Semantic understanding of urban scenes from textured meshes

Gao, W.

DOI
10.4233/uuid:75ffb577-05a6-465f-8d71-92fb3e3f960c

Publication date
2024

Document Version
Final published version

Citation (APA)
Gao, W. (2024). *Semantic understanding of urban scenes from textured meshes*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:75ffb577-05a6-465f-8d71-92fb3e3f960c

Semantic understanding of urban scenes from textured meshes

Semantic understanding of urban scenes from textured meshes

Dissertation

for the purpose of obtaining the degree of doctor

at Delft University of Technology

by the authority of the Rector Magnificus, prof. dr. ir. T.H.J.J. van der Hagen,

chair of the Board for Doctorates

to be defended publicly on

Monday 4 November 2024 at 12:30 o'clock

by

Weixiao GAO

This dissertation has been approved by the promotor and the copromotor.

Composition of the doctoral committee:

Rector Magnificus	chairperson
Dr. H. Ledoux	Delft University of Technology, promotor
Dr. L. Nan	Delft University of Technology, copromotor
Independent members:	
Prof. dr. -Ing. N. Haala	University of Stuttgart, Germany
Dr. F. Lafarge	Inria Centre at Université Côte d'Azur, France
Prof. dr. ing. S. Nijhuis	Delft University of Technology
Dr. ir. S.J. Oude Elberink	University of Twente, Netherlands
Prof. dr. T.G. Vrachliotis	Delft University of Technology

This dissertation was partly supported by Cyclomedia.

Keywords: Textured meshes, urban scene understanding, semantic segmentation, interactive annotation, 3D city modeling

Printed by: Ridderprint | https://www.ridderprint.nl

ISBN: 978-94-6366-942-9

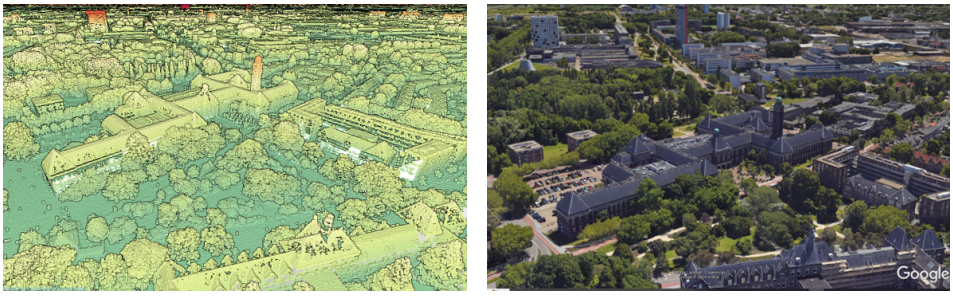
A digital version of this thesis is available at https://repository.tudelft.nl/.

Contents

1 Introduction

1.1. Background and motivation

The increasing availability of 3D datasets, such as LiDAR point clouds [AHN, 2019] and textured meshes [Google, 2012; Helsinki, 2019], plays a vital role in managing and studying urban environments (see Figure 1.1). As cities grow and change, there is an increasing need for effective methods to handle the complexities introduced by these diverse and intricate datasets. Urban 3D datasets provide detailed spatial information that helps in accurately mapping urban spaces, essential for applications such as urban planning [Ran, 2011], spatial analysis [Yaagoubi *et al.*, 2015], and environmental analysis [Yichuan *et al.*, 2016].



(a) AHN4 LiDAR point cloud (b) Google Earth 3D mesh

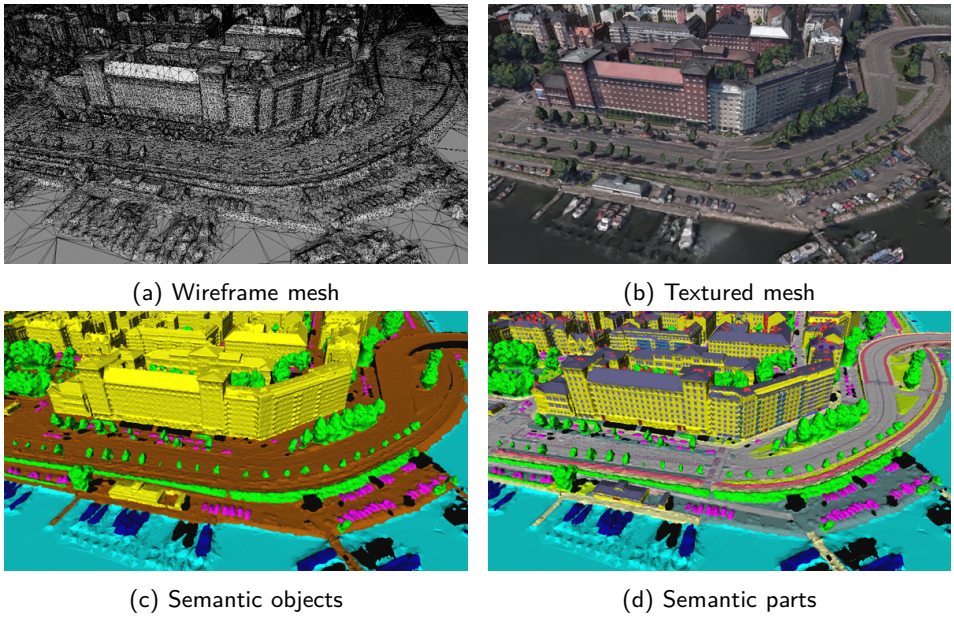
Figure 1.1: Representative 3D urban data

The absence of semantic information in 3D datasets can significantly limit the effectiveness of urban applications. Initially, LiDAR and textured meshes provide only geometries and basic attributes such as intensity or colors. However, they lack semantic information regarding objects and their parts, as well as instance labels. Incorporating semantic information into these datasets allows users to

achieve a detailed and intuitive understanding of urban environments. Significant advances in computer vision and artificial intelligence (AI), particularly in semantic understanding of complex 3D urban datasets, are critical [Q. Hu *et al.*, 2022; G. Yang *et al.*, 2023]. Applying AI techniques, such as neural networks, enables researchers to transform raw data into actionable insights by identifying structural patterns [Krehl *et al.*, 2016], detecting anomalies [M. Zhang *et al.*, 2022], and predicting dynamic changes [Kuru and Yüzer, 2021; Xia *et al.*, 2021]. These enriched semantic urban datasets substantially improve a range of urban applications and enable the simulation of complex urban scenarios, such as energy management [Katal *et al.*, 2022], disaster prevention [Verykokou *et al.*, 2016], microclimate modeling [Garcia-Dorado *et al.*, 2017], traffic flow analysis [Chao *et al.*, 2020], and urban fluid simulation [Maragkogiannis *et al.*, 2014].

Semantic segmentation, a critical AI technology, is often applied to scene interpretation. It significantly enhances the understanding of urban scenes by accurately classifying and locating each object, thereby boosting the usability and interpretability of the data. While extensive resources for semantic segmentation, including datasets, tools, and methods, are available for images and point clouds [Cordts *et al.*, 2016; Martinovi *et al.*, 2015], research on textured meshes has predominantly targeted small-scale indoor scenes [Armeni *et al.*, 2017; Chang *et al.*, 2017; Dai *et al.*, 2017] over expansive outdoor urban settings [Miksik *et al.*, 2015]. Textured meshes (see Figure 1.2), which provide richer physical details than point clouds through continuous surfaces embedded with topological and color information, are lightweight and ideally suited for 3D urban scene interpretation. However, despite progress made by researchers [Rouhani *et al.*, 2017; Verdie *et al.*, 2015], the field of urban scene understanding from textured meshes still faces a significant shortage of methods, datasets, and related applications.

This thesis addresses critical gaps in the semantic understanding of urban scenes from textured meshes. It focuses on three primary challenges: 1) the lack of effective strategies and tools for accurately labeling mesh facets and textures; 2) the limited development of semantic segmentation methods tailored for textured meshes, as most are designed for LiDAR and images; 3) the underexplored potential of semantically enriched meshes in applications. This work introduces significant innovations, including the first large-scale benchmark datasets for semantic urban textured meshes at both object and part levels, an interactive and semi-automatic 3D semantic annotation framework for mesh facets and textures, and a novel deep learning semantic segmentation method designed specifically for these meshes. Additionally, it explores the creation of lightweight 3D city models using semantic information, setting new directions for future research in semantic-based applications of 3D urban scenarios. These contributions collectively advance the field by offering practical solutions and establishing a foundational framework for continued exploration and development.



(a) Wireframe mesh (b) Textured mesh

(c) Semantic objects (d) Semantic parts

Figure 1.2: Examples of urban textured meshes with semantic classes: (a) shows a wireframe mesh with triangles represented by black edges; (b) depicts a mesh with textures; (c) demonstrates a mesh with object-level semantics, including terrain ■, water ■, buildings ■, high vegetation ■, cars ■, boats ■, etc.; (d) illustrates a mesh with part-level semantics, detailing components such as roofs ■, facades ■, balconies ■, chimneys ■, dormers ■, windows ■, doors ■, road markings ■, cycle lanes ■, sidewalks ■.

1.2. Problem statement

To address the limitations of current research, this thesis focuses on the semantic segmentation of urban textured meshes and identifies four key problems related to their interpretation and application in 3D urban scene understanding:

Problem I: *There is a lack of publicly available, large-scale, part-level semantic benchmark datasets for urban textured meshes.*

Semantic benchmark datasets are crucial for evaluating the performance of automatic semantic segmentation models and providing training data for learning-based algorithms. While previous research has primarily concentrated on semantic benchmark datasets for urban point clouds [Q. Hu *et al.*, 2021; Varney *et al.*, 2020; Zolanvari *et al.*, 2019] or indoor scene meshes [Armeni *et al.*, 2017; Chang *et al.*, 2017; Dai *et al.*, 2017], datasets specifically targeting large-scale urban textured meshes are notably rare [Kölle *et al.*, 2021]. Furthermore, although datasets such as DublinCity [Zolanvari *et al.*, 2019] and Hessigheim3D [Kölle *et al.*, 2021] include

some part-level details, their categories and scope remain relatively limited. This gap significantly impedes the development and validation of methods for the semantic segmentation and interpretation of large-scale urban textured meshes.

Problem II: *There is a significant shortage of effective interactive 3D semantic annotation tools specifically designed for textured urban meshes.*

Annotation tools are intended to assign accurate semantic labels to raw data or correct erroneous labels on predicted data through human-computer interaction. Traditional manual annotation methods can annotate textured meshes but often require substantial time investment [Ibrahim *et al.*, 2021; Pütz *et al.*, 2021a]. Most existing research on efficient interactive 3D semantic annotation focuses on point clouds [Kontogianni *et al.*, 2023; K. Liu and Boehm, 2014; Schmitz *et al.*, 2022; Steinlechner *et al.*, 2019; Yue *et al.*, 2024] or solid models [Jayaraman *et al.*, 2021], with limited studies on triangular meshes tailored for scenes that require specific camera parameters and original images [D. T. Nguyen *et al.*, 2018; Romanoni and Matteucci, 2018]. While semantic annotation of mesh textures might employ interactive image segmentation techniques, these are hindered by the limited efficiency of traditional methods [Fan and T. C. Lee, 2015; T. N. A. Nguyen *et al.*, 2012; Rother *et al.*, 2004] and the generalization capabilities associated with deep learning approaches [Kirillov *et al.*, 2023; Q. Liu *et al.*, 2023]. There is a pressing need for tools that can efficiently handle interactive 3D semantic annotation of large-scale urban textured meshes, focusing on both their facets and texture images.

Problem III: *Efficient semantic segmentation methods specifically targeting large-scale urban textured meshes are notably lacking.*

Early methods for semantic segmentation of urban textured meshes primarily used machine learning-based manual feature techniques, which were limited in accuracy, efficiency, and performance [Rouhani *et al.*, 2017; Verdie *et al.*, 2015]. Modern 3D deep learning approaches, primarily focused on point cloud segmentation, often exhibit poor generalization capabilities [Landrieu and Simonovsky, 2018; Qi *et al.*, 2017; Thomas *et al.*, 2018; Thomas *et al.*, 2019]. Methods applicable to triangular meshes are generally confined to segmenting small-scale objects [Fu *et al.*, 2021; L. Gao *et al.*, 2019; Hanocka *et al.*, 2019] or indoor scenes [Nan *et al.*, 2012; Selvaraju *et al.*, 2021]. Consequently, there is a critical need for research into advanced semantic segmentation methods tailored for large-scale urban textured meshes [Rouhani *et al.*, 2017; Verdie *et al.*, 2015].

Problem IV: *There is limited research on using urban textured meshes with semantic information to reconstruct lightweight 3D city models and their applications.*

Urban textured meshes typically comprise numerous facets, consuming substantial

storage and hindering the efficiency of 3D applications that involve heavy geometric processing. Developing lightweight semantic city models could mitigate these challenges. Few studies have explored the reconstruction of detailed semantic 3D city models using semantic-based urban textured meshes, with most focusing either exclusively on building models [Bouzas *et al.*, 2020; J. Han *et al.*, 2021; Zhu *et al.*, 2018] or on template-based object matching [Verdie *et al.*, 2015]. Given the complexity of urban environments, existing techniques fall short in reconstructing large-scale, lightweight models tailored to diverse urban components and semantic classes.

1.3. Research objectives and research questions

The main research objective of this thesis is to develop and implement a robust framework for extracting semantic information from large-scale urban textured meshes at both object and part-level semantics (see Figure 1.2c and Figure 1.2d). For example, objects include buildings, roads, and trees, while parts refer to specific elements like windows and chimneys on buildings or cycle lanes and road markings. This framework will support both semi-automatic and fully automatic segmentation methods. Additionally, the research investigates the application of semantic meshes in generating lightweight city models and explores the various uses of semantic city models in related fields. To this end, I have established the following specific objectives and questions.

I. Semantic urban mesh benchmark dataset

- **Objective:** *Build a comprehensive benchmark dataset of large-scale semantic urban meshes, which includes both object-level and part-level semantics.*
- **Research questions:**
  1. Which semantic objects and parts can be precisely identified by humans and computers from the triangular facets and texture images in urban textured meshes?
  2. What 3D semantic segmentation methods are suitable for performance evaluation in urban semantic meshes?

II. Urban textured mesh annotation framework

- **Objective:** *Design and develop an interactive 3D annotation tool that enables semi-automatic semantic labeling for mesh facets and texture pixels of urban textured meshes.*
- **Research questions:**
  1. How can an interactive annotation user interface be designed for urban

textured meshes?

2. How can we develop a semi-automatic annotation framework that leverages existing annotated data to predict and correct unannotated regions?

3. How can we achieve an efficient selection of 3D objects and parts in urban textured meshes with minimal user interaction?

4. How can regions of interest be efficiently selected in the mesh textures?

III. Semantic segmentation of large-scale urban textured meshes

- **Objective:** *Develop fully automatic semantic mesh segmentation algorithms for large-scale urban scenes.*
- **Research questions:**

1. How can a deep learning framework be developed for semantic segmentation of large-scale urban meshes?

2. How can the generalizability of deep learning be enhanced for 3D semantic segmentation tasks?

3. How to obtain sufficient contextual information for semantic segmentation?

IV. Applications of the semantic urban meshes

- **Objective:** *Develop a semantic-based lightweight city model reconstruction framework and explore other related practical applications.*
- **Research questions:**

1. How can semantic information from textured meshes be utilized for lightweight reconstruction of 3D city models?

2. What applications can be developed using urban meshes and lightweight models equipped with object and part-level semantics?

1.4. Thesis outline

The thesis is organized into seven chapters. Chapter 1 introduces the topic, and the subsequent chapters each address a specific aspect of the research:

- Chapter 2 introduces the principles of urban 3D reconstruction, focusing on the 3D reconstruction pipeline that transforms images into textured meshes. It also covers recent advances in interpreting and encoding urban models and

discusses the role of semantic information in 3D urban applications

- Chapter 3 is dedicated to creating a benchmark dataset for the semantic mesh of urban objects. It begins with an overview of existing datasets and detailed specifications of the input mesh dataset. The chapter then presents a basic annotation framework for efficient ground truth semantic mesh generation at the object level of semantics, including a semi-automatic mesh labeling framework and an automatic semantic mesh segmentation algorithm. The created dataset covers approximately 4 km^2 of the Helsinki center and is used to evaluate state-of-the-art 3D semantic segmentation methods.

- Chapter 4 introduces the proposed semantic mesh segmentation algorithm for large-scale urban objects, summarizing recent research on segmentation of 3D urban data. The chapter presents a solution for processing large-scale urban meshes using planarity-sensible over-segmentation and a novel graph for capturing contextual information. Additionally, a feature embedding module is introduced to integrate handcrafted and learned features. Classification is performed using a graph convolutional network, and the performance is evaluated on the benchmark dataset.

- Chapter 5 focuses on an annotation framework designed for efficient part-level semantic segmentation of large-scale urban textured meshes. It aims to reduce manual effort by incorporating interactive extraction and structure-aware matching. This interactive framework is aimed at advancing deep learning methods for urban analysis. Additionally, the chapter presents a novel dataset for part-level semantic segmentation of large urban scenes, defined by detailed annotations of specific urban components like chimneys and road markings. This dataset is rigorously evaluated against leading 3D semantic segmentation technologies, illustrating its effectiveness in refining deep learning models.

- Chapter 6 extends the exploration of semantic information from urban textured meshes towards practical applications, detailing a new framework for reconstructing lightweight 3D city models that ensure geometric integrity, surface continuity, and semantic retention. Utilizing semantic information for 3D model reconstruction and surface simplification, the framework supports the creation of multi-level city models from LoD2 to LoD3. Experiments conducted on the semantic urban mesh benchmark datasets (SUM) and SUM-Parts confirm the framework's effectiveness, surpassing traditional methods in efficiency and accuracy. These lightweight models facilitate advanced urban applications, enhancing automation in measurements and realism in simulations, showcasing the potential of semantic-rich city models in real-world scenarios.

- Chapter 7 provides a comprehensive summary of the research conducted in this study, including the contributions, key findings, and reflections. It also offers recommendations for future research directions to build upon the current work and further advance the field of 3D urban modeling and analysis.

2

Fundamentals of urban 3D reconstruction

This chapter outlines the foundational principles and concepts related to 3D urban reconstruction. It details the principles of input data generationspecifically, the 3D reconstruction pipeline that transforms images into textured meshes. Additionally, this chapter discusses the latest advancements in techniques for interpreting, reconstructing, and encoding city models within extensive urban environments, as well as the role of semantic information in 3D urban applications.

2.1. Textured mesh generation

Textured mesh generation is a fundamental task in 3D computer vision and photogrammetry, which aims to reconstruct the 3D geometry of an object or scene from 2D images. The pipeline for textured mesh generation typically consists of five steps: keypoints detection and matching, structure from motion (SfM), multi-view stereo (MVS), surface reconstruction, and texture mapping.

2.1.1. Keypoints detection and matching

In the first step, keypoints detection and matching algorithms are applied to identify corresponding points among multi-view images. This involves extracting distinctive features from the images and matching them across different views.

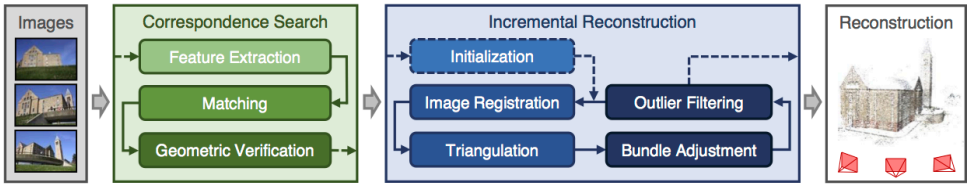


Figure 2.1: Incremental Structure-from-Motion pipeline. Figure taken from Schönberger and Frahm, (2016).

Traditional methods such as SIFT [Lowe, 1999] and SURF [Bay *et al.*, 2006] are commonly used for this task, which typically involves detecting local feature points and computing descriptors to represent them. In recent years, deep learning-based approaches have shown superior performance in feature detection and matching, such as SuperPoint [DeTone *et al.*, 2018] and SuperGlue [Sarlin *et al.*, 2020], which are trained to directly predict feature points and their correspondences in an end-to-end manner.

2.1.2. Structure from motion

The second step in textured mesh generation involves using structure from motion techniques to estimate the camera poses and reconstruct a sparse point cloud (see Figure 2.1). This involves determining the relationship between keypoints in different views using epipolar geometry constraints, as well as estimating camera parameters such as focal length and distortion parameters. Bundle adjustment is a key step in SfM algorithms that optimizes the camera parameters and 3D points by minimizing the reprojection error between the 2D keypoints and their corresponding 3D points in the reconstructed scene.

SfM algorithms can be categorized into incremental and global methods. Some well-known SfM systems include Bundler [Snavely *et al.*, 2008; Snavely *et al.*, 2006],

VisualSFM [C. Wu, 2013; C. Wu *et al.*, 2011], and COLMAP [Schönberger and Frahm, 2016]. Recently, deep learning-based methods have been developed for SfM, such as DeepSfM [Wei *et al.*, 2020] and SfM-Net [Vijayanarasimhan *et al.*, 2017], which have shown promising results in improving camera pose estimation and dense point cloud reconstruction.

In contrast to structure from motion methods, photogrammetry pipelines often employ aerotriangulation to generate sparse point clouds from aerial imagery [Duane, 1971]. Aerotriangulation involves using ground control points to accurately estimate the position and orientation of the cameras, which can then be used to triangulate points in the scene. This approach has the advantage of being able to handle large-scale scenes and can produce accurate point clouds with minimal user intervention. However, it requires careful planning and preparation to collect accurate ground control points and can be sensitive to errors in camera calibration and positioning.

2.1.3. Multi-view stereo

After the sparse point cloud is generated, multi-view stereo algorithms are applied to densify the point cloud and recover the complete 3D shape of the object. The core operation of MVS is to estimate the depth of each pixel in the input images, given the camera poses and sparse point cloud. MVS algorithms can be classified into dense matching and patch-based methods.

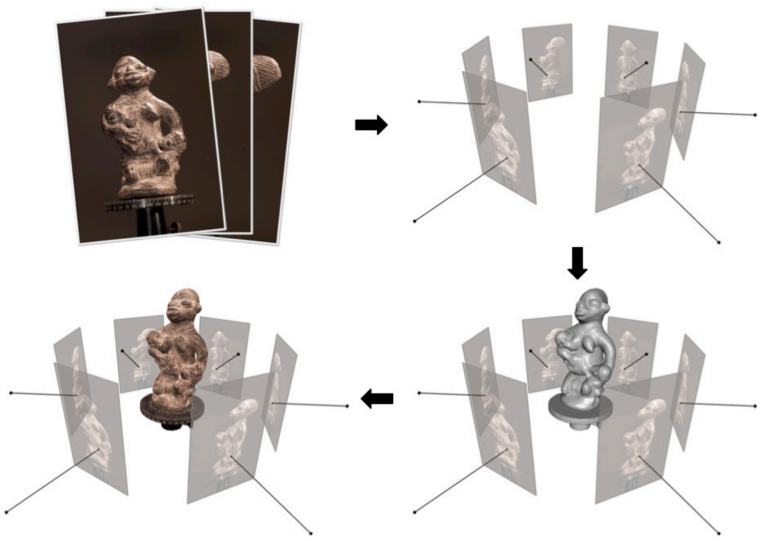


Figure 2.2: Example of textured mesh generation pipeline: input imagery, posed imagery, reconstructed 3D geometry, textured 3D geometry. Figure taken from Furukawa and Hernández, (2015).

Dense matching-based methods aim to find correspondences between pixels in different images by comparing image patches around each pixel. Classical dense matching-based methods include PMVS [Furukawa and Ponce, 2010], which performs multi-view stereo reconstruction based on photometric consistency (see Figure 2.2). Recent deep learning-based dense matching-based methods, such as DeepMVS [P.-H. Huang *et al.*, 2018], have shown superior performance in terms of accuracy and efficiency. However, they require large amounts of training data and computational resources.

In contrast, patch-based methods divide each image into small patches and attempt to match these patches across different views. The 3D structure is then reconstructed by merging the resulting patches. Classical patch-based methods include PatchMatch Stereo [Michael Bleyer and Rother, 2011], which uses a randomized search strategy to find correspondences between image patches. Recent patch-based methods, such as MVSNet [Yao *et al.*, 2018], incorporate deep learning techniques to improve the accuracy and efficiency of patch-based methods. Patch-based methods are slower but more memory-efficient and can handle non-Lambertian surfaces better than dense matching-based methods.

2.1.4. Surface reconstruction

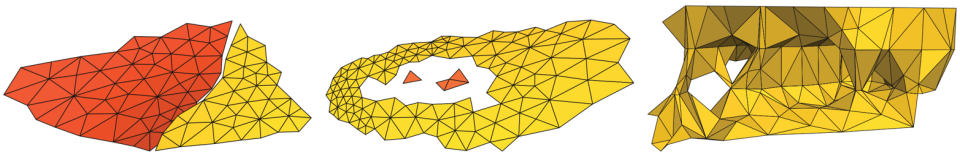
Once the dense point cloud is obtained, surface reconstruction algorithms are applied to create a mesh representation of the object's surface (see Figure 2.2). Surface reconstruction algorithms aim to generate a continuous and smooth surface representation of the 3D point cloud data by estimating the local surface properties such as normals, curvature, and connectivity. These algorithms convert a set of unordered 3D points into a mesh representation, which can be used for visualization, analysis, and simulations. Surface reconstruction methods can be broadly categorized as volumetric, implicit, and explicit methods.

Volumetric methods typically convert the point cloud data into a 3D voxel grid and then extract the surface using techniques such as marching cubes [Lorensen and Cline, 1987]. These methods are computationally efficient but can result in low-quality surfaces due to voxelization artifacts.

Implicit methods, such as the Poisson surface reconstruction [Kazhdan *et al.*, 2006] and its variants, represent the surface as the zero-crossing of an implicit function, which is defined by the point cloud data. These methods generate high-quality surfaces with fewer artifacts but are computationally expensive.

Explicit methods, such as Delaunay triangulation [D.-T. Lee and Schachter, 1980], directly generate a mesh representation of the surface by fitting the points to an initial mesh or constructing a mesh incrementally. While these methods can produce high-quality meshes, they are often computationally intensive and may result in meshes with artifacts.

In recent years, deep learning-based methods have shown promising results in surface reconstruction tasks. For example, DeepSDF [J. J. Park *et al.*, 2019] learns a signed distance function that can represent complex surfaces with high accuracy. Deep marching cubes [Liao *et al.*, 2018] can generate high-quality meshes from 3D point cloud data with improved efficiency compared to traditional marching cubes. However, these methods can require a large amount of training data and computational resources for training and inference.



(a) Gap (with partial overlap) (b) Hole (with islands) (c) Topological noise

Figure 2.3: Examples of three common types of flaws and defects typically found in urban meshes. Figure taken from Attene *et al.*, (2013).

Despite the advances in surface reconstruction, the generated meshes may contain errors and artifacts [Attene *et al.*, 2013] due to incomplete or noisy inputs (see Figure 2.3). To further improve the quality of the reconstructed mesh, various mesh optimization techniques can be applied, which typically involve iteratively refining the mesh geometry and connectivity while preserving important surface properties such as curvature and smoothness [Faugeras and Keriven, 1998; Pons *et al.*, 2007].

It is noteworthy that the mesh dataset used in this thesis is primarily generated by the commercial software Bentley ContextCapture [Bentley, 2015]. The core surface reconstruction method [Vu *et al.*, 2012] behind this encompasses the following three steps:

1. Generation of a Quasidense Point Cloud: Initially, a quasidense point cloud is created using standard multiview stereo techniques. This forms the foundation for constructing the 3D model, involving images captured from multiple viewpoints.

2. Extraction of a Mesh that Respects Visibility Constraints: Based on the point cloud, an initial mesh is constructed through Delaunay triangulation. This mesh is then adjusted using graph-based optimization methods, such as the minimum s-t cut algorithm, to more closely resemble the actual 3D surface while respecting visibility constraints within the scene.

3. Variational Refinement of the Mesh: Finally, this initial mesh is refined through a variational approach, enhancing its photometric consistency with the image data, thus improving the details and accuracy of the reconstruction.

This pipeline employs a more efficient and lightweight Lagrangian framework for energy optimization to enhance processing efficiency for large-scale datasets.

2.1.5. Texture mapping

Surface reconstruction generates a mesh that approximates the geometric shape of an object, and the final step in mesh generation is texture mapping, which plays a crucial role in creating a realistic appearance for the reconstructed object (see Figure 2.2). To achieve high-quality texture mapping, the selection of the best images to cover the entire surface of the mesh with high resolution and minimize texture distortion is important. This process can be done manually [Se and Jasiobedzki, 2006], but it is time-consuming and error-prone. Alternatively, automatic image selection methods can be used to select the best images based on several factors such as image quality, resolution, lighting conditions, and viewpoint coverage [Frueh *et al.*, 2004].

Once the images have been selected, texture mapping involves projecting the 2D texture images onto the 3D mesh surface. The texture mapping methods can be categorized into two types: parameterization-based and atlas-based methods. Parameterization-based methods, such as angle-based flattening (ABF++) [Sheffer *et al.*, 2005] and least squares conformal maps (LSCM) [Lévy *et al.*, 2002], aim to find a mapping between the 3D mesh surface and a 2D domain while preserving the texture details. In contrast, atlas-based methods (see Figure 2.4), such as texture atlas [Purnomo *et al.*, 2004], create a seamless 2D texture map by partitioning the 3D mesh into several charts and then mapping them onto a 2D plane. Each method has its own advantages and limitations, for example, parameterization-based methods may produce distortions, whereas atlas-based methods may result in seams.

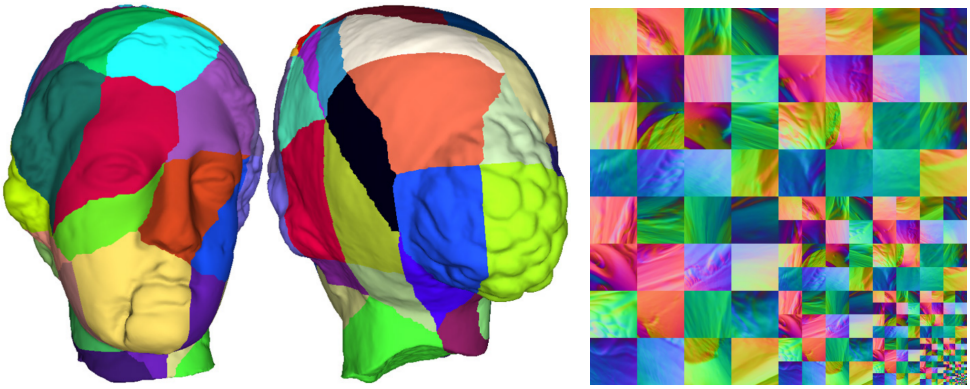


Figure 2.4: An example of Seamless Texture Atlases: where the 3D model is divided into quadrilateral charts, and the corresponding texture is represented as a flat seamless atlas. Figure taken from Purnomo *et al.*, (2004).

Another important step in texture mapping is color correction, which aims to balance the color differences among the input images and create a consistent texture appearance. This step can be done automatically using color transfer techniques, such as histogram matching [A. Neumann and L. Neumann, 2005] and color transfer [Gooch *et al.*, 2001] method, which can achieve good results but may not be effective for complex lighting

conditions. Deep learning-based methods, such as deep image harmonization [Tsai *et al.*, 2017] and deep photo style transfer [Luan *et al.*, 2017], have been proposed to address this issue and can produce high-quality color transfer results.

2.1.6. Summary

In summary, textured mesh generation is a complex and challenging task that requires the integration of various algorithms and techniques. While traditional methods have achieved significant success, recent state-of-the-art deep learning-based approaches have shown great potential in advancing the field. In this research, we use textured meshes from Helsinki3D [Helsinki, 2019], which are reconstructed via the commercial software ContextCapture [BENTLEY, 2016] using traditional methods. These meshes are used as input to extract semantics by leveraging both mesh geometry and texture information. By combining these two types of information, we aim to achieve more accurate and robust semantic segmentation of urban textured meshes.

2.2. Large-scale 3D urban modeling

This section on large-scale urban scene modeling covers a range of topics related to the creation of 3D models of urban environments. The main content of the section includes the understanding of 3D urban scenes, lightweight modeling, and encoded storage. Firstly, a critical aspect of understanding 3D urban scenes is applying basic methods for semantic segmentation of 3D data. These techniques allow for accurate classification of different elements of the urban environment. Subsequently, lightweight 3D urban modeling aims to simplify the representation of different object categories in urban scenes, such as terrain, buildings, vegetation, and infrastructure. This process involves reducing the complexity of 3D models and making them easier to analyze, manipulate, and use in various applications. Lastly, the encoded storage of 3D city models introduces two mainstream 3D urban coding formats, namely CityGML [Gröger and Plümer, 2012; OGC, 2012] and CityJSON [Ledoux *et al.*, 2019]. These formats provide a standardized method for storing and sharing 3D urban models, which facilitates collaboration across different platforms and applications. Overall, the topics covered in this section are fundamental to the creation of precise and practical 3D models of urban environments.

2.2.1. Urban scene understanding

The understanding of 3D urban scenes has been a highly researched topic in photogrammetry and computer vision for decades, thanks to the vast potential that 3D data holds in various fields such as urban planning [Madrazo *et al.*, 2012; Sindram and Kolbe, 2014], architecture [Benner *et al.*, 2005; Kolbe and Donaubauer, 2021], and transportation [Geiger *et al.*, 2014]. Semantic information, or the contextual meaning

and interpretation of 3D data, is critical for extracting meaningful insights from the data and maximizing its potential.

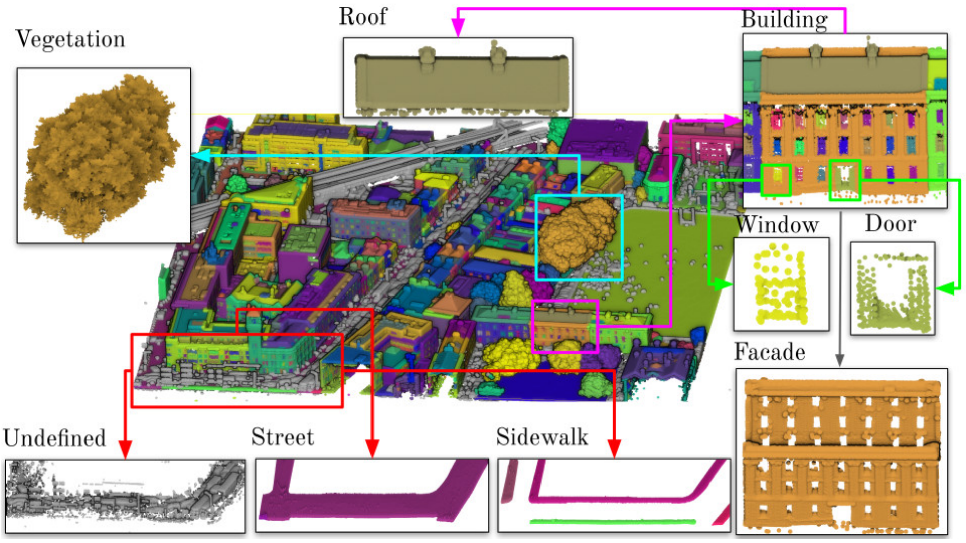


Figure 2.5: Visualization of Dublin city annotated LiDAR point cloud. Figure taken from Zolanvari *et al.*, (2019).

