of these applications require the models to be more than looking realistic : the models geometry needs to be faithful to reality.
Nowadays, different solutions exist in order to represent cities as 3D textured meshes: LiDAR (light detection and ranging [53]) based techniques have been prevalent over the last decade but recent advances on fully automated multi-view stereo (MVS) workflows have made available high-resolution textured surface meshes. For this project, 3D urban models were generated through Carmenta framework from data coming from several different third-party suppliers. The latter data are usually either automatically created by processing of laser scans cloud data points and their very precise associated aerial photos or from a combination of MVS and airborne surveying and mapping [1]. One important thing to note here is that there is no industry standard or common practice for exactly how the data is organised, leading to assumptions during the reconstruction.
Such methods of reconstruction however often raise a problem: the resulting models typically contain 3D artefacts from trees, cars and other objects that make the models less realistic at zoomed levels, as shown

in Fig. 1.1. Also, these artefacts increase the model complexity by adding unwanted elements and leading to extra computations judged unnecessary, for instance shadow mapping of tree artefacts. Within the scope of this project, artefacts include both imperfections in the model and capturing of entities that change over time, such as trees or cars.
The identification of artefacts therefore plays a part not only in increasing a model quality but also in reducing its complexity, leading to better performances for applications. A logical follow-up to identification would be handling the artefacts, either by removing them, ignoring them later on during processing of the model or replacing them by simpler structures.
Identifying artefacts, or more generally objects, in a 3D scene corresponds to performing a semantic analysis of the scene. These analyses are usually carried out by manual assisted approaches INSERT REF, leading to time consuming procedures, unsuitable for large scale applications. Therefore, automatic methods for urban environments analysis are needed.
Finally, even though data become available on a larger scale, their resolution is varying and for a lot of applications it is not conceivable to have very high and long processing times on very detailed structures. If a mesh resolution is sufficient for humans to grasp a scene semantics at first glance, it may also be possible for algorithms to adapt to lower-quality meshes.

Figure 1.1: Artefacts from trees that make the model less realistic at a zoomed-in level

1.1 Problem statement

Problems in this section / thesis : Visual inspection: This is problematic as it does not really allow for a systematic evaluation/comparison.

You should at least formulate or define your own criteria, rather quantitative.
This Master's thesis project belongs to the urban modelling and analysis field, a computer vision problem involving 3D computer graphics and machine learning. The goal of this Master's thesis project is to examine the problem of automatic identification of objects (artefacts) in 3D urban meshes. More particularly, this project will study whether it is possible to generalise an already-existing unsupervised algorithm to lower-resolution meshes. The comparisons will be performed on datasets representing different cities and generated by different providers, with varying levels of detail and the results will be evaluated through visual inspection.
The contributions of this project can be formulated as follows:

- Extending already existing semantic mesh segmentation techniques to object identification
- Studying the influence of mesh resolution on a state-of-the-art unsupervised mesh segmentation technique
- Suggestion of new geometrical features for object description in lower-resolution meshes
- Suggestion of an unsupervised artefact identification framework for meshes

1.2 The Employer

Consider removing / replacing by: it would be desirable to either include a new subsection or mention here about the scope, assumptions in the project
The employer, Carmenta, develops an advanced toolkit named Carmenta Engine to work with geospatial data, including 3D city models. Carmenta is currently interested in improving the quality of their 3D models, which can be achieved by automatic identification of artefacts, so that they can be removed later on.

1.3 Thesis Outline

The thesis starts with an introduction including a problem statement, followed by the Background chapter describing prior knowledge in machine learning and mesh segmentation relevant to the understanding of the thesis scientific content and placement in literature. The thesis then moves on to the Related Works chapter, presenting a more in-depth review of existing and recent literature related to the thesis content. Finally, the Methods, Results and Discussion chapters present the work carried out for this project and the conclusions drawn from it as well as suggestions for further improvements.

Chapter 2

Background

2.1 Machine Learning overview

Machine learning is a field of artificial intelligence that include a wide variety of techniques to give computers the ability to "learn" or infer models from data, without having any relationships in the data or models explicitly programmed. Machine learning techniques usually involve a training phase where the algorithms try to infer relationships or models parameters within the training data, as opposed to testing or prediction phases where the trained algorithm will predict outputs of new input data based on the learned model.
Supervised learning corresponds to when the training data consists in data points x_i and their respective output y_i. Unsupervised learning corresponds to when the training data only consists in data points x_i. Clustering is an example of unsupervised learning and consists in grouping a set of objects in such a way that objects in the same group, called a cluster, are more similar, according to a chosen metric, to each other than to those in other clusters.
Classification is when the model being learned produces discrete outputs, and the outputs are often referred to as labels. Regression is when the model being learned produces continuous outputs, such as function estimation.
A feature is a one dimensional value describing part of a data point and when all features are combined into a feature vector, they then correspond to the input to the algorithm.

2.1.1 Support Vector Machines (Chang et al. [7])

SVM is a supervised learning technique and non-probabilistic binary linear classifier.
Given a training dataset of n points of the form $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$ where the y_i are either 1 or -1, each indicating the class to which the point $\vec{x}_i$ belongs. Each $\vec{x}_i$ is a p-dimensional real vector.
The aim is to find the "maximum-margin hyperplane" that divides the group of points $\vec{x}_i$ for which $y_i = 1$ from the group of points for which $y_i = -1$, which is defined so that the distance between the hyperplane and the nearest point $\vec{x}_i$ from either group is maximized. Any hyperplane can be written as the set of points $\vec{x}$ satisfying $\vec{w} \cdot \vec{x} - b = 0,$.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces via kernel functions and scalar products in said higher-dimensional space.
SVMs are robust classifiers that perform well with limited amount of training data and high–dimensional data.

2.1.2 Data Clustering Techniques

There are fewer unsupervised learning, or data clustering, techniques present in the literature as opposed to supervised techniques. (Borji et al. [4]) presents and compares some common unsupervised algorithm on computer vision-oriented problems.

K-means

K-means (Lloyd [31]), minimises the sum of squared errors between data points and their nearest cluster centres. It is a simple and widely used method.

Fuzzy C-means

Fuzzy C-Means (Dunn [15]) assigns soft labels to data points meaning that each data point can belong to more than one cluster with different degrees of membership.

Spectral Clustering

Spectral Clustering regroups several algorithms and is very commonly used in Data Mining for graph clustering. It consists in a spectral analysis of the matrix of point-to-point similarities instead of estimating an explicit model of data distribution (as in k-Means). Normalised Cut (Shi et al. [54]) and Ng-Jordan-Weiss algorithm (Ng et al. [39]) are two examples of spectral clustering algorithms.

Mean Shift

Mean shift (Comaniciu et al. [14]) seeks the modes of a density function from discrete samples of that function. Mean Shift performs as follows. First, it fixes a window around each data point. Then, computes the mean of data within each window. Finally, shifts the window to the mean and repeats till convergence.

Deep Convolutional Neural Networks (CNNs)

This is a description of the algorithm proposed by Borji et al. [4].
CNNs have sparked a lot of interest recently as they have been very successful with a lot of different machine learning problems, including computer vision problems. CNNs learn representations through several stages of non-linear processing, similarly to how the cortex biologically adapts to visualise and adapt to the visual world. Such a human-perception based sounds promising as computer vision criteria often consist in human intuitions and perceptions of the results.
The authors use back propagation via stochastic gradient descent to optimise a clustering objective to learn the mapping, which is parameterised by a deep neural network. In this way, there is no need to specify parameters like number of clusters, distance measure, scale, cluster centres, etc.

2.2 Data Type

3D point clouds and meshes can both be considered as discretised 3D functions representations.

2.2.1 3D cloud data points

A point cloud is a collection of data points defined by a given coordinates system. In a 3D coordinates system, for example, a point cloud may define the shape of some real or created physical system. Colours can also be associated to the coordinates.
There are many techniques for converting a point cloud to a 3D surface.

2.2.2 Mesh

A polygon mesh is a set of vertex positions and a set of polygonal facets defined by an ordered list of vertex indices. In general, the facets are triangles. A mesh edge corresponds to a polygon edge, that is to say a segment connecting two vertices.
A texture is a 2D rectangular image that is being mapped onto the vertices of the mesh. Intuitively, a mesh is the 3D shape of model while the texture corresponds to a coloration of each facet. A mesh structural properties will be referred to as geometric information and properties extracted from textures as photometric information, given that the textures are usually generated from aerial pictures.

2.2.3 Comparison

Meshes are usually created from processing of point clouds. According to [45], point clouds are simple and unified structures that avoid the combinatorial irregularities and complexities of meshes, and thus are easier to learn from.
Meshes generally do not enclose a volume, as they are mostly used to model surface information for visualisation purposes whereas structures, for instance buildings, are usually characterised by a higher density in point clouds.

2.3 Mesh Segmentation

Mesh segmentation is an active research field of geometry processing robotics and computer vision. A semantic segmentation of a mesh would be dividing it into meaningful components, following human

intuition.

Let $M = \{V, E, F\}$ a mesh where V corresponds to the mesh vertices, E the mesh edges and F the facets and S be either V, E or more usually F. Also called partitioning or clustering, a segmentation is a set of sub-meshes induced by a partition of S into k disjoint subsets. These subsets can be called regions or segments. A mesh segmentation can usually be formulated by specifying two key elements: a function measuring the quality of a partition, eventually under a set of constraints, and a mechanism for finding an optimal partition (Lafarge [27]).

A key aspect in mesh segmentation approaches is the design of feature vectors encoding geometric and photometric information. Geometric information correspond to information contained in the mesh while photometric information correspond to information contained in the textures. There is a very wide variety of features, also called feature descriptors, signatures or shape descriptors, that can be used to design the feature vectors.

2.3.1 Region growing

Region growing is a method commonly used in image processing and more specifically image segmentation. Its goal is to partition an image, or in our case a mesh, by examining initial seed points, determining whether the point that are neighbours to these initial seeds should be added to the region and then iterate on the neighbours of the newly added points until there are no suitable points left to add.
This method calls for the definition of several concepts:

- data points, or the entities we want to regroup through the algorithm,
  - pixels for images.
  - vertices or faces for meshes.
- an adjacency relationship between points, as in Fig. 2.1,
  - In 2D images, we can consider either Von Neumann neighbourhoods (4-connected neighbourhoods) or Moore neighbourhoods (8-connected neighborhoods).

- In meshes, if we are considering vertices, then two vertices are adjacent if they are connected by an edge. If we are considering facets, then two facets are adjacent if they share an edge.

– a similarity measure to compare two points,

– a threshold to decide when to accept or reject a new point into a region.

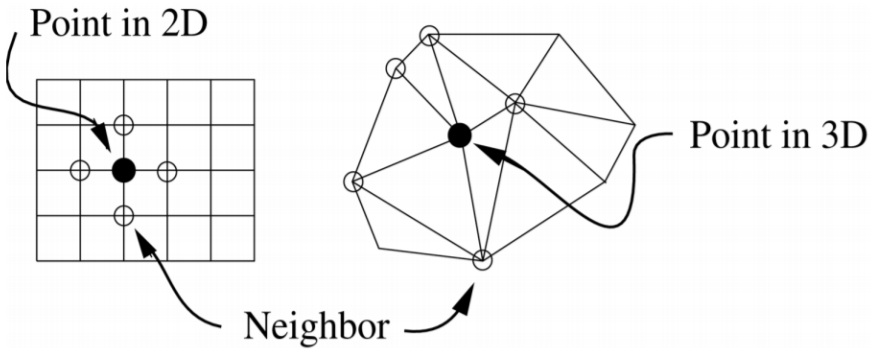


Figure 2.1: Adjacency rules in 2D and in 3D, taken from [32]

A region growing pseudo-algorithm for the segmentation algorithm presented in [41] is then:

Data: Mesh
Result: Labels for each facet
initialisation;
while *the mesh not entirely segmented* **do**
  choose an unlabelled facet f_0;
  label[f_0]:=k;
  neighbours_list:= all f_n adjacent to f_0;
  while *neighbours_list not empty* **do**
    f_n:=neighbours_list.pop(first_element) ;
    if *similarity*(f_n,f_0) < *similarity_threshold* **then**
      label[f_n]:=k;
      neighbours_list.extend(all f_i adjacent to f_n)
    end
  end
  k:=k+1
end

Algorithm 1: Region growing segmentation

2.3.2 Watershed

Watershed is another example of techniques borrowed from image processing. It can be generalised from a 2D rectilinear grid to an arbitrary surface with well-defined neighbour connectivity, as presented in Mangan et al. [32] with meshes mainly. The watershed concept can also be efficiently used with 3D cloud data points (Serna et al. [52] or Hernandez et al. [24]).
The watershed algorithm derives its name from the manner in which regions are segmented into catchment basins. Let $f : X \rightarrow \mathbf{R}$ be a height function, X being the set of all vertices in the mesh. Typically, f can be a curvature function or a regular height function such as $f : (x, y, z) \rightarrow z$.

The algorithm for mesh segmentation using watershed segmentation described in Mangan et al. [32] has the following steps:

1. Compute the height function at each vertex.
2. Find the local minima and assign each a unique label
3. Find each flat area and classify it as a minimum or a plateau

4. Loop through plateaus and allow each one to descend until a labelled region is encountered.
5. Allow all remaining unlabelled vertices to similarly descend and join to labelled regions
6. Merge regions whose watershed depth (Fig. 2.2) is below a preset threshold.