Data labeling is the primary method for obtaining semantic information, involving assigning labels or categories to various elements in the 3D data, such as buildings, trees, and roads (see Figure 2.5). Manual labeling by professionals is the most accurate method, as it captures subtle nuances and variations in the scene. However, it can be time-consuming and expensive for large-scale datasets [Hackel *et al.*, 2017; Serna *et al.*, 2014]. Semi-automatic labeling combines human expertise with automated algorithms, making it more efficient and increasing labeling throughput [Patil *et al.*, 2019; Zimmer *et al.*, 2019]. It requires labeling professionals to possess a higher skill set as they must instruct and fine-tune the algorithm to accurately segment the data. Fully automatic labeling methods are the most efficient, but their accuracy is often lower than manual or semi-automatic labeling methods, especially for complex urban scenes with a high degree of variability and noise [Q. Hu *et al.*, 2021; Verdie *et al.*, 2015]. Therefore, manual correction of mislabeled data is essential for accuracy.

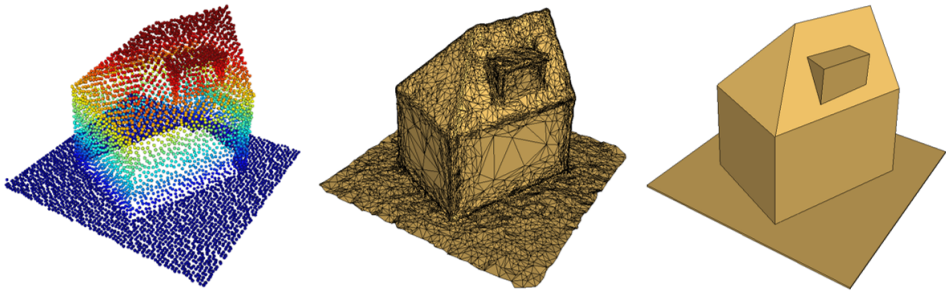
In the task of generating semantic labels for large-scale urban scenes, both unsupervised and supervised methods are commonly employed. Unsupervised methods typically rely on manual feature design or clustering to segment the data and assign labels to scenes, as seen in previous works [Verdie *et al.*, 2015]. On the other hand, supervised methods utilize pre-labeled datasets to learn features and feature distributions and are capable of predicting labels for unlabeled data. However, recent deep learning-based supervised methods, such as PointNet [Charles *et al.*, 2017] and PointNet++[Qi *et al.*,

2017], require a large volume of labeled data to achieve good performance. Therefore, they are often used in conjunction with unsupervised methods to refine and improve labeling accuracy [Landrieu and Boussaha, 2019; Landrieu and Simonovsky, 2018].

In conclusion, labeling 3D data is a complex and challenging task that requires a combination of human expertise and automated algorithms. Despite its challenges, it is a fundamental step towards utilizing the vast potential of 3D urban data in various applications.

2.2.2. Lightweight 3D city model reconstruction

Lightweight 3D city models utilize the minimum number of polygons to accurately represent solid models or surface models, thereby preserving the original geometric structure of the input data (see Figure 2.6). In this context, solid models specifically refer to watertight, 2-manifold 3D solid models, and surface models are defined as non-closed, 2-manifold polygonal patches. When compared to traditional triangular meshes, these lightweight 3D models substantially reduce the storage, transmission, and computational demands in various applications.



(a) Point cloud (b) Triangular mesh (c) Lightweight model

Figure 2.6: Examples of a lightweight building model: (a) shows the input point cloud, where the color gradient from red to blue represents a decrease in height; (b) depicts the reconstructed triangular mesh, with black lines indicating the edges of the triangles; (c) illustrates the lightweight model of a solid building, accompanied by a polygonized ground surface. The black lines depict the edges of the polygons.

The generation of lightweight 3D city models (see Figure 2.7) has become an essential aspect of various applications, such as urban planning [Madrazo *et al.*, 2012; Sindram and Kolbe, 2014], environmental analysis [Leszek, 2015], and virtual reality [Doyle *et al.*, 1998]. The aim of this process is to create lightweight, simplified, and highly accurate 3D city models, which can significantly reduce the computational costs associated with their applications, analysis, rendering, and sharing [W. Li *et al.*, 2011].

The main approaches for lightweight 3D city model reconstruction include manual,

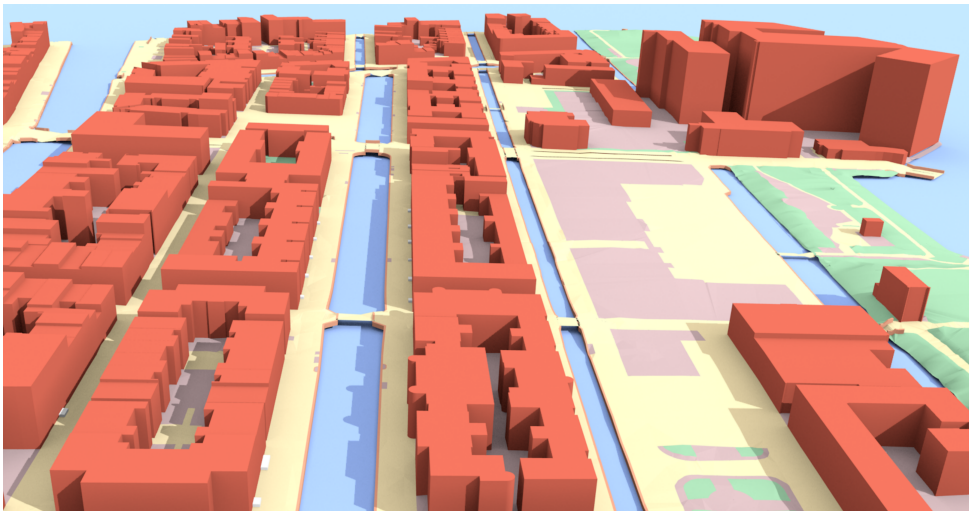


Figure 2.7: 3D city models of Leiden, generated by 3dfier. Figure from Ledoux *et al.*, (2021).

semi-automatic, and fully automatic methods. Manual methods involve the use of specialized software, such as SketchUp [Trimble, 2000], Autodesk Revit [Autodesk, 2000], or Rhinoceros 3D [Rhinocentre, 1992]. However, for large-scale urban scenes, this method can be time-consuming. Semi-automatic methods aim to reduce user interaction by automatically fitting or matching models to the input data on top of the manual process [Arikan *et al.*, 2013; S. N. Sinha *et al.*, 2008]. Although these methods save time and effort, they require more sophisticated algorithms and may not be as accurate as fully manual methods.

Fully automatic methods represent the most advanced approach to 3D urban scene modeling. These methods generate models entirely from 3D data, which can be divided into surface fitting, model fitting, and procedure modeling. Surface fitting, akin to surface reconstruction, focuses primarily on surface-represented categories such as terrain and roads in urban scenes [K. Kumar *et al.*, 2019; K. Kumar *et al.*, 2018]. However, surface fitting typically has specific requirements for compact representations. Model fitting involves matching and fitting input data with simple geometric models, such as vertices [L. Li *et al.*, 2022], wireframes [Tian *et al.*, 2022], planes [Nan and Wonka, 2017], cylinders [Du *et al.*, 2019], or predefined models [Xiong *et al.*, 2014]. This method is usually applicable to objects with complete 3D structures, such as buildings, vegetation, and urban furniture. Procedure modeling is another fully automatic approach, which involves the construction of large-scale virtual urban scenes using certain grammar rules and parameters [I. Demir *et al.*, 2014]. It is suitable for virtual scenes such as urban design, games, and movies, but it may not accurately model real scenes. Nevertheless, inverse procedure modeling can be used for real scene modeling, as it can infer procedural models and parameters from the input geometry and semantics [Vanegas *et al.*, 2012].

In conclusion, lightweight 3D city models are crucial for various applications. Yet, automatically generating accurate and detailed models of complex urban environments remains a formidable challenge. Although progress has been made in automated reconstruction algorithms, capturing the intricate details of urban landscapes accurately continues to be difficult. Further research and development are essential to enhance the precision and efficiency of these methods, enabling broader and more effective implementation of 3D models in urban planning, architecture, and beyond.

2.2.3. Encoding 3D city models

With the growth of smart cities and the increasing availability of 3D data, the demand for 3D city models has grown significantly in recent years. These models are used in a wide range of applications. However, creating and sharing these models can be challenging, as they can be complex and difficult to work with.

To address this issue, the Open Geospatial Consortium (OGC) developed CityGML [Gröger and Plümer, 2012; OGC, 2012], an open data model for 3D urban objects. CityGML provides a standard way to represent and exchange 3D city models, making it easier to manage and analyze data from various sources and applications. It enables the integration of data from multiple sources and provides a common language and framework for storing, exchanging, and sharing 3D city models. Besides, it supports various geometrical elements (including volumes), attributes attached to objects, semantics for surfaces with different levels of detail (LoD) (see Figure 2.8), and texture materials. Despite its benefits, CityGML has some limitations. Its complexity and large file sizes can make it challenging to use and process. In response to these challenges, a more lightweight and streamlined version of CityGML, called CityJSON [Ledoux *et al.*, 2019], has been developed. CityJSON provides a more accessible and efficient way to store and share 3D city models, making it easier to use and integrate with other systems.

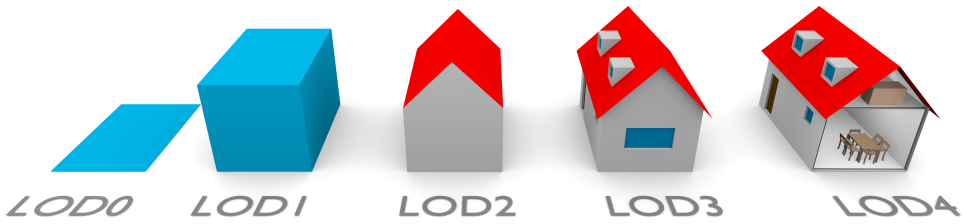


Figure 2.8: An example of the Level of Detail (LoD) in a CityGML building model: CityGML 2.0 [Gröger and Plümer, 2012; OGC, 2012] defines multiple LoDs ranging from simple building footprints (LoD1) to detailed models, including interior and exterior elements (up to LoD4). In CityGML 3.0 [OGC, 2021], LoD4 has been removed, allowing interior details to be represented in LoD2 or LoD3. Figure taken from Biljecki *et al.*, (2014).

Overall, the development of CityGML and CityJSON has significantly contributed to the standardization and accessibility of 3D city models. As the demand for these models continues to grow, it is likely that further innovations will be made to improve their usability and functionality.

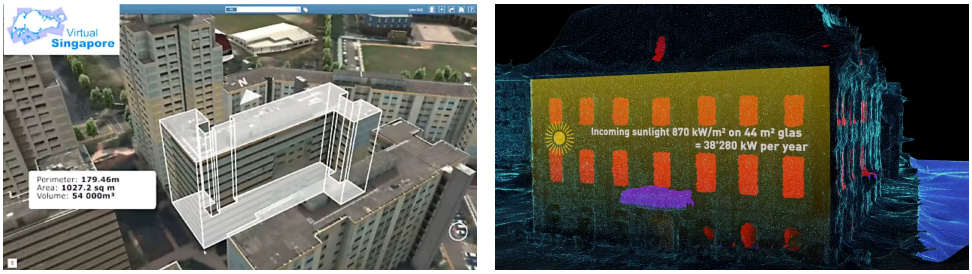
2.3. Applications of 3D city models

The growing prevalence of 3D city models is driven by advances in 3D geoinformation technology, which enhances the creation, analysis, and management of urban environment data. With the rising popularity of 3D data such as LiDAR and photogrammetry, as well as the promotion of CityGML and CityJSON 3D urban data model standards, applications of 3D city models have become more important in various fields. These applications enable professionals to visualize and interact with urban data in a more intuitive and immersive way, leading to better decision-making and improved urban design.

In urban 3D scene applications, 3D city models are often used as base maps, which contain essential information such as the geometry, color, and semantics of urban scenes. By combining them with external data such as meteorological data, environmental monitoring data, and urban surveillance data, complex 3D computational analyses can be performed, and the results can be displayed through rendering based on 3D space. These applications can be further divided into two categories: applications without semantic information and those with semantic information [Biljecki *et al.*, 2015b; Ross, 2011]. The former refers to 3D city models that only contain geometry and color information. Related applications can be built on this basis, such as visualization [Glander and Döllner, 2009; Mao and Ban, 2011; L. Zhang *et al.*, 2014], virtual touring [Koutsoudis *et al.*, 2007; Santana *et al.*, 2017], geometric measurement [Wong and Ellul, 2016], and shadow analysis [Alam *et al.*, 2013; Biljecki *et al.*, 2017]. In the absence of external data, these applications are still useful for basic purposes. However, with external data, more advanced applications can be performed, such as flood simulation analysis by combining precipitation data [Amirebrahimi *et al.*, 2016; Jain *et al.*, 2000], computational fluid dynamics (CFD) analysis of urban scenes by combining weather data [Kazak *et al.*, 2022; Maragkogiannis *et al.*, 2014; Paen *et al.*, 2022], and signal strength analysis by injecting GNSS receiver signals [R. Kumar and Petovello, 2014; L. Wang *et al.*, 2012; L. Wang *et al.*, 2013].

If semantic information is added on the basis of geometry and color, a wider range of applications can be extended (see Figure 2.9). In the absence of external data, semantic information can significantly enhance the level of applications, as mentioned above. For example, based on semantic information, users can display specific geographic elements [Pantoja-Rosero *et al.*, 2022; Rouhani *et al.*, 2017] such as building windows, roof superstructures, etc., assist in the advanced geometric measurement of objects and object components [Over *et al.*, 2010; Ropinski *et al.*, 2005; Soheilian *et al.*, 2013; Willenborg *et al.*, 2018], such as road length, window, and roof area, building volume,

etc., and conduct change detection [Pédrinis *et al.*, 2015; Qin, 2014; Sharkawi and Rahman, 2014] such as comparison of building damage and changes in vegetation area.



(a) (b)

Figure 2.9: Two examples illustrating the application of semantic 3D city models include: (a) shows the geometric calculation of buildings in the Virtual Singapore dataset (Figure taken from Authority, (2021)); (b) shows the computation of incoming sunlight in the VarCity dataset (Figure taken from Vanhoey *et al.*, (2017)).

With external data, more advanced applications can be performed, such as object attribute display and management based on GIS data [Apollonio *et al.*, 2012; Padsala and Coors, 2015] (such as building age, function, and type, vegetation quantity, height, type, etc.), visibility analysis [L. Liu *et al.*, 2010; P. P.-J. Yang *et al.*, 2007] and sunlight analysis [Besuievsky *et al.*, 2014; Y. Liu *et al.*, 2015; Yasumoto *et al.*, 2012] based on facade windows, solar potential analysis [Biljecki *et al.*, 2015a; Eicker *et al.*, 2014; Santos *et al.*, 2014] based on roof, permeability analysis [L. Luo *et al.*, 2021; C. Wang *et al.*, 2019] based on urban low vegetation, air pollution analysis [San José *et al.*, 2012; Uznir *et al.*, 2013] based on roof chimneys, and automatic driving simulation [Schwab *et al.*, 2020; Wagener *et al.*, 2022; Wysocki, 2020] based on roads and road markings. Various sensors can also be combined for real-time detection and prediction, such as traffic detection [Brédif, 2013; Chun *et al.*, 2008; Lothe *et al.*, 2010], intensive crowd detection [Aschwanden *et al.*, 2009; Q. Zhang and Chan, 2020], infrastructure monitoring [Corongiu *et al.*, 2018; Sofia *et al.*, 2020], building energy monitoring [Agugiaro *et al.*, 2018; D. Lee *et al.*, 2016], wind flow simulations [García-Sánchez *et al.*, 2021; Paen *et al.*, 2022], etc.

In conclusion, semantic information can greatly enhance and expand the breadth and depth of applications based on 3D city models, making it possible to finely manage, analyze, predict, and make decisions about the city [Biljecki *et al.*, 2015b]. This enables professionals to obtain valuable insights and make more informed decisions, leading to better urban design and management. The utilization of 3D city models can also improve public participation in urban planning by providing an intuitive and interactive way to understand urban data, promoting transparency and accountability in decision-making. Additionally, integrating diverse data sources and sensors within these models supports smart city development. This integration enables real-time monitoring and analysis of urban activities, ultimately enhancing the quality of life for residents.

3

Benchmarking object-level semantic segmentation*

This chapter delves into object-level semantic understanding within large-scale urban environments. While deep learning methods for semantic segmentation of textured meshes can significantly enhance this understanding, they are heavily dependent on the availability of extensively labeled data. To overcome this limitation, this chapter makes three key contributions: (1) a new benchmark dataset of semantic urban meshes, (2) a novel semi-automatic annotation framework, and (3) an annotation tool for 3D meshes. In particular, our dataset covers about 4 km^2 in Helsinki (Finland), with six classes, and we estimate that we save about 600 h of labeling work using our annotation framework, which includes initial segmentation and interactive refinement. We also compare the performance of several state-of-the-art 3D semantic segmentation methods on the new benchmark dataset. Other researchers can use our results to train their networks: the dataset is publicly available, and the annotation tool is released as open-source.

*This chapter is based on the paper: Gao, W., Nan, L., Boom, B. and Ledoux, H. (2021). 'SUM: A benchmark dataset of Semantic Urban Meshes'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 179, pp. 108–120. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2021.07.008.

3.1. Introduction

Understanding the urban environment from 3D data (e.g. point clouds and 3D meshes) is a long-standing goal in photogrammetry and computer vision [Hackel *et al.*, 2017; Matrone *et al.*, 2020]. The fast recent developments in data acquisition technologies and processing pipelines have allowed us to collect a great number of datasets on our 3D urban environments. Prominent examples are Google Earth [Google, 2012], textured meshes covering entire cities (e.g. Helsinki [Helsinki, 2019]), or point clouds covering entire countries (e.g., the Netherlands AHN [AHN, 2019]). These datasets have attracted interest because of their potential in several applications, for instance, urban planning [Czyska and Rubinowicz, 2014; Ran, 2011], positioning and navigation [Cappelle *et al.*, 2012; Peyraud *et al.*, 2013; Li-Ta *et al.*, 2015], spatial analysis [Yaagoubi *et al.*, 2015], environmental analysis [Yichuan *et al.*, 2016], and urban fluid simulation [García-Sánchez *et al.*, 2014].

To effectively understand the urban phenomena behind the data, a large amount of ground truth is typically required, especially when applying supervised learning-based techniques, such as a deep Convolutional Neural Network (CNN). The recent development of machine learning (especially deep learning) techniques has demonstrated promising performance in semantic segmentation of 3D point clouds [Charles *et al.*, 2017; Landrieu and Simonovsky, 2018; Thomas *et al.*, 2019]. Textured meshes generated through photogrammetric pipelines from multiview images can also represent and interpret urban 3D scenes. However, these derivative products may contain errors that affect the accuracy of semantic segmentation. Nevertheless, compared to point clouds, a surface representation (in the form of a 3D mesh, often with textures, see Figure 3.1 and Figure 3.2 for an example) of the urban scene has multiple advantages: easy to acquire, compact storage, accurate, and with well-defined topological structures. This means that 3D meshes have the potential to serve as input for scene understanding. As a consequence, there is an urgent demand for large-scale urban mesh datasets that can be used as ground truth for both training and evaluating the 3D semantic segmentation workflows.

In this paper, we aim to establish a benchmark dataset of large-scale urban meshes reconstructed from aerial oblique images. To achieve this goal, we propose a semi-automatic mesh annotation framework that includes two components: (1) an automatic process to generate intermediate labels from the raw 3D mesh and (2) manual semantic refinement of those labels. For the intermediate label generation step, we have developed a semantic mesh segmentation method that classifies each triangle into a pre-defined object class. This semantic initialization allows us to achieve an overall accuracy of 93.0% in the classification of the triangle faces in our dataset, saving significant efforts for manual labeling. Then, in the semantic refinement step, a mesh annotation tool (which we have developed) is used to refine the semantic labels of the pre-labeled data (at the triangle and segment levels).

We have used our proposed framework to generate a semantic-rich urban mesh dataset consisting of 19 million triangles and covering about 4 km^2 with six object classes

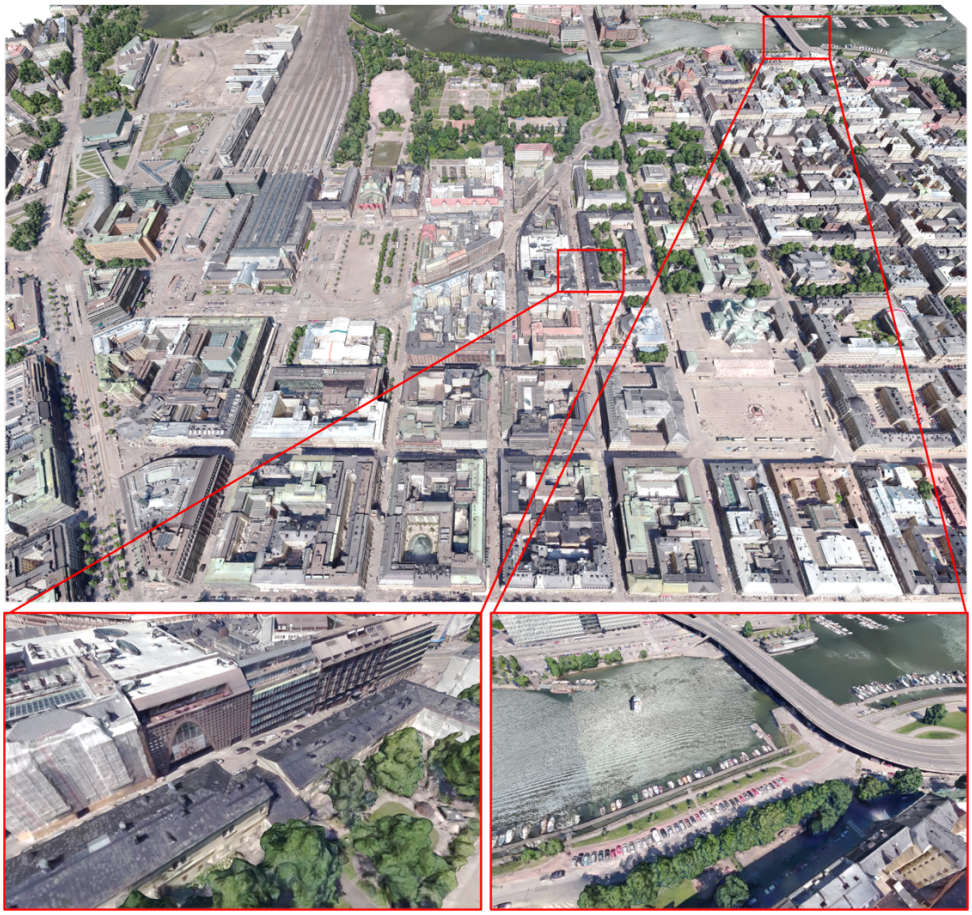


Figure 3.1: Part of the semantic urban mesh benchmark dataset shown as a textured mesh

commonly found in an urban environment: terrain, high-vegetation, building, water, vehicle, and boat (see Figure 3.2 shows an example from our dataset). With our semi-automatic annotation framework, generating the ground truth took only about 400 hours; we estimate that manually labeling the triangles would have taken more than 1000 hours. The contributions of our work are:

- a semantic-rich urban mesh dataset of six classes of common urban objects with texture information;
- a semi-automatic mesh annotation framework consisting of two parts: a pipeline for semantic mesh segmentation and an annotation tool for semantic refinement;
- a comprehensive evaluation and comparison of the state-of-the-art semantic segmentation methods on the new dataset.

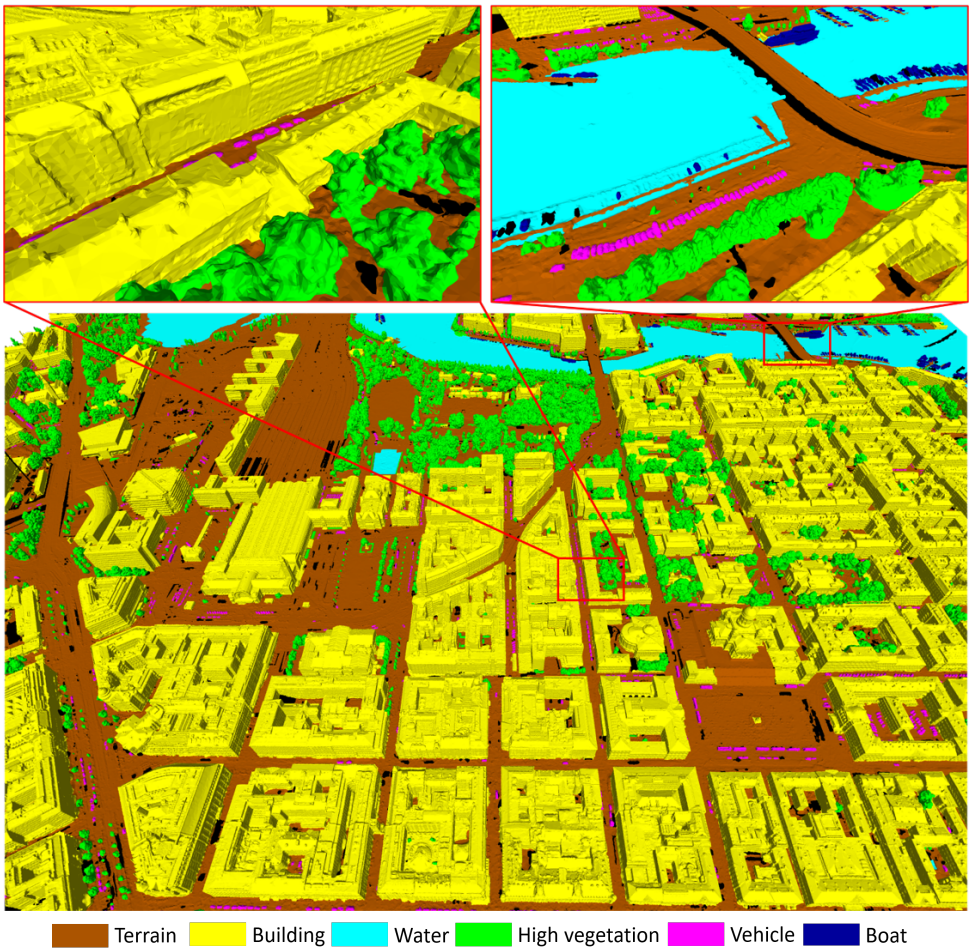


Figure 3.2: Part of the semantic urban mesh benchmark dataset, showing the semantic classes (unclassified regions are in black)

The benchmark dataset is freely available, and the semantic mesh segmentation methods and the annotation software for 3D meshes are released as open-source[1].

3.2. Related work

Urban datasets can be captured with different sensors and reconstructed with different methods, and the resulting datasets will have different properties. Most benchmark urban datasets focus on point clouds, whereas our semantic urban benchmark dataset

[1] https://3d.bk.tudelft.nl/projects/meshannotation/

is based on textured triangular meshes.

The input of the semantic labeling process can be raw or pre-labeled urban datasets such as the automatically generated results from over-segmentation or semantic segmentation (see subsection 3.3.3). Regardless of the input data, it still needs to be manually checked and annotated with a labeling tool, which involves selecting a correct semantic label from a predefined list for each triangle (or point, depending on the dataset) by users. In addition, some interactive approaches can make the labeling process semi-manual. However, unlike our proposed approach, the labeling work of most of the 3D benchmark data does not take full advantage of over-segmentation and semantic segmentation on 3D data and interactive annotation in the 3D space.

We present in this section an overview of the publicly available semantic 3D urban benchmark datasets categorized by sensors and reconstruction types (see Table 3.1). More specifically, we elaborate on the quality, scale, and labeling strategy of the existing urban datasets regarding semantic segmentation.

3.2.1. Photogrammetric products

Dense point clouds. The *Campus3D* [X. Li *et al.*, 2020] is, to our knowledge, the first aerial point cloud benchmark. The coarse labeling is conducted in 2D projected images with three views, and the grained labels are refined in 3D with user-defined rotation angles. The dataset covers only the campus of the National University of Singapore and is thus not representative of a typical urban scene.

SensatUrban [Q. Hu *et al.*, 2021] is another example of the photogrammetric point clouds covering various urban landscapes in two cities of the UK. The semantic points are manually annotated via the off-the-shelf software tool CloudCompare [Girardeau-Montaut, 2016], and the overall annotation is reported to have taken around 600 hours. The dataset also contains several areas without points, especially for water surfaces and regions with dense objects. The leading causes are the Lambertion surface assumption during the image matching and the inadequate image overlapping rate during the flight.

Similarly, the *Swiss3DCities* [Can *et al.*, 2021] was recently released that covers three cities in Zurich but twice smaller than the SensatUrban. The annotation work was conducted on a simplified mesh in the software Blender [Foundation, 2002], and then the semantics were transferred to the mesh vertices, which are regarded as point clouds, via the nearest neighbor search. The mesh simplification may result in the loss of small-scale objects such as building dormers and chimneys, and the automatic transfer of the labels could have introduced errors in the ground truth.

Triangle meshes. To the best of our knowledge, the *ETHZ RueMonge 2014* [Riemenschneider *et al.*, 2014] is the first urban-related benchmark dataset available as surface meshes. The label for each triangle is obtained from projecting

Name	Platforms	Year	Data Type	Area[a]/ Length	Classes	Points / Triangles	RGB	Automatic Pre-labelling	Annotation	Time Cost (hours)
Oakland 3D	MLS	2009	Point Cloud	1.5 km	5	1.6 M	No	No	3D Manually	Not reported
Paris-rue-Madame	MLS	2014	Point Cloud	0.16 km	17	20 M	No	2D semantic segmentation	3D Semi-manually	Not reported
iQmulus	MLS	2015	Point Cloud	10 km	8	300 M	No	No	2D Semi-manually	Not reported
Semantic3D	TLS	2017	Point Cloud	-	8	4000 M	Yes	No	2D & 3D Semi-manually	Not reported
Paris-Lille-3D	MLS	2018	Point Cloud	1.94 km	9	143 M	No	No	3D Manually	Not reported
SemanticKITTI	MLS	2019	Point Cloud	39.2 km	25	4549 M	No	No	3D Manually	1700
Toronto-3D	MLS	2020	Point Cloud	1.0 km	8	78.3 M	Yes	No	3D Manually	Not reported
ISPRS	ALS	2012	Point Cloud	0.1 km^2	9	1.2 M	No	No	3D Manually	Not reported
AHN3	ALS	2019	Point Cloud	41,543 km^2	4	415.43 B[b]	No	3D semantic segmentation	3D Manually	Not reported
DublinCity	ALS	2019	Point Cloud	2.0 km^2	13	260 M	No	No	3D Manually	2500
DALES	ALS	2020	Point Cloud	10.0 km^2	8	505.3 M	No	3D semantic segmentation	3D Manually	Not reported
LASDU	ALS	2020	Point Cloud	1.02 km^2	5	3.12 M	No	No	3D Manually	Not reported
ETHZ RueMonge	Auto-mobile camera	2014	Mesh	0.7 km	9	1.8 M (lowres)[c]	Yes (per vertex)[d]	2D over-segmentation	2D Semi-manually	230 (701 frames)[e]
Campus3D	UAV camera	2020	Point Cloud	1.58 km^2	14	937.1 M	Yes	No	2D & 3D Manually	Not reported
SensatUrban	UAV camera	2020	Point Cloud	6 km^2	13	2847.1 M	Yes	No	3D Manually	600
Swiss3DCities	UAV camera	2020	Point Cloud	2.7 km^2	5	226 M	Yes	No	3D Manually (on mesh)	144 (1 M Triangles)[f]
Hessigheim 3D	UAV Lidar & camera	2021	Point Cloud & Mesh	0.19 km^2	11	125.7 M / 36.76 M[g]	Yes (texture)[h]	No	3D Manually[i]	Not reported
SUM-Helsinki (Ours)	Airplane camera	2021	Mesh	4 km^2	6	19 M	Yes (texture)[h]	3D over-segmentation & 3D semantic segmentation	3D Semi-manually	400

[a] The area was measured in a 2D map.
[b] The number of total points (i.e., 415.43 billion) is estimated.
[c] The low-resolution meshes contain 1.8 million triangle faces, according to the publications.
[d] An RGB color was assigned to each triangle vertex.
[e] The frames were from video sequences.
[f] About one million triangles (16 tiles) from simplified mesh were labeled, which took around 6 to 12 hours per tile.
[g] The number of LiDAR points is 125.7 million and the number of triangle faces is 36.76 million.
[h] The color of each triangle face corresponds to a patch of the texture image.
[i] The LiDAR point clouds were manually annotated and the labels were transferred to the mesh.

Table 3.1: Comparison of existing 3D urban benchmark datasets, including Oakland 3D [Munoz *et al.*, 2009], Paris-rue-Madame [Serna *et al.*, 2014], iQmulus [Vallet *et al.*, 2015], Semantic3D [Hackel *et al.*, 2017], Paris-Lille-3D [Roynard *et al.*, 2018], SemanticKITTI [Behley *et al.*, 2019], Toronto-3D [Tan *et al.*, 2020], ISPRS [Niemeyer *et al.*, 2014], AHN3 [AHN, 2019], DublinCity [Zolanvari *et al.*, 2019], DALES [Varney *et al.*, 2020], LASDU [Ye *et al.*, 2020], ETHZ RueMonge [Brostow *et al.*, 2009; Riemenschneider *et al.*, 2014], Campus3D [X. Li *et al.*, 2020], SensatUrban [Q. Hu *et al.*, 2021], Swiss3DCities [Can *et al.*, 2021], and Hessigheim 3D [Kölle *et al.*, 2021; Laupheimer *et al.*, 2020]

selected images that are manually labeled from over-segmented image sequences [Brostow *et al.*, 2009]. In fact, due to the error of multi-view optimization and the ambiguous object boundary within triangle faces, the datasets contain many misclassified labels, making them unsuitable for training and evaluating supervised learning algorithms.

Hessigheim 3D [Kölle *et al.*, 2021; Laupheimer *et al.*, 2020] is a small-scale semantic urban dataset consisting of highly dense LiDAR point clouds and high-resolution textured meshes. In particular, the mesh is generated from both LiDAR point cloud and oblique aerial images in a hybrid way. The labels of point clouds are manually annotated in CloudCompare [Girardeau-Montaut, 2016], and the labels of the mesh are transferred from the point clouds by computing the majority votes per triangle. However, if the mesh triangle has no corresponding points, some faces may remain unlabelled, which results in about 40% unlabelled area. In addition, this dataset contains non-manifold vertices, which makes it difficult to use directly.

3.2.2. LiDAR point clouds

Unlike photogrammetric point clouds, LiDAR point clouds usually do not contain color information. To annotate them properly, additional information is often required, e.g. images or 2D maps. LiDAR point cloud benchmark datasets are more common than photogrammetric ones.

Street-view datasets. The *Oakland 3D* [Munoz *et al.*, 2009] is one of the earliest mobile laser scanning (MLS) point cloud datasets, which was designed for the classification of outdoor scenes. It has five hand-labeled classes with 44 sub-classes but without color information and semantic categories like roof, canopy, or interior building block, which are typical for all street-view captured datasets.

Compared to *Oakland 3D, Paris-rue-Madame* [Serna *et al.*, 2014] is a relatively smaller dataset that used the 2D semantic segmentation results for 3D annotation. Specifically, the point clouds were projected onto images to extract the objects hierarchically with several unsupervised segmentation and classification algorithms. Although the 2D pre-labelled generation is fully automatic, different semantic categories require different segmentation algorithms resulting in difficulties in the classification of multiple classes.

The *iQmulus dataset* [Vallet *et al.*, 2015] is a 10 km street dataset annotated based on projected images in the 2D space. Specifically, the user first needs to extract objects by editing the image with a polyline tool and then assign labels to the extracted object regions. Some automatic functions are made for polyline editing in this framework, but the entire annotation pipeline is still complicated.

Unlike other street view datasets, *Semantic3D* [Hackel *et al.*, 2017] is a dataset consisting of terrestrial laser scanning (TLS) point clouds (the scanner is not moving and scans

are made from only a few viewpoints). It has eight classes, and colors were obtained by projecting the points onto the original images. There are two annotation methods: (1) annotating in 3D with an iterative model-fitting approach on manually selected points; (2) annotating in a 2D view by separate background from a drawn polygon in CloudCompare [Girardeau-Montaut, 2016]. Although it covers many urban scenes and includes RGB information, the acquired objects are incomplete because of the limited viewpoints and occlusions.

The other three typical MLS point cloud datasets that were manually labeled are *Paris-Lille-3D* [Roynard *et al.*, 2018], *SemanticKITTI* [Behley *et al.*, 2019], and *Toronto-3D* [Tan *et al.*, 2020].

Aerial-view datasets. As for ALS benchmark point clouds, representative datasets are *ISPRS* [Niemeyer *et al.*, 2014], *DublinCity* [Zolanvari *et al.*, 2019], and *LASDU* [Ye *et al.*, 2020] covering various scales of city landscapes and were annotated manually with off-the-shelf software. Instead of fully manual annotation, the *Dayton Annotated LiDAR Earth Scan (DALES)* [Varney *et al.*, 2020] used digital elevation models (DEM) to distinguish ground points with a certain threshold, the estimated normal to label the building points roughly, and satellite images to provide contextual information as references for annotators to check and label the rest of data. Similarly, the AHN3 dataset [AHN, 2019] was semi-manually labeled by different companies with off-the-shelf software. Besides, since the ALS measurement is conducted in the top view direction, unlike oblique aerial cameras, the obtained point clouds often miss facade information to a certain degree.

3.3. The semantic urban mesh dataset

3.3.1. Dataset specification

We have used Helsinki's 3D textured meshes as input and annotated them as a benchmark dataset of semantic urban meshes. Helsinki's raw dataset covers about 12 km^2, and it was generated in 2017 from oblique aerial images that have about a 7.5 cm ground sampling distance (GSD) using an off-the-shelf commercial software, namely ContextCapture [BENTLEY, 2016]. The source images have three color channels (i.e., red, green, and blue) and are collected from an airplane with five cameras that have 80% length coverage and 60% side coverage. To recover the 3D water bodies that do not fulfill the Lambertian hypothesis, 2D vector maps and ortho-photos are used when performing the surface reconstruction. Furthermore, processing like aerial triangulation, dense image matching, and mesh surface reconstruction were all performed with ContextCapture. It should be noticed that the entire region of Helsinki is split into tiles, and each of them covers about 250 m^2 [KIGA-digi, 2019]. As shown in Figure 3.3, we have selected the central region of Helsinki as the study area, which includes 64 tiles and covers about 4 km^2 map area (8 km^2 surface area) in total.

3.3.2. Object classes

We define the semantic categories for urban meshes by the most common objects in the urban environment with unambiguous geometry and texture appearance. Moreover, each triangle face is assigned to a label of one of the six semantic classes. Ambiguous regions (which account for about 2.6% of the total mesh surface area), such as shadowed regions or distorted surfaces, are labeled as unclassified (see Figure 3.4). The object classes we consider in the benchmark dataset are:

- **terrain**: roads, bridges, grass fields, and impervious surfaces;
- **building**: houses, high-rises, monuments, and security booths;
- **high vegetation**: trees, shrubs, and bushes;
- **water**: rivers, sea, and pools;
- **vehicle**: cars, buses, and lorries;
- **boat**: boats, ships, freighters, and sailboats;
- **unclassified**: incomplete objects like buses and trains, distorted surfaces like tables, tents and facades, construction sites, and underground walls.

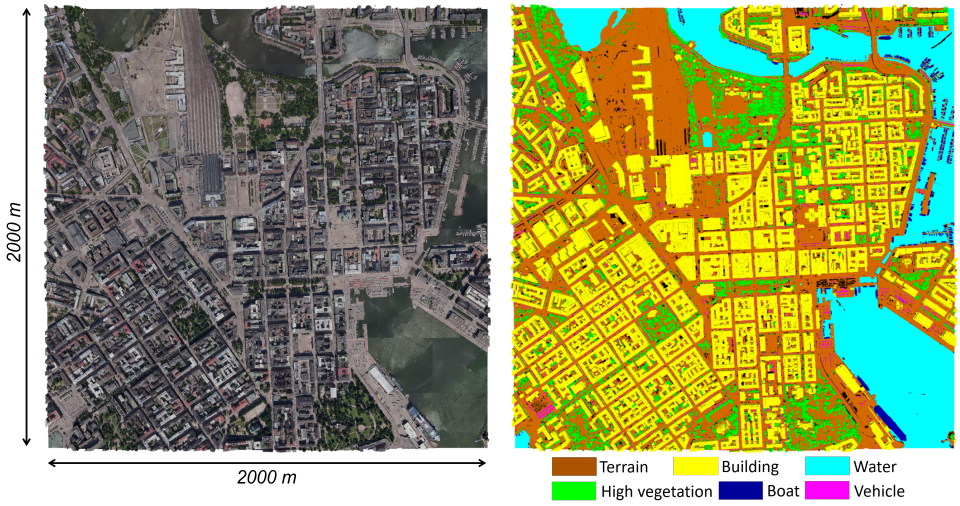


Figure 3.3: Overview of the semantic urban mesh benchmark: the left one shows the textured meshes covering about 4 km^2 map area; the right one shows the ground truth meshes. More views of the same scene (with different visualization styles) are shown in Figure 3.1 and Figure 3.2.

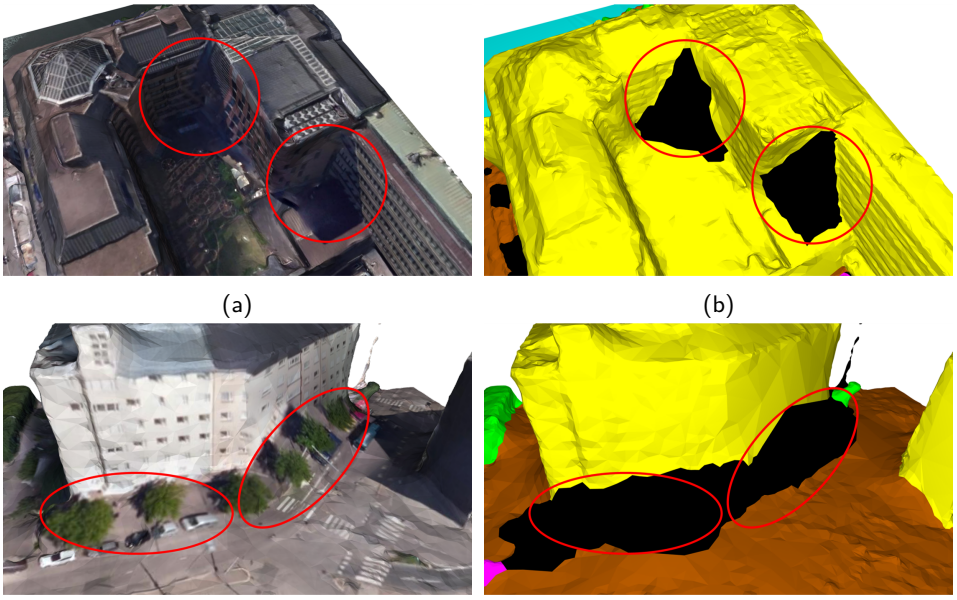


Figure 3.4: Ambiguous regions are labeled as unclassified (in black): (a) shows a shadow region with texture; (b) depicts a shadow region with semantic color; (c) shows a distorted region with texture; (d) illustrates a distorted region with semantic color.

3.3.3. Semi-automatic mesh annotation

Rather than manually labeling each triangle face of the raw meshes, we design a semi-automatic mesh labeling framework to accelerate the labeling process. Figure 3.5 shows the overall pipeline of our labeling workflow.

Given the fact that urban environments consist of a large number of planar regions in the data, we opt to label the data at the segment level instead of individual triangle faces. Specifically, we over-segment the input meshes into a set of planar segments. These segments can enrich local contextual information for feature extraction and serve as the basic annotation unit to improve annotation efficiency.

Instead of randomly choosing a mesh tile as input for annotation and refinement, which is insufficient for manual annotation progress, we favor picking a mesh tile that is more difficult to classify. Similar to active learning, we first compute the feature diversity (see Equation 3.1) to optimally select a mesh tile containing a variety of classes and objects at different scales and complexity. The feature diversity F_m of tile m is computed as

$$F_m = \frac{\sum_{i=1}^{N_f} \left(f_i - \bar{f}\right)^2}{N_f} \tag{3.1}$$

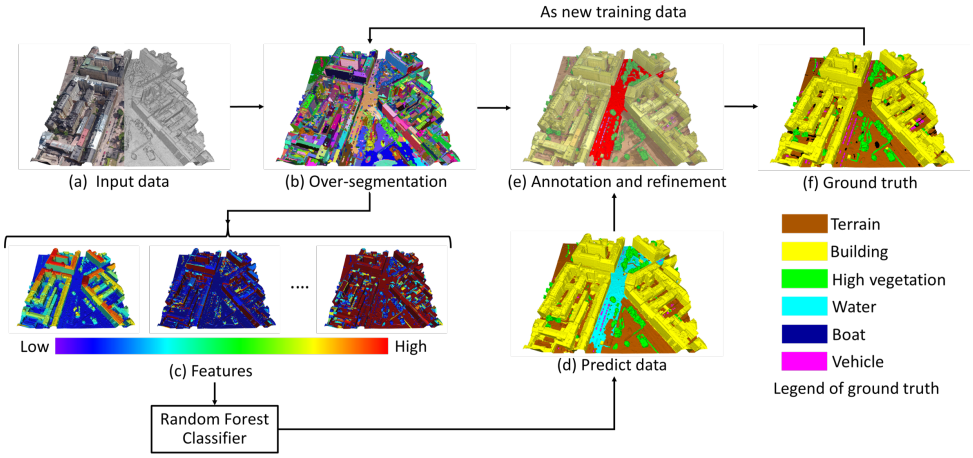


Figure 3.5: The pipeline of the labeling workflow

where f_i represents each handcrafted feature described in 'Initial segmentation', and $\bar{f}$ is the mean value of a N_f dimensional feature vector. To acquire the first ground truth data, we manually annotate the mesh (with segments) that is selected with the highest feature diversity. Then, we add the first labeled mesh into the training dataset for the supervised classification. Specifically, we use the segment-based features as input for the classifier, and the output is a pre-labeled mesh dataset. Next, we use the mesh annotation tool to manually refine the pre-labeled mesh according to the feature diversity. Finally, the new refined mesh will be added to the training dataset to improve the automatic classification accuracy incrementally.

Initial segmentation. To avoid redundant computations of numerous triangles, we first apply mesh over-segmentation (i.e., linear least-squares fitting of planes) based on region growing on the input data to group triangle faces into homogeneous regions [Lafarge and Mallet, 2012]. Such grouped regions are beneficial for computing local contextual features. We then extract both geometric and radiometric features from those mesh segments as follows:

- *Eigen-based features* are computed from the covariance matrix of the triangle vertices with respect to the average centre within each segment, which is beneficial for identifying urban objects with various surface distributions. The linearity $= (\lambda_1 - \lambda_2)/\lambda_1$, sphericity $= \lambda_3/\lambda_1$ and change of curvature $= \lambda_3/(\lambda_1 + \lambda_2 + \lambda_3)$ are computed based on the three eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$. The local eigenvectors $\mathbf{n}_i$ and the unit normal vector $\mathbf{n}_z$ along Z-axis are used to compute the verticality $= 1 - |\mathbf{n}_i \cdot \mathbf{n}_z|$ [Hackel *et al.*, 2016]. Note that many eigen-based features have been studied in literature [Hackel *et al.*, 2016; Weinmann *et al.*, 2013; West *et al.*, 2004], and some of them were designed for and tested on LiDAR point clouds. These eigen-based features are mostly computed per point based

on its spherical neighborhood, which often contains noise and does not form a surface. Our chosen eigen-based features are defined on a segment representing the surface of a mesh, and thus they can capture non-local geometric properties of an object. Additionally, in this work, we have tested all eigen-based features from the literature [Hackel *et al.*, 2016], and we only present the ones that are effective for textured meshes.

- *Elevation* is divided into absolute elevation z_a, relative elevation z_r and multiscale elevations z_m. Where z_a is the average elevation of the segment; the relative elevation is computed as $z_r = z_a - z_{r_{min}}$; the multiscale elevation [Rouhani *et al.*, 2017; Verdie *et al.*, 2015] is defined as $z_m = \sqrt{\frac{z_a - z_{min}}{z_{max} - z_{min}}}$. And $z_{r_{min}}$ denotes the lowest elevation of the local largest ground segment computed within a cylindrical neighborhood with 30 meters radius around the segment center. z_{min} and z_{max} represent the local minimum and maximum elevation values of a cylindrical neighborhood within the scale of 10 meters, 20 meters, and 40 meters. Such large cylindrical neighborhoods allow to find the local ground considering the resilience to hilly environments, and the square root ensures that small relative height values (i.e., values smaller than 1 m) get a larger elevation attribute to enlarge elevation differences between small objects and the local ground (e.g., cars against the ground, boats against the water surfaces). More importantly, due to the influence of terrain fluctuations and various scales of urban objects, the elevation of these three categories can complement each other.

- *Segment area* is computed as $area(S_k) = \sum_{i=1}^{N} area(f_i)$, where f_i denotes a triangle of the segment S_k, and N denotes the total number of triangles in S_k.

- *Triangle density* is defined as $density(S_k) = \frac{N}{area(S_k)}$, which reveals the object complexity, especially for adaptive urban meshes.

- *Interior radius of 3D medial axis transform (InMAT)* [Ma *et al.*, 2012; Peters and Ledoux, 2016] of a segment S_k is formulated as $r_k = \frac{\sum_{i=1}^{M} r_i}{M}$, where M denotes the total number of triangle vertices of S_k, and r_i denotes the interior radius of the shrinking ball that touches the vertex v_i within the segment S_k. It is designed to distinguish objects with different scales.

- *HSV color-based features* are derived from the RGB channel of the entire texture map. We use the HSV color space since it can better differentiate different objects than RGB. We compute the average color, the variance of the color distribution of all pixels within each segment, and we further discretize it into a histogram that consists of 15 bins of the hue channel, five bins of the saturation channel, and five bins of the value channel.

- *Greenness* a_g is used to classify objects that are similar to green vegetation [McKinnon and Hoff, 2017]. Specifically, it is computed according to the averaged RGB color of each segment via $a_g = G - 0.39 \cdot R - 0.61 \cdot B$.

All the above features are concatenated into a 44-dimensional feature vector used by our random forest (RF) classifier in the initial segmentation.

Annotation tool for refinement. Because of the under-segmentation errors and the imperfect results of the semantic mesh segmentation process, we design a mesh annotation tool (see Figure 3.6) to manually correct the labeling errors. Our mesh annotation tool is developed based on the labeling tool of CGAL [The CGAL Project, 2024].

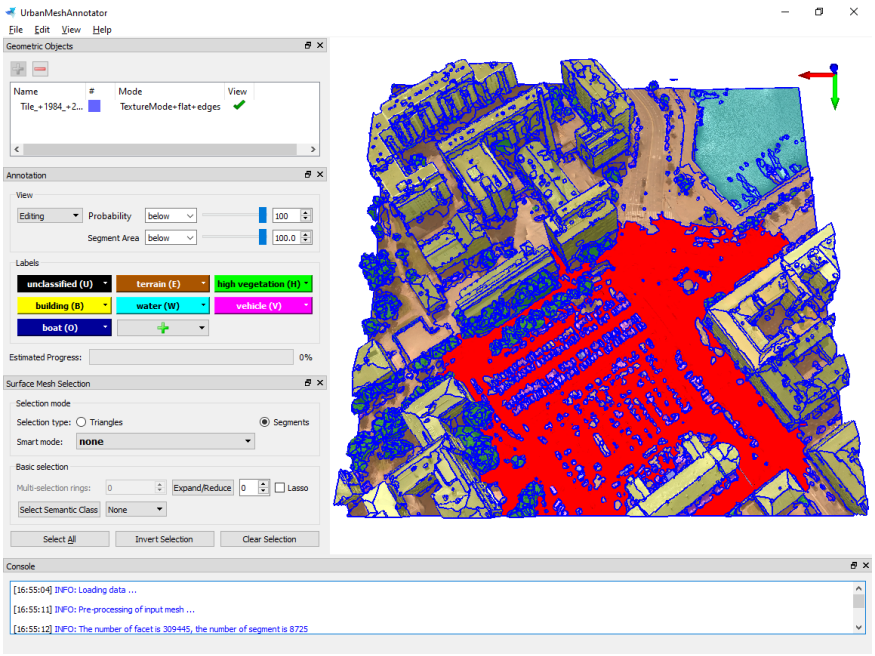


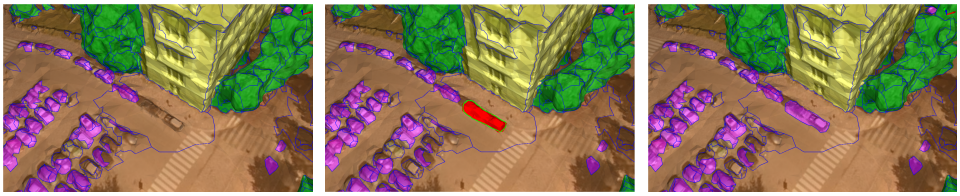
Figure 3.6: The interface of our annotation tool for 3D textured meshes

As shown in Table 3.2, it consists of three operation categories: view, selection, and annotation. The view operations provide essential functions for the user to manipulate the scene camera, such as translating, rotating, zooming, or setting the new pivot for the scene. In addition, to use textures as a reference for labeling, we map texture and face color with a certain degree of transparency, and we visualize the segment border to differentiate each segment.

The selection operations allow the user to select or deselect either triangle faces (see Figure 3.7) or segments (see Figure 3.8) freely via a brush or a lasso. Specifically, the face selection operation is used to fix the under-segmentation errors and generate new segments, and the segment selection operation is to fix incorrect segment labels.

Categories	Operations	Objects
View	Translate	Camera
	Rotate	Camera
	Zoom in / out	Camera
	Set pivot	Camera
Selection	Multi-selection / Lasso	Triangles / Segments
	Expand / Reduce	Triangles / Segments
	Semantic selection	Segments
	Split region	Segments
	Planar region extraction	Triangles
	Split mesh	Triangles
Annotation	Probability slider	Segments
	Segment area slider	Segments
	Progress bar	Triangles
	Switch semantic view	Triangles
	Labelling	Triangles / Segments

Table 3.2: Basic operations in our annotation tool

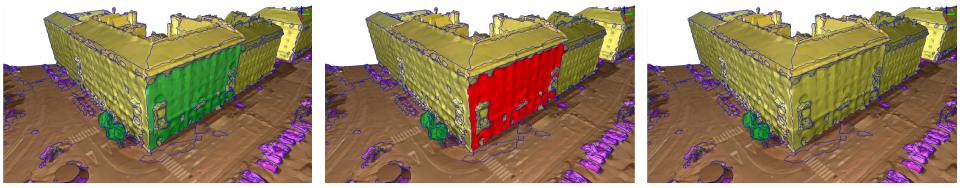


(a) (b) (c)

Figure 3.7: An example of labelling by selecting triangles using the lasso tool (blue edges: segment boundaries): (a) shows the input; (b) depicts Lasso selection result (in red); (c) demonstrates the correct label has been assigned to the selected region. In this example, the label of the selected region has been changed from 'ground' to 'vehicle'.

We also allow the user to edit the selection of each individual segment with splitting functions (see Figure 3.9) and automatic extraction of the most planar region (see Figure 3.10). As for splitting, we first detect the potential planar and non-planar segments marked by user strokes, and then the non-planar one is split according to the vertex-to-plane distance. It allows the generating candidate non-planar regions (with respect to the detected planar segment) for the user to edit, and it is useful to split a segment that covers large non-planar regions or contains more than one dominant planar area.

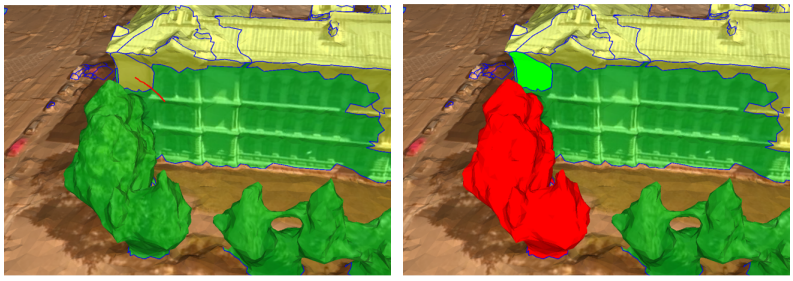
To extract the most planar region, we apply the region growing algorithm [Lafarge and Mallet, 2012] within the selected segment to automatically generate the candidate



(a) (b) (c)

Figure 3.8: An example of segment labeling: (a) shows part of a wall of the building was previously labeled as 'high vegetation' (in green); (b) depicts segment selection results (in red); (c) demonstrates the label of the selected segment has been corrected with the new label 'building'.

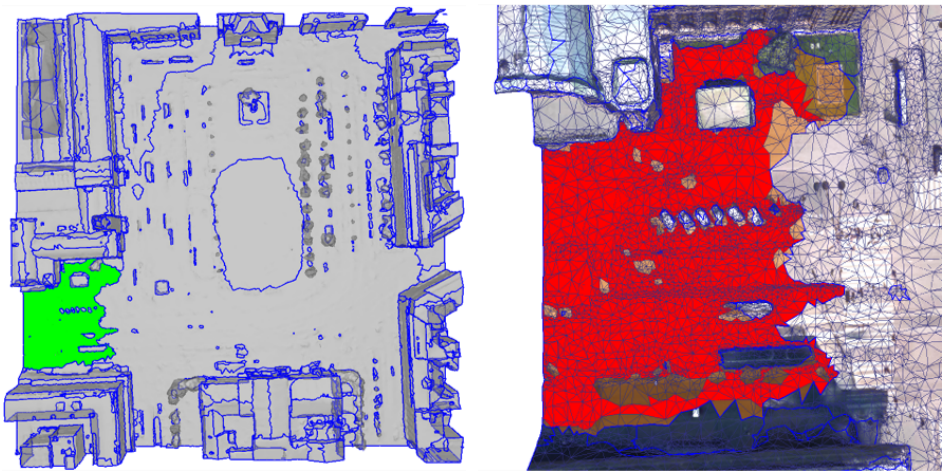
triangle faces with user-defined thresholds (i.e., the maximum distance to the plane, the maximum accepted angle, and the minimum region size). Such an operation allows the user to filter out some small bumpy regions of the selected segment.



(a) (b)

Figure 3.9: An example splitting planar and non-planar regions: (a) illustrates the user draws a stroke (in red) across the border of the non-planar segment and the planar segment; (b) shows the detected non-planar segment has been split into two parts (i.e., a non-planar region shown in red and a planar segment shown in green).

Besides, probability and area-based sliders and a progress bar are provided in the annotation panel to improve annotation efficiency and experience, respectively. Specifically, the probability slider is introduced for the user to visually inspect the segments that are most likely misclassified. Moreover, the user can further use it to inspect a specific class by switching the view to highlight a specific semantic class. The segment area slider is used to identify isolated tiny segments, which commonly appear as errors. The progress bar is used to indicate the estimated labeling progress during the annotation. After performing the selection, the user can easily assign the corresponding label to the selected area.



(a) (b)

Figure 3.10: Editing an individual segment: (a) shows a segment is selected (highlighted in green) for splitting; (b) demonstrates automatic extraction of the most planar region (shown in red) within the selected segment according to user-defined thresholds.

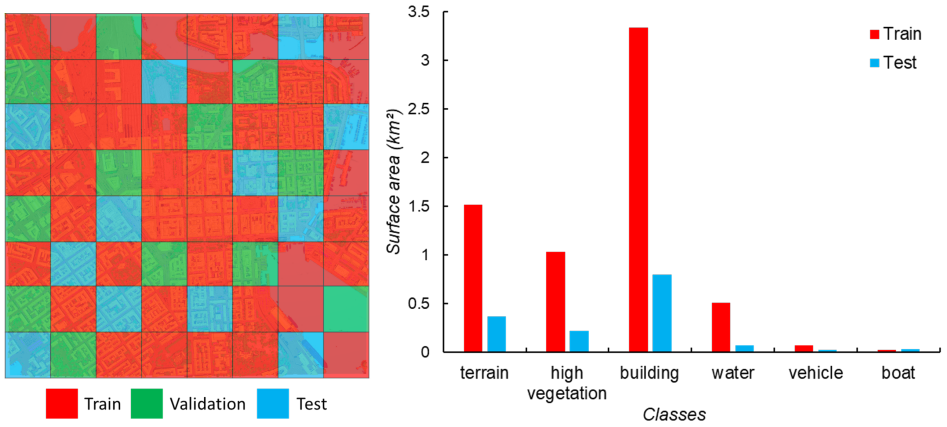
3.4. Experiments

3.4.1. Data split

To perform the semantic segmentation task, we randomly select 40 tiles from the annotated 64 tiles of Helsinki as training data, 12 tiles as test data, and 12 tiles as validation data (see Figure 3.11 (a)). For each of the six semantic categories, we compute the total area in the training and test dataset to show the class distribution. As shown in Figure 3.11 (b), some classes, like vehicles and boats, only account for less than 5% of the total area, while the building and terrain together comprise more than 70%. The unbalanced classes impose significant challenges for semantic segmentation based on supervised learning.

3.4.2. Evaluation metrics

Since the triangle faces in the meshes have different sizes, we compute the surface area for semantic evaluation instead of using the number of triangles. The performance of semantic mesh segmentation is measured in precision, recall, F1 score, and intersection over union (IoU) for each object class. The evaluation of the whole test area is applied with overall accuracy (OA), mean per-class accuracy (mAcc), and mean per-class intersection over union (mIoU).



(a) (b)

Figure 3.11: Overview of the data used in our experiment: (a) shows the distribution of the training, test, and validation dataset; (b) shows semantic categories of training (including validation dataset) and test dataset.

3.4.3. Evaluation of initial segmentation

We have implemented the semantic mesh segmentation and annotation tool in C++ using the open-source libraries, including CGAL [The CGAL Project, 2024], Easy3D [Nan, 2021a], and ETHZ random forest [Walk, 2014].

Our proposed pipeline for initial segmentation only takes a few input parameters, which are shown in Table 3.3. The over-segmentation is intended to find all planar regions in the model, for which we set the distance threshold to 0.5 meters. This threshold value specifies the minimum geometric features we would like the over-segmentation method to identify. In other words, the region-growing-based over-segmentation method will not be able to distinguish two parallel planes with a distance smaller than this threshold. We set the angle threshold to 90 degrees, which is large enough to cope with high levels of noise (e.g., the distance value is small, but the angle between the triangle normal and the plane normal is large). Moreover, the minimum area is set to zero to allow planar segments of any arbitrary size. As for the random forest classifier, we set the parameters initially to those of Rouhani et al. [Rouhani *et al.*, 2017] followed by fine-tuning using the validation data. Specifically, using 100 trees is sufficient to guarantee the stability of the model, and using a depth of 30 is adequate to avoid over-fitting and under-fitting for training.

Rather than classifying about 19 million triangle faces (i.e., the entire dataset), we use 515,176 segments that are clustered during over-segmentation. Although both semantic segmentation and labeling refinement can benefit from mesh over-segmentation, the degree of the under-segmentation error cannot be avoided. Since our mesh over-

Method	Parameters	Value
Region Growing	Minimum area	0 m^2
	Distance to plane	0.5 m
	Accepted angle	90°
Random Forest	Number of trees	100
	Maximum depth	30

Table 3.3: Parameters used in our approach

segmentation does not intend to retrieve the individual objects and the purpose is to perform semantic segmentation, we measure the maximum achievable performance by calculating the upper bound IoU instead of using under-segmentation errors to evaluate it. In other words, the upper bound IoU indicates the perfect classification results of each segment. The upper bound IoU of each class we could achieve for semantic segmentation is presented in Table 3.4, and the upper bound mean IoU (mIoU) over all classes is about 90.9% as shown in Table 3.5. In addition, the results of our experiment in Table 3.4 and Table 3.5 are reported based on the average performance of ten times experiments with the same configuration.

Class	Precision (%)	Recall (%)	F1 scores (%)	IoU (%)	Upper bound IoU (%)
Terrain	87.7	94.3	90.9	83.3	93.9
High Vegetation	96.3	93.8	95.0	90.5	96.2
Building	94.6	97.7	96.1	92.5	99.0
Water	97.0	88.3	92.5	86.0	92.7
Vehicle	77.9	41.7	54.4	37.3	73.2
Boat	77.9	7.5	13.7	7.4	90.5

Table 3.4: Overall evaluation of our method. The *Upper bound IoU* refers to the maximum achievable IoU in theory.

For semantic segmentation, a detailed evaluation of each class is listed in Table 3.4, and we achieve about 93.0% overall accuracy and 66.2% mIoU as shown in Table 3.5. The qualitative evaluation of it is shown in Figure 3.12. As shown in Figure 3.12 (e), most of the prediction errors occur at small-scale objects such as vehicles and boats due to fewer training samples and errors from over-segmentation.

To better understand the relevance of the features, we measure the feature importance and perform ablation studies (see Table 3.5). We can observe that the radiometric features (which account for 62.8%) are more important than geometric ones (which account for 37.2%). Moreover, after removing individual feature vectors, the performance will decline, indicating each feature contributes to the best results.

Model	OA (%)	mAcc (%)	mIoU (%)	ΔmIoU (%)
Upper bound (Perfect)	98.1	91.6	90.9	
Ours (best)	93.0	70.6	66.2	0.0
Without sphericity	93.0	70.5	66.1	-0.1
Without segment area	92.9	70.5	66.0	-0.2
Without triangle density	92.9	70.4	66.0	-0.3
Without variance HSV	92.9	70.3	65.9	-0.3
Without absolute elevation	93.0	70.2	65.9	-0.3
Without relative elevation	92.9	70.3	65.8	-0.4
Without curvature	92.9	70.2	65.8	-0.4
Without multiscale elevations	92.8	69.8	65.1	-1.1
Without linearity	91.8	66.6	62.0	-4.2
Without greenness	91.9	66.6	61.9	-4.3
Without InMat	91.6	66.4	61.6	-4.6
Without average HSV	91.7	66.1	61.4	-4.8
Without verticality	91.4	66.1	61.3	-4.9
Without HSV histogram bins	91.5	66.0	61.1	-5.1

Table 3.5: Ablation study of the features in our approach. The *Upper bound (Perfect)* refers to the maximum achievable performance in theory.

3.4.4. Evaluation of competition methods

To the best of our knowledge, none of the state-of-the-art deep learning frameworks of 3D semantic segmentation can directly be used on large-scale textured meshes. Additionally, although the data structures of point clouds and meshes are different, the inherent properties of geometry in the 3D space of the urban environment are nearly identical. In other words, they can share the feature vectors within the same scenes. Consequently, we sample the mesh into colored point clouds (see Figure 3.13) with a density of about 10 pts/m^2 as input for the competing deep learning methods. In particular, we use Montecarlo sampling [Cignoni *et al.*, 1998] to generate randomly uniform dense samples, and we further prune these samples according to Poisson distributions [Corsini *et al.*, 2012] and assign the color via searching the nearest neighbor from the textures.

To evaluate and compare with the current state-of-the-art 3D deep learning methods that can be applied to a large-scale urban dataset, we select five representative approaches (i.e., PointNet [Charles *et al.*, 2017], PointNet++ [Qi *et al.*, 2017], SPG [Landrieu and Simonovsky, 2018], KPConv [Thomas *et al.*, 2019], and RandLA-Net [Q. Hu *et al.*, 2020]). We perform all the experiments on an NVIDIA GEFORCE GTX 1080Ti GPU. Note that these deep learning-based methods downsample the input point clouds significantly as a pre-processing step. In our experiments, the point sampling density is limited by the GPU memory, and increasing or decreasing the sampling density within a reasonable range may lead to slightly different performance.