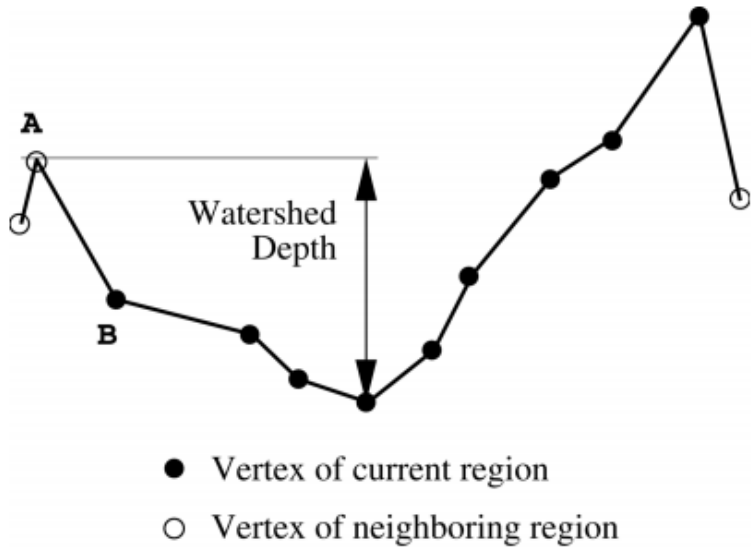


Figure 2.2: Defining the depth of a region based on its lowest vertex and lowest boundary vertex, taken from [32]

This solution strategy can be extended by bringing additional information such as triangle coloring or texture coordinate, or other saliency measures.
Although used as a mesh segmentation technique in the original article, it can also be used as an intermediate step, either for grouping similar facets as pre-processing or isolating objects, in a similar manner as in [52].

2.3.3 Markov Random Field

Markov Random Fields find a wide range of applications in computer vision and machine learning, from image processing with denoising

applications to semantic scene partitioning.

A Markov Random Field consists in a set of random variables having a Markov property described by an undirected graph.
A Markov property refers to the memoryless property of a stochastic process, that is if the conditional probability distribution of future states of the process (conditional on both past and present values) depends only upon the present state; that is, given the present, the future does not depend on the past.
A MRF is always associated to a neighbourhood system defining the dependency between graph nodes. $N = \{n(i)\,, i \in V\}$ is a neighbourhood system if

- $i \notin n(i)$
- $i \in n(j) \Leftrightarrow j \in n(i)$

The markovian property and the neighbourhood system ensure that each random variable cannot be dependent to all the other ones and reduce complexity by spatial considerations.

MRF formulation for meshes

Markov Random Fields for meshes

- Graph nodes := vertices & graph edges := edges
- Graph nodes := facets & graph edges := edges
- Graph nodes := edges & graph edges := facets

MRF formulations usually correspond to minimising an energy. The energy is commonly composed of two terms: a data term that measures the coherence of each datum with respect to a label, and a pairwise potential that favours label smoothness. Let $G = (V, E)$ a graph and $l = (l_i)_{0 \leq i < |V|}$ a label configuration for the graph nodes

$$U(l) = \underbrace{\sum_{i \in V} D_i(l_i)}_{\text{Data term}} + \gamma \underbrace{\sum_{\{i,j\} \in E} V_{ij}(l_i, l_j)}_{\text{Pairwise potential}}, \qquad \gamma > 0$$

When in a Bayesian case,

data term $= -log($likelihood$)$ and pairwise potential $= -log($pairwise interaction prior$)$

MRF formulated as an energy minimisation makes it possible to use efficient algorithms to find the optimal labelling such as simulated annealing, graph-cut based approaches, Monte Carlo sampling...

MRF solving through graph cuts

Graph Cuts finds the optimal solution to a binary problem. However when each node can be assigned many labels, finding the solution can be computationally expensive. For the aforementioned type of energy, graph cuts can be used subsequently to find a convenient local minimum.
General algorithm from Boykov et al. [6]:

Let L a label set and E an energy function

1. Start with an arbitrary labelling f
2. Set success := 0
3. For each pair of labels $\{\alpha, \beta\} \subset L$
   (a) Find $\hat{f} = \arg\min E(f')$ among f' within one $\alpha - \beta$ swap of f
   (b) If $E(\hat{f}) < E(f)$, set $f := \hat{f}$ and success := 1
5. If success = 1 goto 2
6. Return f

Figure 2.3: Energy minimisation algorithm with Graph Cuts

The main idea of the alpha-beta swap algorithm is to successively segment all α nodes from β nodes with graph cuts and the algorithm will change the $\alpha - \beta$ combination at each iteration. The algorithm will iterate through each possible combination until it converges.
Within an iteration (step 3. of algorithm 2.3), a graph (cf Fig. 2.4) is constructed: it consists in all vertices with label α and β, two additional terminal nodes named α and β. There are then two types of edges: n-links or edges between vertices (the ones corresponding to the mesh) and t-links or edges between a vertex and a terminal node. Details on the weights can be found in (Boykov et al. [6]).

The energy of a configuration is then equivalent to the capacity of the minimum cut in the graph, and can be found through max-flow/min-cut capacity as proposed in Boykov et al. [5].

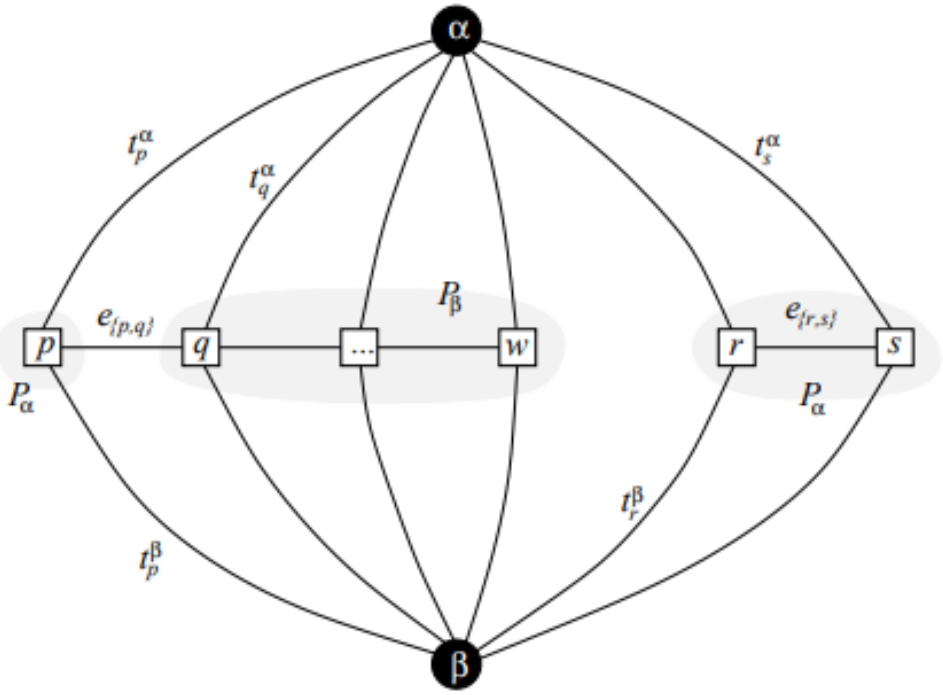


Figure 2.4: Taken from [6]: An example of the graph $G_{\alpha\beta}$ for a 1D image. The set of pixels in the image is $P_{\alpha\beta} = P_\alpha \cup P_\beta$ where {vertices with label α} $= P_\alpha = \{p, r, s\}$ and {vertices with label β} $= P_\beta = \{q, ..., w\}$

2.3.4 Metrics

Zhang et al. [64] provide a study on quality assessment of *2D images* segmentation, classifying methods in five groups:

- **Analytical methods**. These take into account the characteristics of an algorithm such as principles, requirements, complexity, ...
- **Subjective methods**. The evaluation is performed in a subjective way with segmentation results judged by a human operator. This evaluation is very biased and therefore requires a large set of objects and a large group of humans and cannot be integrated in an automatic metric evaluation.

- **System level evaluation methods**. This kind of methods indicates if the characteristics of the results obtained by a segmentation algorithm are suited for the over-all system which uses this segmentation algorithm.
- **Empirical goodness or unsupervised methods**. The performance of the algorithms is evaluated by judging the quality of the segmented images themselves. To achieve this task, a set of quality is established according to human intuition about what conditions should be satisfied by an ideal segmentation. It is however difficult to specify a priori criteria for the evaluation of a segmentation quality.
- **Empirical discrepancy or supervised methods**. A set of references images representing the ideal segmentations constitutes a ground-truth. The evaluation is then carried out by measuring the discrepancy between the ground-truth and the segmentation algorithm results.

These categories can be extended to 3D segmentations. However, lacking a ground-truth segmentation, a lot of semantic segmentations of urban meshes are evaluated through visual inspection and designing metrics is an active research field. Benhabiles et al. [3] designed metrics following a set of principles for segmentations comparison. A reliable measure of mesh segmentation similarity has to possess the following set of properties:

- **No degenerative cases**. The score's measure must be proportional to the similarity degree between an automatic segmentation and the ground-truth segmentations of the same model.
- **Tolerance to refinement**. Some humans can provide a coarse segmentation while others will provide a finer one. The common element being that both segmentations remain consistent. Thus a segmentation measure has to be invariant to granularity differences between segmentations.
- **Cardinality independence**. Two segmentations compared can have different numbers of segments and different sizes of segments.

- **Tolerance to cut boundary imprecision**. The boundaries tend to vary between two similar segments and thus do not have such an importance from a semantic point of view.
- **Tolerance to multiple ground-truth**.
- **Meaningful comparison**. The scores obtained by the measure have to allow a meaningful comparison between different segmentations of the same model and between segmentations of different models. For the first case (segmentations of the same model), the scores have to vary according to the segmentation quality, then, the more the automatic segmentation is similar to the ground-truth segmentations of the same model, and better the score is. For the second case (segmentations of different models), the scores have to indicate which kind of 3D-models is the most convenient to segment by an automatic algorithm

Finally, Benhabiles et al. [3] categorise mesh segmentation measures into three categories:

- **Boundary matching**. This kind of measure computes the mapping degree between the extracted region boundaries of two segmentations.
- **Region differencing**. These measures compute the consistency degree between the regions produced by two segmentations.
- **Non-parametric tests.** Different non-parametric measures can be found in the statistical literature. Some of them are variants of the Rand index. This index converts the problem of comparing two segmentations with different numbers of segments into a problem of computing pairwise label relationships. The Rand index can be computed as the ratio of the number of pairs of vertices or faces having the compatible label relationship in both segmentations.

Chapter 3

Related Work

This chapter presents a review of the most recent and related works on 3D automatic analysis of urban environments, which is to say semantic segmentation with focus on methods dealing with meshes of urban scenes and objects detection in urban 3D point clouds. Even though 3D acquisition systems have reached a high maturity level, 3D automatic analysis of urban areas is still an active research area.

3.1 Mesh and Image oriented techniques

Classification

Image, or point cloud, partitioning in computer vision is a recurrent problem that many different classification approaches have tried to solve. In general, the aim is to partition the input data into labelled areas with criteria following a human intuition of the result. Classification approaches differ mainly in level of supervision, chosen features to extract and use of spatial dependencies and contextual information. Many supervised approaches can be found through literature and Rouhani et al. [47] propose a detailed and urban mesh segmentation-oriented overview of these approaches.
Clustering and classification tend to be studied separately in computer vision or even machine learning in general. Neural Networks, in principle, learn from the data the same way humans do and therefore hold potential for applications with human-oriented standards for results. Borji et al. [4] propose a Deep Convolutional Neural Network (CNNs) for human-like clustering inspired after the use of CNNs in semantic

segmentation problems and compare their solution to already existing clustering algorithms (presented in Section 2.1.2)..

MRFs and CRFs

MRFs and CRFs are preferred classification techniques for semantic segmentation: in MRFs and CRFs, a datum classification depends on the rest of the data, as the classification decision relies on non-local information accounting for spatial consistency between neighbouring areas.
Generally, the results of semantic segmentation can be improved with information derived form the context. Ladický et al. [26] use co-occurence statistics by recording which pairs of classes are likely to occur in the same image. Myeong et al. [38] describe region pairwise relationship through a similarity graph. Finally, semantic segmentation and object detection can be improved by combining both global and local contexts (Mottaghi et al. [36]) to improve both semantic segmentation and object detection.

Mesh segmentation

There is a large variety of mesh segmentation algorithms in the literature, shared with image processing and computer vision fields. One of the simplest way to deal with mesh segmentation is by representing it as an unsupervised clustering problem based on specific geometric criteria (Shlafman et al. [55]). Region growing (Page et al. [41]) and spectral analysis (Zhang et al. [63]) are other examples of deterministic approaches. MRFs and CRFs are probabilistic approaches that help capture contextual and spatial consistency (Lafarge et al. [28]). These probabilistic approaches allow to have solutions ranging from unsupervised segmentations to supervised segmentations: Verdie et al. [58] design three geometric attributes for a labelling cost function to define the unary term while Rouhani et al. [47] predict it.
Finally, Neural Networks and Deep Learning techniques have also been applied to mesh segmentation problems. George et al. [18] propose Multi-Branch 1-dimensional CNNs on 11-feature vectors extracted from the input mesh. Shu et al. [56] perform 3D shape segmentation and co-segmentation using deep learning with the following steps: i) generation of primitive patches for each shape input; ii) extraction of several shape descriptors from each patch and concatenate them into

a resulting feature vector; iii) use the previously obtained feature vectors as input for a deep neural network in order to general a high-level feature space; iv) segmentation by performing a clustering operation in the high-level feature space.

Image-based techniques in Multi-View Stereo

As 2D image segmentation and analysis is an important segment of the image processing literature, in MVS contexts, rather than processing the output mesh, some methods first perform classification directly from the images before mapping it to the output 3D model (He et al. [22]). Lafarge et al. [28] propose a hybrid approach, refining the output model while detecting regular urban objects. The incremental approach proposed by Vineet et al. [59] operates in near real time and delivers a rough reconstruction with a street-based semantics. Xiao et al. [61] propose a larger MRF that includes all the views, and models all connections between the associated areas. The smoothness term between two views is defined either based on the colour similarity or using the number of common feature tracks between the two associated images. These image-based methods are compute-intensive and insufficiently exploit the geometric properties of the observed scene (Rouhani et al. [47]).