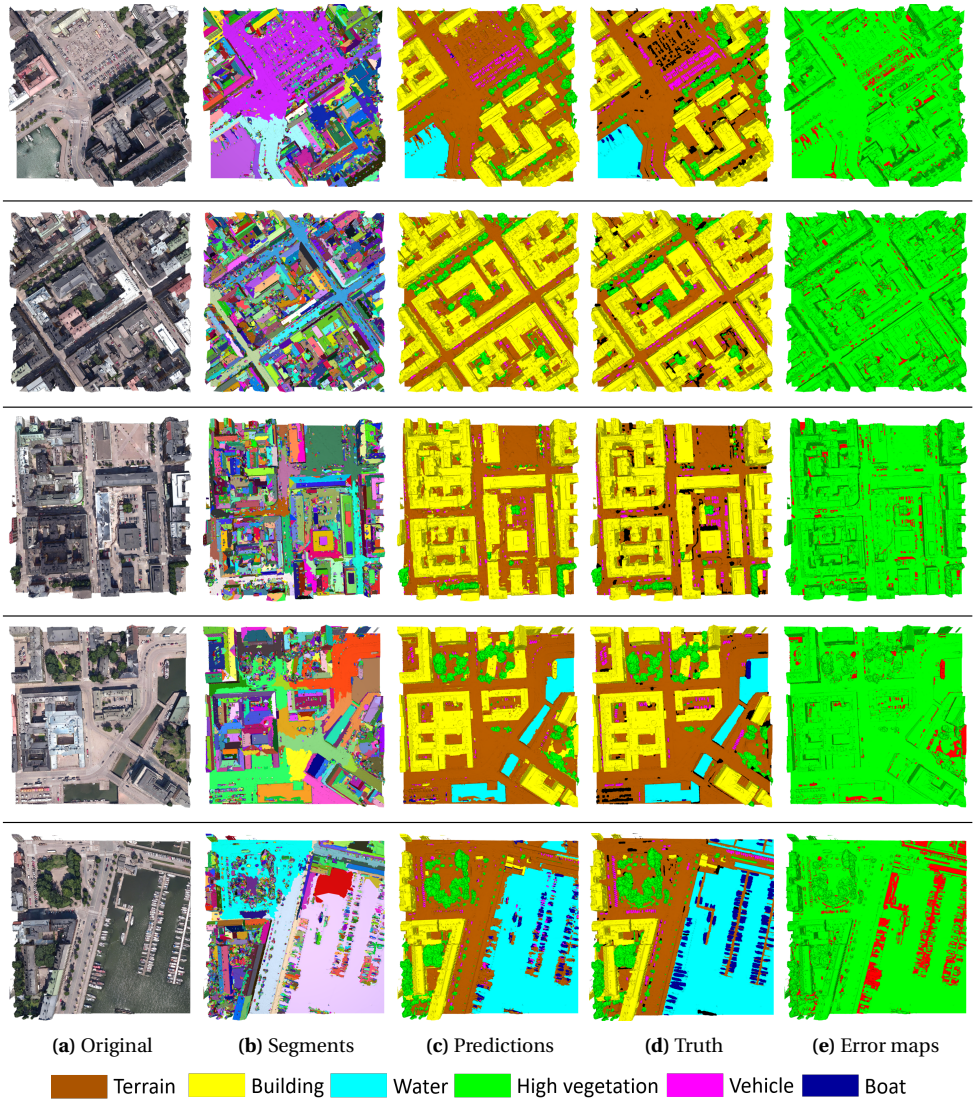
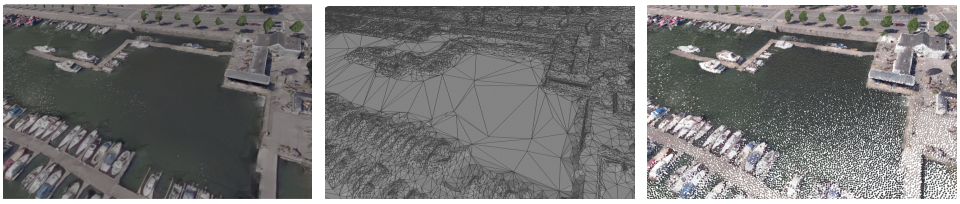


Figure 3.12: Part of our semantic segmentation results: the first column shows the input meshes; the second column shows the over-segmentation results; the third column shows the predicted semantic meshes; the fourth column shows the ground truth meshes; and the last column shows the error maps (red: errors; green: correct labels).

It should be noted that no matter how dense the input point clouds are, almost all state-of-the-art deep learning architectures (such as PointNet, PointNet++, RandLaNet, KPConv, and SPG, etc.) downsample the input point clouds significantly, and they are



(a) Textured mesh (b) Wireframe (c) Sampled point cloud

Figure 3.13: Sampling point cloud from textured meshes: our sampled points preserve both geometric and radiometric information of the original mesh.

still able to learn effective features for classification. Besides, different deep learning-based point cloud classification frameworks exploit different strategies for downsampling the input points. In addition, we also compare with the joint RF-MRF [Rouhani *et al.*, 2017], which is the only competition method that directly takes the mesh as input without using GPU for computation.

The hyper-parameters of all the competing methods are tuned according to the validation data to achieve the best results we can acquire. Besides, the results of each competitive method (see Table 3.6) are demonstrated in average performance based on ten times experiments with the same setting. From the comparison results, as shown in Table 3.6, we found that our baseline method outperforms other methods except for KPConv. Specifically, our approach outperforms RF-MRF with a margin of 5.3% mIoU, and deep learning methods (not including KPConv) from 16.7% to 29.3% mIoU. Compared with the KPConv, the performance of our method is much more robust, which can be observed from Table 3.6 that the standard deviation of our method is close to zero (i.e., the standard deviation of mIoU of our method is about 0.024%). The reason is that in our method, we set 100 trees in the random forest to ensure the stability of the model, but in KPConv, the kernel point initialization strategy may not be able to select some parts of the point cloud, which leads to the instability of the results. Furthermore, compared with all deep learning pipelines, our method is conducted on a CPU and uses much less time for training (including feature computation). This can be explained by the fact that we have fewer input data (triangles versus points), and the time complexity of our handcrafted features computation is much lower than the features learned from deep learning.

3.4.5. Evaluation of annotation refinement

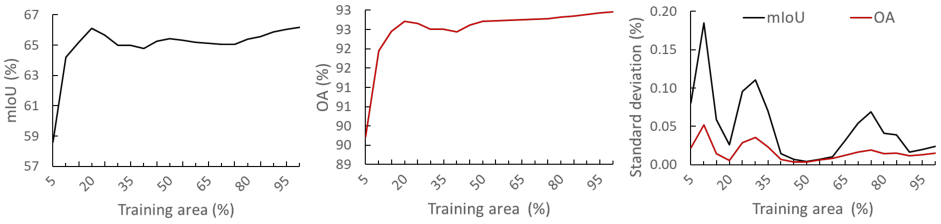
Following the proposed framework, a total of 19,080,325 triangle faces have been labeled, which took around 400 working hours. Compared with a triangle-based manual approach, we estimate that our framework saved us more than 600 hours of manual labor. Specifically, we have measured the labeling speed with these two different approaches on the same mesh tile consisting of 309,445 triangle faces and 8,033 segments. It took around 17 hours for manual labeling based on triangle faces, while

	Terrain	High Vegetation	Building	Water	Vehicle	Boat	mIoU	OA	mAcc	mF1	t_{train}
PointNet [Charles *et al.*, 2017]	56.3	14.9	66.7	83.8	0.0	0.0	36.9 ś 2.3	71.4 ś 2.1	46.1 ś 2.6	44.6 ś 3.2	1.8
RandLaNet [Q. Hu *et al.*, 2020]	38.9	59.6	81.5	27.7	22.0	2.1	38.6 ś 4.6	74.9 ś 3.2	53.3 ś 5.1	49.9 ś 4.8	10.8
SPG [Landrieu and Simonovsky, 2018]	56.4	61.8	87.4	36.5	34.4	6.2	47.1 ś 2.4	79.0 ś 2.8	64.8 ś 1.2	59.6 ś 1.9	17.8
PointNet++ [Qi *et al.*, 2017]	68.0	73.1	84.2	69.9	0.5	1.6	49.5 ś 2.1	85.5 ś 0.9	57.8 ś 1.8	57.1 ś 1.7	2.8
RF-MRF [Rouhani *et al.*, 2017]	77.4	87.5	91.3	83.7	23.8	1.7	60.9 ś 0.0	91.2 ś 0.0	65.9 ś 0.0	68.1 ś 0.0	**1.1**
KPConv [Thomas *et al.*, 2019]	**86.5**	88.4	**92.7**	77.7	**54.3**	**13.3**	**68.8** ś 5.7	**93.3** ś 1.5	**73.7** ś 5.4	**76.7** ś 5.8	23.5
Baseline	83.3	**90.5**	92.5	**86.0**	37.3	7.4	66.2 ś 0.0	93.0 ś 0.0	70.6 ś 0.0	73.8 ś 0.0	1.2

Table 3.6: Comparison of various semantic segmentation methods on the new benchmark dataset: the results reported in this table are per-class IoU (%), mean IoU (mIoU, %) ś standard deviation, Overall Accuracy (OA, %) ś standard deviation, mean class Accuracy (mAcc, %) ś standard deviation, mean F1 score (mF1, %) ś standard deviation, and the time cost of training (t_{train}, hours). The running times of SPG include both feature computation and graph construction, and RF-MRF and our baseline method include feature computation. We repeated the same experiment ten times and presented the mean performance.

with our segment-based semi-automatic approach, it took only 6.5 hours.

We also evaluate the performance of semantic segmentation with different amounts of input training data on our baseline approach with the intention of understanding the required amount of data to obtain decent results. Specifically, we use ten sets of different training areas with ten times experiments with the same configuration of each set, and we linearly interpolate the results as shown in Figure 3.14. From Figure 3.14a, Figure 3.14b, and Figure 3.14c, we can observe that our initial segmentation method only requires about 10% (equal to about 0.325 km^2) of the total training area to achieve acceptable and stable results. In other words, using a small amount of ground truth data, our framework can provide robust pre-labeled results and significantly reduce manual labeling efforts.



(a) mIoU (b) OA (c) Standard deviation

Figure 3.14: Effect of the amount of training data on the performance of the initial segmentation method used in the semi-automatic annotation: we repeated the same experiment ten times for each set of training areas and presented the mean performance.

3.5. Conclusion

We have developed a semi-automatic mesh annotation framework to generate a large-scale semantic urban mesh benchmark dataset covering about 4 km^2. In particular, we first used a set of handcrafted features and a random forest classifier to generate the pre-labeled dataset, which saved us around 600 hours of manual labor. Then, we developed a mesh labeling tool that allows the users to interactively refine the labels at both the triangle face and the segment levels. We have further evaluated the current state-of-the-art semantic segmentation methods that can be applied to large-scale urban meshes, and as a result, we have found that our classification based on handcrafted features achieves 93.0% overall accuracy and 66.2% of mIoU. This outperforms the state-of-the-art machine learning and most deep learning-based methods that use point clouds as input. Despite this, there is still room for improvement, especially on the issues of imbalanced classes and object scalability. For future work, we plan to label more urban meshes of different cities and extend our Helsinki dataset to include parts of urban objects (such as roofs, chimneys, dormers, and facades). We will also investigate smart annotation operators (such as automatic boundary refinement and structure extraction),

which involve more user interactivity and may help reduce further the manual labeling task.

4

Planarity-sensible semantic mesh segmentation*

Building upon the dataset and tools developed in Chapter 3, this chapter delves deeper into the semantic segmentation process to enhance the accuracy and efficiency of interpreting 3D urban environments. Specifically, we introduce a novel deep learning-based framework to interpret 3D urban scenes represented as textured meshes. Based on the observation that object boundaries typically align with the boundaries of planar regions, our framework achieves semantic segmentation in two steps: planarity-sensible over-segmentation followed by semantic classification. The over-segmentation step generates an initial set of mesh segments that capture the planar and non-planar regions of urban scenes. In the subsequent classification step, we construct a graph that encodes the geometric and photometric features of the segments in its nodes and the multi-scale contextual features in its edges. The final semantic segmentation is obtained by classifying the segments using a graph convolutional network. Experiments and comparisons on two semantic urban mesh benchmarks demonstrate that our approach outperforms the state-of-the-art methods in terms of boundary quality, mean IoU (intersection over union), and generalization ability. We also introduce several new metrics for evaluating mesh over-segmentation methods dedicated to semantic segmentation, and our proposed over-segmentation approach outperforms state-of-the-art methods on all metrics.

*This chapter is based on the paper: Gao, W., Nan, L., Boom, B. and Ledoux, H. (2023). 'PSSNet: Planarity-sensible Semantic Segmentation of large-scale urban meshes'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 196, pp. 32–44. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2022.12.020.

4.1. Introduction

Recent advances in photogrammetry and 3D computer vision have enabled the generation of textured meshes of large-scale urban scenes that contain buildings, trees, vehicles, etc. [W. Gao *et al.*, 2021; Google, 2012; Helsinki, 2019]. Deriving semantic information from the mesh models is critical to allowing the use of these meshes in diverse applications, e.g., energy estimate, noise modelling, and solar potential [Besuievsky *et al.*, 2018; Biljecki *et al.*, 2015b; Saran *et al.*, 2015].

There exists a large volume of machine learning-based algorithms for the semantic segmentation of 3D data, and they are designed mainly for 3D point clouds [Charles *et al.*, 2017; Demantké *et al.*, 2011; Hackel *et al.*, 2016; Qi *et al.*, 2017; Thomas *et al.*, 2018; Thomas *et al.*, 2019]. A few recent works also address deep learning for surface meshes [Fu *et al.*, 2021; L. Gao *et al.*, 2019; Hanocka *et al.*, 2019; Selvaraju *et al.*, 2021] but are limited to individual objects or small indoor scenes (e.g., living room, kitchen). Unlike point clouds that are usually obtained as the raw input from typical data acquisition devices, textured meshes (see Figure 4.2a) provide topological information, have continuous surfaces, yield better visualization, and are lightweight, which makes them an ideal representation for urban scenes. Surprisingly, the semantic segmentation of urban meshes has rarely been investigated, [W. Gao *et al.*, 2021; Rouhani *et al.*, 2017; Verdie *et al.*, 2015] are exceptions.

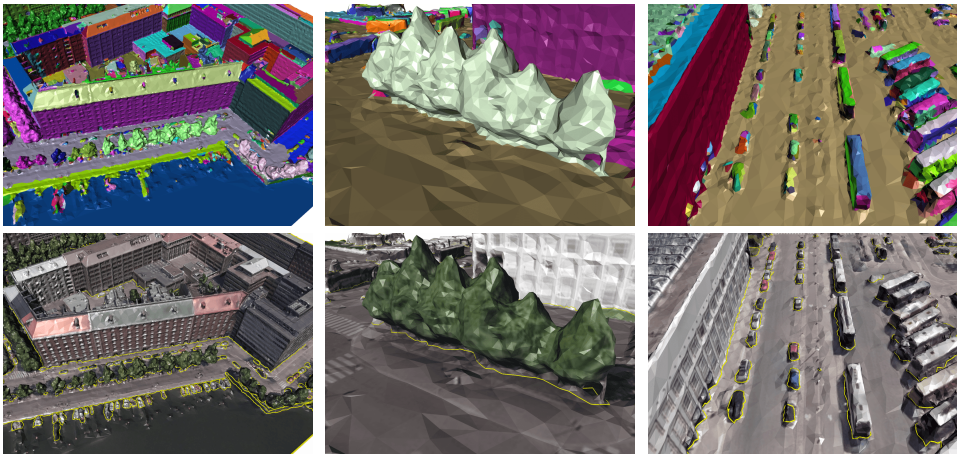


Figure 4.1: Object boundaries often align with the boundaries of planar regions: the top figures show planar and non-planar segmentation results; the bottom figures depict the corresponding ground truth object boundaries (shown as yellow lines).

In this work, we address the semantic segmentation of urban meshes by introducing a two-step framework using deep learning. Our framework is designed to improve the following three aspects of semantic segmentation:

1) Segmentation quality. Urban scenes typically contain piecewise regions, which can

already inspire the separation of man-made objects (e.g., roads, buildings) from organic objects (e.g., trees). We observe that semantic segmentation algorithms usually perform well in the interior of large smooth surfaces (including planar surfaces), but that they perform poorly for the identification of object boundaries. Given the fact that object boundaries typically align with the boundaries of planar regions (see Figure 4.1), our framework achieves semantic segmentation by first exploiting a planarity-sensible over-segmentation step that separates *planar* and *non-planar* surface patches.

2) Descriptiveness of geometric features. Existing methods for semantic segmentation of 3D data commonly rely on features defined on local primitives (i.e., points or triangles) [Charles *et al.*, 2017; J. Huang *et al.*, 2019; S. Li *et al.*, 2019; Schult *et al.*, 2020; Weinmann *et al.*, 2015; Weinmann *et al.*, 2013] or segments (i.e., a group of points or triangles) [Cohen-Steiner *et al.*, 2004; Lafarge and Mallet, 2012; Landrieu and Simonovsky, 2018; Y. Lin *et al.*, 2018; Rouhani *et al.*, 2017; Verdie *et al.*, 2015]. Features from local primitives are limited to a certain distance in the local neighborhood, while features used in existing segment-based approaches do not effectively capture the contextual relationships between segments. Thus, they are less descriptive in representing the complex shapes of diverse objects and in revealing the relationships between objects. In our work, by initially decomposing a mesh model into *planar* and *non-planar* segments, both local geometric features of individual segments and global relationships between segments can be captured.

3) Efficiency. Existing deep learning-based methods for processing 3D data are limited by the data size, especially for large-scale urban scenes. This has already motivated over-segmentation for semantic segmentation [Hui *et al.*, 2021; Landrieu and Boussaha, 2019; Landrieu and Simonovsky, 2018; Weinmann *et al.*, 2015]. Following the spirit of the previous work for improving efficiency, our over-segmentation facilitates better object boundaries and strengthens semantic segmentation by distinctive local and non-local features, which is suitable for the subsequent classification using graph convolutional networks (GCN).

Besides the two-step semantic segmentation framework, we also introduce several new metrics for evaluating mesh over-segmentation techniques. We believe the proposed metrics will further stimulate the improvement of over-segmentation for semantic segmentation. Experiments on two benchmarks show that our approach outperforms recently developed methods in terms of boundary quality, mean IoU (intersection over union), and generalization ability.

In summary, our contributions are 1) a novel mesh over-segmentation approach for extracting planarity-sensible segments that are dedicated to GCN-based semantic segmentation; 2) a new graph structure that encodes both local geometric and photometric features of segments, as well as global spatial relationships between segments; 3) several novel metrics for evaluating mesh over-segmentation techniques in the context of semantic segmentation.

4.2. Related work

While there is a large volume of research on the over-segmentation and semantic segmentation of urban images [Cordts *et al.*, 2016; M. Yang *et al.*, 2018], we focus in this sole section on methods designed to process large-scale 3D data, i.e., point clouds and meshes of urban scenes. Methods specially designed for handling individual objects or small scenes [Fu *et al.*, 2021; L. Gao *et al.*, 2019; Hanocka *et al.*, 2019; Nan *et al.*, 2012; Selvaraju *et al.*, 2021] usually do not scale to large-scale urban scenes and thus are not covered.

4.2.1. Over-segmentation of 3D data

Many methods for over-segmentation of 3D data are inspired by image over-segmentation algorithms [M.-Y. Liu *et al.*, 2011] and can be divided into four categories: (1) primitive-based fitting [Ben-Shabat *et al.*, 2018; Lafarge and Mallet, 2012; Schnabel *et al.*, 2007], (2) graph-based partitioning [Ben-Shabat *et al.*, 2018; Landrieu and Simonovsky, 2018], (3) local region expansion [Cohen-Steiner *et al.*, 2004; Lafarge and Mallet, 2012; Y. Lin *et al.*, 2018; Melzer, 2007; Papon *et al.*, 2013; Rouhani *et al.*, 2017; Vosselman *et al.*, 2017], and (4) learning-based methods [Hui *et al.*, 2021; Landrieu and Boussaha, 2019]. Over-segmentation often serves as pre-processing for tasks such as semantic segmentation, instance segmentation, or reconstruction and aims at reducing the complexity of subsequent tasks by using fewer segments having local homogeneity. Due to the complexity of real-world scenes and the irregularity of the data, it is challenging to obtain over-segmentation results with a desired number of segments and clear object boundaries. The aforementioned methods are either limited by the primitive types (e.g., plane, sphere, and cylinder) or suffer from severe under-segmentation errors when the number of segments is reduced or by the type of available labels in the training data (e.g., a few methods require instance labels [Hui *et al.*, 2021; Landrieu and Boussaha, 2019]). We propose to partition the input meshes into a relatively small number of homogeneous regions with clear object boundaries based on both geometric and photometric characteristics (see Section 4.3), which is beneficial to semantic segmentation (see Section 4.4).

4.2.2. Semantic segmentation of 3D data

An important step in semantic segmentation is feature extraction. Based on the methods used for feature extraction, semantic segmentation approaches can be roughly categorized into three groups: handcrafted-feature-based [Demantké *et al.*, 2011; Hackel *et al.*, 2016; Rouhani *et al.*, 2017; Thomas *et al.*, 2018; Verdie *et al.*, 2015; Vosselman *et al.*, 2017; Weinmann *et al.*, 2015; Weinmann *et al.*, 2013], learning-based [Charles *et al.*, 2017; Q. Hu *et al.*, 2020; Lei *et al.*, 2021; Qi *et al.*, 2017; Thomas *et al.*, 2019], and hybrid methods [Landrieu and Simonovsky, 2018; S.-C. Wu *et al.*, 2021]. Handcrafted features are often effective when limited training data is available. In contrast, deep-

learning techniques are more effective when sufficient training data is available [Guo *et al.*, 2021]. These methods usually require contextual information to compute or learn features. However, it is difficult to capture effective global contextual features. Inspired by SPG [Landrieu and Simonovsky, 2018], our graph structure encodes various local geometric, photometric, and contextual features, and we apply a GCN for semantic segmentation. Our method exploits enriched spatial relationships in the graph at both local and global scales, which greatly facilitates the GCN model in capturing contextual information and learning distinctive features for semantic segmentation.

4.3. Methodology

Our framework for semantic segmentation of urban meshes has two steps (as shown in Figure 4.2):

Planarity-sensible over-segmentation. This step decomposes the urban mesh into a set of *planar* and *non-planar* surface patches because object boundaries often align with the boundaries of planar regions. This step not only enhances the descriptiveness of the features learned through local context but also significantly reduces the number of segments to be classified.

Segmentation classification. We construct a graph with its nodes encoding the local geometric and photometric features of the segments and its edges encoding global contextual features. We achieve semantic segmentation of the mesh by classifying the segments using a graph convolutional network.

(a) Input mesh (b) Over-segmented mesh (c) Semantic mesh

Figure 4.2: The workflow of our method: we first decompose the input mesh (a) into a set of *planar* and *non-planar* segments (b); then we classify the segments using graph convolutional networks to obtain the results of semantic segmentation (c). In (b), the segments are randomly colorized. In (c), the colors are: terrain, building, high vegetation, water, vehicle, boat.

4.3.1. Planarity-sensible over-segmentation

This step aims to decompose the urban mesh into a set of homogeneous segments in terms of geometric and photometric characteristics, see Figure 4.3. Compared with *planar* segments generated by classical region growing methods [Lafarge and Mallet, 2012], our segments can accommodate more complex surfaces (i.e., trees and vehicles).

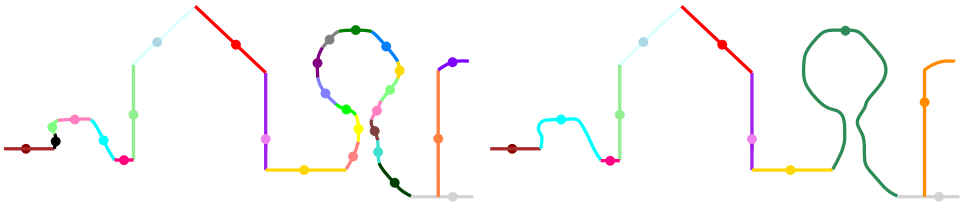
Our over-segmentation, further detailed below, is achieved in two steps: (1) *planar* and *non-planar* classification, and (2) incremental segmentation.

Planar and non-planar classification We classify the triangle faces of a mesh as either *planar* or *non-planar*. Following Gao et al. [W. Gao *et al.*, 2021], we design a set of features including Eigen-based (i.e., *linearity*, *planarity*, *sphericity*, *curvature*, and *verticality*), elevation-based (i.e., *absolute*, *relative*, and *multi-scale*), scale-based (i.e, *InMAT radius* [Ma *et al.*, 2012; Peters and Ledoux, 2016]: interior shrinking ball radius of 3D medial axis transformation), density-based (i.e., the number of vertices and the density of triangle faces), and color-based (i.e., *greenness* and *HSV histograms*) features, and we concatenate these features into a feature vector $\mathbf{F}_i$. We then use random forest (RF) [Geurts *et al.*, 2006] to learn the probability of a face being *non-planar* as

$$G_i(L) = \frac{1}{|\tau|} \sum_{t \in \tau} \log\big(P_t(l_i \mid \mathbf{F}_i)\big), \tag{4.1}$$

where τ is a set of decision trees, and the predicted probability from decision tree t is denoted by $P_t \in [0,1]$. $L = \{0,1\}$ represents the potential labels of a face i (i.e., $l_i = 0$ for *planar* and $l_i = 1$ for *non-planar*). We learn a probability map (instead of binary classification) for the subsequent segment aggregation.

Incremental segmentation Grouping all triangles into segments in one step using graph cuts would require the total number of segments, which is often not a priori. Therefore, we use the learned *planar* and *non-planar* probability maps to incrementally aggregate the mesh faces into a set of locally homogeneous segments. Inspired by Lafarge and Mallet [Lafarge and Mallet, 2012], we accumulate faces for a segment by solving a binary labelling problem. Starting from the face with the highest *planar* probability (i.e., the current region r has only a starting face at the beginning), we incrementally gather its neighboring face i to the current region r based on the labeling outcome of face i. The growing process is illustrated in Figure 4.4. Our idea is to grow a region r if its neighboring face i receives the same label. We exploit a Markov Random Field (MRF) formulation to select the most suitable face for the aggregation in each



(a) Planar segments (b) Planarity-sensible segments

Figure 4.3: 2D illustrative comparison between planar and planarity-sensible segments. Each dot and its line denote a segment.

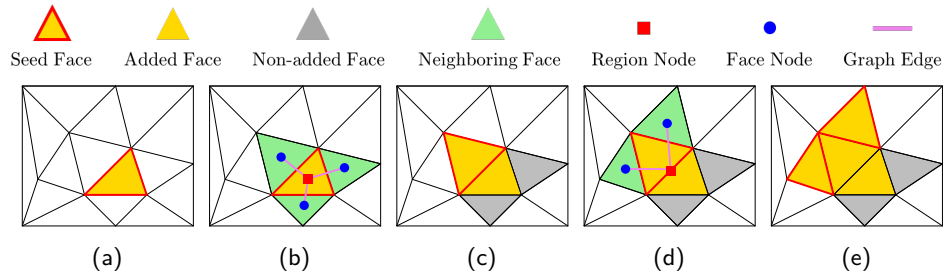


Figure 4.4: An illustration of the first few steps of incremental segmentation: (a) shows the seed face with the highest *planar* probability is shown in gold color; (b) depicts the local graph is constructed on the seed face (represented by the region node) and its three neighboring faces (represented by the face node); (c) illustrates the labeling outcome of the Markov random field (MRF): one newly added face is used as a seed face, and two non-added faces will be labeled as visited faces for the current growth step; (d) demonstrates a local graph is constructed based on the growing region (represented by the region node) and two neighboring faces (represented by the face node); (e) shows the new labeling outcome, where the newly added two faces will be used as seed faces for the next growing step.

growing iteration. The energy function $U(X)$ is defined as the sum of a unary term $\psi_i(x_i)$ and a pairwise term $\varphi_{i,r}(x_i, x_r)$, i.e.,

$$U(X) = \lambda_d \cdot \sum_{i \in A} \psi_i(x_i) + \lambda_m \cdot \sum_{i \in A} \varphi_{i,r}(x_i, x_r), \tag{4.2}$$

where A denotes the neighboring faces of the current growing region (i.e., the faces directly connected to r). x_i and x_r denote the binary labels that will be received by face i and region r, respectively. A neighboring face can be added to the current region only if it receives the same label as the current region. In our implementation, we fix the label of the current region to 0 (i.e., $x_r \equiv 0$) before minimizing the energy function. The face i is added to r only when $x_i = 0$ after the optimization. $\lambda_d \geq 0$ and $\lambda_m \geq 0$ are the weights balancing the unary and pairwise terms. A larger λ_d can lead to an excessive number of segments with smaller under-segmentation errors (see Figure 4.5a and Figure 4.5b). In contrast, a larger λ_m can result in fewer segments but may introduce larger under-segmentation errors (see Figure 4.5c).

The **unary term** $\psi_i(x_i)$ measures the penalty of assigning a label x_i to a face i. To define this term, we consider the geometric distance (for *planar* regions) and the probability map (for *non-planar* regions), which is formulated as

$$\psi(x_i) = \begin{cases} \min\{\mathrm{d}(f_i, p_r), C_i\}, & \text{if } x_i = 0 \\ 1 - \min\{\mathrm{d}(f_i, p_r), C_i\}, & \text{if } x_i = 1 \end{cases}, \tag{4.3}$$

$$C_i = \begin{cases} 1 - \lambda_g \cdot G_i, & \text{if } l_i = 1 \wedge l_r = 1 \\ \infty, & \text{otherwise} \end{cases}, \tag{4.4}$$

where $\mathrm{d}(f_i, p_r)$ measures the Euclidean distance between the farthest vertex of face i and the fitted plane p_r of the region r. The plane is obtained by linear least squares fitting using all the vertices of the region and dynamically updated when a new face has been added. During growing, when $x_i = 0$, $\min\{\mathrm{d}(f_i, p_r), C_i\}$ measures the cost of assigning face i the same label as the current region r (i.e., the cost of adding face i to the current region r). On the contrary, the cost is measured by $1 - \min\{\mathrm{d}(v_i, p_r), C_i\}$ when $x_i = 1$. In particular, for the *planar* case, since $C_i = \infty$, the geometric distance $\mathrm{d}(f_i, p_r)$ is actually used as the cost measure. For the *non-planar* case, the prior term G_i (see Equation 4.1) is considered to define the cost C_i. $\lambda_g \geq 0$ is a weight that controls the relative numbers of *planar* and *non-planar* segments (see Figure 4.5a and Figure 4.5d).

The **pairwise term** $\varphi_{i,r}(x_i, x_r)$ is designed to control the smoothness degree during the growing process,

$$\varphi_{i,r}(x_i, x_r) = \angle(\mathbf{n}_i, \mathbf{n}_r) \cdot \mathbb{1}(x_i \neq x_r), \tag{4.5}$$

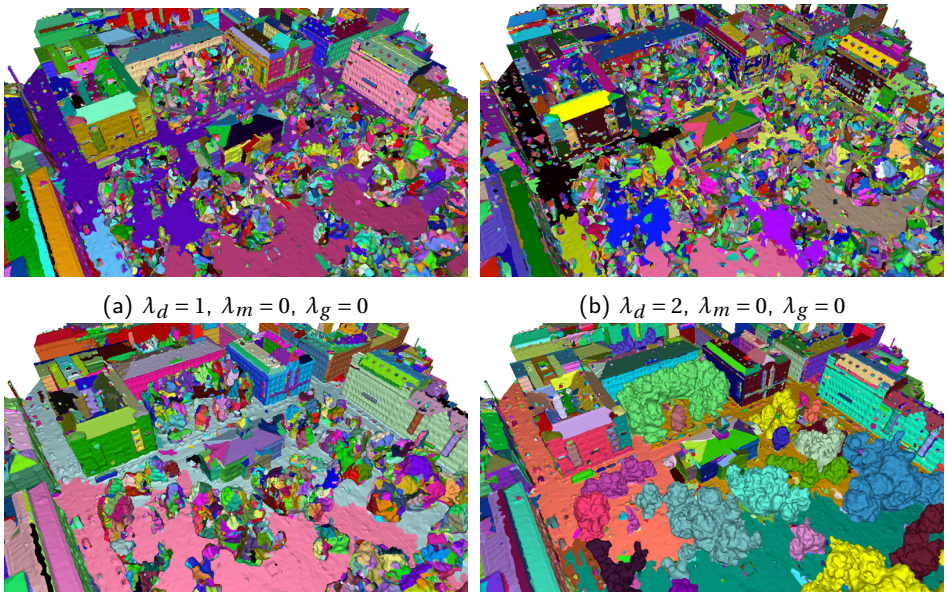
where $\mathbf{n}_i$ and $\mathbf{n}_r$ denote the normals of a neighboring triangle face i and the region r, respectively. This term encodes the angle between these normal vectors to reduce the normal deviation within the local neighborhood in the segmentation. $\mathbb{1}(x_i \neq x_r)$ is an indicator function that measures the coherence between x_i and x_r.

The energy $U(X)$ is minimized using the $\alpha - \beta$ swap graph cut algorithm [Boykov *et al.*, 2001] to accumulate a face for the current segment. The growth of a segment stops if no more faces can be accumulated. We then restart growing a new segment from the face with the highest *planar* probability in the remaining set of faces. The growing of segments is repeated until all mesh faces have been processed.

4.3.2. Classification

We construct a graph whose nodes encode features of the segments and edges encode interactions between segments. With this graph, the semantic segmentation of the mesh is achieved by classifying the segments using GCN.

Node feature embedding In our graph, each node represents a segment, and it encodes two types of features generated based on the vertices and face centroids of the segment: 1) $F_l(s_k)_{256}$ is learned using PointNet [Charles *et al.*, 2017], and the input to it is a point cloud of randomly sub-sampled points from mesh vertices and face centroids. The size of the feature vector is 128×6 and consists of XYZ and RGB; 2) $F_h(s_k)_{48}$ is generated from the handcrafted feature generator (HFG), and it contains the same type of features used for *planar and non-planar classification* (see in subsection 4.3.1) and four additional shape-based features capturing local geometric differences (see Table 4.1).



(a) $\lambda_d = 1,\ \lambda_m = 0,\ \lambda_g = 0$ (b) $\lambda_d = 2,\ \lambda_m = 0,\ \lambda_g = 0$

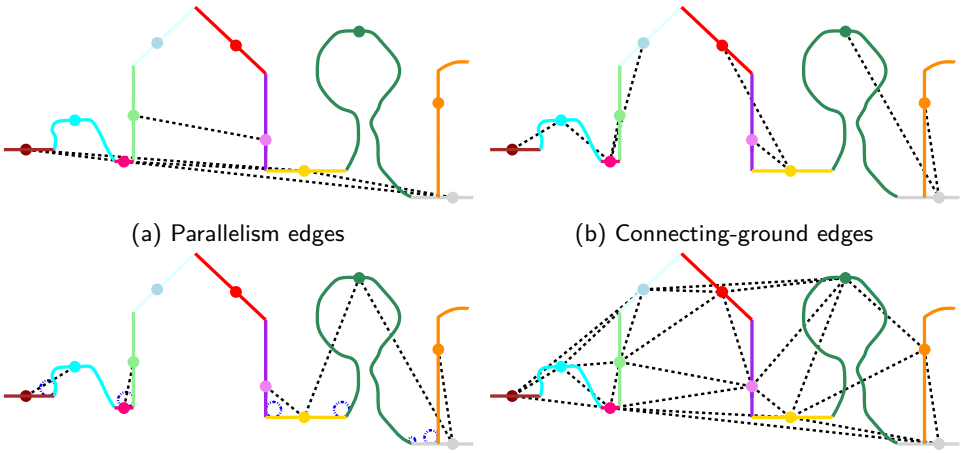
(c) $\lambda_d = 1,\ \lambda_m = 1,\ \lambda_g = 0$ (d) $\lambda_d = 1,\ \lambda_m = 0,\ \lambda_g = 1$

Figure 4.5: The effect of the parameters λ_d, λ_m, and λ_g on the over-segmentation: these parameters provide control over the size and boundary smoothness of the segments.

Compactness	$CP_k = \frac{4 \cdot \pi \cdot area(s_k)}{C(s_k)^2}$	**Shape Index**	$SI_k = \frac{C(s_k)}{\sqrt[4]{area(s_k)}}$
Straightness	$SD_k = \frac{\sum_{i=1}^{m} \frac{\lambda_2}{\lambda_3}}{m}$	**Avg Distance**	$D_k = \frac{\sum_{n}^{i=1} dist(p_i, P_k)}{n}$

Table 4.1: Shape-based features defined on segments: $C(s_k)$ is the circumference of a segment s_k. λ_2 and λ_3 are the eigenvalues derived from the linear fitting line of m boundary points in 3D [Pearson, 1901]. *Avg Distance* measures the average distance from n mesh vertices p_i to the supporting plane P_k of the segment.

Edge feature embedding In contrast to graphs defined on 3D points or triangle faces, graphs defined on certain segmentation of the data can better capture global contextual relationships than simple adjacency connections. The idea behind the designed graph is to give more prominence to the segment differences that are usually present in urban scenarios. To this end, we intend to include global features that fulfill two conditions. First, the global features should be generalizable, i.e., they can be captured in different scenarios. Second, the global features should be established between graph nodes that have large feature differences. To make full use of the *planar* and *non-planar* segments and establish meaningful relationships between the segments, we propose a graph consisting of the following four types of edges (see Figure 4.6):



(a) Parallelism edges (b) Connecting-ground edges

(c) ExMAT edges (d) Spatially-proximate edges

Figure 4.6: An illustration of the four types of edges in our graph: each color indicates a segment encoded as a node (i.e., the colored dot on each segment) in the graph. The dash lines denote the graph edges. In (c), the blue circles represent the exterior shrinking balls.

1) *parallelism edges*: edges connecting parallel *planar* segments. Two *planar* segments are considered parallel if the angle between their supporting planes is smaller than a threshold (5° in our experiments). These edges mainly connect the *planar* segments belonging to man-made objects.

2) *connecting-ground edges*: edges connecting segments and their local ground planes. A local ground plane is identified as the lowest and largest *planar* segment in a cylindrical neighborhood (30 m in our experiments) around the boundary vertices of the segment. These edges primarily capture the relationship between the ground and all non-ground objects.

3) *exterior medial axis transform (ExMAT) edges.* We first build the ExMAT (i.e., exterior shrinking ball radius of 3D medial axis transformation) [Ma *et al.*, 2012; Peters and Ledoux, 2016] on the segments, and we introduce graph edges that link the segments connected by the exterior shrinking ball (see Figure 4.6c). Since the external skeleton usually corresponds to the joints between objects, ExMAT edges allow connecting segments that are adjacent but belong to different objects.

4) *spatially-proximate edges.* We first build a 3D Delaunay triangulation [Jaromczyk and Toussaint, 1992] with the input mesh vertices and the centroids of mesh faces. Two segments are connected by an edge if at least one pair of points from the two segments are connected by a Delaunay edge. This type of edges allow the encoding of contextual information on different scales. Particularly, these edges contribute to capturing the relationships between urban objects from short-range (for objects

that are close to the ground) to long-range (for objects that are far away from the ground).

With the above graph edges defined, we establish the edge feature as $F_h(e_{k,k+1}) = log(F_h(s_k)/F_h(s_{k+1}))$, where s_k and s_{k+1} are the two segments connected by an edge $e_{k,k+1}$. We also introduced two additional edge features defined as the mean and standard deviation of the vertex offsets of the segment boundaries, in which the offset is defined using the closest point pair between two segments.

Segment classification Based on the graph and feature embedding (see Figure 4.7), we exploit a GCN [Y. Li *et al.*, 2017] to classify the segments. The node features learned from PointNet [Charles *et al.*, 2017] and the handcrafted features are concatenated and fed to MLP (Multilayer perceptron) to output a 64D feature vector that serves as the hidden state of the Gated Recurrent Unit (GRU: a gating mechanism in recurrent neural networks for updating and resetting hidden states to capture short-term and long-term dependencies in sequence) [Cho *et al.*, 2014]. We apply ReLU activation [Nair and Hinton, 2010] and batch normalization [Ioffe and Szegedy, 2015] for each hidden layer of all MLPs. The computed edge features are used as the input to the Filter Generating Network (FGN: a sequence of MLPs with widths of 32, 128, and 64 to output a 64D edge feature vector) [Landrieu and Simonovsky, 2018]. The output edge weights are then used to update the hidden state and refine the GRUs via Edge-Conditioned Convolution (ECC: a dynamic edge-conditioned filter that computes element-wise vector-vector multiplication for each edge and averages the results over respective nodes) [Simonovsky and Komodakis, 2017]. To alleviate class imbalance, we apply a standard cross-entropy loss [Szegedy *et al.*, 2016] $l_n = -w_{y_n} \log \frac{\exp(x_{n,y_n})}{\sum_{c=1}^{C} \exp(x_{n,c})}$ weighted by $w_{y_n} = \sqrt{N/n_c}$, where x is the input, and y is the target. N denotes the total number of segments, and n_c represents the number of segments in each class c. The final output is per-segment labels that are then transferred to the faces of the input mesh.

4.4. Evaluation

4.4.1. Data split

We have implemented our mesh over-segmentation with CGAL [The CGAL Project, 2024] and Easy3D [Nan, 2021b], and the semantic classification with PyTorch [Paszke *et al.*, 2019]. All experiments were carried out on a desktop PC with a 3.5GHz CPU and a GTX 1080Ti GPU.

We have used the SUM dataset [W. Gao *et al.*, 2021] and the H3D dataset [Kölle *et al.*, 2021] to evaluate our method. To the best of our knowledge, SUM is the largest benchmark dataset for semantic urban meshes, which covers about 4 km^2 of Helsinki (Finland) with six object classes: *terrain* (*terra.*), *high vegetation* (*h-veg.*), *building* (*build.*), *water*, *vehicle* (*vehic.*), and *boat*. The whole dataset contains 64 tiles, each

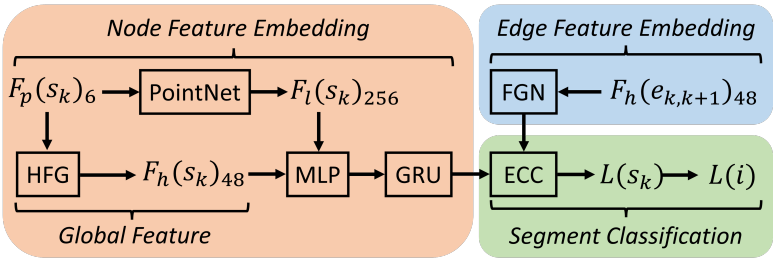


Figure 4.7: The feature embedding components for segment classification: our network takes mesh vertices and face centers (with XYZ and RGB) of each segment, denoted as $F_p(s_k)_6$, as input. PointNet [Charles *et al.*, 2017] is used to learn features $F_l(s_k)_{256}$ for each segment. The handcrafted features $F_h(s_k)_{48}$ are computed using the feature generator (HFG). These two types of features are then processed jointly by the MLP and then refined in the GRU. The $F_h(e_{k,k+1})_{48}$ are handcrafted edge features and input to a filter-generating network (FGN). The final classification is obtained using edge-conditioned convolution (ECC), which takes both node and edge features as input. The output segment labels $L(s_k)$ are then transferred to face labels $L(i)$.

covering a 250 m × 250 m area. Following the SUM baseline, we used 40 tiles (62.5% of the whole dataset) for training, 12 tiles (18.75%) for the test, and 12 tiles for validation. The H3D dataset covers about 0.19 km^2 area of the village of Hessigheim (Germany) with 11 classes: *Low Vegetation*, *Impervious Surface*, *Vehicle*, *Urban Furniture*, *Roof*, *Facade*, *Shrub*, *Tree*, *Soil/Gravel*, *Vertical Surface*, and *Chimney*. We follow the data splits in H3D [Kölle *et al.*, 2021], and we further merge small mesh tiles into a large one to obtain more contextual information.

4.4.2. Evaluation metrics

Metrics for over-segmentation Our over-segmentation aims to produce homogeneous segments to better facilitate semantic segmentation. We propose three novel evaluation metrics focusing on the impact of the over-segmentation on the final semantic segmentation: *object purity (OP)*, *boundary precision (BP)*, and *boundary recall (BR)*.

Since our goal is semantic segmentation, the best achievable over-segmentation is identical to the ground truth semantic segmentation. In this ideal situation, each segment covers exactly an individual object, and its boundaries perfectly align with the object boundaries. Thus, similar to *intersection over union*, we define *object purity* as

$$OP(S,G) = \frac{\sum_k purity(s_k, G)}{area(G)}, \tag{4.6}$$

where $S = \{s_k\}$ denotes the set of segments in our over-segmentation, and $G = \{g_k\}$ are the segments extracted as connected components from the ground truth semantic

segmentation. $purity(s_k, G)$ measures the surface area of the largest overlapping region between a segment and the ground truth segments (see Figure 4.8).

Boundary precision measures the correctness of the segment boundaries. Thus, it is defined to quantify how much the segment boundaries overlap with the boundaries of the ground truth semantic segmentation,

$$BP(B_S, B_G) = \frac{length(B_S \cap B_G)}{length(B_S)}, \tag{4.7}$$

where B_S and B_G denote the boundaries of the over-segmentation and those of the ground truth of the semantic segmentation, respectively. The function $length(\cdot)$ quantifies the total length of a set of segment boundaries. To handle noisy and dense meshes, we allow a tolerance when looking for overlapping boundary edges. Specifically, two edges e_1 and e_2 are considered overlapping if the two endpoints of e_2 fall within the 2-ring neighborhood of the endpoints of e_1 (see Figure 4.8).

Boundary recall measures the completeness of the segment boundaries, defined as

$$BR(B_S, B_G) = \frac{length(B_S \cap B_G)}{length(B_G)}. \tag{4.8}$$

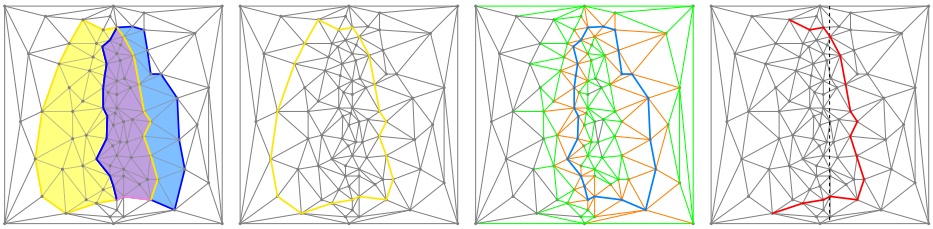


Figure 4.8: 2D schematic of *object purity*, *boundary precision*, and *boundary recall*. In (a), the yellow region represents the ground truth segment G. The blue region represents the generated segment S. The pink region represents the largest overlapping region between a segment and the ground truth segments. In (b), the yellow edges represent the border B_G of the ground truth segment. In (c), the blue edges represent the border B_S of the generated segment. The orange edges represent the first ring of B_S. The green edges represent the second ring of B_S. In (d), the red edges are the intersection of B_G and B_S (as well as its first two rings). The black dashed line represents the true border between objects.

Metrics for semantic segmentation To evaluate semantic segmentation results, we measure the precision, recall, F1 score, and intersection over union (IoU) for each object class, and we also record the overall accuracy (OA), mean accuracy (mAcc), and mean intersection over union (mIoU) of all object classes.

4.4.3. Evaluation of over-segmentation

We evaluate over-segmentation on SUM dataset [W. Gao *et al.*, 2021] because it covers a larger area and contains fewer unlabeled areas compared to H3D dataset [Kölle *et al.*, 2021]. Figure 4.9 presents our planarity-sensible over-segmentation result and comparison with seven other commonly used over-segmentation techniques, namely region growing (RG) [Lafarge and Mallet, 2012], efficient RANSAC (RA) [Schnabel *et al.*, 2007], geometric partition (GP) [Landrieu and Simonovsky, 2018], supervised superpoint generation (SSP) [Landrieu and Boussaha, 2019], variational shape approximation (VSA) [Cohen-Steiner *et al.*, 2004], supervoxel generation (SPV) [Y. Lin *et al.*, 2018], voxel cloud connectivity segmentation (Vccs) [Papon *et al.*, 2013], superface clustering (SC) [Verdie *et al.*, 2015], and superface partitioning (SP) [Rouhani *et al.*, 2017]. RG, VSA, SC, SP, and our method use meshes as input, and the other methods (originally developed for point clouds) perform over-segmentation on points that we densely sampled (10 pts/m^2) from the input mesh. We can see from Figure 4.9

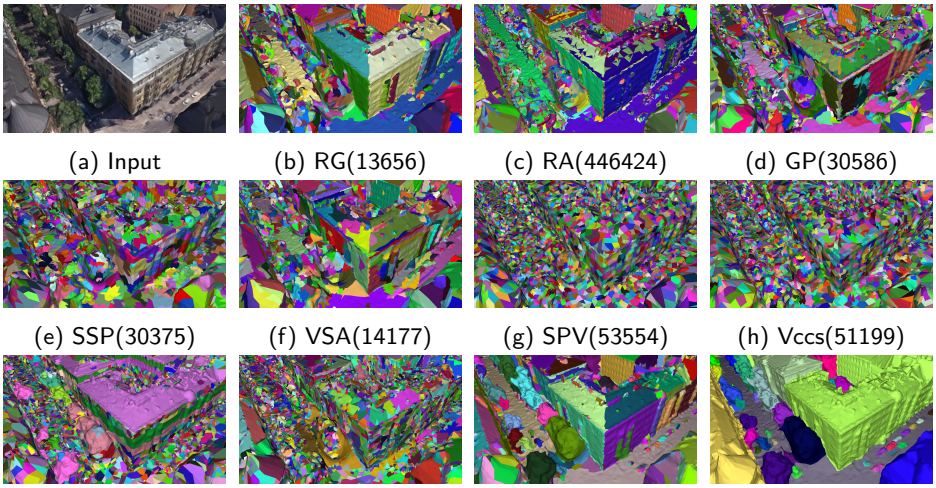
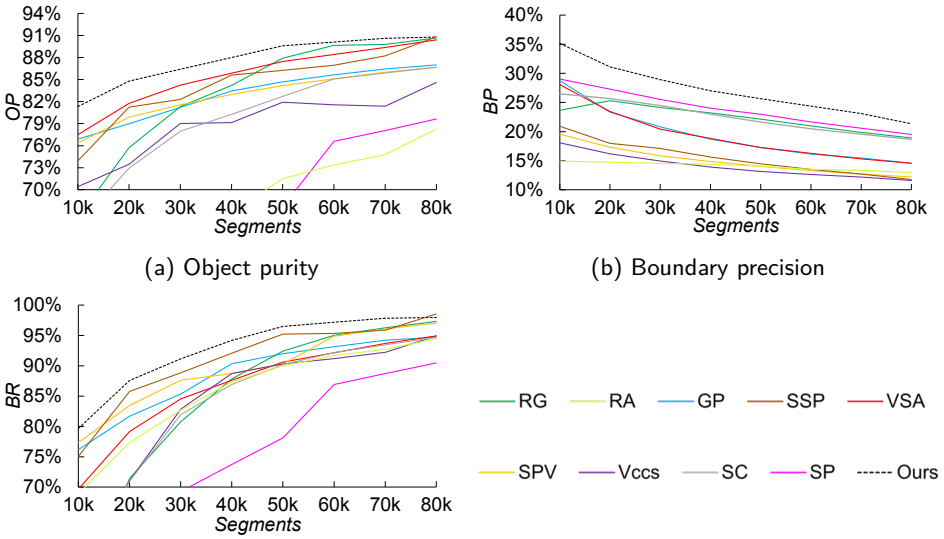


Figure 4.9: Comparison of mesh over-segmentation methods on a tile of the SUM dataset [W. Gao *et al.*, 2021]: (b) to (k) show the over-segmentation results with the same *object purity* (around 92%) for region growing(RG) [Lafarge and Mallet, 2012], efficient RANSAC (RA) [Schnabel *et al.*, 2007], geometric partition (GP) [Landrieu and Simonovsky, 2018], supervised superpoint generation (SSP) [Landrieu and Boussaha, 2019], variational shape approximation (VSA) [Cohen-Steiner *et al.*, 2004], supervoxel generation (SPV) [Y. Lin *et al.*, 2018], voxel cloud connectivity segmentation (Vccs) [Papon *et al.*, 2013], superface clustering (SC) [Verdie *et al.*, 2015], and superface partitioning (SP) [Rouhani *et al.*, 2017]. The number below each result denotes the number of segments required to achieve the desired *object purity*. (l) shows the connected components extracted from the ground truth semantic segmentation.

that the segment boundaries of our method are largely aligned with object boundaries. RG and VSA perform similarly but generate excessive segments for non-planar objects such as trees. Our over-segmentation generates segments that are closer to semantically meaningful objects. In terms of the number of segments, RA, SC, SP, Vccs, and SPV generate relatively large numbers of segments, which are unfavorable to the subsequent classification using GCN. In contrast, our method generates the least segments, which is a strong advantage for the subsequent classification step in terms of efficiency. We have used the entire test split of SUM dataset [W. Gao *et al.*, 2021] to



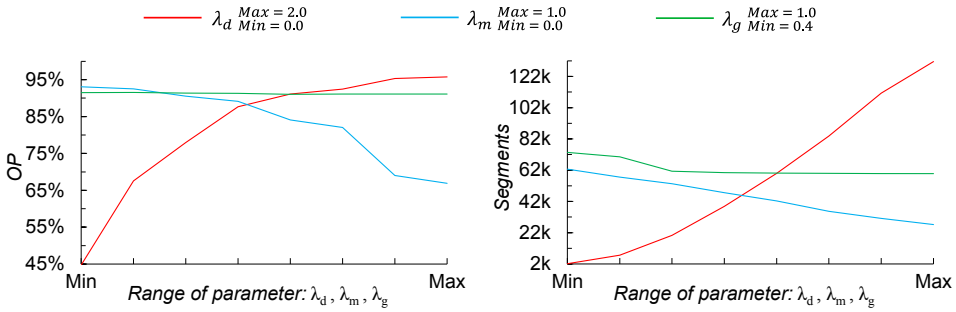
(a) Object purity

(b) Boundary precision

(c) Boundary recall

Figure 4.10: Comparison of different over-segmentation method in terms of *OP*, *BP*, and *BR* on the test split of SUM dataset [W. Gao *et al.*, 2021]. For each method, we tuned their parameters so that all methods generated a similar number of segments. The data was recorded at different numbers of segments for all methods.

evaluate our over-segmentation method in terms *OP*, *BP*, and *BR*, and we have compared them with those of the other seven over-segmentation methods. In the comparison, we tuned the parameters of each method such that all methods generated a similar number of segments, and we then computed the *OP*, *BP*, and *BR* for each method. We recorded the performance of all methods for different numbers of segments, and the results are shown in Figure 4.10. It can be observed that our method outperforms the others for all three metrics. Specifically, as the number of segments increases, the *OP* of VSA, RG, and SSP get closer to ours. However, VSA underperforms our method in terms of *BP* and *BR*, which indicates that our method generated segments with better boundary qualities. For RG and SSP, its *OP*, *BP*, and *BR* are rather low when the number of segments is small, indicating that our method is more robust with a relatively small number of segments. Other methods like RA, SP, GP, SPV, Vccs, and SC also require a larger number of segments to produce satisfactory results. To



(a) Object purity (b) Number of Segments

Figure 4.11: Comparison of over-segmentation with different parameter configurations in terms of object purity and the number of segments on the SUM dataset [W. Gao *et al.*, 2021]. Note that the range of parameters depends on the quality of the input data.

understand the potential of each method, Table 4.2 provides the maximum achievable performance of semantic segmentation for each over-segmentation method. The maximum achievable performance is measured by the maximum IoU and mIoU that can be achieved in theory. We can see that our method significantly outperforms the other methods with a considerable margin ranging from 4.8% to 30.1% in terms of mIoU (which reflects the under-segmentation errors). It is also worth noting that VSA has slightly better results on very few object classes (e.g., *water* and *boat*) while our method can better distinguish small non-planar objects such as vehicles which are very common in urban textured meshes.

Performance analysis For the impact of planarity-sensible over-segmentation in different configurations, we experimented with different settings for each weight term based on the default parameters (i.e., $\lambda_d = 1.2$, $\lambda_m = 0.1$, and $\lambda_g = 0.9$). In each experiment, only one weight is tuned while the others remain unchanged. Figure 4.11 shows its results in terms of *OP* and the number of *segments*. We can observe that increasing λ_d leads to a larger *OP* but also encourages the splitting of segments into smaller planar ones. In contrast, increasing λ_m results in over-smoothed segments (i.e., the smaller segments are merged into a larger one). However, λ_g performs differently from the other two since our aim is to use λ_g to reduce the number of *segments* without decreasing *OP* (as demonstrated in Figure 4.11). The performance of applying λ_g is limited to the results of *planar* and *non-planar* classification, while the performance of the other two terms is limited to the quality of the mesh.

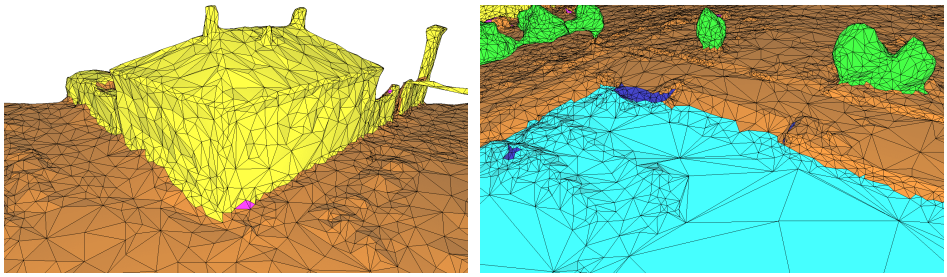
Methods	Terra.	H-veg.	Build.	Water	Vehic.	Boat	mIoU
SP [Rouhani *et al.*, 2017]	73.4	88.1	96.8	12.5	15.7	68.1	59.1
RANSAC [Schnabel *et al.*, 2007]	82.3	88.6	97.0	57.3	29.8	69.7	70.8
Vccs [Papon *et al.*, 2013]	77.4	86.3	94.3	87.4	35.9	84.5	77.6
SC [Verdie *et al.*, 2015]	86.8	92.5	97.8	75.5	46.1	79.3	79.7
SPV [Y. Lin *et al.*, 2018]	83.7	91.5	97.0	87.1	37.5	84.5	80.2
GP [Landrieu and Simonovsky, 2018]	86.5	91.2	96.6	87.1	46.9	84.6	82.1
RG [Lafarge and Mallet, 2012]	90.9	93.9	98.4	84.5	53.9	75.6	82.9
SSP [Landrieu and Boussaha, 2019]	85.3	91.2	96.1	91.1	49.6	88.3	83.6
VSA [Cohen-Steiner *et al.*, 2004]	91.1	95.1	98.7	**93.4**	39.3	**88.9**	84.4
Ours	**93.3**	**95.6**	**98.9**	91.6	**67.1**	88.8	**89.2**

Table 4.2: Comparison of different over-segmentation methods in terms of maximum achievable performance of semantic segmentation on test data from the SUM dataset [W. Gao *et al.*, 2021], with 50,000 segments. Evaluation metrics are reported as per-class IoU (%) and mean IoU (mIoU, %).

4.4.4. Evaluation of semantic classification

We have tested our semantic classification method on both the SUM [W. Gao *et al.*, 2021] and the H3D [Kölle *et al.*, 2021] datasets.

Evaluation on SUM. Figure 4.13 shows the results for two tiles from the SUM dataset. From the extensive experiments, we also observed that our method is robust against non-uniform triangulation of mesh, for which an example is shown in Figure 4.12.

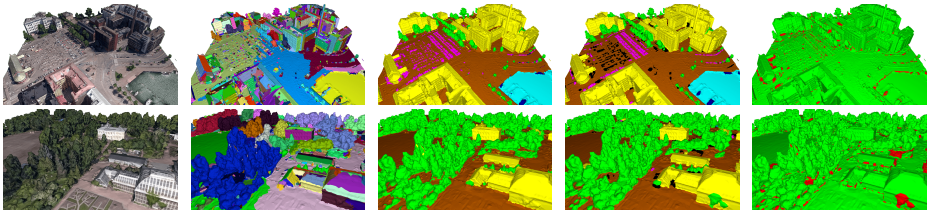


(a) Near-uniform triangulation (b) Non-uniform triangulation

Figure 4.12: Robustness against triangulation

We have also compared our method with several state-of-the-art semantic segmentation approaches, among which RF-MRF [Rouhani *et al.*, 2017] and SUM-RF [W. Gao *et al.*, 2021] directly consume meshes. To compare with methods originally developed for semantic segmentation of point clouds, e.g., PointNet [Charles *et al.*, 2017], PointNet++ [Qi *et al.*, 2017], SPG [Landrieu and Simonovsky, 2018], KPConv [Thomas *et al.*, 2019], and RandLA-Net [Q. Hu *et al.*, 2020], we sampled points from the meshes by following [W. Gao *et al.*, 2021]. For each deep learning method

designed for point clouds, we feed it with the colored point clouds densely sampled from the texture meshes. We tune the hyper-parameters starting with their default setting. For a fair comparison, we use the same weight for all the competing methods involved in the comparison. Due to the randomness of deep learning, we ran each method ten times with the same settings to record its average performance.



(a) Input (b) Segments (c) Semantic (d) Truth (e) Error (red)

Figure 4.13: Semantic segmentation results of our method on two tiles from the SUM dataset [W. Gao *et al.*, 2021]

We report the results in Table 4.3, and we can see the results of the top three best methods in Figure 4.14. Our method achieves the highest per-class IoU on the majority of object classes, and it outperforms all other methods in all overall metrics, with a margin from 4% to 35.9% in terms of mIoU. Compared to KPConv and SPG, our method requires less training time, and our results are more stable (i.e., with smaller standard deviations).

Evaluation on H3D. We also attempted to train and test on the H3D dataset [Kölle *et al.*, 2021] (see for an overview of the results in Figure 4.15). It should be noted that the H3D dataset is much smaller (0.19 km^2) than SUM (4 km^2). In addition, 40% of the area of the mesh in H3D is unlabeled, and many triangles have incorrect labels, which was due to the limitation in their cloud-to-mesh labeling process where the correspondence was either not completed or had ambiguities when transferring the labels from the points to the mesh faces. This prevents H3D from being an ideal training dataset for us, as both our method and other deep learning methods require a large amount of labeled data. Besides, to distinguish between different classes that are geometrically coplanar in H3D (e.g., Low Vegetation, Impervious Surface, Soil, and Gravel), the *planar* class is divided into sub-classes and used as a prior for over-segmentation in our approach. The test results show that our method achieves about 52.2% mIoU, outperforming the KPConv (45.5% mIoU), SPG (29.9% mIoU), and PointNet++ (15.6% mIoU).

4.4.5. Ablation study

To understand the effect of several design choices made in the graph construction, and the contributions of the hand-crafted features and the learned features, we have conducted an ablation study on the SUM dataset [W. Gao *et al.*, 2021].

Methods	Terra.	H-veg.	Build.	Water	Vehic.	Boat	mIoU	OA	mAcc	mF1	Training (h)	Testing (min)
PointNet [Charles *et al.*, 2017]	56.3	14.9	66.7	83.8	0.0	0.0	36.9 § 2.3	71.4 § 2.1	46.1 § 2.6	44.6 § 3.2	1.8	**1**
RandLaNet [Q. Hu *et al.*, 2020]	38.9	59.6	81.5	27.7	22.0	2.1	38.6 § 4.6	74.9 § 3.2	53.3 § 5.1	49.9 § 4.8	10.8	52
SPG [Landrieu and Simonovsky, 2018]	56.4	61.8	87.4	36.5	34.4	6.2	47.1 § 2.4	79.0 § 2.8	64.8 § 1.2	59.6 § 1.9	17.8	26
PointNet++ [Qi *et al.*, 2017]	68.0	73.1	84.2	69.9	0.5	1.6	49.5 § 2.1	85.5 § 0.9	57.8 § 1.8	57.1 § 1.7	2.8	3
RF-MRF [Rouhani *et al.*, 2017]	77.4	87.5	91.3	83.7	23.8	1.7	60.9 § 0.0	91.2 § 0.0	65.9 § 0.0	68.1 § 0.0	**1.1**	15
SUM-RF [W. Gao *et al.*, 2021]	83.3	90.5	92.5	**86.0**	37.3	7.4	66.2 § 0.0	93.0 § 0.0	70.6 § 0.0	73.8 § 0.0	1.2	18
KPConv [Thomas *et al.*, 2019]	**86.5**	88.4	92.7	77.7	**54.3**	13.3	68.8 § 5.7	93.3 § 1.5	73.7 § 5.4	76.7 § 5.8	23.5	42
Ours	84.9	**90.6**	**93.9**	84.3	50.9	**32.3**	**72.8** § 2.0	**93.8** § 0.4	**79.2** § 3.0	**81.6** § 2.3	16.3	62

Table 4.3: Comparison with state-of-the-art semantic segmentation methods on the SUM dataset [W. Gao *et al.*, 2021]: per-class IoU (%), mean IoU (mIoU, %), Overall Accuracy (OA, %), mean class Accuracy (mAcc, %), mean F1 score (mF1, %), and the running times for training (over-seg: 1.8 hours, graph: 2.7 hours, classification: 11.8 hours, total: 16.3 hours) and testing (over-seg: 15 minutes, graph: 40 minutes, classification: 7 minutes, total: 62 minutes) are included. Note that each method was run ten times, and the mean performance is reported here.

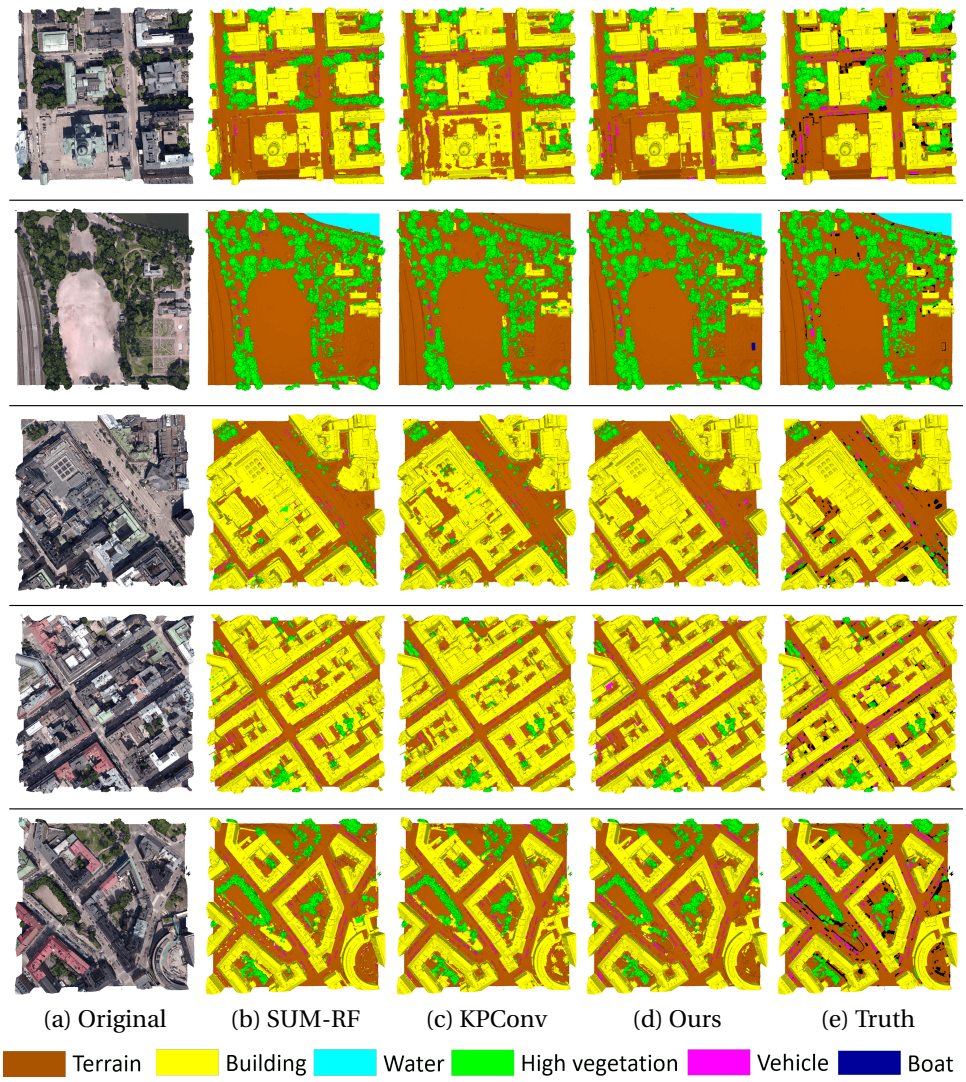
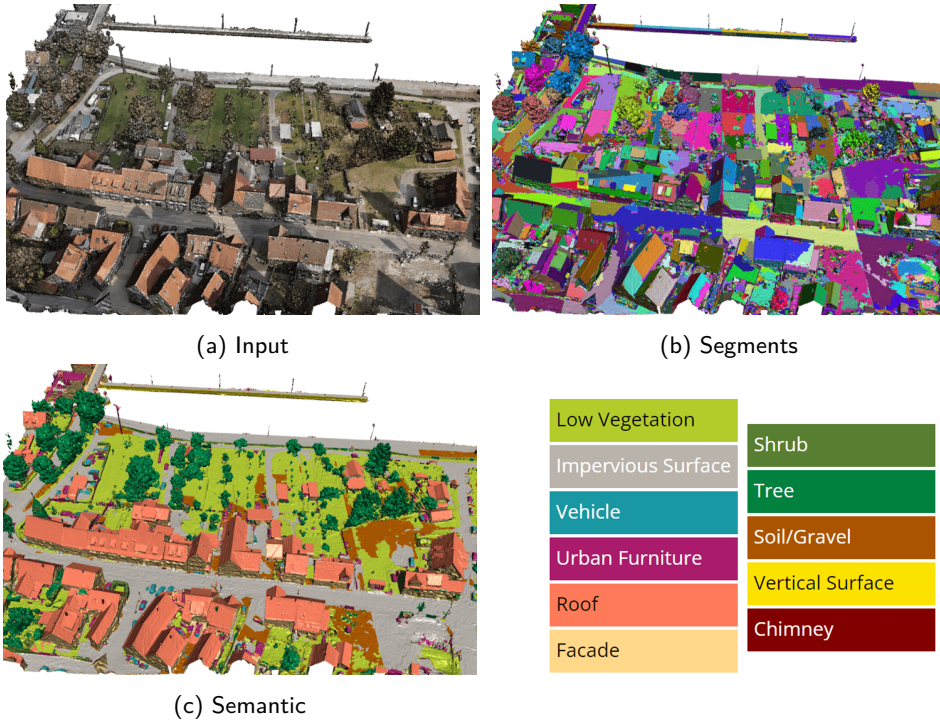


Figure 4.14: Semantic segmentation results of SUM-RF [W. Gao *et al.*, 2021], KPConv [Thomas *et al.*, 2019], and our method on five tiles from the SUM dataset [W. Gao *et al.*, 2021]

Table 4.4 summarizes the ablation result of the graphs and the features. The upper part of Table 4.4 reveals the impact of removing each type of graph edge (i.e., connections between segments) on the final semantic segmentation. It reveals that every type of graph edge contributes to the performance, and removing any of them results in a drop in mIoU in the range [2.9%,4.3%]. This implies that both local and global interactions between segments provide useful information and play an important role in semantic



(a) Input (b) Segments

(c) Semantic

Figure 4.15: Semantic segmentation results of our method on the test area of the H3D dataset [Kölle *et al.*, 2021]

segmentation. To understand the effectiveness of the designed graph, we compared it to using a graph with random connections (with the same number of edges as in our designed graph). Although these connections may overlap with the edges we have designed, the presence of randomness significantly reduces the capability of the network. This is because the random set of edges does not convey the effective features captured by our carefully designed edges. Besides, the orthogonal edges were also tested (see Table 4.4), from which we can see that they are less useful than the other types of edges. This is because the orthogonal relationship is more often incident to man-made structures within the same object (such as building parts) than between segments from different objects.