3.2 Point cloud oriented techniques

ALS: Aerial Laser Scan;
MLS: Mobile Laser Scan;
TLS: Terrestrial Laser Scan;

Elevation images

Point clouds are usually very dense and their processing lead to high computation time and complexity. Therefore, 3D information is commonly projected onto a 2D grid. When the information contained in each pixel corresponds to elevation information, the grid is commonly referred to as range or elevation image, or digital elevation model. These 2.5D images have a long tradition in the scientific community (Hoover et al. [25]) and are still of interest due to technological advances in remote sensing, with the Kinect for instance (Saponara Iri-

arte Paniagua [50]). Gorte [20] presents a method to segment planes on TLS data using range images. They obtain a so-called panoramic range image and estimate planes for each pixel of the image. They then regroup pixels belonging to a same plane through region-growing. Hernandez et al. [24], improved later on by Serna et al. [52] propose a solution with the following steps: i) projection of 3D point cloud to elevation images; ii) ground segmentation through λ-flat zones algorithm (Meyer [34]); iii) object detection based on mathematical morphology transformations; iv) object classification using an SVM classifier.

Real-time applications and autonomous driving

Real-time applications favour elevation images processing as it is both precise and fast. Falling in that category, approaches for autonomous vehicles generally require average accuracy and high speed in order to detect and predict obstacles in real time. Recent LIDAR-based methods place 3D windows in 3D voxel grids to score the point cloud [62, 16] or apply CNNs to the front view point map in a dense box prediction scheme Li et al. [30]. Image-based methods [8, 9] typically first generate 3D box proposals and then perform region-based recognition. Methods based on LIDAR point cloud usually achieve more accurate 3D locations while image-based methods have higher accuracy in terms of 2D box evaluation. Chen et al. [10] combine LIDAR and images for 3D detection by employing a deep fusion scheme.

General semantic segmentation

Several general segmentation and classification frameworks can be also found in the literature. Golovinskiy et al. [19] develop a set of algorithms to detect, segment, characterise and classify urban objects. Their pipeline is as follows: i) ground segmentation using graph cuts, ii) object detection and segmentation using hierarchical clustering, iii) object characterisation using geometrical and contextual descriptors, and iv) object classification using SVM. More recently, Velizhev et al. [57] have improved this workflow including spin images and implicit shape models. The major problems of these approaches are noise, sparse sampling and proximity between objects. Moreover, some prior knowledge about the object scale is required to set up thresholds.
Schnabel et al. [51] present a semantic system for 3D shape detection. Their algorithm consists in two main steps: i) a topology graph is built

with primitive shapes extracted from the data; ii) a search is carried out in order to detect characteristic subgraphs of semantic entities. The main drawback is the graph complexity when dealing with non-trivial objects. Pu et al. [44] propose a framework for segmenting and classifying urban objects from MLS data. This work starts with a rough classification into three large categories: ground, on-ground objects and off-ground objects. Then, based on geometrical attributes and topological relations, more detailed classes such as traffic signs, trees, building walls and barriers are recognised.

3.3 Urban modelling

3D urban analysis is a topic which is often included in urban modelling or urban reconstruction, which regroups a high volume of literature entries while still facing many unsolved problems (Musialski et al. [37]): extracting semantics from the input data is an important and challenging step of urban reconstruction. The classes of interest generally consist in stationary elements such as buildings, roads or eventually trees.
Rottensteiner et al. [46] provide a comprehensive survey on both 3D urban reconstruction and urban object classification (2D outlines of urban objects in the input data). Non-local strategies exploiting spatial and contextual information in urban scenes have proven rather efficient, both on images (Volpi et al. [60] and Montoya-Zegarra et al. [35]) and on 3D point clouds (Lai et al. [29] and Niemeyer et al. [40]).
Trees are a special object of focus for urban modelling as it is usually complicated to capture their shape and usually results in complex models calling for simplification. Rutzinger et al. [49] describe an automated workflow to segment and model trees from MLS data: i) input point cloud is segmented into planar regions using the 3D Hough Transform and surface growing algorithms; ii) remaining small segments are merged through connectivity analysis; iii) non-tree objects are excluded from the analysis; iv) trees are thinned and realistic 3D models are generated. Babahajiani et al. [2] propose an algorithm for urban 3D segmentation and modelling from street view images and Lidar point clouds. Their proposal for urban segmentation is the following: they first isolate road points from the point cloud by locally

computing minimum z-values and then performing plane fitting on the lowest points; then, they segment the building façades through a rule-based segmentation using height and density features. They then use a boosted decision tree detector for super-voxel features to classify the different objects present in the remaining point ensemble.
Literature on analysis of urban meshes is more marginal. Verdie et al. [58] segment an input mesh into four classes: ground, facades, roof and vegetation. This unsupervised classification relies on three geometric features: elevation, planarity and verticality used in labelling cost function modelling the unary term of a MRF. Rouhani et al. [47] propose a supervised approach based on Verdie et al. [58] geometric features and introduce a joint-labelling strategy as well as photometric features. Martinović et al. [33] use a supervised classifier modelling a labelling cost function. They also use a corrective post-processing step.

Chapter 4

Methods

The iterature on object detection in meshes is rather marginal. Therefore, the present work suggests a solution adapted from object-detection-and-classification algorithms on point clouds (Hernandez et al. [24] and Serna et al. [52] or Babahajiani et al. [2]). The suggested algorithm pipeline will have the following steps:
Maybe use a chart here

1. Pre-processing
2. Primary clustering (Ground detection)
3. Post-processing
4. Objects detection
5. Objects classification

The input data corresponds to an urban textured mesh.

4.1 Pre-processing

This section covers computations performed at the beginning of the pipeline, either in order to achieve better consistency in measures, reduce computation time or to gather information about the whole mesh.

4.1.1 Connectivity analysis

Meshes are usually unified and internally-connected structures. However, it is rather common for automated reconstruction methods to fail

to capture certain details leading to the presence of disconnected components within the mesh.
The general idea is then to perform region growing on the triangles, with adjacency being the only constraint. To be considered adjacent, two triangles must share an edge. The algorithm result then consists in all of the connected components present in the mesh. Whether a cluster of triangles should be considered as unwanted is then decided based on comparing the number of triangles contained in the component to an arbitrary threshold.
There are several things that should be kept in mind while conducting such a connectivity analysis, depending mainly on the reconstructed model. If the acquisition and triangulation methods are precise enough, the resulting model might not contain any disconnected components. When the model contains different levels of detail, higher levels of detail can correspond to such disconnected component, and therefore do not need to be removed. Finally, it is common during the data acquisition and the reconstruction steps to process smaller areas or tiles one at a time. One direct implication is that objects at the border between two areas might be divided in two separate groups only matching visually.

4.1.2 Division into sub-meshes

Urban meshes modelling an entire city contain a lot of triangles and direct processing of the entire model would lead to unpractical computation time and memory requirements. The original mesh is therefore divided into smaller meshes. Hernandez et al. [24] separate city blocks using the Hough transform to detect façades direction, under the assumption that the façades of a same street are aligned. This assumption is however not always verified across the datasets used for this work.
The only considerations for this division are memory and computation time reduction, therefore the mesh is divided into arbitrary-sized rectangular meshes, judged large enough to contain semantic information about large structures such as buildings.

4.1.3 Elevation and density maps

This step is actually performed before dividing the original mesh into sub-meshes. It is useful either in order to obtain an elevation image

and perform a mathematical morphology-oriented strategy as in [52, 24] or to help compute the elevation and density features for the mesh segmentation strategy. For the latter, computing values before division of the mesh ensures consistency near the separation lines.

Elevation image

The following paragraphs describe how to get an elevation image from the mesh vertices. Unlike point clouds, mesh vertex ensembles are sparse and would not be enough to estimate surfaces without the additional information provided by the faces. Therefore, in an attempt to adapt point cloud solutions to meshes, additional vertices need to be generated. One way is to sample new vertices from triangles according to the grid we want to project the vertices on and the constrained plane equation defined by a triangle.
We consider the rectangle bounding the projection of the triangle T on the XY plane (in red in figure 4.1). For each grid point (i, j) contained in the bounding rectangle, if (i, j) is within the projection of the triangle, then a new vertex $v_{new} = (i, j, z_{new})$ is created from the equation of the plane passing through T (the plane is defined by the triangle normal and one vertex from the triangle). One can determine whether a point is inside a triangle using a barycentric coordinate system in the triangle:
A point (p_x, p_y) is in triangle $T = \{(v_{0x}, v_{0y}), (v_{1x}, v_{1y}), (v_{2x}, v_{2y})\}$
$\iff \forall x_i$ in $x, x_i \geq 0$ with x solution of $Ax = b$
where $A = \begin{bmatrix} v_{0x} & v_{1x} & v_{2x} \\ v_{0y} & v_{1y} & v_{2y} \\ 1 & 1 & 1 \end{bmatrix}$ and $b = \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}$

Once the vertices are obtained, they are projected onto the grid similarly to [52] and we obtain two arrays: one for the minimal height in a local neighbourhood represented by the grid resolution and another one for the maximal height in the same local neighbourhood.

4.1.4 Superfacet partitioning

As input meshes are very dense, labelling each facet requires a lot of computations and leads to impractical computation times with an MRF formulation. Superfacet partitioning [47, 58] consists in grouping facets into small clusters, an over-segmentation analogous to su-

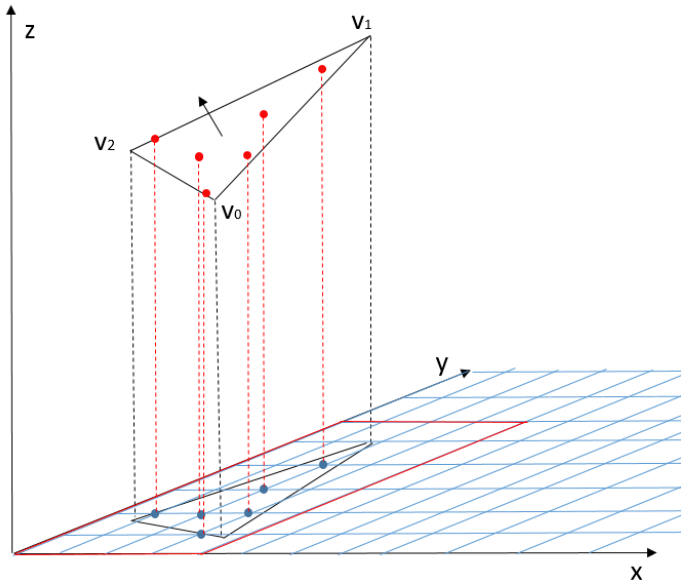


Figure 4.1: New vertices generation from triangle sampling

perpixels in image analysis.
Clustering is performed through a region growing approach over all triangles based on a similarity measure. A region grows until the similarity measure exceeds a user-specified threshold. Additionally, a constraint on superfacet areas can be formulated.
In a lot of applications, choosing a similarity measure and the corresponding threshold corresponds to the challenging part. The following sections present different choices for similarity measures.

Shape operator

The shape operator is a measure of how a regular surface bends in $\mathbb{R}^3$. Evaluated at a point on a surface, it is a linear transformation of the tangent space that measures how the surface bends in different directions [21]. The normal curvature of a surface in some direction is the reciprocal of the radius of the circle that best approximates a normal slice of surface in that direction.
Rusinkiewicz [48] provides an algorithm for the estimation of the curvature and the second fundamental form, whose matrix formulation is the same as the shape operator's. This estimation method outperforms Cohen-Steiner and Morvan algorithm Cohen-Steiner et al. [12], used by Verdie et al. [58].

This metric identifies the nearly planar components and preserves sharp features. Comparison can be performed via the Frobenius norm on the estimated second fundamental form matrix.

Normal comparison

Used in Rouhani et al. [47], Cohen-Steiner et al. [13] introduce $\mathcal{L}^{2,1}$ as Shape metric. It is essentially an $\mathcal{L}^2$ measure of the normal field:

$$\mathcal{L}^{2,1}(R_i, P_i) = \iint_{x \in R_i} ||\mathbf{n}(x) - \mathbf{n}_i||^2 dx$$

with R_i a region of a partition and $P_i = (x_i, \mathbf{n}_i)$ a planar proxy passing through x_i with normal $\mathbf{n}_i$ approximating R_i.

Intuitively, normal orientation and co-planar regions give a lot of information on the structure of an object. The previously explained measure can simply amount to a normal comparison during region growing, with the threshold being a maximum angle: a facet is admitted if the angle difference between its normal and the average normal of the region is smaller than the threshold.

Colour comparison

Jointly used in Rouhani et al. [47] with the covariance-based shape estimator from Cohen-Steiner et al. [13], photometric information helps preserving image discontinuities of the texture map. It is computed as the L1-distance in the colour space. Colours help us perceive the boundaries of objects and recognise some of them, trees for instance are characterised by green tones. Therefore, if available, trying to use photometric information for clustering similar facets follows human intuition and perception.

4.2 Primary clustering

This section mainly presents two suggestions for ground detection in a sub-mesh. The first one is an adaptation of a point cloud strategy adopted in [24, 52] to meshes. The second one extends [58] to meshes with varying resolutions, as it struggles with tree detection as the mesh resolution decreases.

4.2.1 Feature extraction

In order to capture the information contained in triangles or the mesh in general, features must be computed. They are used mainly for triangles labelling in the mesh segmentation approach but were also used for vertices clustering in the adaption of the point cloud approach. Elevation, planarity and horizontality (or verticality) were first introduced by Verdie et al. [58].
When using superfacets, for each superfacet, the feature value is the area-weighted sum of the feature value of its triangle facets.

Elevation

Elevation discriminates the ground from higher objects such as roofs. It measures the height of a point with respect to the ground.

$$\text{Elevation}(p(x,y,z)) = \sqrt{\frac{z - z_{min}(x,y)}{z_{max}(x,y) - z_{min}(x,y)}}$$

where z_{min} and z_{max} are respectively the minimal and maximal z-coordinates in a local window on the XY plane and z is the z-coordinate of the point. If we are looking at facets, z is the z-coordinate of the barycenter of the triangle.
The z_{min} and z_{max} are pre-computed during pre-processing. The size of the local window must be large enough to meet ground areas and small enough to offer robustness to terrain height variation. Using open source data such as Open Street Map, the window sizes could be estimated automatically based on buildings dimensions: the window size L such that the window has an area of $L \times L$ should be larger than the smaller dimension of the larger building bounding rectangle in the scene.

Planarity

Planarity was derived from the surface variation introduced by Pauly et al. [43]: it is computed from the minimum eigenvalues of the covariance matrix computed in closed form over all triangle facets of the superfacet and measures how much the superfacet deviates from the local tangent plane, yielding 1 for planar superfacets and 0 for isotropic superfacets.

In order to get a facet-oriented measure, this work suggests for planarity measure the average cosine angle between a facet normal and its neighbouring facets unit normals:

$$\text{Planarity}(t) = \frac{1}{|\mathcal{N}(t)|} \sum_{t_i \in \mathcal{N}(t)} |\mathbf{n}(t) \cdot \mathbf{n}(t_i)|$$

where $\mathcal{N}(t)$ is the neighbourhood ensemble of a facet t.
In Verdie et al. [58], planarity is used to discriminate vegetation from the other three classes: roof, façade, ground. However, this feature might not be very efficient for lower-resolution meshes for tree identification.

Horizontality

This feature measures the deviation of the unit normal $\mathbf{n}$ of a facet to the vertical axis $\mathbf{z}$:

$$\text{Horizontality} = |\mathbf{n} \cdot \mathbf{z}|$$

A low horizontality value corresponds to vertical components, which corresponds to façades mainly. A high horizontality value corresponds to horizontal components, typically roofs and roads.

Density

This feature comes from analyses performed on point clouds: LiDAR reconstructions provide dense and high regions for building façades [2]. In meshes, façades and trees lead to a higher number of triangles in a small window of the XY plane while flat areas are usually sparse in terms of triangles and vertices.
In the pre-processing part, vertices are assigned to $1m \times 1m$ areas. The resulting vertices count array is then clamped and normalised. The clamping rule is the following:

if vertices_count[i,j] > mean(vertices_count) + $\sqrt{\text{variance(vertices\_count)}}$
then vertices_count[i,j]=mean(vertices_count) + $\sqrt{\text{variance(vertices\_count)}}$

Which results in a value close to 0 for sparse areas and close to 1 for areas with a high number of vertices.

Greenness

This feature also intends to discriminate trees from other components. Greenness corresponds to how green a triangle is, computed from the Hue value in Hue-Saturation-Value (HSV) colour space. HSV space describes colour in way that is similar to human perception. The Hue value corresponds to an angle on the chromatic circle, on which green is represented by 120°. In computer vision, the sensibility s for a colour perception is commonly 30-40°around the colour. This yields:

$$\text{Greenness}(\text{Hue}, s) = \frac{|\text{Hue} - 120|}{s} \mathbb{1}_{|\text{Hue}-120|)<s}$$

Identifying tree triangles based on their colours feels rather intuitive, although not systematic as trees are quite obviously not stationary and depending on when the acquisition was performed, they might not be green. This feature is however a rather simple way to use texture information in an unsupervised approach.

Photometric features

The photometric information contained in textured meshes provide complementary clues for facet classification. HSV space is usually preferred over RGB space as it is more effective to discriminate objects with different reflectivity properties. Rouhani et al. [47] use photometric features in HSV space, designed for each superfacet:

- the average colour,
- the standard deviation of its colour distribution,
- discretised colour distribution by clustering HSV colour values of the whole texture map into a color palette [11].

4.2.2 Elevation map and mathematical morphology strategy adaptation

This section is inspired by the elevation map approach in [52, 24]. Their strategy for ground segmentation is based solely on elevation images and morphological operations on the images. Unlike point cloud, urban meshes are mostly surface representations and therefore discriminating maximal and minimal elevation images can not be generated from them. Adapting a point cloud approach hence also means

adapting its use case: here, the elevation map is intended for road and planar areas discrimination from buildings. Then, having this ground model, object detection can be performed on triangles present in this model.

λ-flat zones labelling algorithm

Based on Meyer [34] labelling algorithm in image processing, [24, 52] obtain the ground mask as the largest λ-flat zones (Definition 1) in the elevation image.

Definition. *Let f be a digital gray-scale image* $f : D \subset \mathbb{Z}^2 \rightarrow V = [0, \ldots, R]$. *Two neighbouring pixels p,q belong to the same λ-flat zone of f , if their difference* $|f_p - f_q|$ *is smaller than or equal to a given* λ *value.* $\forall x \in D$, *let* $A_x(\lambda)$ *be the λ-flat zones of image f containing pixel x.*
$A_x(\lambda) = \{x\} \cup \{q | \exists P = (p_1 = x, \ldots, p_n = q) \, such \, that \, |f_{p_j} - f_{p_{j+1}}| \leq \lambda\}$

With this definition, the ground mask $g_m(f)$ is $g_m(f) = \arg \max_{x \in D}(|A_x(\lambda)|$.

In the eventuality of a finely captured mesh, this approach can be used as such to generate the ground mask. However, as the resolution decreases, the ground disappears under large objects such as trees. This has for effect to stop the region growing in the middle of the role and to have several $A_x(\lambda)$ corresponding to the ground.
In order to choose relevant regions, it is possible to look for local minima in the elevation image and consider regions containing these minima.
Optionally, the resulting ground mask can be smoothed via morphological closing.

Clustering pixels using k-means

Alternatively to using region growing via λ-quasi-flat zones, it is also possible to cluster the pixels of the elevation image using features computed from triangles mapped to the image: elevation and horizontality. Once k-means is performed, the cluster centre with the minimal elevation corresponds to the ground label. The ground mask is then all pixels having the ground label. Finally, k-means can be used in both superfacet- and a facet-oriented approaches.

Backprojection

Backprojection consists in going from the binary image to labelled triangles. Triangles mapped onto the ground mask pixels are labelled as ground (and the other ones as 'other'). However, some pixels might contain façades triangles and in order to label them correctly, an additional clustering, similar to the one applied on pixels, separates ground triangles from façades triangles: minimal horizontality corresponds to façades and minimal elevation to the ground.

4.2.3 Unsupervised triangles classification

Directly labelling triangles corresponds to a mesh segmentation approach.

Clustering using k-means

Clustering triangles using k-means is fast and only requires to extract features from the triangles as introduced previously. k-means also allows a flexible use of features and works in a predictable way and semantic classes labels can be inferred from the cluster centres. The predicted classes however are dependent on the data to be clustered and non-representative sub-meshes might lead to errors in the expected results.

Clustering using MRF

An MRF with pairwise interactions formulation is a more complicated approach than k-means, but it also corresponds to a more guided and comprehensive approach: by designing unary terms similarly to Verdie et al. [58], classes formulation does not depend on the data while providing contextual and spatial consistency. Such an approach however raises the problem of how well features and unary terms can represent the desired classes.
Four classes are considered: $\{façade, roof, vegetation, ground\}$. Using the same notations as in Section 2.3.3, the energy of a label configuration on superfacets can be written as:

$$U(l) = \sum_{i \in S} D_i(l_i) + \gamma \sum_{\{i,j\} \in E} V_{ij}(l_i, l_j), \qquad \gamma > 0$$

where S denotes the set of superfacets and E denotes all pairs of adjacent superfacets, which is to say superfacets sharing at least one edge in the input mesh.
The data term D combines the previously-described features weighted by the area A_i of the superfacet i.
We refer to planarity as a_p, horizontality as a_h, elevation as a_e , density as a_d and greenness as a_g and $\overline{a} = 1 - a$.

$$D_i(l_i) = A_i \times \begin{cases} 1 - \overline{a_e} \cdot a_p \cdot a_h \cdot \overline{a_d} & \text{if } l_i = ground \\ 1 - \overline{a_p} \cdot a_h \cdot a_d & \text{if } l_i = vegetation \\ 1 - a_p \cdot \overline{a_h} \cdot a_d & \text{if } l_i = façade \\ 1 - a_e \cdot a_p \cdot a_h \cdot \overline{a_d} & \text{if } l_i = roof \end{cases}$$

The vegetation unary term can also be formulated as : $1 - \overline{a_p} \cdot a_h \cdot a_g$.
The pairwise interaction V between two adjacent superfacets i and j favors label smoothness:

$$V_{i,j}(l_i, l_j) = C_{i,j} \cdot w_{i,j} \cdot \mathbb{1}_{\{l_i \neq l_j\}}$$

where C_{ij} is the sum of interface (mutual) edge lengths between superfacet i and j. w_{ij} is a weight introduced to prevent label propagation over sharp creases and is defined as the angle cosine between the estimated normals of two superfacets. As the transition between roof and ground is judged impossible, it has a fixed high cost. Such a measure offers additional robustness for the elevation feature when encountering large flat areas.
As the unary data term and pairwise potential are weighted by the superfacet areas and interface lengths, this energy formulation behaves similarly to a triangle facet-based energy with grouping constraints. An approximate solution to this energy minimisation problem is solved through the $\alpha - \beta$ swap algorithm (Boykov et al. [6]).

4.3 Post-processing rules

Post-processing rules play a dual role. In the event of completely unsupervised methods, these rules correspond to the inclusion of the human rationale. They also help achieve a smoother and more inclusive ground mesh, in order to facilitate the object detection step.

4.3.1 Surrounding rule

This rule has smoothing and consistency-enforcing motivations: if all neighbours of a triangle T have the same label l_n and the label of T is $l_T \neq l_n$, then the label of T is changed to the label of its neighbours.

4.3.2 Watershed rule

This rule makes sure the ground label contains all lower triangles. It also smooths the ground mesh in a watershed-like manner: if a triangle are 'below' a ground triangle, then they are ground triangles as well.
Definition. t_1 below $t_2 \iff \text{PointOfMass}(t_1).z < \text{PointOfMass}(t_2).z$

4.3.3 Verdie et al. [58] semantic rules

Two types of errors frequently occur when dealing with complex urban scenes: structures on top of roofs such as chimneys often get mislabelled as vegetation and vertical components in large vegetation components get mislabelled as façades.

- **Rule 1.** Superfacets labelled as vegetation and adjacent to only superfacets labelled as roof are re-labelled as roof.
- **Rule 2.** Superfacets labelled as façade and adjacent to superfacets labelled as vegetation and ground are re-labelled as vegetation.

4.4 Objects detection

The previous steps segmented an over-inclusive ground mesh from the input sub-mesh. For the two following steps, it is critical for this ground mesh to be correctly segmented. Additionally, if the previous steps are perfectly calibrated and executed, then the ground mesh should only include small objects close to the ground. For this section, analogously to Hernandez et al. [24], artefacts, or objects, are assimilated to humps on the ground. The suggested method to detect and segment them is a constrained watershed from local maxima, based on a height function and additional similarity criteria. Mangan et al. [32] propose insights for watershed on surface meshes. In essence, constrained watershed on meshes is growing regions whose

seeds are local maxima and with constraints both on height and similarity. The height function used here is simply the z-coordinate of a point: height$(x, y, z) = z$. It was chosen over alternative height functions such as curvature as it is easier to expect results and anticipate problems.

Definition. $A\,vertex\,v\,is\,a\,local\,maximum$
$\iff v.z > w.z, \forall\, vertex\,w\,connected\,to\,v\,by\,an\,edge\,(of\,a\,triangle)$

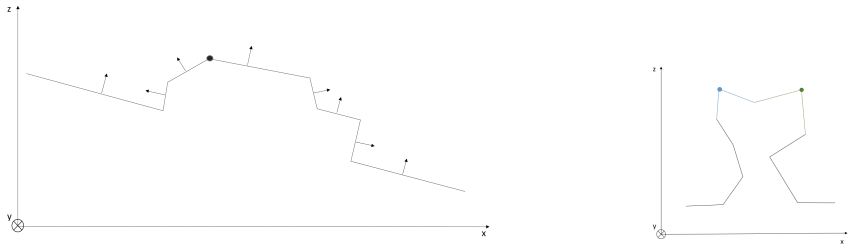
For the watershed, local maxima are used as markers and one marker will correspond to one object. Under the assumption that one triangle should belong to only one object at a time, rules are introduced to implement this rationale:

- If two local maxima are connected by an edge, they correspond to the same object.
- All triangles connected to a local maxima are considered part of the object. Let's refer to them as seed triangles.
- If two seed triangles from two different objects are adjacent, the two objects are merged into one (Fig 4.2b).

In simple cases, where the ground is flat, segmenting objects is similar to [24]: from a marker, the object corresponds to all triangles below that marker until the ground is met. However, on sloping ground, the height constraint is no longer sufficient when a labelled ground is not available (Fig 4.2a). Constraints based on angle changes between two adjacent triangle normals would also fail to completely capture certain types of objects such as cars (Fig 4.2a). Therefore, the suggestion for propagation constraint is the correlation distance between colour distributions of two triangles.
The correlation distance between u and v is defined as:

$$correlation(u, v) = 1 - \frac{(u - \overline{u}) \cdot (v - \overline{v})}{||u - \overline{u}||_2 \cdot ||v - \overline{v}||_2}$$

Colour distributions approach the human intuition perception of an object limits based on texture information: colour helps the visual system analyse complex images more efficiently, improving object recognition, and is commonly used in computer vision problems [42]. The correlation distance is a common way to compare histograms, the discretised representation of distributions.



(a) Car profile on sloping ground

(b) A case of two markers corresponding to one object

Figure 4.2: Watershed for object detection. Local maxima are represented as dots

4.5 Object categorisation

Research has been carried out on mesh objects supervised classification and has proven efficient [PUT A REF HERE]. However, it was not possible due to the thesis context and schedule to generate and label manually a large and representative enough set of objects. In the eventuality of the presence of labelled data, the classification approach could be using an SVM classifier with features similar to those used in Serna et al. [52], adapted to meshes.
The proposed approach is more oriented towards an engineering way to deal with the humps: by flattening them. The borders of the mesh contain local maxima and therefore objects often corresponding to lower chunks of façades. In order to prevent flattening these, a clustering based on both elevation and horizontality features using k-means is chosen.