The lower part of Table 4.4 details the ablation analysis of different features. We have evaluated the importance of each feature by removing it from the experiment and recording the performance of the semantic segmentation. These experiments show that the combination of all the features outperforms all degraded features with a margin of mIoU from 2.4% to 7.8%, which means every feature contributes to the performance. It is also interesting to notice that the results are more stable without RGB information as input to PointNet for feature learning, which indicates the low quality (e.g., distortion, shadow) of mesh textures in our training datasets.

	Model	OA (%)	mAcc (%)	mIoU (%)	ΔmIoU (%)
Graph Edges	No *parallelism*	92.9	75.3	68.5 ś 1.9	-4.3
	No *ExMAT*	93.1	76.0	69.3 ś 1.3	-3.5
	No *spatial-proximity*	93.1	76.3	69.4 ś 1.9	-3.4
	No *connecting-ground*	93.3	76.3	69.9 ś 1.6	-2.9
	Only *random*	92.5	74.9	67.8 ś 2.3	-5.0
	With *orthogonal*	93.0	75.3	68.9 ś 1.8	-3.9
Features	No *All Handcrafted*	89.7	75.9	65.0 ś 3.8	-7.8
	No *Offset*	92.4	75.0	67.2 ś 1.5	-5.6
	No *PointNet*	92.8	74.5	68.2 ś 1.3	-4.6
	No *Eigen*	93.0	75.3	68.3 ś 2.3	-4.5
	No *Color*	93.1	75.0	68.9 ś 1.4	-3.9
	No *Density*	93.1	76.4	69.4 ś 2.0	-3.4
	No *Scale*	93.0	76.1	69.7 ś 0.9	-3.1
	No *Shape*	93.2	75.6	69.7 ś 0.8	-3.1
	No *RGB with PointNet*	93.5	76.4	70.4 ś 0.4	-2.4
	Ours	**93.8**	**79.2**	**72.8** ś 2.0	-

Table 4.4: Ablation study of graph edges and features on the SUM dataset [W. Gao *et al.*, 2021]: *PointNet* denotes features learned from PointNet [Charles *et al.*, 2017] with XYZ and RGB as input, and 'No *RGB with PointNet* means RGB is not used as input.

4.4.6. Generalization ability

We have conducted experiments to test the generalization ability of our method and the competing methods. Such tests can indicate the applicability of models trained by different methods on practical test datasets. SUM [W. Gao *et al.*, 2021] and H3D [Kölle *et al.*, 2021] are well qualified for generalization ability tests as they both represent urban scenes. As the original classes of these two datasets do not match, we merged them into four common classes that are typical of urban scenarios for testing, i.e., terrain (including water, low vegetation, impervious surface, soil, and gravel), high-vegetation (including shrub and tree), building (including roof, facade, and chimney), and vehicle (including car and boat). For all methods, we trained the model on the SUM dataset and performed the testing on the H3D dataset. In particular, for training and validation, we have used the training and validation splits of the SUM dataset as they cover a larger area than H3D. For testing, we have used training and validation splits of the H3D dataset that has publicly available ground truth labels. To compare the differences in results, we also tested the same model in the SUM test area.

As shown in the top eight rows of Table 4.5, the mIoU of all methods has improved compared to the previous six classes in the SUM test area because the task of classification has become relatively easy due to the reduction in the number of classes. The bottom eight records of Table 4.5 demonstrate the generalization ability of different methods. We can see that our method outperforms all competing methods with a margin from 5.8% to 49.3% in terms of mIoU. Except for our approach, almost all other

	Methods	Terra.	H-veg.	Build.	Vehic.	mIoU
SUM	PointNet [Charles *et al.*, 2017]	66.8	13.8	65.7	0.0	36.6
	PointNet++ [Qi *et al.*, 2017]	77.7	76.7	86.3	1.3	60.5
	SPG [Landrieu and Simonovsky, 2018]	86.0	73.9	88.5	13.9	65.6
	RF-MRF [Rouhani *et al.*, 2017]	86.8	86.7	90.5	20.7	71.2
	RandLaNet [Q. Hu *et al.*, 2020]	83.0	91.6	90.1	22.0	71.7
	KPConv [Thomas *et al.*, 2019]	89.4	84.7	91.5	**34.1**	75.0
	SUM-RF [W. Gao *et al.*, 2021]	88.0	90.2	92.3	30.4	75.2
	Ours	**89.5**	**92.0**	**93.9**	33.0	**77.1**
H3D	PointNet++ [Qi *et al.*, 2017]	0.0	0.0	0.0	1.1	0.3
	KPConv [Thomas *et al.*, 2019]	0.0	0.0	0.0	1.1	0.3
	SPG [Landrieu and Simonovsky, 2018]	0.0	0.0	18.3	0.0	4.6
	RandLaNet [Q. Hu *et al.*, 2020]	0.0	0.0	20.7	0.0	5.2
	PointNet [Charles *et al.*, 2017]	56.9	24.1	0.0	0.0	20.3
	RF-MRF [Rouhani *et al.*, 2017]	73.9	46.0	40.0	4.7	41.1
	SUM-RF [W. Gao *et al.*, 2021]	**80.2**	44.0	41.6	9.2	43.8
	Ours	74.2	**66.2**	**44.5**	**13.4**	**49.6**

Table 4.5: Generalization ability comparison: all methods are trained on four classes of the SUM dataset [W. Gao *et al.*, 2021]. The top eight rows show the scores on the testing area of the SUM dataset, while the bottom eight records show the testing results on the four classes H3D dataset. Per-class IoU (%) and mean IoU (mIoU, %) are reported here.

deep learning-based methods (i.e. PointNet++ [Qi *et al.*, 2017], SPG [Landrieu and Simonovsky, 2018], KPConv [Thomas *et al.*, 2019], and RandLA-Net [Q. Hu *et al.*, 2020]) failed to predict the classes in a new urban scene. This is because these methods all learn global features by having a large receptive field, and these global features can lead to overfitting of the model to the training data and result in degradation of generalization ability. In contrast, the global features of PointNet [Charles *et al.*, 2017] are based on aggregated local features and do not correspond to a larger receptive field, which does not have the overfitting problem. In other words, training with only local features avoids the degradation of model generalization ability, which is better illustrated by RF-MRF [Rouhani *et al.*, 2017] and SUM-RF [W. Gao *et al.*, 2021] as they only use features extracted on local segments. Whereas our approach includes global features derived from local features, our proposed graph is based on the spatial distribution of the object components in the urban scene, which facilitates the generalization of the global features.

From Table 4.4 and Table 4.5, we can conclude that compared with features learned by the neural network, the proposed handcrafted features and edges lead to better semantic segmentation results and contribute to a stronger generalization ability, especially for data with domain gaps. In particular, the domain gaps are attributed to the differences between training data and test data (i.e., different feature distributions), and the reasons for these differences can be grouped into two main categories: 1) same data acquisition and processing pipeline, but covering different urban scenarios (e.g., from dense urban area to rural or forest area); 2) covering the same urban scenarios but with different

data acquisition (e.g., using different sensors or different parameters for data collection) and processing pipelines (e.g., using different approaches or parameter configurations for generating the 3D data). Nevertheless, the features learned by the existing neural network architectures cannot cope with such differences without adding new training samples from the test area. Our segment-based handcrafted features and edges capture the intrinsic characteristics of the object (e.g., the facade is usually perpendicular to the ground or the surface of the tree is undulating and non-planar) and can better cope with the variance in feature distribution.

4.4.7. Limitations

Our method is based on the observation that urban scenes consist of objects demonstrating both planar and non-planar regions and that object boundaries lie in the connections between the planar and non-planar regions. In special cases where adjacent objects contain only non-planar regions (e.g., vehicles underneath trees), our method will not be able to differentiate them. In addition, our approach generates segments for semantic classification, which reduces memory consumption as well as the number of samples. Specifically, if the training data covers a small area (e.g., H3D [Kölle *et al.*, 2021]) and only a few segments are generated after over-segmentation, the number of samples may not be sufficient to train a competent model. Possible solutions include data augmentation of segments and graph connections or adding more labeled data. Besides, our method requires adjacency information of the mesh for incremental region growing. Additional preprocessing is necessary for meshes that are non-manifold or contain duplicated vertices, as they destroy the topological adjacency information. Potential solutions can be to split non-manifold vertices or to reconstruct adjacency information of duplicated vertices. In our experiments, the meshes in the SUM dataset [W. Gao *et al.*, 2021] are 2-manifold, while the meshes in the H3D dataset [Kölle *et al.*, 2021] are not.

4.5. Conclusion

We have presented a two-stage supervised framework for semantic segmentation of large-scale urban meshes. Our planarity-sensible over-segmentation algorithm favors generating segments largely aligned with object boundaries, closer to semantically meaningful objects, can deliver descriptive features, and can represent urban scenes with a smaller number of segments. A thorough analysis reveals that our planarity-sensible over-segmentation plays a key role in achieving superior performance in semantic segmentation. We have also shown that exploiting multi-scale contextual information better facilitates semantic segmentation. Furthermore, we have demonstrated that our proposed approach achieves better generalization abilities in comparison with other methods, owing to the segment-based local features and unique connections in graphs. Our proposed new metrics are effective for evaluating mesh over-segmentation methods dedicated to semantic segmentation. We believe the

proposed metrics will further stimulate improving other over-segmentation techniques. In future work, we would like to extend our framework to part-level (e.g., dormers, balconies, roofs, and facades of buildings) urban mesh segmentation. In addition, we will also investigate how the semantics learned from multi-view images can be used for the semantic segmentation of urban meshes.

5

Structure-aware interactive annotation

As we further refine the level of detail in semantic labeling, as discussed in Chapter 3 and Chapter 4, the focus now shifts to acquiring detailed, part-level semantics for urban textured meshes. Consequently, an innovative annotation framework is introduced, designed specifically for efficient semantic segmentation of large-scale urban textured meshes. The framework focuses on reducing manual annotation efforts by integrating interactive component extraction and structure-aware matching tailored to the complexities of urban settings. Employing a semi-automatic, unsupervised, and real-time approach meets the urgent need to accelerate the development of deep learning methods for analyzing urban scenes. The framework not only significantly reduces the time and interaction required for precise labeling but also ensures high accuracy in segmenting detailed urban components. Additionally, we provide a novel part-level semantic mesh dataset for large-scale urban scenes, where 'part-level' refers to the detailed annotation of object components such as building chimneys, windows, or road markings, etc. This dataset has undergone comprehensive comparison with state-of-the-art 3D semantic segmentation technologies. The results demonstrate the practical effectiveness of our method in enhancing the training and refinement of deep learning methods.

5.1. Introduction

The growing availability and detail of 3D urban data, particularly point clouds and textured meshes have paved the way for enhanced rendering and simulation of large-scale urban environments. Resources such as Google Earth's 3D imagery [Google, 2012] and Helsinki's 3D city models [Helsinki, 2019] demonstrate that these datasets can significantly advance visualization techniques, environmental simulations, and interactive graphic applications [Kelly *et al.*, 2017; Verdie *et al.*, 2015]. The complexity of these urban scenes poses challenges to current computer graphics methods, requiring innovations in efficient data handling, real-time rendering, and detailed texturing techniques. Recent advancements in semantic understanding of these datasets [Q. Hu *et al.*, 2022; G. Yang *et al.*, 2023] through computer vision and deep learning not only refine environmental modeling but also enrich applications directly relevant to spatial intelligence. These include 3D city modeling, detailed dynamic simulations for video games, and virtual reality, and realistic scenario creation for AI training in graphic settings. Enhanced semantic understanding facilitates the creation of more interactive and responsive graphic environments, capable of dynamically adapting to changes in urban landscapes [Kolbe and Donaubauer, 2021]. Moreover, these advancements underpin crucial applications that are integral to the development of smart cities, facilitating enhanced visualization for urban governance and planning, alongside detailed simulations that address energy management, microclimate variability, disaster response, and traffic flow optimization [Biljecki *et al.*, 2015b; Chao *et al.*, 2020; Garcia-Dorado *et al.*, 2017; Katal *et al.*, 2022; Verykokou *et al.*, 2016].

Semantic segmentation is crucial in understanding urban scenes by accurately classifying objects within these environments, thus significantly improving data usability and interpretation. Numerous tools, datasets, and methodologies for semantic segmentation have been developed, particularly for images and point clouds [Cordts *et al.*, 2016; Martinovi *et al.*, 2015]. However, research on textured meshes tends to focus on semantic understanding within small-scale indoor settings [Armeni *et al.*, 2017; Chang *et al.*, 2017; Dai *et al.*, 2017], with less emphasis on extensive outdoor urban scenes [Miksik *et al.*, 2015]. Despite some progress in urban mesh object segmentation by researchers, the segmentation of semantic parts, crucial for detailed semantic analysis, is often neglected [W. Gao *et al.*, 2021; Rouhani *et al.*, 2017]. Such parts include detailed components like dormers, windows, chimneys, road markings, etc., which are essential for comprehensive urban scene interpretation. Complementing these works, this paper presents a large-scale urban scene benchmark dataset with part-level semantic labels (see Figure 5.1).

Obtaining ground truth labels for large-scale urban scenes often relies on manual annotation, which is both time-consuming and expensive [W. Gao *et al.*, 2021]. Although automated methods like weakly supervised [B. Zhou *et al.*, 2016] and transfer learning [Pan and Q. Yang, 2010] can reduce labour costs, they require extensive training and frequently introduce errors that necessitate substantial manual revision to satisfy the accuracy needs of downstream applications. Semi-automatic interactive annotation tools offer a more efficient alternative for acquiring accurate

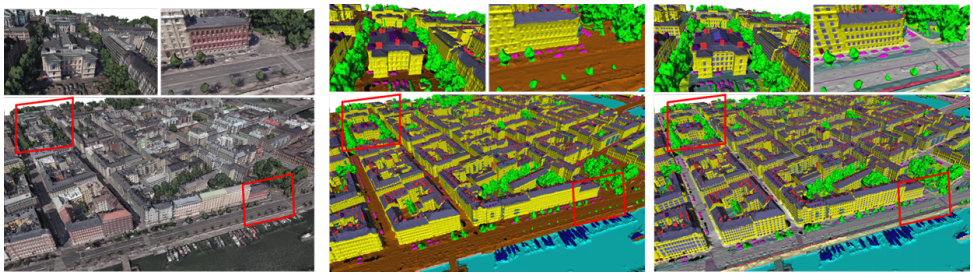


Figure 5.1: Utilizing our proposed structure-aware interactive annotation method, we created the first part-level semantic textured urban mesh dataset spanning approximately $2.5km^2$. From left to right, the images correspond to textured triangular meshes, semantic meshes with face-based labels (comprising 13 classes), and semantic meshes with texture-based labels (encompassing 21 classes). Specific semantic classes include unclassified ■, terrain ■, high vegetation ■, water ■, car ■, boat ■, wall ■, roof surface ■, facade surface ■, chimney ■, dormer ■, balcony ■, roof installation ■, window ■, door ■, low vegetation ■, impervious surface ■, road ■, road marking ■, cycle lane ■, and sidewalk ■.

labels [Kontogianni *et al.*, 2023; Q. Liu *et al.*, 2023; Rother *et al.*, 2004]. These tools aim to provide real-time accuracy with minimal user interactions. Despite advancements in semi-automatic interactive annotation for both 2D and 3D data, the quality of the output from these methods, whether traditional energy optimization or newer deep learning approaches, often requires multiple manual adjustments to reach the desired level of precision. This issue is particularly pronounced in urban textured meshes with complex geometries and class distributions, where existing methods lack effective real-time interactive annotation strategies for mesh triangles and their textures.

This paper presents an interactive, unsupervised, real-time annotation framework for urban textured meshes to obtain ground truth semantic labels efficiently. The core of this annotation framework relies on utilizing structure-aware algorithms designed to analyze and utilize the geometric and textural patterns of user-interacted regions, thereby facilitating the extraction of components in areas of interest and enabling the matching of similar structures. The process involves two main steps:

1) Interactive component extraction. For mesh faces, users can extract components in areas of interest by analyzing the local structural differences between planar areas and protrusions using a single lasso or a stroke operation. For mesh textures, our method utilizes local color feature consistency, allowing users to achieve component extraction with minimal interactions using a specially designed local expansion algorithm.

2) Structure-aware matching. To streamline the annotation of similar structures, we calculate structural features that enable rapid matching of user-extracted components with others in the scene, allowing users to annotate multiple similar components in a single interaction.

Experimental results demonstrate that our method surpasses others in interaction convenience, efficiency, and accuracy for face-based annotation. For annotation on the textures, our method also outperforms traditional methods in terms of efficiency and exceeds deep learning-based methods concerning accuracy. Using our method, we have created the first part-level semantic textured mesh dataset tailored for large-scale urban scenes. This dataset encompasses up to 21 categories and covers three urban areas, totaling approximately $2.5km^2$, as illustrated in Figure 5.2. Our key contributions include:

- A semantic annotation framework for urban textured meshes that supports both triangle and texture annotations;
- A structure-aware interactive selection strategy that facilitates rapid extraction and matching of 2D and 3D structures;
- A new evaluation paradigm for assessing the performance of 2D and 3D interactive annotation methods, which effectively demonstrates their effectiveness in practical applications;
- A part-level benchmark dataset for semantic segmentation of urban textured meshes, with a detailed analysis of advanced methods on this dataset.

Figure 5.2: Our textured mesh dataset encompasses three major urban areas in Helsinki, covering a total area of approximately $2.5km^2$.

5.2. Related work

This section reviews research on interactive segmentation and annotation for both 2D images and 3D data and also discusses studies concerning benchmark datasets for 3D semantic urban scenes.

5.2.1. Interactive image segmentation

Early studies on interactive image segmentation are primarily categorized into four methods [Ramadan *et al.*, 2020]: region-growing, contour-based, graph-cut, and random walk. Region-growing methods start with initial seeds and merge similar areas based on consistency criteria, offering intuitive use and easy adaptability [Bischof and Adams, 1994; Fan and T. C. Lee, 2015; Ning *et al.*, 2010]; however, they require sufficient initial user inputs to ensure accuracy, especially in complex images. Contour-based methods leverage edge features and user inputs to precisely delineate object contours, but they demand significant user involvement and are time-intensive [Badshah and K. Chen, 2010; Bresson *et al.*, 2007; Mortensen and Barrett, 1998; T. N. A. Nguyen *et al.*, 2012]. Graph-cut methods, utilizing energy optimization within a Markov Random Field (MRF) framework, effectively distinguish between foreground and background by balancing pixel characteristics with spatial continuity [Blake *et al.*, 2004; Boykov and Jolly, 2001; Rother *et al.*, 2004; Tang *et al.*, 2013; Veksler, 2008]. Although robust and precise, these methods are computationally intensive and rely on initial user inputs. Random walk models, employing graph theory and probability, use local pixel correlations for accurate segmentation, yet struggle with efficiency in complex textured or blurred images [X. Dong *et al.*, 2016; Grady, 2006; T. H. Kim *et al.*, 2008]. Our proposed method integrates superpixel-based region-growing with graph-cut optimization for efficient and precise segmentation with minimal user input and computational resources.

Deep learning has significantly advanced interactive image segmentation (IIS). Convolutional Neural Networks (CNNs) are widely used in this field, and various models such as DEXTR [Maninis *et al.*, 2018], Deep GrabCut [Xu *et al.*, 2017], and DeepCut [Rajchl *et al.*, 2017] have enhanced performance over traditional models. These methods often use complex network architectures like ResNet-101 [K. He *et al.*, 2016] and SegFormer [Xie *et al.*, 2024], incorporating region, contour, or multi-scale information to optimize segmentation results. Interactive methods like BRS [Jang and C.-S. Kim, 2019] and f-BRS [Sofiiuk *et al.*, 2020] enhance segmentation by optimizing images or distance maps, while FCA-Net [Z. Lin *et al.*, 2020] emphasize the importance of the first click, as it is often crucial in determining the subject's position. Advanced models like FocalClick [X. Chen *et al.*, 2022] and the Segment Anything Model (SAM) [Kirillov *et al.*, 2023] demonstrate how various inputs, including clicks, text, or masks, can increase segmentation accuracy and flexibility. They often utilize sophisticated architectures such as Graph Convolutional Networks (GCN) [Acuna *et al.*, 2018; Castrejón *et al.*, 2017; Ling *et al.*, 2019] and Vision Transformers (ViT) [Kirillov *et al.*, 2023; Q. Liu *et al.*, 2023], alongside deep reinforcement learning [K. M. Lee *et al.*, 2018], to manage more intricate interactions and boost performance. Methods like RITM [Sofiiuk *et al.*, 2022] use iterative sampling strategies during training to enhance results. However, these techniques have drawbacks, primarily their poor performance with new categories not seen in the training set and their dependency on large datasets, which may restrict their practical use. Despite attempts to mitigate these issues through dataset expansion and data augmentation, their full integration into sectors like urban scene understanding remains under development. Our unsupervised, real-time

interactive textured mesh segmentation method offers substantial generalizability, contrasting with the limitations of deep learning approaches.

5.2.2. Interactive 3D annotation

Interactive 3D annotation involves various input data types, including images with camera parameters and depth maps, point clouds, textured meshes, and CAD models. Research methods in this field include purely manual interaction, graph-cut based approaches, region-based segmentation, and deep learning methods. Manual methods involve users selecting objects directly in point clouds or meshes using tools like brushes or lassos, which can be labor-intensive for large-scale data [Ibrahim *et al.*, 2021; Pütz *et al.*, 2021a]. Graph-cut methods rely on user interactions to distinguish between foreground and background in 3D data, using MRF and energy function optimizations [Jayaraman *et al.*, 2021; K. Liu and Boehm, 2014; Miksik *et al.*, 2015; Romanoni and Matteucci, 2018; Schmitz *et al.*, 2022; J. Valentin *et al.*, 2015]. These have been effectively applied in semantic annotation of point clouds, meshes, and CAD models, though their success depends heavily on the quality of user inputs. Region-based methods reduce interaction by merging similarly colored or geometric areas in the data, followed by detailed refinement [W. Gao *et al.*, 2021; D. T. Nguyen *et al.*, 2018; Steinlechner *et al.*, 2019], but may lead to increased costs due to over- or under-segmentation issues. Recently, deep learning methods have used advanced neural networks and attention mechanisms to streamline interactive 3D segmentation, providing real-time updates to prediction masks [Kontogianni *et al.*, 2023; Lang *et al.*, 2024; Yue *et al.*, 2024]. However, they require extensive training data and struggle with accuracy on new, unseen data. In contrast, our approach for interactive mesh annotation is unsupervised, requiring no prior sample labeling and independent of original imagery or camera parameters. Users make approximate selections in areas of interest, enabling precise component annotations. Additionally, our technique employs structure-aware matching to swiftly locate and annotate similar structures within the scene, improving both efficiency and accuracy.

5.2.3. Semantic 3D urban datasets

3D urban datasets are collected through various technologies such as Terrestrial Laser Scanning (TLS), Mobile Laser Scanning (MLS), Aerial Laser Scanning (ALS), and aerial or drone-mounted cameras. The product of semantic 3D urban data mainly consists of LiDAR point clouds [AHN, 2019; Behley *et al.*, 2019; Hackel *et al.*, 2017; Roynard *et al.*, 2018; Tan *et al.*, 2020; Varney *et al.*, 2020], photogrammetric point clouds [Can *et al.*, 2021; Q. Hu *et al.*, 2022; X. Li *et al.*, 2020; G. Yang *et al.*, 2023], textured meshes [Brostow *et al.*, 2009; J. Chen *et al.*, 2022a; W. Gao *et al.*, 2021], and synthetic data [M. Chen *et al.*, 2022; Q. Gao *et al.*, 2020; J. Zhou *et al.*, 2024]. Compared to indoor methods using RGBD cameras and structured light [Armeni *et al.*, 2017; Chang *et al.*, 2017; Dai *et al.*, 2017; Song *et al.*, 2015], these technologies provide

broader coverage of urban landscapes. However, datasets established by ground-based systems like TLS and MLS struggle to capture the upper parts of urban objects such as rooftops and tree canopies. Meanwhile, despite their extensive urban coverage, point cloud datasets from ALS and drones often miss details such as building facades and windows due to occlusion, light reflection, transmission issues, sensor resolution, and limitations in data processing methods. Although synthetic datasets can capture urban details and automatically generate semantic labels, their deviation from real scenes restricts their practical applicability. On the other hand, textured meshes from oblique photogrammetry offer more comprehensive detail capture and rich potential semantic information. Most existing 3D semantic urban datasets focus on specific urban objects, restricting their broader application. Although datasets like DublinCity [Zolanvari *et al.*, 2019] and Hessigheim3D [Kölle *et al.*, 2021] contain some information about object parts, their categories and scale are relatively small. This paper introduces the first large-scale, part-level semantic urban textured mesh benchmark dataset, designed to cover and depict urban environments' complexity thoroughly.

5.3. Overview

(a) (b) (c) (d)

Figure 5.3: Semantic textured mesh annotation framework: here, (a) and (b) demonstrate selections based on mesh faces. (c) and (d) illustrate pixel selections based on textured planar segments (i.e., the green highlighted area in the bottom left of the image (c)). In (a), the user employs a lasso tool (green trajectory) to select and extract protrusions (red faces). (b) shows the automatically matched protrusions on the same roof (red faces). In (c), a user click (red star) extracts the local region (red pixels). (d) shows the automatically matched regions (red pixels).

We aim to create ground truth data through interactive semantic annotation of large-scale urban textured meshes. Our input data consists of triangular meshes and their texture images, while our output includes triangular meshes with semantic labels and semantic texture masks. Our labeling targets include mesh triangles and texture pixels to achieve part-level semantic annotation. Since manually annotating urban textured meshes containing a large amount of semantic information is very time-consuming, we propose a semi-automatic annotation framework. This framework supports interactive selections based on local features and global structure matching (see Figure 5.3). Our annotation framework includes two main modules: face-based annotation and texture-based annotation.

The face-based annotation aims to assign a semantic label to each triangle through the user's interactive selection. We provide basic interactive selection tools such as brushes, lassos, expand/reduce, semantic class selection, and inverse selection options. To enhance efficiency and reduce the need for user interactions, we introduced interactive protrusion extraction and a structure-aware matching algorithm. Specifically, we first overs-segment the input mesh through region growing to obtain planar segments. These segments enable quick selection of large planar areas, while protrusions can be semi-automatically selected based on user interactions and local geometric features. Due to the repetitive nature of urban structures, we calculate structural features from user-selected planar segments and protrusions to enable global matching, thereby facilitating rapid selection and annotation.

The objective of interactive annotation for mesh textures is to enable pixel-level semantic labeling of textures on planar segments. We also offer basic selection tools such as brushes, rectangles, circles, polygon selections, lassos, semantic class selection, and inverse selection options. We developed a method that extracts components using a local expansion strategy coupled with a structure-aware matching algorithm to enhance semantic annotation efficiency. Initially, we over-segment texture images derived from planar segments into superpixels, then employ user clicks to acquire seeds and perform local expansions based on color distribution characteristics. We utilize the structural features of the expanded areas or the user's drawn shape for global matching, recommending regions with high similarity for users to assign semantic labels.

We reduce user input and enhance semantic annotation efficiency for large-scale urban meshes by computing structural features from interactively extracted components and applying structure-aware matching. Additionally, our proposed annotation framework is unsupervised and real-time, and our structure-aware matching method is invariant to translation, rotation, and scale, adding robustness to the process.

5.4. Face-based annotation

Large-scale urban meshes consist of numerous triangular faces. Manually selecting and labeling each face is time-consuming. We aim to efficiently select and label geometric structures within these meshes through user interactions. Our approach comprises two key components: protrusion extraction and structure-aware matching. We assume that all geometric structures in the urban mesh are composed of planar segments and protrusions, with components sharing similar geometric features typically bearing identical semantic labels. As shown in Figure 5.4, planar segments refer to areas made up of geometrically flat or nearly flat faces, typically forming the main surfaces of buildings, grounds, or other structures; protrusions refer to 3D geometric bodies that are significantly raised from the main planes, such as balconies, chimneys, cars, or trees.

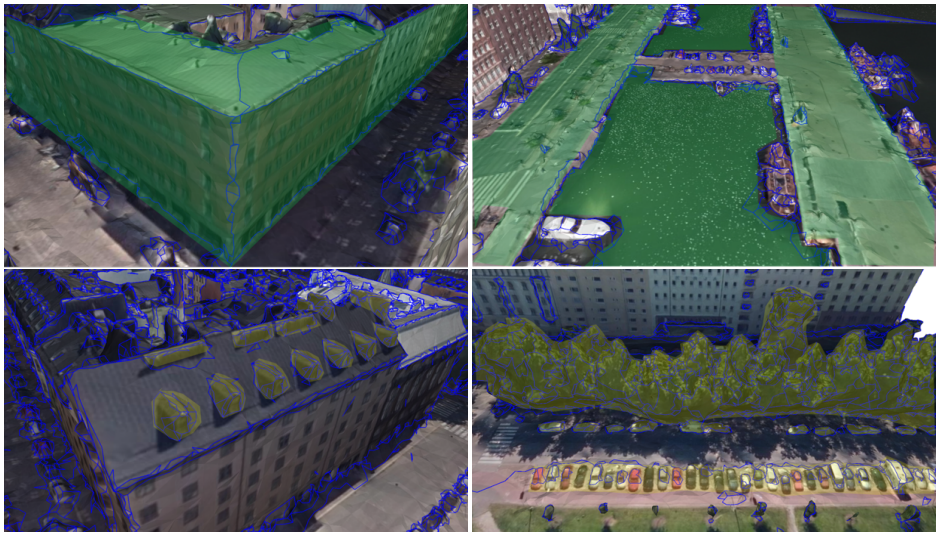


Figure 5.4: Examples of planar segments and protrusions: planar segments are highlighted in green in the upper images, with boundaries delineated by blue lines; the lower images display protrusions, marked in yellow, which significantly bulge from the main plane and can be broken down into several planar segments.

5.4.1. Protrusion extraction

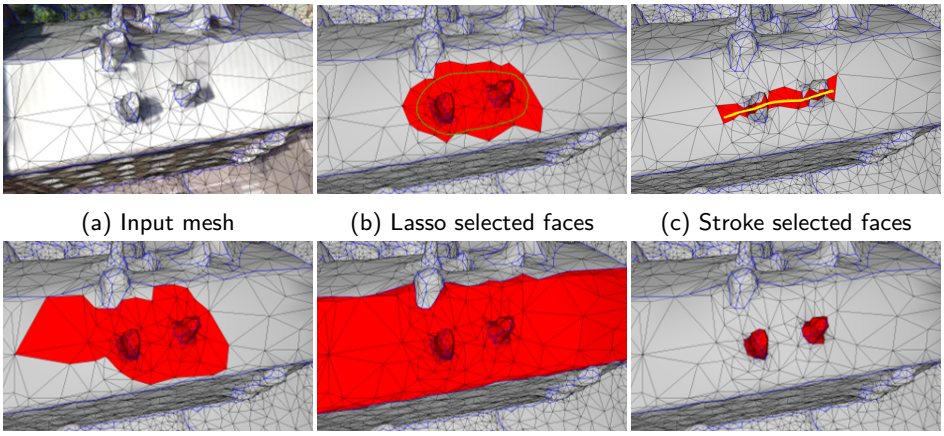
First, we over-segment the input mesh to derive geometric structures represented by planar segments. Thus, larger continuous planar areas like the ground and building facades can be grouped together. Although these segments facilitate direct selection and labeling and reduce interactions compared to selecting individual triangles [W. Gao *et al.*, 2021], it still poses three challenges:

- Non-planar areas like trees are represented by several small planar segments, necessitating significant user judgment and selection.
- Sharp features suffer from under-segmentation, mainly due to the typically over-smoothed problem in urban textured meshes. Users may need to re-select at the face level to correct these errors.
- Small-scale structures are divided into multiple smaller planar segments, such as chimneys and cars, requiring frequent user interactions for accurate selection.

These challenges are inherent in all over-segmentation algorithms for triangular meshes [W. Gao *et al.*, 2023]. To overcome these, we introduce an interactive protrusion structure extraction method that allows users to quickly and efficiently select protrusions at the face level with few interactions, maintaining accurate boundaries.

The core idea of our algorithm is to identify protrusions that are not part of the current

support plane in a manner akin to how interactive image segmentation differentiates between the background (analogous to support planes in our context) and the foreground (corresponding to protrusions) [Rother *et al.*, 2004]. Using tools such as a lasso or a stroke, users traverse and establish a candidate set of faces, which are subsequently subjected to binary labeling. Unlike traditional interactive segmentation methods that necessitate separate inputs for foreground and background, our approach requires only one user input to extract a collection of protrusion faces automatically. Our method consists of three main steps: interactive selection, binary labeling, and boundary refinement.



(a) Input mesh (b) Lasso selected faces (c) Stroke selected faces

(d) Stroke expanded faces (e) Candidate faces (f) Extracted protrusions

Figure 5.5: Interactive selection diagram of protrusions: in (b), the green trajectory represents the user's lasso selection, while in (c), the yellow trajectory denotes the user's stroke selection. The red area visible from (b) to (f) indicates the selected faces. Additionally, the blue lines in all images mark the boundaries of the planar segments, and the black lines represent the edges of the triangles.

1) Interactive selection. Interactive selection aims to produce candidate faces for binary classification from user inputs. Our system recognizes two input modes: lasso and stroke. The lasso tool is primarily designed for selecting small-scale protrusions, such as chimneys and cars, while the stroke tool is intended for easily selecting larger or more complex protrusions, such as trees, entire buildings, and large cruise ships. We analyze the user's drawn path to compute geometric metrics, which help discern whether the input is a lasso or a stroke. This metric is derived from the ratio of the distance between the contour endpoints to the diagonal length of the contour's bounding box, taking a ratio of less than 0.7 as a lasso, otherwise as a stroke.

Given the fundamental differences between the two selection modes, we implement distinct processing strategies for each. We aim to ensure that both modes extract the same candidate faces, as identical candidate faces will yield the same protrusions. If identified as a lasso, all the planar segments corresponding to the faces within the lasso

are selected as candidate faces (see Figure 5.5b). If a stroke is identified, expansion is applied to the neighboring faces of those traversed by the stroke's trajectory. This involves calculating the distance from the centers of the faces to the trajectory. The expansion limit is set based on the farthest distance from the segments corresponding to the faces crossed by the trajectory to the trajectory itself (see Figure 5.5c and Figure 5.5d). Then, we select the expanded planar segments as candidate faces (consistent with Figure 5.5e).

2) Binary labeling. To select surfaces belonging to protrusions from the interactively chosen candidate faces, we formulate this as a binary labeling problem: $l^f = \{\text{support plane, protrusion}\}$. We start by arranging the planar segments associated with the candidate faces $\{f_i\}_{i=1...m}$ in descending order based on their area, selecting the largest segment to serve as the reference. Next, guided by a predetermined structural scale ratio $s$fixed at $s = 2$ throughout this studywe identify suitable planar segments $\{P_k^f\}_{k=1...n}$ by comparing their sizes with that of the reference segment. These selected segments are then utilized to construct support surfaces. Subsequently, we build a dual graph $\mathcal{G}^f = \{\mathcal{V}^f, \xi^f\}$ for all candidate faces $f = \{f_i\}$, with each face f_i representing a node in the graph and connected to adjacent faces by edges.

5

Next, we define the data term D^f to evaluate the likelihood that face f_i belongs to a protrusion, given by:

$$D^f(l_i^f) = \eta \times \begin{cases} p_i & \text{if } l_i^f = \text{support plane} \\ 1 - p_i & \text{if } l_i^f = \text{protrusion} \end{cases}, \tag{5.1}$$

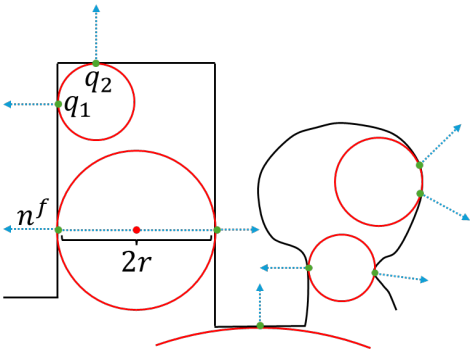
where η modulates sensitivity to various geometric characteristics, such as areas with high noise or inconsistent data quality. Specifically, for outdoor scenes discussed in this paper, η is set to 1. p_i denotes the protrusion score of face f_i. We observed that faces further from the support plane or those with a larger angle to the normal of the support plane are more likely to be protrusions. Therefore, by measuring the distance and angle from face f_i to support plane P_k^f, we define protrusion score $p_i = d_i + \omega_i \cdot \theta_i$, where $d_i = \max_{t\in\{0,1,2\}} \left(\text{dist}(v_{t,i}, P_k^f)\right)$ is the maximum Euclidean distance from the vertices v_t of the face f_i to the support plane P_k^f. The angle weight θ_i is calculated by measuring the angle $\hat{\theta}_i = \cos^{-1}(\mathbf{n}_i \cdot \mathbf{n}_k)$ between the normal $\mathbf{n}_i$ of face f_i and the normal $\mathbf{n}_k$ of support planar segment P_k^f, defined as:

$$\theta_i = \frac{\min(\hat{\theta}_i, 180° - \hat{\theta}_i)}{90°}. \tag{5.2}$$

The distance weight ω_i is defined as follows:

$$\omega_i = \begin{cases} 1 & \text{if } d_i > 1 \\ 1 - d_i & \text{otherwise} \end{cases}, \tag{5.3}$$

indicating that a smaller d_i has a greater influence on the angle.



To measure the geometric similarity between adjacent faces, we utilize an internal shrinking sphere algorithm derived from the 3D medial axis transform to compute the sphere radii for each face [Peters, 2018; Sherbrooke *et al.*, 1996]. In urban mesh scenarios, larger shrinking balls typically correspond to major geometric structures such as the ground or main surfaces of buildings, whereas smaller balls indicate sharp structures or protrusions (as shown in the cross-sectional diagram). Consequently, the size of these spheres can indirectly reflect the local structural scale, suggesting that adjacent faces within the same geometric structure should have similar radii. Our smoothness term, V_f, is defined as follows:

$$V^f(l_i^f, l_j^f) = R_{i,j} \cdot 1_{\{l_i^f \neq l_j^f\}}, \tag{5.4}$$

where $1_{\{\cdot\}}$ represents the characteristic function. Local geometric similarity is denoted by

$$R_{i,j} = 1 - \min\left(1, \frac{2|r_i - r_j|}{z_{max} - z_{min}}\right), \tag{5.5}$$

where z_{max} and z_{min} correspond to the range of z-values of all vertices in $\{f_i\}$. The mesh shrinking ball radius can be derived as

$$r = \frac{\|q_1 - q_2\|^2}{2(\mathbf{n}^f \cdot (q_1 - q_2))}, \tag{5.6}$$

where r refers to the radius r_i or r_j, and $\mathbf{n}^f$ to the normal $\mathbf{n}_i$ or $\mathbf{n}_j$, of the respective faces f_i or f_j; q_1 and q_2 are the tangent points on the faces.

We finally define the objective function as a combination of the data term and smoothness term, as shown below:

$$E^f(l^f) = \sum_{i \in \nu^f} D^f(l_i^f) + \lambda^f \cdot \sum_{\{i,j\} \in \xi^f} V^f(l_i^f, l_j^f), \tag{5.7}$$

where l^f represents the labels for the candidate faces, and λ^f is utilized to adjust the weight of the smoothing term, thus regulating the impact of geometric similarity in the surrounding area. The objective function is efficiently minimized using a graph-cut algorithm [Boykov *et al.*, 2001].

3) Boundary refinement. To address the issue of irregular boundaries (i.e., jagged edges) resulting from the shapes of triangles during the selection process, we employ

morphological operations to refine the protrusion boundaries, comprising two phases: erosion and dilation. Erosion primarily aims to eliminate isolated faces at the boundaries. The criterion is that if a face is adjacent to other faces and one or none of these is selected, the face will be removed. Dilation, conversely, aims to identify unselected faces closely surrounded by selected ones. A face is marked as selected if it has two or more selected neighbors. Erosion and dilation are in two separate loops until no more faces can be updated. Figure 5.6 illustrates the visual impact of boundary refinement before and after.

(a) Without boundary refinement (b) With boundary refinement

Figure 5.6: Examples of protrusion boundary refinement

5.4.2. 3D structure-aware matching

Given the presence of repetitive structures in the scene, we believe that employing structural matching can significantly reduce the frequency of annotations. We aim to extract features from the user-selected faces and match them with similar structures within the scene. We propose two structure-aware matching strategies: segment matching and protrusion matching.

Segment matching. Planar segments typically cover flat, continuous surface areas exhibiting significant two-dimensional expansiveness. Based on this characteristic, We take the user-selected segment $P^{(t)}$ as a template to match with the set of candidate segments $\{P_k^{(c)}\}_{k=1...n'}$ in the scene. Our matching strategy is based on the feature similarity between the template $P^{(t)}$ and the candidate segments $P_k^{(c)}$. We describe this similarity by constructing a feature vector $\mathbf{F}^{(\mathbf{seg})}$, which includes:

- Geometric homogeneity: The area of a planar segment reflects its local geometric homogeneity. Typically, planar segments that are geometrically homogeneous exhibit minor differences in area. This is calculated by $\Delta A^{(seg)} = \frac{|area^{(c)} - area^{(t)}|}{area^{(t)}}$, where $area^{(c)}$ is the area of the candidate segment $P_k^{(c)}$, and $area^{(t)}$ is the area of the template segment $P^{(t)}$.
- Spatial distribution: We argue that similar components usually have a similar

vertical distribution in urban scenes. We assess this similarity by comparing the weighted average heights of the segments, calculated as $\Delta H^{(seg)} = \left| \frac{\sum_{i=1}^{m'} z_i \cdot a_i}{area^{(c)}} - \frac{\sum_{j=1}^{m''} z_j \cdot a_j}{area^{(t)}} \right|$. Here, z_i and a_i represent the z-coordinate and area of each face f_i in the candidate segment $P_k^{(c)}$, respectively, while z_j and a_j denote those in the template segment $P^{(t)}$.

- Spatial orientation: In urban environments, man-made structures like facades typically share similar vertical orientations. We quantify this similarity using the difference in vertical orientation, denoted as $\Delta R^{(seg)} = \left| \left| \mathbf{n}^{(c)} \cdot \mathbf{n}^{(z)} \right| - \left| \mathbf{n}^{(t)} \cdot \mathbf{n}^{(z)} \right| \right|$, where $\mathbf{n}^{(z)}$ is the unit vector along the z-axis, and $\mathbf{n}^{(c)}$ and $\mathbf{n}^{(t)}$ are the normals of segments $P_k^{(c)}$ and $P^{(t)}$, respectively.

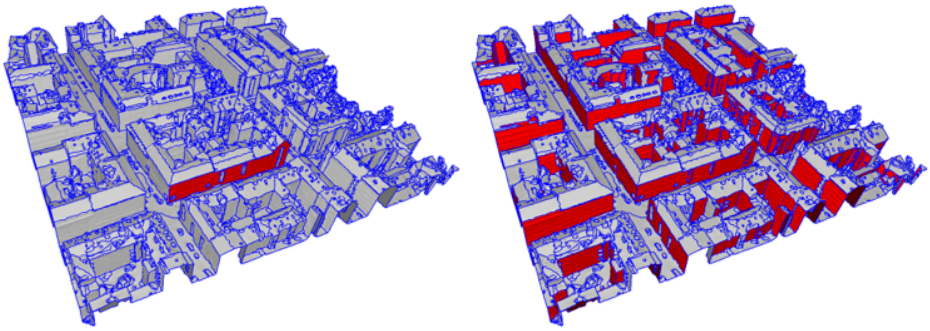
- Shape sphericity: Because we set distance and angle thresholds when generating planar segments through segmentation, the planar segments exhibit some distribution variation within the threshold range. This distribution variation can be represented by sphericity. We believe the sphericity of the segment shape can be used to assess their similarity. We reflect this through the calculation of eigen-based sphericity $\Delta S^{(seg)} = \left| \lambda_3^{P^{(c)}} / \lambda_1^{P^{(c)}} - \lambda_3^{P^{(t)}} / \lambda_1^{P^{(t)}} \right|$ computed based on the eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$ from triangle vertices of the segment $P_k^{(c)}$ and $P^{(t)}$.

- Photometric coherence: While vegetation and other urban components may lack notable geometric similarity, they often resemble each other in color. We measure this resemblance using the average CIELAB [M. R. Luo *et al.*, 2001] color distance $\Delta C^{(seg)} = \| \mathbf{C}_k^{(c)} - \mathbf{C}^{(t)} \|$, and the greenness [McKinnon and Hoff, 2017] difference $\Delta G^{(seg)} = \left| G_k^{(c)} - G^{(t)} \right|$. Here, $\mathbf{C}_k^{(c)}$ and $\mathbf{C}^{(t)}$ denote the CIELAB coordinate vectors of segments $P_k^{(c)}$ and $P^{(t)}$, respectively, while $G_k^{(c)}$ and $G^{(t)}$ indicate their greenness values. Note that incorporating color information in matching is optional and can be adjusted based on the presence of vegetation or similar characteristics in the matching category.

Based on these characteristics, a match between $P_k^{(c)}$ and $P^{(t)}$ is indicated when $\| \mathbf{F}^{(\mathbf{seg})} \| < \epsilon^{(seg)}$. The value of $\epsilon^{(seg)}$ depends on the user interactions and the quality of the input data. An example of segment matching in a large-scale urban scene is shown in Figure 5.7.

Protrusion matching. Protrusions, with their complex 3D shapes, differ significantly in features from planar segments. In light of this, our goal in this step is to use the protrusions extracted by the user in Section 4.1 as templates to match with other similar structures within the scene. Our matching process is divided into (1) seed matching and expansion and (2) structural matching.

1) Seed matching and expansion. This step aims to generate candidate regions through seed matching using template segments and to extract protrusions. We start by decomposing the template protrusion's face collection $f^{(t)} = \{f_i^{(t)}\}_{i=1...m^{(t)}}$ into several



(a) User-selected segment (b) Matched segments

Figure 5.7: Example of segment matching: the red area indicates the selected segment, and the blue lines represent the boundaries of the planar segment.

planar segments. These segments are then matched with others in the scene to identify all qualifying initial seed segments $\{P_k^{(e)}\}_{k=1...n^{(e)}}$, using the strategy from segment matching.

Next, we expand these seed segments to their neighboring segments $\{P_k^{(a)}\}_{k=1...n^{(a)}}$ to generate candidate regions. Spatial and segment scale constraints are applied to limit the scope and size of these expansions, respectively. The spatial constraint is given by the following formula:

$$\left\| O_k^{(e)} - O_j^{(a)} \right\| < \sqrt{s} \cdot \max_i \left\| O^{(t)} - O_i^{(t)} \right\|, \tag{5.8}$$

where $O_k^{(e)}$ and $O_j^{(e)}$ represent the centers of the seed segment $P_k^{(e)}$ and the face $f_j^{(a)}$ in its neighboring segment $P_k^{(a)}$, respectively. $O^{(t)}$ and $O_i^{(t)}$ denote the centers of the template face collection $f^{(t)}$ and its individual face $f_i^{(t)}$, with s as the structural scale (same as in subsection 5.4.1). The segment scale constraint is defined by the following formula:

$$\frac{area_k^{(a)}}{area_k^{(e)}} < s \cdot \frac{\max_j area_j^{(t)}}{\min_j area_j^{(t)}}, \tag{5.9}$$

where $area_k^{(e)}$ and $area_k^{(a)}$ represent the areas of the seed segment $P_k^{(e)}$ and its neighboring segment $P_k^{(a)}$, respectively; $area_j^{(t)}$ denotes the area of the planar segment j that is used to extract template protrusion.

Additionally, topology constraints based on adjacency relationships are provided to focus on areas of interest for annotation. We calculate the planar segments of the template support surface as the neighborhood search space. For instance, topological constraints can confine the search space to the planar segments of the current facade for annotations of facade installations. Following seed expansion to generate candidate regions, the method from subsection 5.4.1 is employed to extract the candidate

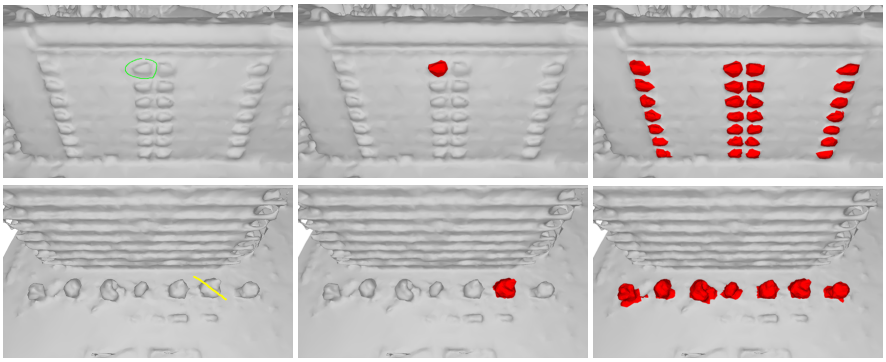
protrusion face collection $f^{(c)}$ and to structurally match the extracted protrusions with the template protrusions.

2) Structural matching. The purpose of this step is to further filter the protrusions extracted from step 1), retaining only those that closely resemble the template protrusion. Similar to segment matching, our strategy is based on the feature similarities between the template and candidate protrusions, and we construct a feature vector $\mathbf{F^{(str)}}$ to describe these characteristics, which includes:

- Spatial compactness: We quantify the spatial compactness of a protrusion by measuring its volume. Structurally similar protrusions are expected to exhibit high spatial compactness similarity, which is defined as $\Delta V^{(str)} = \left| \frac{vol^{(c)}}{vol^{(c)}_{box}} - \frac{vol^{(t)}}{vol^{(t)}_{box}} \right|$, where $vol^{(c)}$ and $vol^{(c)}_{box}$ represent the volume of $f^{(c)}$ and its bounding box volume, respectively, $vol^{(t)}$ and $vol^{(t)}_{box}$ are the corresponding values for $f^{(t)}$.

- Surface complexity: We consider that more complex 3D shapes typically decompose into numerous planar segments. We characterize the similarity of surface complexity by the ratio of the number of planar segments of the template and candidate protrusions, which is shown as $\Delta N^{(str)} = \left(\frac{\max(n^{(t)}, n^{(c)})}{\min(n^{(t)}, n^{(c)})} \right)^{\mu}$, where $n^{(t)}$ and $n^{(c)}$ respectively represent the number of planar segments for the template and candidate protrusions, and $\mu = \min(n^{(t)}, n^{(c)})$.

- Structural features: Measuring the similarity of protrusions involves comparing their structural features through eigenvalue analysis, such as linearity = $(\lambda_1 - \lambda_2)/\lambda_1$, planarity = $(\lambda_2 - \lambda_3)/\lambda_1$, and sphericity = λ_3/λ_1. We determine structural similarity by the L1 feature distance, including differences in linearity $\Delta L^{(str)}$, planarity $\Delta P^{(str)}$, and sphericity $\Delta S^{(str)}$.



(a) User interaction (b) Extracted protrusion (c) Matched protrusions

Figure 5.8: Example of protrusion matching: the green trajectory indicates the user's lasso selection, the yellow trajectory represents the user's stroke selection, and the red faces denote the selected area.

The similarity of photometric coherence $\Delta C^{(str)}$ is based on CIELAB [M. R. Luo *et al.*, 2001] calculations, an optional feature for users. The matching result between the template and candidate protrusions depends on $\|\mathbf{F}^{(\mathbf{str})}\| < \epsilon^{(str)}$, where $\epsilon^{(str)}$ is also determined by user interaction and data quality. We present two examples of protrusion matching in Figure 5.8.

5.5. Texture-based annotation

Due to resolution limitations, triangle meshes often fail to capture detailed planar components and their boundaries, such as doors, windows, and road markings. Consequently, semantic annotations based on triangle faces fail to describe these intricate structures adequately. However, mesh textures can more effectively capture these geometric details. Compared to annotating original multi-view images, choosing mesh textures as the annotation subject avoids redundant annotation and label discontinuities caused by image projections and more accurately represents the structure in 3D space, simplifying the annotation process.

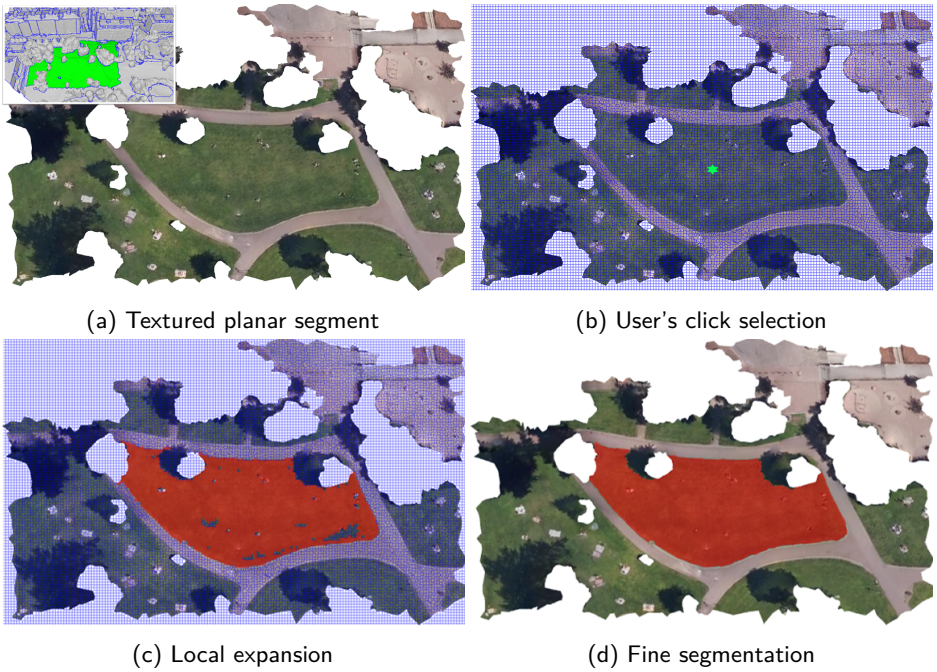
Compared to annotating 2D images, directly annotating the whole continuous texture images on the surface of 3D models is relatively difficult. This is mainly because the texture coordinates between the triangular faces are discontinuous and involve much computation for 3D to 2D space conversion. And the direct annotation of individual triangular faces lacks context information. Textured segments formed by texture boundaries are unsuitable for texture annotation due to the presence of texture seams. These segments have relatively lower geometric accuracy compared to planar segmentation on meshes. Additionally, without the original images, it is impossible to arbitrarily re-segment or merge them.

Therefore, we propose a mesh texture annotation strategy based on planar segments. Our strategy permits users to flexibly split or merge planar segments to adjust the size of the texture annotation areas. Moreover, using basic selection tools like strokes, shapes, and lassos for direct pixel-level texture annotation remains time-consuming. To address this, we have developed an efficient interactive semantic annotation algorithm that leverages local region extraction and matching. Our selection strategy mainly includes local region extraction and 2D structure-aware matching for selected textured planar segments. Our approach assumes that semantic components within textured planar segments consist of multiple superpixels with uniform local color features, and components sharing the same semantic label display pronounced similarities in color and geometric structure.

5.5.1. Local region extraction

We aim to capture the entire region by allowing users to click on the area of interest. Unlike traditional interactive segmentation methods, our strategy does not require users

to select negative samples (i.e., background); it operates solely through clicks on positive samples (i.e., foreground). Our method can be divided into two main steps: 1) Local expansion and 2) Fine segmentation.



(a) Textured planar segment (b) User's click selection

(c) Local expansion (d) Fine segmentation

Figure 5.9: Example of local region extraction based on textured planar segments: in (a), the green area in the upper left represents the textured planar segment selected by the user; the green star in (b) indicates the user's click input. Blue lines in (b) and (c) delineate the boundaries of superpixels; the red areas in (c) and (d) highlight the selected pixels.

1) Local expansion. Our aim is to expand the seed superpixels selected by user clicks to encompass the entire area of interest. Initially, we apply the Simple Linear Iterative Clustering (SLIC) [Achanta *et al.*, 2012] to the input textured planar segments to generate superpixels that exhibit homogeneous color characteristics. Subsequently, we establish a topological adjacency matrix for these superpixels to define their spatial relationships. Seed superpixels are obtained via user clicks, and their first-order neighborhood is used to establish a local adjacency graph $\mathcal{G}^s = \{v^s, \xi^s\}$. Each superpixel represents a node in the graph and is connected through edges to adjacent superpixels. We transform the local expansion process into a binary labeling problem for adjacent superpixels: $l^s = \{\text{similar}, \text{non-similar}\}$, and use the adjacency graph to initialize an MRF. The labeling outcomes determine which adjacent superpixels are similar to the first seed S_0, and similar superpixels are then added to the seed queue $\{S_i\}_{i=1...n}$. Like the region-growing algorithm, a local adjacency graph is re-established for the expanded superpixels to conduct the next binary labeling. This expansion process continues until the seed queue

is empty. It is important to note that during the entire expansion process, the seed S_0 consistently acts as the initial reference for comparisons in each expansion, rather than the superpixels in the seed queue $\{S_i\}_{i=1...n}$. The local binary labeling problem is addressed by defining the data and smoothness terms.

We define the data term D^s by measuring the similarity of the pixel value distribution between the seed superpixel S_i and its adjacent superpixels $\{S_j\}_{j=1...n'}$. The formula is given as follows:

$$D^s(l_j^s) = \alpha \times \begin{cases} 1 - w_j & \text{if } l_j = \text{non-similar} \\ w_j & \text{if } l_j = \text{similar} \end{cases}, \tag{5.10}$$

where the weight α controls data fidelity and is pre-calculated using the local color difference (CIEDE2000) [M. R. Luo *et al.*, 2001] based on the input textured planar segment. The term w_j, representing the average Wasserstein distance, is used to quantify the similarity between the Gaussian mixture distributions of two superpixels, denoted as $w_j = \frac{1}{K}\sum_{k=1}^{K} W(G_k(S_j), G_k(S_0))$. Here, $K = 3$ represents the image's three RGB channels, and $W(G_k(S_j), G_k(S_0))$ represents the Wasserstein distance between superpixels in the k-th channel. G_k denotes the Gaussian mixture model for the k-th channel, defined as $G_k(S) = \sum_{m=1}^{M} \pi_{km} \mathcal{N}(x; \mu_{km}, \Sigma_{km})$, where S represents the superpixel S_0 or S_j, and x is a pixel sample point of S, and M is the number of components in GMM ($M = 5$ in all experiments in this paper). $\mathcal{N}(s; \mu, \Sigma)$ denotes the multivariate normal distribution, with μ representing the mean for superpixels S_0 or S_j, and Σ as their respective covariance matrices. To acquire additional contextual information for the seed superpixel S_0, the samples for computing $G_k(S_0)$ are taken from its first-order neighborhood superpixels. Samples for $G_k(S_j)$ are derived from the pixels within S_j.

To measure the similarity between neighboring pixels, we have formulated a smoothness term V^s based on the color differences among neighboring superpixels, defined as follows:

$$V^s(l_j^s, l_{j'}^s) = H_{j,j'} \cdot 1_{\{l_j^s \neq l_{j'}^s\}}, \tag{5.11}$$

where $1_{\{\cdot\}}$ represents the characteristic function. Local color similarity is defined as $H_{j,j'} = \sum_{j,j' \in \{S_j\}} |\rho_j - \rho_{j'}|$, where $\rho_j = \Delta E_{00}(U_0, U_j)$ is the color distance (i.e., CIEDE2000 [M. R. Luo *et al.*, 2001]) from the superpixel S_j to its seed S_0. To more accurately capture the intrinsic structure and variability within superpixels' color distributions, we employ a GMM to compute the average Lab color, represented by $U = \sum_{m=1}^{M} \pi_m \mu_m$, where U could represent U_0 or U_j, with π_m as the mixing weight and μ_m as the mean for the m-th Gaussian component in Lab color space. Additionally, seed samples for U_0 are taken from its first-order neighborhood, whereas samples for U_j come from its own pixels.

The energy function E^s is defined as the sum of the data term D^s and the smoothness term V^s, as follows:

$$E^s(l^s) = \sum_{j \in \nu^s} D^s(l_j^s) + \lambda^s \cdot \sum_{\{j,j'\} \in \xi^s} V^s(l_j^s, l_{j'}^s), \tag{5.12}$$

where l^s is the label of superpixels, and λ^s is used to adjust the weight of the smoothness term to control the impact of neighborhood color similarity. The objective function is solved using graph cuts [Boykov *et al.*, 2001].

2) Fine segmentation. Given that coarse segmentation results from superpixel expansion are influenced by superpixel resolution and user inputs, these results tend to be rough at object boundaries. Thus, this step aims to refine these coarse results. Our approach, akin to GrabCut [Rother *et al.*, 2004], employs foreground and background samples for detailed pixel-level segmentation. However, GrabCut uses a user-drawn bounding box to identify the outside region as background, which may fail for multi-target segmentation within an image. Therefore, our method requires only the coarse results from the previous step as foreground samples, with background samples generated automatically via a buffer area around the coarse results. Specifically, we use the buffer area extending beyond the optimal bounding box of the coarsely segmented object to produce background samples for detailed segmentation.

5.5.2. 2D structure-aware matching

For texture images based on planar segments, there are still a large number of repetitive structures, such as windows and road markings. To improve annotation efficiency, interactive sessions can be reduced by adopting fast matching techniques based on structural awareness. We propose two matching strategies based on structural awareness: template matching and region matching.

Template matching. Template matching uses user-defined template structures to find and select similar structures in the current textured planar segment. This template can be a user-generated local region or any arbitrary shape drawn by the user. We employ a common image template traversal strategy for matching, i.e., Normalized Cross-Correlation (NCC) [Briechle and Hanebeck, 2001], which efficiently identifies potential candidate areas. To filter matching regions, we utilized an adaptive thresholding strategy [Bradley and Roth, 2007] based on fine-tuned constants. Once the matching structures are identified, users can select them using the template shape or apply the detailed segmentation method described in subsection 5.5.1. However, this strategy performs poorly when dealing with rotation and scale changes, especially in 3D urban scenes facing targets with different orientations, such as zebra crossings and lane lines, where this limitation is particularly evident.

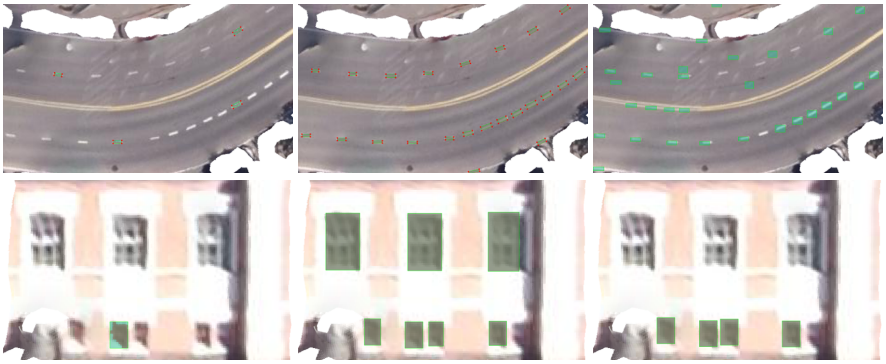
Region matching. Our objective is to extract structural features from the region $R^{(t)}$ created by local expansion and match them to similar regions within the textured planar segments. We first calculate the GMM $G_k(R^{(t)})$ of the overall region for the set of superpixels generated through local expansion selected by the user. Next, we use the Wasserstein distance based on these GMMs to filter candidate superpixels. The criterion is defined by the formula: $W(G_k(R^{(t)}), G_k(S_i^{(c)})) < \epsilon^{seed}$, where ϵ^{seed} represents the user-adjustable seed matching threshold, and $G_k(S_i^{(c)})$ represents the GMM model for candidate superpixel $S_i^{(c)}$. We then extract a collection of candidate regions $\{R_i^{(c)}\}_{i=1...m^{(c)}}$ from the qualified superpixels using the method described in subsection 5.5.1, and calculate their similarity to the template region to form a feature vector $\mathbf{F^{(reg)}}$, including:

- Shape index: To differentiate the shape similarities across regions, we utilize a shape index reflecting an object's elongation or flatness. Defined as $r = \frac{\min(w,h)}{\max(w,h)}$, where w and h represent the width and height of the object's bounding box, respectively. Specifically, we define the formula for calculating the similarity of shape indices between regions as $\Delta I^{(reg)} = |r_c - r_t|$, where r represents the ratio r_c of the candidate region or r_t of the template region.

- Shape regularity: We consider the regularity of a region's shape also to calculate the similarity between areas. Similar to structural matching, to demonstrate the complexity of a shape by assessing how well it fills its bounding box, we describe the shape's regularity using compactness, defined as: $\Delta A^{(reg)} = \left| \frac{area^{(c)}}{area^{(c)}_{box}} - \frac{area^{(t)}}{area^{(t)}_{box}} \right|$, where $area^{(c)}$ and $area^{(t)}$ represent the area of the candidate and template regions, respectively, and $area^{(c)}_{box}$ and $area^{(t)}_{box}$ are the areas of their bounding boxes.

- Contextual features: The contextual information within similar regions should exhibit similar characteristics, particularly in their internal and external color distributions. We evaluate these differences using the Wasserstein distance, calculated as follows: $\Delta D^{(reg)} = \left| W(G_k(R_{in}^{(c)}), G_k(R_{out}^{(c)})) - W(G_k(R_{in}^{(t)}), G_k(R_{out}^{(t)})) \right|$, where $R_{in}^{(c)}$ and $R_{out}^{(c)}$ denote the interior and exterior pixel collections of the candidate region, respectively, and $R_{in}^{(t)}$ and $R_{out}^{(t)}$ for the template region. The external region R_{out} includes pixels covered but not selected during local expansion in subsection 5.5.1.

Based on the above features, we use the formula $\mathbf{F^{(reg)}} < \epsilon^{reg}$ to filter the matching areas that meet the criteria, where the value of ϵ^{reg} depends on user input and image resolution. Additionally, we introduce a scaling range based on the quantity of template pixels, expressed as $s^{(range)} \in \left[\frac{N(R^{(t)})}{s^{(reg)}}, s^{(reg)} \cdot N(R^{(t)}) \right]$, to constrain the scale of the matching regions. Here, N denotes the number of pixels within the region, and the scaling threshold $s^{(reg)}$ is set to 2 in all experiments. Notably, our method offers rotational and scale invariance, unlike NCC-based template matching [Briechle and Hanebeck, 2001] (see Figure 5.10). This allows it to match targets within textured planar segments that vary in orientation and size.