4.6 Evaluation + Implementation

Important part, needs to be elaborated on

Chapter 5

Results

Implementation

The work was implemented in Python using Carmenta Engine to process the 3D models, OpenCV for textures processing, Networkx for graph structures and min-cut implementation and Sklearn for clustering purposes.

Datasets

For this report, four datasets coming from three different providers were used. Bastia and Paris datasets were generated using Acute3D and aerial mapping technology by Ubick [1]. Marseille dataset was generated using Pixel Factory [17], a Data Management and Processing Software by Airbus. Finally, Helsinki dataset is an open source 3D city model of Finland's capital [23].
In terms of quality of reconstruction, based on average size of triangles and overall aspect, Marseille and Helsinki datasets have finer resolutions than Paris and Bastia.

Evaluation

The results are examined through visual inspection mostly. The background section on metrics for mesh segmentation presents several metrics, however these can not be meaningfully used without any ground-truth. In such cases, visual inspection of the results remains the dominating criterion of evaluation.

5.1 Elevation images and mathematical morphology

The result pictures for this section were generated on the following segment of the Bastia mesh (the bounds are symbolised by the red box), as seen in Fig. 5.1. The segment corresponds to an urban segment, with narrow streets and high buildings opening onto a square.

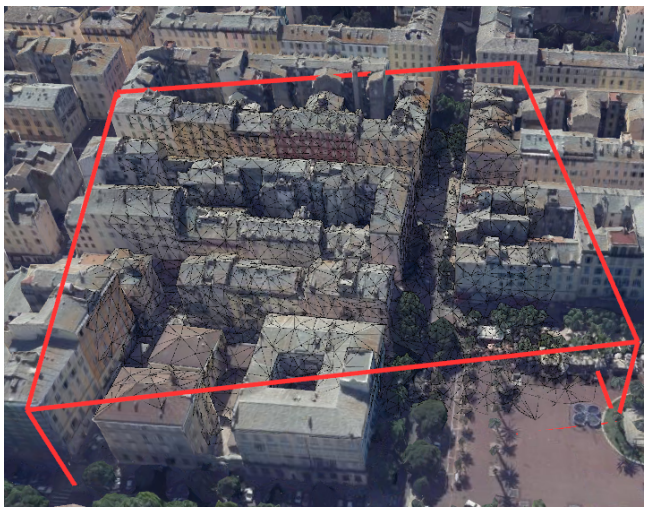


Figure 5.1: Bastia segment for elevation image strategy

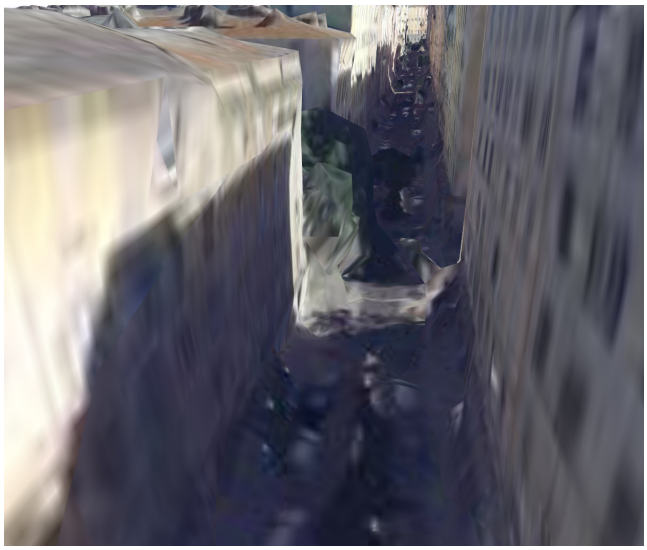


Figure 5.2: Side view of a street from Bastia dataset

5.1.1 Vertices interpolation for elevation images

Fig. 5.3 illustrates the effects of interpolating more vertices for an elevation image approach. In a mesh representation, the vertices distribution is much sparser than in point clouds and the resulting elevation image is not usable. However, vertices interpolation leads to a satisfying elevation image, where it is clearly possible to distinguish objects outline.

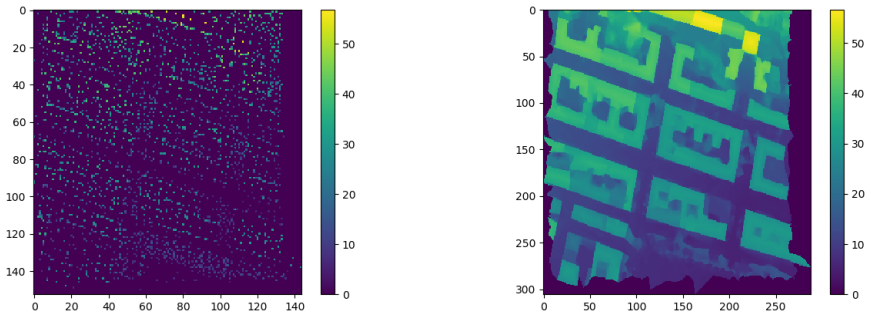


Figure 5.3: Vertices interpolation for minimal elevation images

Although such a method does not make up for the fact that the ground is not always represented, especially under wide objects such as trees, it seems promising enough to infer ground location as opposed to buildings location.

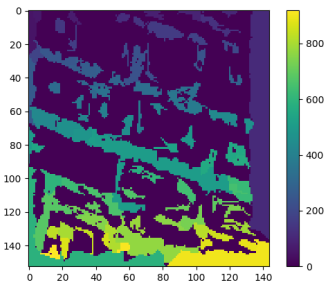
5.1.2 Ground model generation

As presented in the Methods chapter, two strategies were considered for obtaining a ground model from the elevation image.

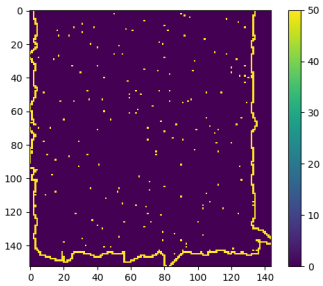
Using $\lambda-$flat zone algorithm

The results of the $\lambda-$flat zone algorithm for $\lambda = 0.5$ are represented in Fig. 5.4.a, consisting in many segments due to the lack of continuity of the ground in the mesh representation. Fig. 5.4.c represents the final ground model obtained by selecting only 0.5m-quasi flat zones containing a local minimum of the elevation image (Fig. 5.4.b). This approach is very promising for high-resolution datasets. The irregularities around the buildings outline come from the presence of cars

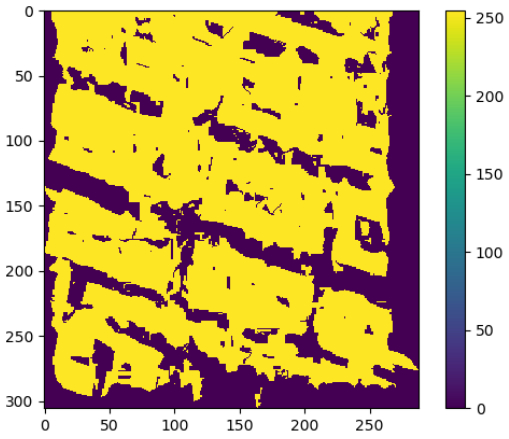
parked on the side of the street (as for instance in Fig. 5.2).
However, in the case of Bastia dataset, areas that should be flat are sometimes captured as bumps and finding the perfect λ gets complicated as it becomes almost impossible to discriminate between bumps corresponding to the road and other objects such as cars.



(a) Local minima



(b) $\lambda-$zones



(c) Resulting ground model for λ=0.5m (ground in purple)

Figure 5.4: $\lambda-$zone algorithm based ground detection

Using k-means clustering

The ground model is depicted in Fig. 5.5. The result approaches the intuition of the result, with the model indicating the zones where the objects we want to detect and identify will be located. It however struggles in the top right corner that corresponds to a steep slop into a court yard with high trees surrounded by buildings.

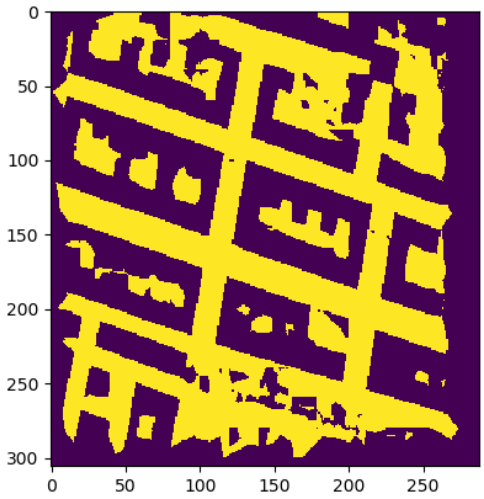


Figure 5.5: Ground model (yellow) using k-means clustering on the vertices

5.1.3 Backprojection

The backprojection, represented on Fig. 5.6, is carried out on triangles projected onto the ground mask. However, using only verticality and elevation as clustering criteria, the façades are clustered but also some vertical triangles with the ground section.

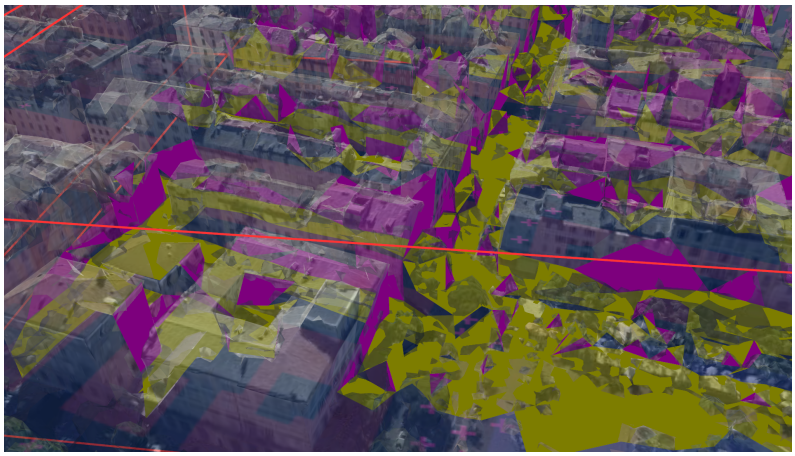


Figure 5.6: Backprojection of triangles belonging to the groundmodel

5.1.4 Conclusion

Although efficient on high-resolution meshes with rather low terrain elevation changes, the use of the $\lambda-$algorithm strategy is rather greedy in terms of computations and suffers on lower-resolution meshes. Performing clustering on interpolated vertices yields rather satisfying results, offering robustness to variability in a mesh resolution. However, when adopting such a strategy, the backprojection represents a challenge.
Overall, this approach is worth investigating and represents a historical step in this degree project. However, when faced with performing clustering on triangle properties for backprojection, the logical following step was to investigate direct clustering of the triangles.

5.2 Feature extraction

Feature extraction is a critical step in mesh segmentation, especially more so for an unsupervised approach, such as the one used in this project. The features need to be able to describe and discriminate different objects. Although geometrical features such as elevation and horizontality perform very well for structures such as roofs and façades in flat urban environments, complications arise as smaller structures are captured on roads and the ground slope gets steeper.

5.2.1 Elevation

Elevation is a rather straight-forward feature but the choice of the window size is critical to adapt to buildings in hilly environments and large flat terrains. Needless to say that a unique window size is most of the time not sufficient.
Fig 5.7 shows the elevation feature of each triangle with a window size of 50m (which means an rectangular area of 50m x 50m around the triangle). This window size is not enough to accommodate for very large flat areas such as the middle square where slight changes in height lead to high variations of the elevation features. It can also be noticed that lower structures close to very high buildings will have lower elevation scores as well.

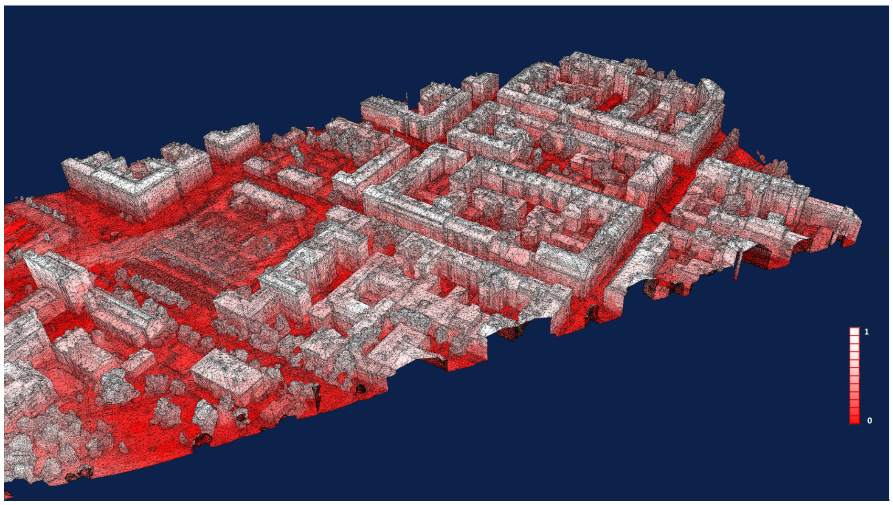


Figure 5.7: Elevation feature on Helsinki data using a window of 50m

5.2.2 Horizontality

Of all the features presented in this project, the horizontality feature is probably the most straight-forward and the easiest to compute, as it does not require any user-input parameters or approximations. Presented on a segment of Bastia (Fig. 5.13), the horizontality value of the triangles is visualised on Fig. 5.8. It appears to be very efficient for the clustering of very geometrically shaped structures such as buildings.

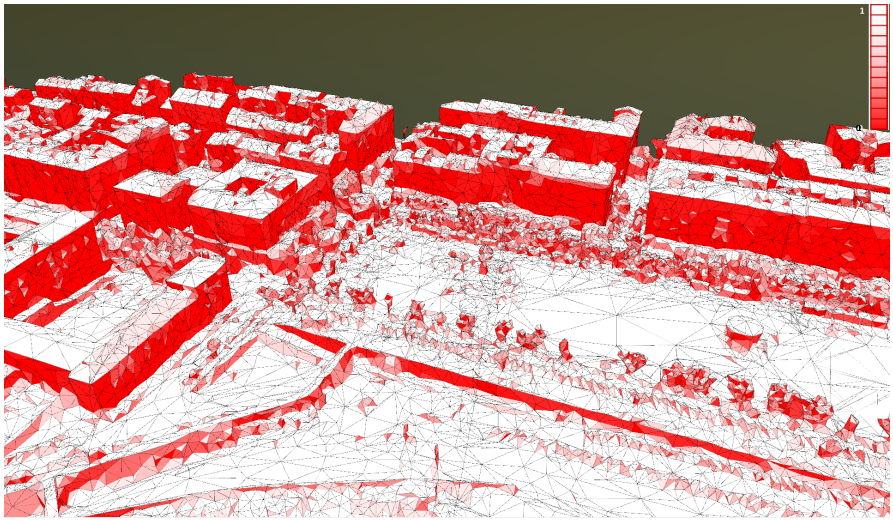


Figure 5.8: Horizontality feature visualisation on Bastia dataset

5.2.3 Planarity

Planarity is a more problematic feature. It is represented on Fig. 5.9, on the same segment of Bastia mesh as previously (Fig. 5.13). In ideal cases or high-resolution models, buildings and grounds would score high while irregular objects such as trees would score low. However, as in the case of Bastia dataset, the reconstruction quality leads to bumps in place of flat areas, therefore leading to lower-scoring for areas that were initially planar.

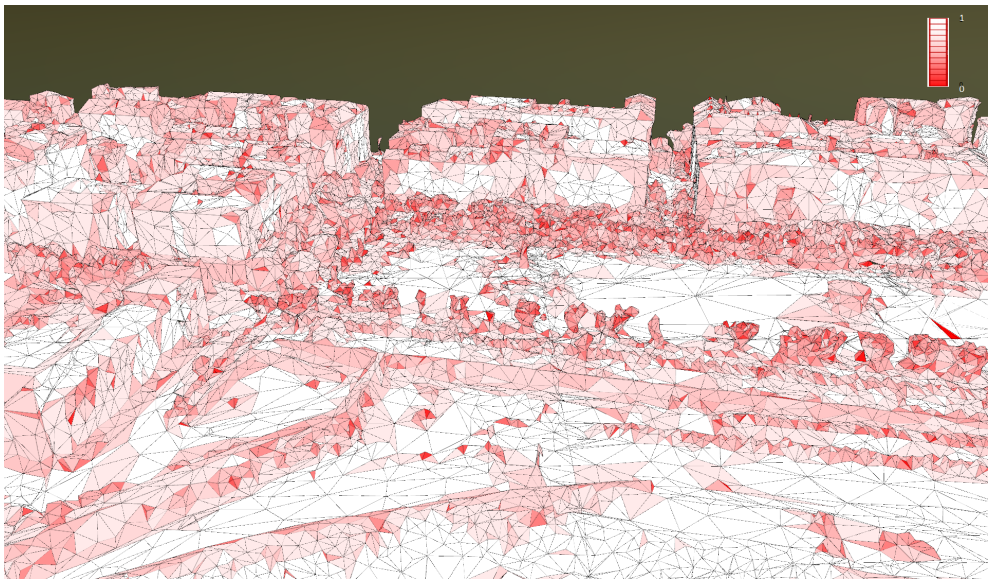


Figure 5.9: Planarity feature visualisation on Bastia dataset

5.2.4 Greenness

The greenness feature was forgotten rather fast when exploring relevant features for trees detection. One of the main reasons, that can be observed on Fig. 5.10 and Fig. 5.11, is that although the overall colour of the texture mapping of trees triangles is green, it contains a lot of black pixels whose hue value is not green and therefore the average HSV colour value of the triangle would not score high in terms of greenness even though it contains green.

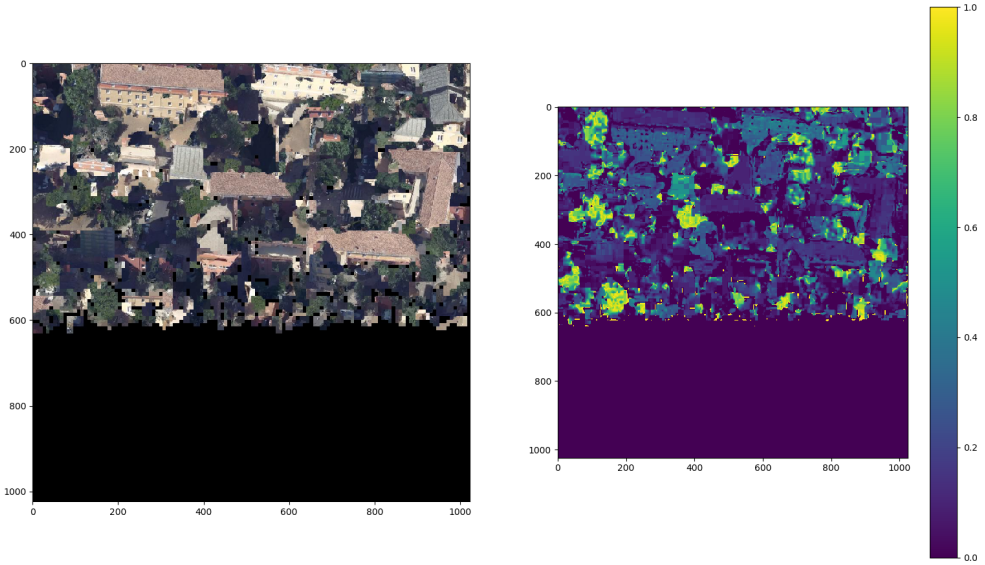


Figure 5.10: Texture from Bastia dataset and associated pixel-wise greenness

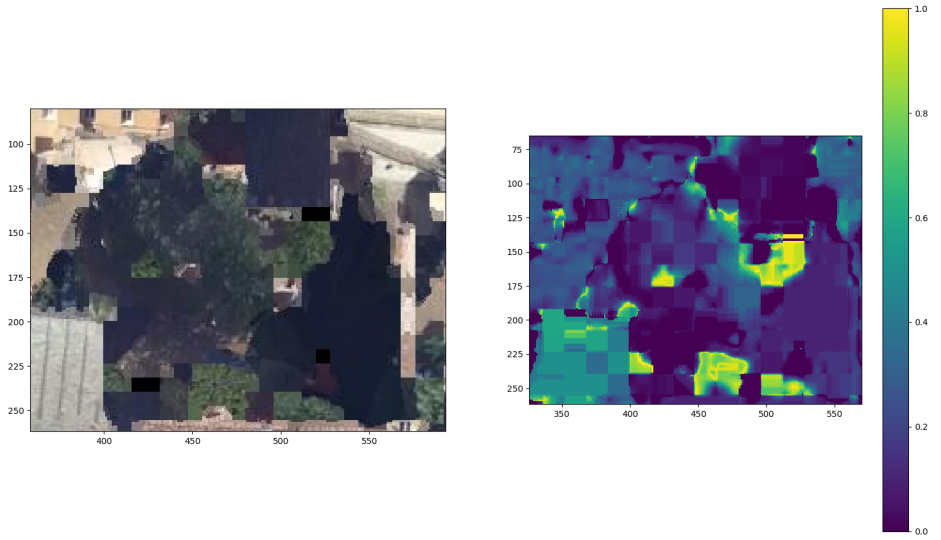


Figure 5.11: Zoom-up on a portion of Fig. 5.10

5.2.5 Density

The density feature feature seems very promising. Computed from the vertices count on 1m x 1m windows, the resulting image was smoothed using an average filter with kernel of size 10x10 before computing the

density feature value as presented in the Methods section.
Bastia and Helsinki datasets present different resolutions, with Bastia dataset having a lower-resolution leading to many bumps in flat areas and poorly reconstructed trees while Helsinki dataset has a higher resolution, with more triangles for objects representation. Fig. 5.12 and Fig. 5.13 represent the density value for 1m x 1m areas. On the picture, it is rather evident that trees and façades mark the density map with high scores while grounds and roofs score lower values. On Helsinki dataset, smaller objects such as cars parked in a square can also be recognised from the pictures, opening to the possibility of using it for objects detection within the mesh as well.

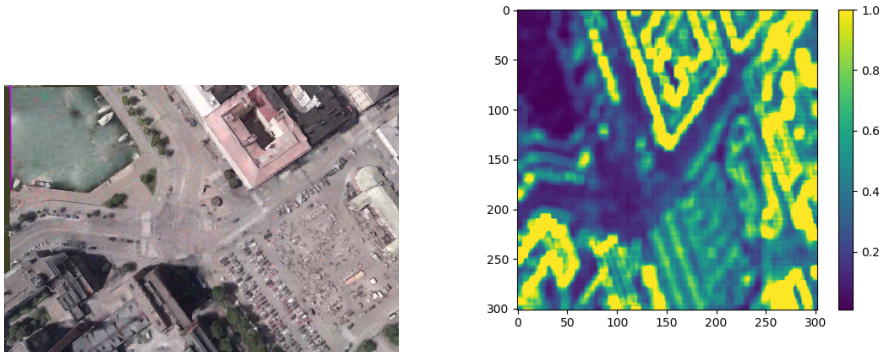


Figure 5.12: Density feature evaluation on Helsinki dataset

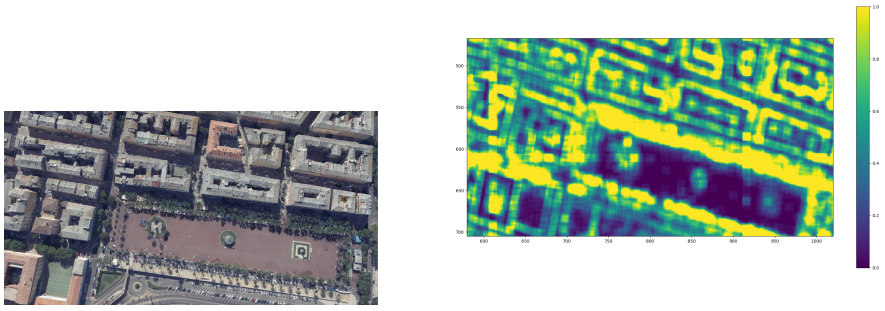


Figure 5.13: Density feature evaluation on Bastia dataset

5.2.6 Conclusion

Features selection is critical for correct classification of triangles, in order to capture discriminative information. The set of features used

in this work are mostly geometric and seem suitable to adapt to both higher- and lower-resolution meshes for artefact detection.

5.3 Artefacts detection algorithm

This section presents the results for several steps of the algorithm as well general conclusions on the techniques used and their potential for artefact detection in lower-resolution meshes.

5.3.1 Connectivity analysis

For finer resolution datasets such as Helsinki dataset, there are not any significant results while performing this analysis. However, when carried out on Bastia dataset, two main types of disconnected components were identified (Fig. 5.15). The first ones (Fig. 5.14), whose detection and identification can be interesting, tree artefacts, small bushes-like components either visually connected to the rest of the tree or simply flying above the location of the tree in the original data. The second ones correspond to manual fixes so that the data visually matches, such as the building corner segment seen on Fig. 5.15.

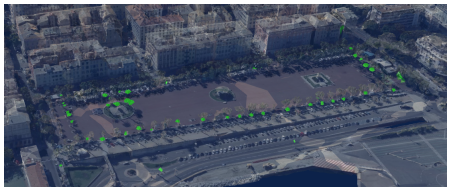


Figure 5.14: Connectivity analysis in Bastia: tree artefacts (bright green)

Figure 5.15: Connectivity analysis in Bastia: Two main types of disconnected components

Connectivity analysis results could be used to isolate objects and add them to the pool of objects to be classified in the last step of the algorithm. Additionally, the number of found disconnected components could be used as an indicator of the quality of a model reconstruction.

5.3.2 k-means

The k-means approach represented a chronological step in the project timeline: this method is easier to apprehend and available in libraries. As the results (see Fig.) were rather encouraging at first, the k-means approach helped put more weight and focus on feature extraction and serves as a control indicator as it provides information on the data being clustered. Additionally, k-means formulation allows to evaluate faster the influence and discriminating power of features without having to design unary terms.
Performed on Fig. 5.1, the results of the triangles clustering is presented in Fig. 5.16

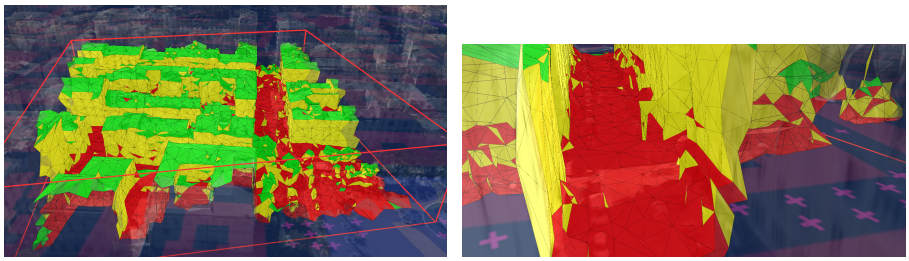


Figure 5.16: k-means clustering on Bastia dataset

5.3.3 Postprocessing rules

First designed for k-means as a way to include contextual information, they were kept even with the MRF formulation as they help pally minor misclassifications. Postprocessing rules were applied on Fig. 5.16 and the result is presented in Fig. 5.17.

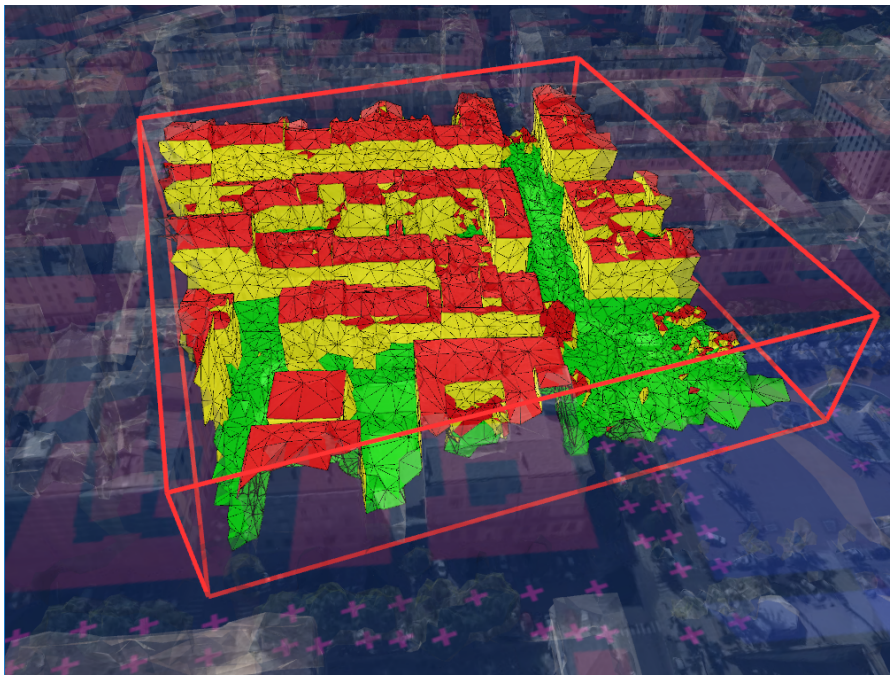


Figure 5.17: Postprocessing rules on k-means clustering

5.3.4 Superfacet clustering

The urge for superfacets arose when the MRF formulation was chosen, as it significantly improved computation times without . However, due to the resolution of meshes and that triangles are in general larger in the datasets used in this work compared to those used in the literature: for instance, the 300mx300m Paris mesh Rouhani et al. [47] use contains around 1M triangular facets while a similar portion of the Paris dataset used in this work contains around 10 times less triangles, that is to say about 100k triangles.
Therefore, the superfacet clustering, otherwise less needed, used with a size threshold of 100m^2, a geometrical threshold of 30°and a photometric threshold of 60 in the colour space, performed in Paris has the following characteristics in terms of superfacets size (number of triangles) and area: $mean_{size} = 2.15, variance_{size} = 5.17, minmax_{size} = (1, 22)$ and $mean_{area} = 16.05, variance_{area} = 635.88, minmax_{area} = (7.42e{-}05, 148.09)$.
Note that the maximal area exceeding the threshold comes from single triangles representing flat terrain.

5.3.5 MRF formulation

The MRF formulation performs rather well according to an anticipated rationale. In this section, we show that the formulation used in Verdie et al. [58] is not enough to accommodate to lower resolution meshes. In order to demonstrate this statement, the four-classes formulation was run onto a segment of Paris dataset, similar to the one used in the original paper. The results are shown in Fig. 5.18. On Fig.5.19, a similar formulation is used, removing the vegetation class.
The latter formulation yields rather satisfying results across the different datasets, with in general the same problems and misclassification as in the other papers: vegetation misclassified as partial roof and façade, large flat areas having parts misclassified as façade and buildings. The buildings detection with this formulation still holds and allows to generate a ground mesh from which objects can be segmented.

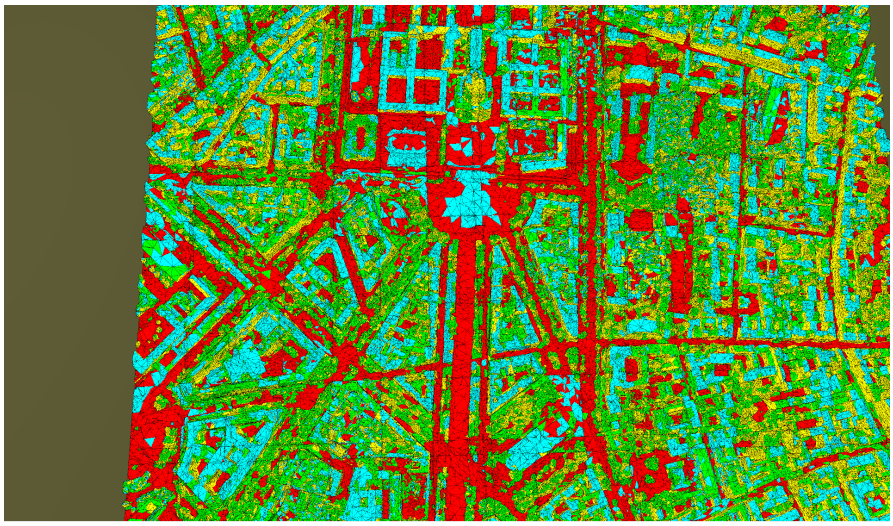


Figure 5.18: Verdie orginal four-class-MRF formulation on Paris dataset

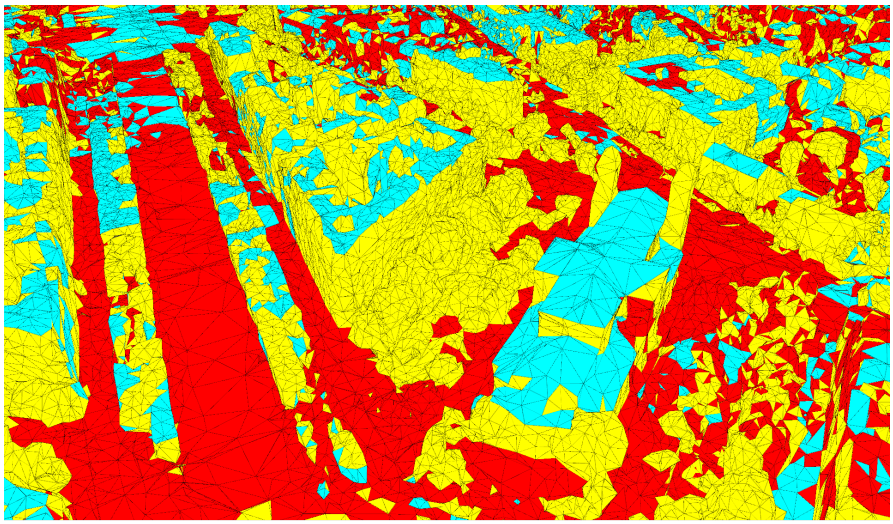


Figure 5.19: Verdie three-class-MRF formulation on Paris dataset

5.3.6 Object identification

Textures comparison is a good compromise with limited resources and hilly grounds but is definitely far from optimal. The results are presented on Fig. 5.20 and Fig. 5.21 (local maxima are represented by red spheres).
In Fig. 5.21, the two main types of objects detected through the method are represented: ground objects, such as cars or bumps in the ground representation and segments from façades or low roofs, coming mostly from maxima present on the edge of the ground mesh.
The classification steps would then cluster these results from an automatic processing point of view, which corresponds to flattening the objects closer to the ground while leaving the higher ones intact.

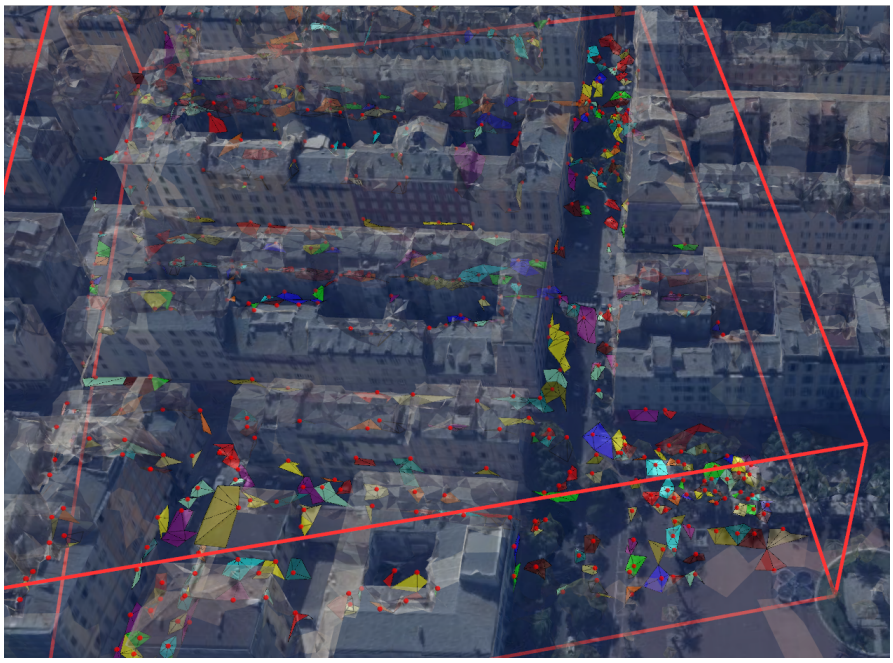


Figure 5.20: Object detection: object detection in Bastia

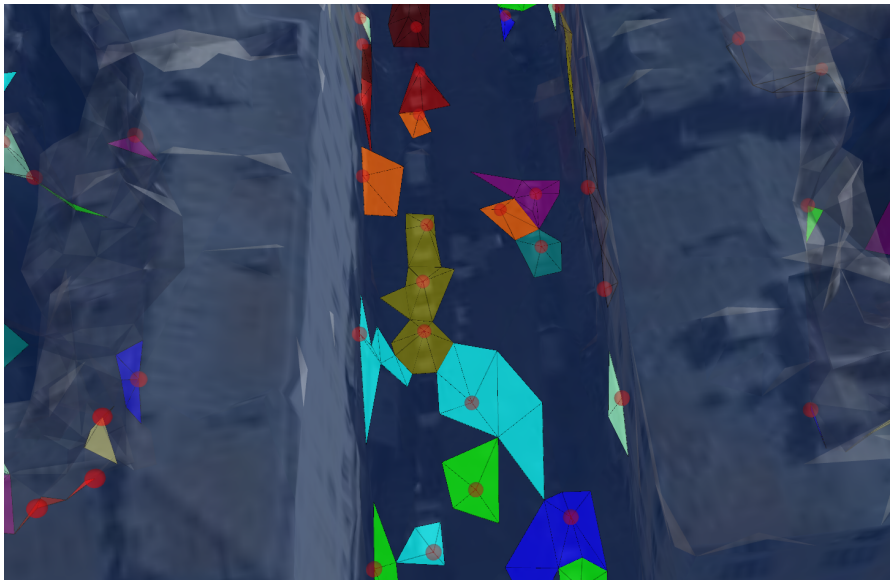


Figure 5.21: Object detection: types of detected objects

5.3.7 Conclusion

On ideal cases, k-means and the MRF formulation perform quite similarly for three classes-identification. The fourth class, vegetation, identification requires further improvement using a MRF formulation, while usually leading to a split in the façade class when using k-means. As k-means performs a clustering based on distance in the feature space, this is not surprising and this clustering behaviour is even more striking when classifying non-representative segments.
The suggested algorithm still faces most of the issues presented in Rouhani et al. [47] and Verdie et al. [58]: misclassification of ground as roof or façade in large flat areas. When using Verdie et al. [58] unsupervised approach on lower resolution meshes, vegetation identification fails most of the time and calls for new, more discriminative features such as density-based suggestions.
As a general conclusion, the detection of artefacts in meshes is a rather complicated but solvable problem, on both higher and lower resolution meshes. The resolution however has to be high enough so that information is contained in geometric properties of the mesh rather than only in textures.

Chapter 6

Discussion

6.1 Degree project limitations

Several problems and constraints arose due to the degree project constraints. The first was that the investigated problem was shaped according to the available and usable data at Carmenta. This means that neither ground-truths nor detailed-enough elevation data were usable in association with the available data in the short time span of the project. Time evidently represented a constraint as well, as some results and suggestions presented in the algorithm were not included in the final results, such as effectively including the density feature in the MRF formulation. Finally, more on the engineering constraints, various approximations in computations arose due to the structure and format of the mesh data.

6.2 Future works

Density holds potential for vegetation detection in meshes with varying resolutions. Due to time constraints, it was however not possible to effectively include it in the final proposal. As vegetation processing for artefact removal is usually rather specific, carrying on to a better-performing vegetation detection in the primary clustering via MRF step, it is important for the primary clustering to efficiently distinguish ground, buildings and trees.
In order to improve object detection, using available and accurate elevation data could allows to perform watershed without requiring

texture-based constraints. Finally, in general, constituting ground-truths for supervised approach or simply performance evaluation could be of great benefit.

6.3 Conclusion

From an engineering point of view, artefacts identification in urban meshes is a solvable problem. Given results and works on the detection of artefacts in 3D point clouds and that urban meshes are usually generated from point cloud surveying, it feels like performing such a detection step for quality purposes should be performed beforehand. However, given a mesh, semantic segmentation is a relevant challenge for automatic simplification of the given mesh or for other applications.

6.4 Ethical and societal aspects

Improving an urban model's quality by removing artefacts allows faster processing of the data which can be used towards efficient rescue missions planning. Digital models also play a very important in urban planning, allowing to visualise and anticipate risks through accurate simulations, as well as mitigating climate change and improving energy-efficiency as performed with the Helsinki data: paired with other energy-consumption-related sources, the model helps examine the solar energy potential of buildings.

Bibliography

[1] Acute3D. *City mapping*. https://www.acute3d.com/city-mapping/. Accessed: 2018-11-12.

[2] Pouria Babahajiani et al. "Urban 3D segmentation and modelling from street view images and LiDAR point clouds". In: *Machine Vision and Applications* 28.7 (Oct. 2017), pp. 679–694. ISSN: 1432-1769. DOI: 10.1007/s00138-017-0845-3. URL: https://doi.org/10.1007/s00138-017-0845-3.

[3] Halim Benhabiles et al. "A comparative study of existing metrics for 3D-mesh segmentation evaluation". In: *The Visual Computer* 26 (2010), pp. 1451–1466. DOI: 10.1007/s00371-010-0494-2.

[4] Ali Borji and Aysegul Dundar. "A new look at clustering through the lens of deep convolutional neural networks". In: *CoRR* abs/1706.05048 (2017). arXiv: 1706.05048. URL: http://arxiv.org/abs/1706.05048.

[5] Y. Boykov and V. Kolmogorov. "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.9 (Sept. 2004), pp. 1124–1137. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2004.60.

[6] Y. Boykov, O. Veksler, and R. Zabih. "Fast approximate energy minimization via graph cuts". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23.11 (Nov. 2001), pp. 1222–1239. ISSN: 0162-8828. DOI: 10.1109/34.969114.

[7] Chih-Chung Chang and Chih-Jen Lin. "LIBSVM: A library for support vector machines". In: *ACM Transactions on Intelligent Systems and Technology* 2 (2001).

[8] X. Chen et al. “Monocular 3D Object Detection for Autonomous Driving”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016, pp. 2147–2156. DOI: 10.1109/CVPR.2016.236.

[9] Xiaozhi Chen et al. “3D Object Proposals using Stereo Imagery for Accurate Object Class Detection”. In: *arXiv*.

[10] Xiaozhi Chen et al. “Multi-View 3D Object Detection Network for Autonomous Driving”. In: *CoRR* abs/1611.07759 (2016). arXiv: 1611.07759. URL: http://arxiv.org/abs/1611.07759.

[11] Andrea Cirillo. *How to build a color palette from any image with R and k-means algo*. http://www.milanor.net/blog/build-color-palette-from-image-with-paletter/. Published: 2017-06-20, Accessed: 2018-11-21.

[12] David Cohen-Steiner and Jean-Marie Morvan. “Restricted Delaunay Triangulations and Normal Cycle”. In: *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*. SCG ’03. San Diego, California, USA: ACM, 2003, pp. 312–321. ISBN: 1-58113-663-3. DOI: 10.1145/777792.777839. URL: http://doi.acm.org/10.1145/777792.777839.

[13] David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. “Variational Shape Approximation”. In: *ACM Trans. Graph.* 23.3 (Aug. 2004), pp. 905–914. ISSN: 0730-0301. DOI: 10.1145/1015706.1015817. URL: http://doi.acm.org/10.1145/1015706.1015817.

[14] Dorin Comaniciu and Peter Meer. “Mean shift: A robust approach toward feature space analysis”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), pp. 603–619.

[15] J. C. Dunn. “A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters”. In: *Journal of Cybernetics* 3.3 (1973), pp. 32–57. DOI: 10.1080/01969727308546046. eprint: https://doi.org/10.1080/01969727308546046. URL: https://doi.org/10.1080/01969727308546046.

[16] Martin Engelcke et al. “Vote3Deep: Fast Object Detection in 3D Point Clouds Using Efficient Convolutional Neural Networks”. In: *CoRR* abs/1609.06666 (2016). arXiv: 1609.06666. URL: http://arxiv.org/abs/1609.06666.

[17] Pixel Factory. *Pixel Factory Neo - Taking Earth Observation Data Processing to the Next Level*. https://www.intelligence-airbusds.com/processing-software/. Accessed:2018-11-22.

[18] David George, Xianghua Xie, and Gary K. L. Tam. "3D Mesh Segmentation via Multi-branch 1D Convolutional Neural Networks". In: *CoRR* abs/1705.11050 (2017). arXiv: 1705.11050. URL: http://arxiv.org/abs/1705.11050.

[19] A. Golovinskiy, V. G. Kim, and T. Funkhouser. "Shape-based recognition of 3D point clouds in urban environments". In: *2009 IEEE 12th International Conference on Computer Vision*. Sept. 2009, pp. 2154–2161. DOI: 10.1109/ICCV.2009.5459471.

[20] Ben Gorte. "Planar Feature Extraction in Terrestrial Laser Scans Using Gradient Based Range Image Segmentation". In: 2007.

[21] A. Gray, E. Abbena, and S. Salamon. "Shape and Curvature". In: *Modern Differential Geometry of Curves and Surfaces with Mathematica, Third Edition*. Ed. by Chapman & Hall/CRC. 2006. Chap. 13, pp. 385–419.

[22] H. He and B. Upcroft. "Nonparametric semantic segmentation for 3D street scenes". In: (Nov. 2013), pp. 3697–3703. ISSN: 2153-0858. DOI: 10.1109/IROS.2013.6696884.

[23] City of Helsinki. *How were the 3D models made?* https://www.intelligence-airbusds.com/processing-software/. Accessed:2018-11-22.

[24] J. Hernandez and B. Marcotegui. "Point cloud segmentation towards urban ground modeling". In: (May 2009), pp. 1–5. ISSN: 2334-0932. DOI: 10.1109/URS.2009.5137562.

[25] A. Hoover et al. "An experimental comparison of range image segmentation algorithms". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18.7 (July 1996), pp. 673–689. ISSN: 0162-8828. DOI: 10.1109/34.506791.

[26] L'ubor Ladický et al. "Inference Methods for CRFs with Co-occurrence Statistics". In: *Int. J. Comput. Vision* 103.2 (June 2013), pp. 213–225. ISSN: 0920-5691. DOI: 10.1007/s11263-012-0583-y. URL: http://dx.doi.org/10.1007/s11263-012-0583-y.

[27] Florent Lafarge. *Mesh segmentation*. https://team.inria.fr/titane/files/2015/04/mesh_segmentation.pdf. Accessed: 2018-11-06.

[28] Florent Lafarge et al. "A hybrid multi-view stereo algorithm for modeling urban scenes". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.1 (Jan. 2013), pp. 5–17. DOI: 10.1109/TPAMI.2012.84. URL: https://hal.inria.fr/hal-00759261.

[29] K. Lai, L. Bo, and D. Fox. "Unsupervised feature learning for 3D scene labeling". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. May 2014, pp. 3050–3057. DOI: 10.1109/ICRA.2014.6907298.

[30] Bo Li, Tianlei Zhang, and Tian Xia. "Vehicle Detection from 3D Lidar Using Fully Convolutional Network". In: *CoRR* abs/1608.07916 (2016). arXiv: 1608.07916. URL: http://arxiv.org/abs/1608.07916.

[31] S. Lloyd. "Least squares quantization in PCM". In: *IEEE Transactions on Information Theory* 28.2 (Mar. 1982), pp. 129–137. ISSN: 0018-9448. DOI: 10.1109/TIT.1982.1056489.

[32] A. P. Mangan and R. T. Whitaker. "Partitioning 3D surface meshes using watershed segmentation". In: *IEEE Transactions on Visualization and Computer Graphics* 5.4 (Oct. 1999), pp. 308–321. ISSN: 1077-2626. DOI: 10.1109/2945.817348.

[33] A. Martinović et al. "3D all the way: Semantic segmentation of urban scenes from start to end in 3D". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015, pp. 4456–4465. DOI: 10.1109/CVPR.2015.7299075.

[34] F. Meyer. "From connected operators to levelings". In: *Mathematical Morphology and its Applications to Image and Signal Processing*. Ed. by Kluwer Academic Publishers. Vol. 12. 1998, pp. 191–198.

[35] Javier A. Montoya-Zegarra et al. "Semantic Segmentation of Aerial Images in Urban Areas with Class-specific Higher-order Cliques". In: 2015.

[36] R. Mottaghi et al. "The Role of Context for Object Detection and Semantic Segmentation in the Wild". In: (June 2014), pp. 891–898. ISSN: 1063-6919. DOI: 10.1109/CVPR.2014.119.

[37] P. Musialski et al. “A Survey of Urban Reconstruction”. In: *Computer Graphics Forum* 32.6 (), pp. 146–177. DOI: 10.1111/cgf.12077. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12077. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12077.

[38] H. Myeong, J. Y. Chang, and K. M. Lee. “Learning object relationships via graph-based context model”. In: (June 2012), pp. 2727–2734. ISSN: 1063-6919. DOI: 10.1109/CVPR.2012.6247995.

[39] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. “On Spectral Clustering: Analysis and an Algorithm”. In: *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*. NIPS’01. Vancouver, British Columbia, Canada: MIT Press, 2001, pp. 849–856. URL: http://dl.acm.org/citation.cfm?id=2980539.2980649.

[40] Joachim Niemeyer, Franz Rottensteiner, and Uwe Soergel. “Contextual classification of lidar data and building object detection in urban areas”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 87 (2014), pp. 152 –165. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2013.11.001. URL: http://www.sciencedirect.com/science/article/pii/S0924271613002359.

[41] David Page, Andreas Koschan, and Mongi Abidi. “Perception-based 3D Triangle Mesh Segmentation Using Fast Marching Watersheds.” In: *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on* 2 (Jan. 2003), pp. 27–32. DOI: 10.1109/CVPR.2003.1211448.

[42] Massimiliano Patacchiola. *The Simplest Classifier: Histogram Comparison*. https://mpatacchiola.github.io/blog/2016/11/12/the-simplest-classifier-histogram-intersection.html. Published: 2016-11-16, Accessed: 2018-11-22.

[43] Mark Pauly, Markus H. Gross, and Leif Kobbelt. “Efficient simplification of point-sampled surfaces”. In: *IEEE Visualization, 2002. VIS 2002.* (2002), pp. 163–170.

[44] Shi Pu et al. “Recognizing basic structures from mobile laser scanning data for road inventory studies”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 66.6, Supplement (2011). Advances in LIDAR Data Processing and Applications, S28 –S39.

ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2011.08.006. URL: http://www.sciencedirect.com/science/article/pii/S0924271611000955.

[45] Charles Ruizhongtai Qi et al. "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation". In: *CoRR* abs/1612.00593 (2016). arXiv: 1612.00593. URL: http://arxiv.org/abs/1612.00593.

[46] F. Rottensteiner et al. "The Isprs Benchmark on Urban Object Classification and 3d Building Reconstruction". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* (July 2012), pp. 293–298. DOI: 10.5194/isprsannals-I-3-293-2012.

[47] Mohammad Rouhani, Florent Lafarge, and Pierre Alliez. "Semantic Segmentation of 3D Textured Meshes for Urban Scene Analysis". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 123 (2017), pp. 124 –139. DOI: 10.1016/j.isprsjprs.2016.12.001. URL: https://hal.inria.fr/hal-01469502.

[48] Szymon Rusinkiewicz. "Estimating Curvatures and Their Derivatives on Triangle Meshes". In: *Symposium on 3D Data Processing, Visualization, and Transmission*. Sept. 2004.

[49] Martin Rutzinger et al. "Tree modelling from mobile laser scanning data-sets". In: *The Photogrammetric Record* 26.135 (2011), pp. 361–372. DOI: 10.1111/j.1477-9730.2011.00635.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1477-9730.2011.00635.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1477-9730.2011.00635.x.

[50] Favio Saponara Iriarte Paniagua. *Object Recognition using the Kinect*. 2011. URL: http://www.nada.kth.se/utbildning/grukth/exjobb/rapportlistor/2011/rapporter11/saponara_favio_11115.pdf.

[51] Ruwen Schnabel et al. "Shape Recognition in 3D Point-Clouds". In: (May 2012).

[52] Andrés Serna and Beatriz Marcotegui. "Detection, segmentation and classification of 3D urban objects using mathematical morphology and supervised learning". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 93 (July 2014), pp. 243–255. DOI: 10.1016/j.isprsjprs.2014.03.015. URL: https://hal.archives-ouvertes.fr/hal-01010012.

[53] National Ocean Service. *What is LIDAR?* https://oceanservice.noaa.gov/facts/lidar.html. Accessed: 2018-11-12.

[54] Jianbo Shi and Jitendra Malik. "Normalized Cuts and Image Segmentation". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 22.8 (Aug. 2000), pp. 888–905. ISSN: 0162-8828. DOI: 10.1109/34.868688. URL: https://doi.org/10.1109/34.868688.

[55] Shymon Shlafman, Ayellet Tal, and Sagi Katz. "Metamorphosis of Polyhedral Surfaces using Decomposition". In: *Comput. Graph. Forum* 21 (2002), pp. 219–228.

[56] Zhenyu Shu et al. "Unsupervised 3D shape segmentation and co-segmentation via deep learning". In: *Computer Aided Geometric Design* 43 (2016). Geometric Modeling and Processing 2016, pp. 39 –52. ISSN: 0167-8396. DOI: https://doi.org/10.1016/j.cagd.2016.02.015. URL: http://www.sciencedirect.com/science/article/pii/S0167839616300164.

[57] Alexander Velizhev, Roman Shapovalov, and Konrad Schindler. "Implicit Shape Models for Object Detection in 3 D Point Clouds". In: 2012.

[58] Yannick Verdie, Florent Lafarge, and Pierre Alliez. "LOD Generation for Urban Scenes". In: *ACM Transactions on Graphics* 34.3 (2015), p. 15. URL: https://hal.inria.fr/hal-01113078.

[59] V. Vineet et al. "Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction". In: (May 2015), pp. 75–82. ISSN: 1050-4729. DOI: 10.1109/ICRA.2015.7138983.

[60] M. Volpi and V. Ferrari. "Semantic segmentation of urban scenes by learning local class interactions". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. June 2015, pp. 1–9. DOI: 10.1109/CVPRW.2015.7301377.

[61] J. Xiao and L. Quan. "Multiple view semantic segmentation for street view images". In: *2009 IEEE 12th International Conference on Computer Vision*. Sept. 2009, pp. 686–693. DOI: 10.1109/ICCV.2009.5459249.

[62] Dominic Zeng Wang and Ingmar Posner. "Voting for Voting in Online Point Cloud Object Detection". In: (2015). DOI: 10.15607/RSS.2015.XI.035.

[63] H. Zhang, O. Van Kaick, and R. Dyer. "Spectral Mesh Processing". In: *Computer Graphics Forum* (2010). ISSN: 1467-8659. DOI: 10.1111/j.1467-8659.2010.01655.x.

[64] Hui Zhang, Jason E. Fritts, and Sally A. Goldman. "Image segmentation evaluation: A survey of unsupervised methods". In: *Computer Vision and Image Understanding* 110 (2008), pp. 260–280.