(a) User extracted regions (b) Our matched regions (c) NCC matched regions

Figure 5.10: Example of region matching: the extracted regions are highlighted with green bounding boxes; the top images display optimal bounding boxes, while the bottom image features vertically aligned bounding boxes. Compared to methods based on NCC [Briechle and Hanebeck, 2001], it is evident from the top images that our method possesses rotational invariance, while the bottom images demonstrate its scale invariance.

5.6. Experimental results

In this section, we detail the part-level semantic textured mesh dataset we have developed and discuss the performance of state-of-the-art 3D semantic segmentation methods on this dataset. Additionally, we evaluate the effectiveness of our annotation tools and discuss their application domains and limitations.

5.6.1. Benchmark dataset

We use the textured mesh of Helsinki city [Helsinki, 2019] as our input, serving as a benchmark dataset for part-level semantic urban meshes. The textured meshes were generated using the commercial software ContextCapture [BENTLEY, 2016], reconstructed from oblique aerial imagery with a ground sampling distance of approximately $7.5cm$ and 2D vector maps. Detailed semantic annotations were conducted on three representative areas in central Helsinki as shown in Figure 5.2, comprising 40 tiles of $62,500m^2$ each, covering a total area of approximately $2.5km^2$.

In the semantic annotation process, we defined two label types: face and pixel labels. Face labels assign a semantic class to each face of the triangle mesh, encompassing 13 distinct classes listed as follows:

- *unclassified*: Includes elements and objects that do not fit into any specific predefined categories.
- *terrain*: Includes elements such as roads, cycle lanes, sidewalks, bridges, low

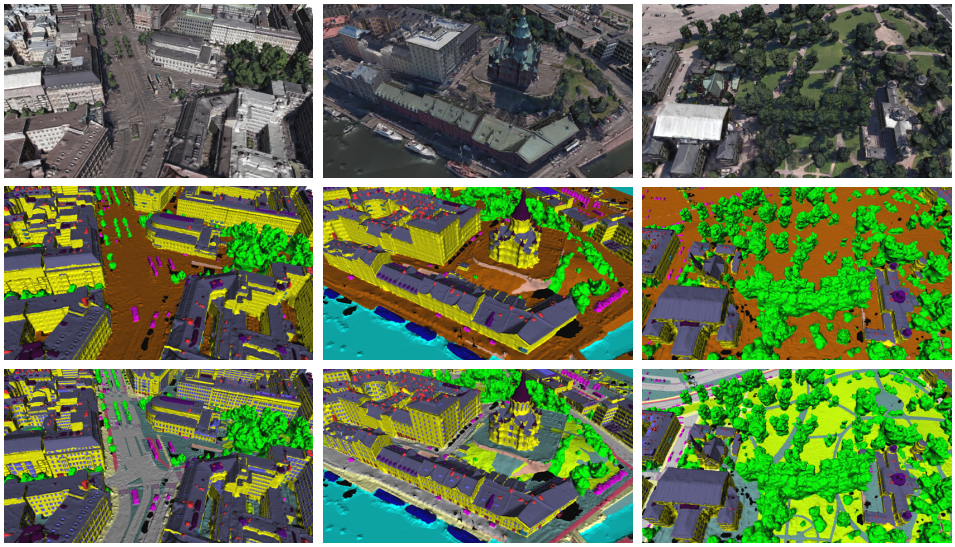


Figure 5.11: Examples of part-level semantic annotation of urban meshes: the first row displays the original textured meshes, the second row shows the semantic meshes with face-based labels (comprising 13 classes), and the third row shows semantic meshes with texture-based labels (encompassing 21 classes). Semantic classes are represented by colors, including unclassified ■, terrain ■, high vegetation ■, water ■, car ■, boat ■, wall ■, roof surface ■, facade surface ■, chimney ■, dormer ■, balcony ■, roof installation ■, window ■, door ■, low vegetation ■, impervious surface ■, road ■, road marking ■, cycle lane ■, and sidewalk ■.

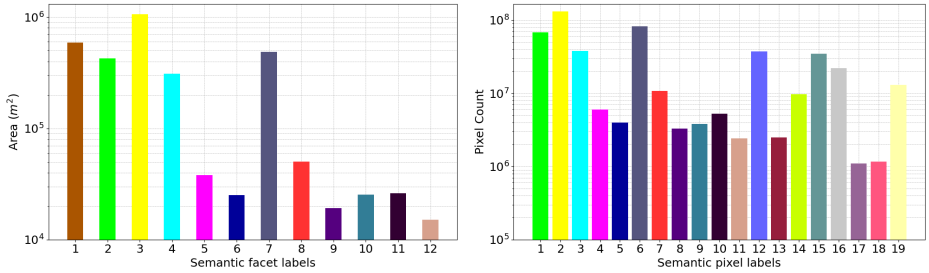


Figure 5.12: Statistical distribution of semantic classes in the entire dataset: the upper chart illustrates the distribution of face labels weighted by area, while the lower chart depicts the distribution of pixel-level labels. The colors correspond to the same semantic classes described in subsection 5.6.1.

vegetation, and impervious surfaces.

- *high vegetation*: Consists of taller plant life such as trees, shrubs, and bushes.
- *water*: Encompasses bodies of water like rivers, seas, and pools.

- *car*: Covers all types of road vehicles, including cars, buses, and lorries.
- *boat*: Comprises various types of watercraft such as boats, ships, freighters, and sailboats.
- *wall*: Refers to any vertical barrier, including fences, monuments, bollards, seawalls, and stone walls.
- *roof surface*: Pertains to the external upper covering of a building.
- *facade surface*: Relates to the exterior front walls of buildings, often facing a street or open space.
- *chimney*: Includes structures protruding above roofs for venting smoke or exhaust.
- *dormer*: Refers to a structural element of a building that projects from a sloping roof.
- *balcony*: Encompasses outdoor platforms on the outside of buildings, enclosed by walls or balustrades, accessible through an upper-floor door or window.
- *roof installation*: Covers fixtures installed on roofs, such as solar panels, satellite dishes, and HVAC units.

Based on face labels, we have further defined pixel labels for mesh textures that encompass part-level details within textured planar areas, introducing 8 new categories:

- *window*: Includes openings in a building that are fitted with glass or transparent material, allowing light and air to enter.
- *door*: Pertains to movable barriers used to close off entrances, typically swinging on hinges.
- *low vegetation*: Covers shorter plant life such as grasses, ground covers, and flowering plants.
- *impervious surface*: Refers to hard surfaces that water cannot penetrate, like concrete and asphalt.
- *road*: Includes thoroughfares for vehicular travel, excluding paths primarily meant for walking or cycling.
- *road marking*: Refers to all markings painted or embedded into road surfaces for regulation and guidance of traffic.
- *cycle lane*: Designated pathways on roads specifically for the use of bicycles.
- *sidewalk*: Pedestrian paths beside roads, usually paved and separated from the roadway by a curb.

According to the label definition, we differentiate between two types of scenes: meshes with only face labels, featuring 12 semantic classes (excluding 'unclassified'), and meshes with both face and pixel labels, comprising 19 semantic classes (excluding 'unclassified', and 'terrain' as these are broken down into more specific pixel labels). Notably, within datasets that include pixel labels, if an area displays both face and pixel labels, the pixel labels take precedence. For instance, if an area is labeled as a "facade surface" for the face but a "window" for the pixel, it is ultimately defined as a "window." Figure 5.11 shows examples of visualized annotated meshes, while Figure 5.12 presents the details of the class distribution. To evaluate state-of-the-art 3D semantic segmentation methods, we divided the entire dataset into three splits by randomly selecting tiles: 24 for training, 8 for validation, and 8 for testing.

5.6.2. Evaluation of semantic mesh segmentation

In this section, we first explore the impact of mesh sampling strategies on semantic segmentation. We then examine the performance of existing 3D semantic segmentation methods on our dataset.

Mesh sampling strategy. Existing research shows that in addition to methods directly performing semantic segmentation on triangle mesh surfaces, point cloud semantic segmentation techniques can utilize point clouds generated from mesh surface sampling as input data [W. Gao *et al.*, 2021; Selvaraju *et al.*, 2021]. This data is then mapped back to the triangle mesh using the nearest neighbor or voting mechanisms. To adapt point cloud segmentation techniques for use with triangle meshes, studies commonly employ Poisson-disk sampling [Cook, 1986] to generate input data. However, research into the effects of various mesh surface point cloud sampling techniques on semantic segmentation performance is still limited.

To evaluate the sensitivity and robustness of these methods to different types of point cloud sampling, we compared four types of mesh sampling strategies: face-centered sampling, random sampling, Poisson-disk sampling [Cook, 1986], and our proposed superpixel texture sampling. Face-centered clouds are obtained by calculating the centroid of each triangular face, nearly equivalent to direct semantic segmentation on triangle faces, but they are unsuitable for expressing pixel labels. Random and Poisson-disk sampling [Cook, 1986] are based on a set number of points or density, introducing challenges in complex urban scenarios because different semantic categories vary in their sensitivity to the number and density of sampling points. Especially for smaller-scale part-level semantic classes, fewer sampling points might not capture features adequately, while excessive points increase the computation load.

To ensure a precise representation of semantic categories with minimal sampling points, we adopted a novel approach. Initially, we applied the SLIC [Achanta *et al.*, 2012] algorithm for superpixel-level over-segmentation on the mesh's entire texture image (superpixel size set at 5 in all experiments). This process accurately captures the boundaries of part-level objects based on color information. Subsequently, we compute

the 3D coordinates of each superpixel center from the texture coordinates of each triangle, generating a superpixel texture point cloud. This approach ensures data accuracy and significantly reduces the computational burden, making it particularly effective for translating pixel labels into point labels. We also provided visual examples of the four mesh sampling strategies, as shown in Figure 5.13.

Figure 5.13: Comparison of textured mesh sampling methods: from the top row to the bottom row, the images demonstrate the original textured meshes, face center point clouds, random sampled point clouds, Poisson-disk sampled point clouds [Cook, 1986], and our proposed superpixel texture sampled point clouds.

Benchmark evaluation. We evaluated a range of representative 3D semantic segmentation methods on our proposed dataset. These methods encompass semantic segmentation methods designed for triangle mesh, including RF-MRF [Rouhani *et al.*, 2017], SUM-RF [W. Gao *et al.*, 2021], and PSSNet [W. Gao *et al.*, 2023], along with diverse point cloud segmentation approaches such as PointNet [Charles *et al.*, 2017], PointNet++ [Qi *et al.*, 2017], superpoint graphs (SPG) [Landrieu and Simonovsky, 2018],

SparseConvUnet [Graham *et al.*, 2018], Randla-net [Q. Hu *et al.*, 2020], KPConv [Thomas *et al.*, 2019], PointNext [Qian *et al.*, 2022], Point transformer V3 (PointTransV3) [X. Wu *et al.*, 2024], and PointVector [Deng *et al.*, 2023]. It is worth noting that there are currently no semantic segmentation methods specifically designed for meshes with textured-based pixel labels. These point cloud segmentation methods were evaluated on both the face and pixel labeling tracks. Here, the face labeling track refers to semantic labels for faces, encompassing 12 classes (excluding 'unclassified'); the pixel labeling track pertains to semantic labels for texture pixels, including 19 classes (excluding 'terrain' and 'unclassified').

To address class imbalance within the dataset, we applied class weights to each method tested to ensure fair consideration and evaluation of all categories. Additionally, to minimize errors resulting from randomness in network predictions, all experimental results are based on the average outcomes from five trained models.

We used established semantic segmentation evaluation metrics, conducting a detailed analysis that included Intersection over Union (IoU) per class, overall accuracy (OA), mean accuracy (mAcc), and mean IoU (mIoU). For the face labeling track, we determine the final labels of triangles using a voting method based on point cloud prediction results, with the area of each triangle serving as a weight for semantic evaluation metrics. For the pixel labeling track, we assign final labels to each pixel in the texture image using the nearest neighbor method based on point cloud prediction results, applying pixel-level semantic evaluation metrics.

1) Face labeling track. The face labelling track includes 12 face semantic labels mentioned in subsection 5.6.1, excluding 'unclassified'. We first evaluated the impact of four different mesh surface point cloud sampling strategies on the performance of point cloud semantic segmentation methods, including face-centered, random sampling, Poisson-disk sampling [Cook, 1986], and superpixel texture sampling point clouds. To avoid the effects of different point cloud densities or quantities, we set the number of points for random and Poisson-disk sampled point clouds based on the number of points in superpixel texture sampled point clouds (totaling 25,169,441 points). The number of points in face-centered point clouds consistently matches the number of mesh faces (totaling 11,779,034 points).

Table 5.1 shows that most methods achieve optimal performance with face-centered point cloud inputs, benefitting from adaptability to the geometric characteristics of triangulated meshes. Specifically, because the density of triangles is lower in flat areas and higher in non-flat areas, this enables deep learning networks to learn rich geometric features based on the distribution of triangle density. However, this does not apply to uniform triangular meshes. This also indicates that the impact of point cloud sampling density on semantic segmentation is far less significant than the impact of point cloud distribution. The average mIoU in Table 5.1 also indicates that our proposed superpixel texture sampling method outperforms other mesh sampling methods (except for face-centered point clouds). Additionally, the standard deviation results indicate that SPG [Landrieu and Simonovsky, 2018] is the most robust across different sampling

	Fc.	Rd.	Po.	Sp.	Std. (%)
PoinNet	5.1	8.5	5.1	**15.1**	4.7
PoinNet++	27.1	17.3	18.4	**33.1**	7.5
SPG	29.9	29.7	31.5	**31.7**	**1.0**
SparseUNet	**60.5**	47.6	38.6	49.9	9.0
Randla-net	**57.4**	49.8	49.1	54.4	3.9
KPConv	**57.5**	56.4	46.5	52.9	5.0
PointNext	**65.3**	51.3	50.4	47.7	7.9
PointTransV3	**59.1**	49.5	51.7	54.0	4.1
PointVector	**70.0**	56.1	52.8	57.1	7.6
Average	**48.0**	40.7	38.2	44.0	-

Table 5.1: Evaluation of the impact of four mesh sampling methods on semantic segmentation performance, using mIoU (%) as the evaluation metric: where 'Fc.' stands for face-centered, 'Rd.' for random sampling, 'Po.' for Poisson-disk sampling [Cook, 1986], 'Sp.' for superpixel texture sampling, 'Std.' for standard deviation in mIoU across different types of sampling, and 'Average' for the average mIoU of all segmentation methods. The highest mIoU achieved under different sampling methods for each approach, the lowest standard deviation among all semantic segmentation methods, and the average of the highest mIoUs across all sampling methods are highlighted in bold.

methods. This is due to its over-segmentation method being unaffected by the density and distribution of the sampled point clouds.

From Table 5.2, it can be observed that among all methods, PointVector [Deng *et al.*, 2023] performs best overall, with a mIoU of 70.0%, significantly outperforming other methods. Due to class imbalance, most methods show better performance in categories with more samples and poorer performance in categories with fewer samples. We also conducted a qualitative analysis of the top three methods, as shown in the Figure 5.14.

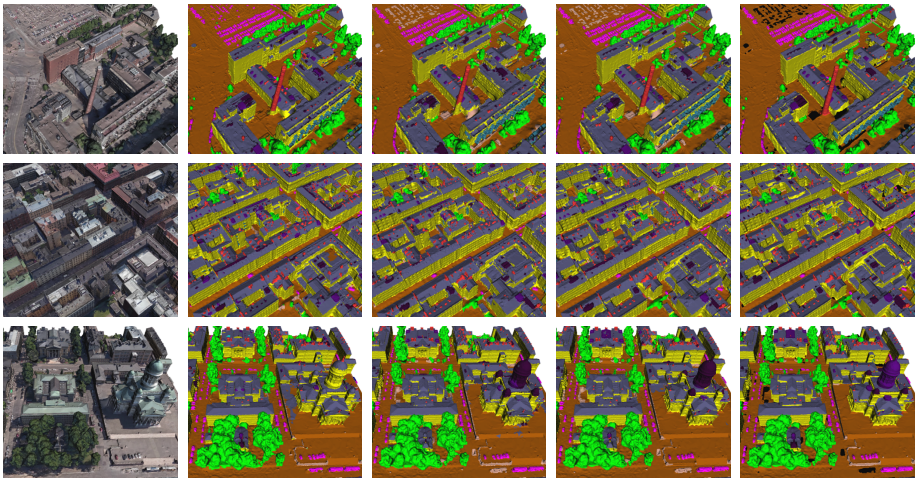
2) Pixel labeling track. The pixel labeling track consists of 19 semantic classes, as described in subsection 5.6.1, including all semantic labels except for 'terrain' and 'unclassified'. We evaluated the impact of three sampling methodsrandom, Poisson-disk, and superpixel texture samplingon semantic segmentation performance. Because face-centered point clouds cannot represent semantic components with pixel labels, they were excluded from testing.

The average mIoU in Table 5.3 shows that most methods achieve optimal performance with our proposed superpixel texture sampling as input, which precisely captures the boundaries of part-level objects. Additionally, the standard deviation results indicate that PointNext [Qian *et al.*, 2022] and Point Transformer V3 [X. Wu *et al.*, 2024] perform robustly across various sampling methods.

Table 5.4 shows that PointVector [Deng *et al.*, 2023] continues to outperform all other methods, achieving a mIoU of 47.9%, significantly higher than the others.

	terr.	hveg.	faca.	wate.	car	boat	roof.	chim.	dorm.	balc.	roin.	wall	OA	mAcc	mIoU
RF_MRF	81.6	86.6	81.3	84.5	24.8	3.7	73.3	27.6	0.0	4.8	0.4	5.9	85.2	45.3	39.5
SUM_RF	84.8	88.1	84.0	79.0	42.5	10.6	77.7	42.4	3.5	22.2	4.7	12.7	86.9	53.6	46.0
PSSNet	80.7	90.5	85.2	64.2	52.6	13.0	78.1	44.0	6.6	25.7	6.9	16.6	86.3	56.4	47.0
PoinNet$^{Sp.}$	52.6	7.1	38.6	59.9	0.0	0.0	22.8	0.0	0.0	0.0	0.0	0.0	50.6	22.0	15.1
PoinNet++$^{Sp.}$	67.9	68.7	59.2	86.1	24.2	11.1	51.1	24.9	0.0	0.0	3.3	1.1	69.0	46.9	33.1
SPG$^{Sp.}$	53.4	55.3	62.5	40.5	27.4	13.1	64.3	33.9	5.1	11.3	3.9	9.9	64.9	55.0	31.7
SparseUNet$^{Fc.}$	88.6	91.7	88.6	76.7	75.6	14.6	82.3	70.1	27.0	49.0	28.0	33.9	90.3	71.7	60.5
Randla-net$^{Fc.}$	86.7	92.3	81.6	**87.1**	82.9	**41.2**	71.6	55.6	21.6	27.6	19.0	21.1	86.7	76.3	57.4
KPConv$^{Fc.}$	86.9	90.8	88.3	81.5	66.4	16.5	81.9	66.7	16.1	45.3	21.2	28.2	90.1	64.7	57.5
PointNext$^{Fc.}$	91.0	95.0	90.4	81.6	91.2	17.9	83.1	74.6	33.8	56.0	30.0	39.3	91.8	77.2	65.3
PointTransV3$^{Fc.}$	88.6	90.1	87.9	78.9	72.1	16.1	81.0	66.2	21.4	45.2	25.0	36.4	89.9	70.2	59.1
PointVector$^{Fc.}$	**92.3**	**96.8**	**91.7**	85.1	**95.2**	22.0	**85.9**	**82.6**	**47.9**	**62.4**	**38.6**	**40.0**	**93.1**	**80.7**	**70.0**

Table 5.2: Comparison of different 3D semantic segmentation methods on the face labeling tracks. Each point cloud semantic segmentation method uses its best-performing sampling method from Table 5.1, where 'terr.' for terrain, 'hveg.' for high vegetation, 'faca.' for facade surface, 'wate.' for water, 'roof.' for roof surface, 'chim.' for chimney, 'dorm.' for dormer, 'balc.' for balcony, and 'roin.' for roof installation. Additionally, $^{Fc.}$ denotes the face-centered sampling, and $^{Sp.}$ represents superpixel texture sampling. The results for different semantic categories are presented as IoU (%), while Overall Accuracy (OA %), mean Accuracy (mAcc %), and mean IoU (mIoU %) are displayed as percentages. The highest values of IoU, OA, mAcc, and mIoU are highlighted in bold.



(a) Input (b) SpaseU$^{Fc.}$ (c) PtNext$^{Fc.}$ (d) PtVec$^{Fc.}$ (e) Truth

Figure 5.14: Qualitative analysis of semantic segmentation results in face labeling track, including the three best-performing methods: SparseUNet$^{Fc.}$(SpaseU) [Graham *et al.*, 2018], PointNext$^{Fc.}$(PtNext) [Qian *et al.*, 2022], and PointVector$^{Fc.}$(PtVec) [Deng *et al.*, 2023], where $^{Fc.}$ represents the face-centered sampled point cloud. Semantic classes are distinguished by different colors (same as in subsection 5.6.1).

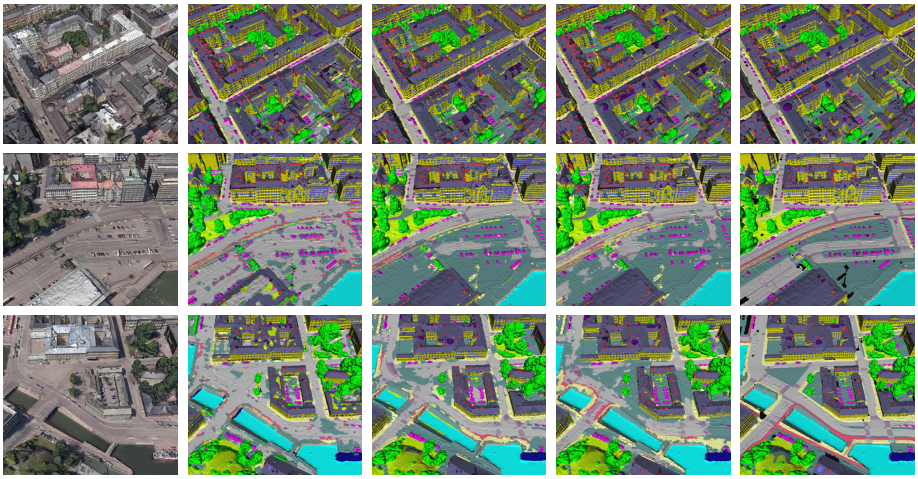
	Rd.	Po.	Sp.	Std. (%)
PoinNet	2.5	5.7	2.6	1.8
PoinNet++	20.6	19.5	**24.7**	2.7
SPG	16.0	20.0	**19.2**	2.1
SparseUNet	**34.5**	13.3	23.3	10.6
Randla-net	36.9	39.9	**42.1**	2.6
KPConv	38.9	26.1	**42.6**	8.7
PointNext	42.9	**44.7**	43.0	**1.0**
PointTransV3	**38.0**	36.1	37.8	**1.0**
PointVector	44.7	45.1	**47.9**	1.8
Average	30.6	27.8	**31.5**	-

Table 5.3: Evaluation of three mesh sampling methods on semantic segmentation performance, measured by mIoU (%): where Rd. represents random sampling, 'Po.' represents Poisson-disk sampling [Cook, 1986], 'Sp.' represents superpixel texture sampling, 'Std.' represents the standard deviation of mIoU across different types of sampling, 'Average' is the average mIoU of all segmentation methods. The highest mIoU achieved by each method under different sampling conditions, the lowest standard deviation across all methods, and the average of the highest mIoUs for all sampling methods are all highlighted in bold.

	hveg.	faca.	wate.	car	boa.	roof.	chim.	dorm.	balc.	roin.	wall	wind.	door	lveg.	impe.	road	roma.	cycl.	side.	OA	mAcc	mIoU
PoinNet$^{Sp.}$	0.5	13.3	16.5	0.0	2.1	7.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	8.5	0.0	0.0	0.0	0.2	17.3	9.8	2.6
PoinNet++$^{Sp.}$	72.7	47.5	86.4	34.9	12.4	52.4	28.1	0.0	5.3	5.6	0.4	13.0	5.0	42.4	31.2	14.6	9.5	0.0	7.3	55.4	35.2	24.7
SPG$^{Sp.}$	58.2	50.8	18.4	24.1	2.7	60.4	39.9	3.1	13.6	4.4	10.5	2.4	4.0	13.4	14.6	31.0	0.0	1.7	12.1	51.5	34.5	19.2
SparseUNet$^{Rd.}$	88.8	**70.0**	5.9	51.6	2.5	**79.8**	55.0	12.3	**45.4**	22.6	**31.5**	32.0	12.3	15.2	43.8	44.6	5.2	0.6	35.8	72.9	45.1	34.5
Randla-net$^{Sp.}$	90.5	60.9	84.6	67.6	**22.7**	74.7	53.3	0.6	29.3	16.2	26.3	33.4	12.8	59.8	48.8	50.2	31.5	0.0	37.1	73.5	57.7	42.1
KPConv$^{Sp.}$	84.0	68.5	81.7	68.6	21.8	78.2	**66.4**	**25.0**	41.8	**29.6**	31.5*	36.1	14.9	21.4	35.8	50.0	7.3	13.4	34.1	74.4	58.3	42.6
PointNext$^{Po.}$	90.1	66.2	87.9	68.1	16.3	74.5	59.7	14.9	35.6	19.1	31.0	33.2	13.7	55.5	51.4	55.5	29.0	6.9	40.0	76.0	57.6	44.7
PointTransV3$^{Rd.}$	85.9	59.9	74.6	64.7	17.8	75.9	58.7	15.3	37.2	16.2	29.3	11.8	7.9	27.1	43.3	51.5	3.5	7.2	33.4	70.6	54.1	38.0
PointVector$^{Sp.}$	**92.7**	66.6	**92.0**	**70.2**	19.8	76.8	60.8	21.8	37.0	20.6	30.8	**37.1**	**16.5**	**59.8**	**53.9**	**57.4**	**35.0**	**16.4**	**45.0**	**77.0**	**63.8**	**47.9**

Table 5.4: Comparative results of various 3D semantic segmentation methods on the pixel labeling tracks. Each point cloud semantic segmentation method employs its most effective sampling strategy from Table 5.3, where hveg. for high vegetation, faca. for facade surface, wate. for water, roof. for roof surface, chim. for chimney, dorm. for dormer, balc. for balcony, roin. for roof installation, wind. for window, lveg. for low vegetation, impe. for impervious surfaces, roma. for road marking, cycl. for cycle lane, side. for sidewalk. Additionally, $^{Sp.}$ represents superpixel texture sampling, $^{Rd.}$ denotes the random sampling, and $^{Po.}$ represents the Poisson-disk sampling [Cook, 1986]. Results for different semantic categories are shown as IoU (%), while Overall Accuracy (OA %), mean Accuracy (mAcc %), and mean IoU (mIoU %) are displayed as percentages. *KPConvs IoU for the wall category is 31.46%, marginally lower than SparseUnets 31.47%. The highest values of IoU, OA, mAcc, and mIoU are highlighted in bold.

PointVector [Deng *et al.*, 2023] surpasses other methods in all categories, particularly with pixel labels. However, compared to the categories shared with Table 5.2, the IoU results for most methods have decreased. This is mainly because the three types of mesh sampling point clouds are relatively uniform and fail to capture the geometric density differences reflected by adaptive meshes. Additionally, the increase in the number of classes has exacerbated the issue of class imbalance. We also performed a qualitative analysis on the top three methods, as illustrated in the Figure 5.15.



(a) Input (b) KPCon$^{Sp.}$ (c) PtNext$^{Po.}$ (d) PtVec$^{Sp.}$ (e) Truth

Figure 5.15: Qualitative analysis of semantic segmentation results in pixel labeling track, including the three best-performing methods: KPConv$^{Sp.}$(KPConv) [Thomas *et al.*, 2019], PointNext$^{Po.}$(PtNext) [Qian *et al.*, 2022], and PointVector$^{Sp.}$(PtVec) [Deng *et al.*, 2023], where $^{Po.}$ represents the Poisson-disk sampling [Cook, 1986], and $^{Sp.}$ for the superpixel texture sampling. Semantic classes are distinguished by different colors (same as in subsection 5.6.1).

5.6.3. Evaluation of interactive annotation

This section discusses the performance of existing interactive annotation tools for triangular meshes and textured images in real-world scenarios. Additionally, we propose new evaluation criteria for interactive annotations, based on user studies.

Evaluation metrics. To more accurately evaluate interactive annotation efficiency, we have developed a comprehensive set of evaluation criteria that reflect real-world effects. Traditional evaluation methods, such as counting the number of clicks required to achieve a certain Intersection over Union (IoU) or Average Precision (AP), provide quantifiable metrics, but these do not fully capture the true efficiency of the annotation process. The main shortcomings of these methods include:

- Evaluation limitations: Relying solely on IoU or AP as the only evaluation metric does not accurately reflect the comprehensiveness of annotation. For example, even if the IoU might have reached 90%, the user may still need to make multiple adjustments to the boundaries to match the real situation accurately.

- Interaction limitations: Interactions based solely on clicks cannot perfectly annotate boundaries, often requiring the combination of other tools like lasso or polygon editing for corrections, hence a single click count assessment does not fully reflect the actual complexity of interactions. Additionally, traditional evaluations often standardize user click positions for comparisons. However, each user's interaction choices vary, making it challenging to realistically assess the efficiency of methods in practical applications.

- Efficiency limitations: The average number of clicks does not equate to real interaction efficiency, as users vary in operational speed. Evaluating total annotation time offers a more accurate reflection of interaction efficiency.

In response to these issues, we designed a comprehensive evaluation system including Intersection over Union (IoU), Boundary IoU (BIoU), number of operations (Oper), annotation time (Time), and the ratio of non-manual interaction usage, i.e., smart interaction ratio (SR). BIoU specifically assesses the accuracy of boundary annotations. For image and mesh boundary evaluation, we refer to the works of [Cheng *et al.*, 2021] and [W. Gao *et al.*, 2023], respectively. Oper counts operations involving mouse clicks and keyboard keystrokes. Time measures the time needed to complete annotations, with all experiments in this paper measured in seconds. SR measures the frequency of non-manual interaction tool use. To assess the usage of different annotation tools in real scenarios, our evaluation approach is entirely based on user studies. Assuming the participation of u users across n test scenes, each with c categories, the averages of these metrics are calculated as follows:

- Evaluating a single scenario. For a given scenario s, annotated by u users across c categories, $\overline{mIoU}$, $\overline{mBIoU}$, $\overline{mOper}$, $\overline{mTime}$, and $\overline{mSR}$, can be obtained by averaging the IoU, $mBIoU$, $mOper$, $mTime$, and mSR values across all users and categories:

$$\overline{mIoU}_s = \frac{1}{u \times c} \sum_{i=1}^{u} \sum_{j=1}^{c} IoU_{i,j,s} \qquad \overline{mBIoU}_s = \frac{1}{u \times c} \sum_{i=1}^{u} \sum_{j=1}^{c} BIoU_{i,j,s}$$

$$\overline{mOper}_s = \frac{1}{u} \sum_{i=1}^{u} Oper_{i,s} \qquad \overline{mTime}_s = \frac{1}{u} \sum_{i=1}^{u} Time_{i,s}$$

$$\overline{mSR}_s = \frac{1}{u} \sum_{i=1}^{u} SR_{i,s}$$

  The formulas above represent the average performance per test scene, aggregated across multiple users.

- Evaluating multiple scenarios. To obtain $\overline{M}$, $\overline{B}$, $\overline{O}$, $\overline{T}$, and $\overline{S}$, the averages of for multiple scenarios, we take the average of each scenarios $\overline{mIoU}$, $\overline{mBIoU}$, $\overline{mOper}$,

$\overline{mTime}$, $\overline{mSR}$ and then average these values:

$$\overline{M} = \frac{1}{n}\sum_{s=1}^{n}\overline{mIoU}_s \qquad \overline{B} = \frac{1}{n}\sum_{s=1}^{n}\overline{mBIoU}_s$$

$$\overline{O} = \frac{1}{n}\sum_{s=1}^{n}\overline{mOper}_s \qquad \overline{T} = \frac{1}{n}\sum_{s=1}^{n}\overline{mTime}_s$$

$$\overline{S} = \frac{1}{n}\sum_{s=1}^{n}\overline{mSR}_s$$

The above formulas denote the average performance across all test scenes.

In our interactive testing experiments detailed in this paper, we invited five users to participate (i.e., u=5), representing diverse backgrounds. The group included two experts with knowledge in photogrammetry and geoinformation, and three non-experts without such expertise. To ensure the reliability of the results, all participants received the same training course prior to the study, providing them with a basic understanding of the tasks and ensuring a consistent operational standard. We recorded each user's annotation progress and interaction count in real-time using a script running in the background, requiring them to complete at least 95% of the content annotation in each scene (annotation progress was based on the total surface area of the mesh or the total number of texture pixels). This evaluation standard is applicable to both 2D and 3D interactive annotations and ensures that our evaluation criteria are practical and scientifically valid.

Evaluation of interactive mesh annotation. To our knowledge, semantic annotation methods for mesh faces are currently limited. As detailed in subsection 5.2.2, existing mesh annotation methods typically annotate only the vertices of dense triangle meshes [Lang *et al.*, 2024; Selvaraju *et al.*, 2021] or require additional inputs like camera parameters, original images, and depth maps [Miksik *et al.*, 2015; D. T. Nguyen *et al.*, 2018; Romanoni and Matteucci, 2018]. Given that our method only requires triangle mesh surfaces as input, we have chosen to compare it with methods using similar inputs. These include a manual annotation method [Rouhani *et al.*, 2017] and a segment-based [W. Gao *et al.*, 2021] interactive annotation method. For evaluation, we selected four representative scenes: a courtyard complex, a street with vehicles, a park with trees, and a harbor with ships. All users independently used purely manual annotation, segment-based interactive annotation, and our method to annotate the four test samples. Notably, we did not consider any single user's purely manual annotation results as the absolute ground truth; instead, we used the cross-validation average of all users' purely manual annotation results as the reference standard.

From Table 5.5, it can be observed that our method outperforms both manual annotation [Rouhani *et al.*, 2017] and segment-based annotation [W. Gao *et al.*, 2021] across all evaluation metrics. Table 5.6 reveals that, compared to segment-based annotation [W. Gao *et al.*, 2021], our method slightly excels in the quality of object region and boundary annotations. Across all four test scenarios, our method significantly reduces both the number of interactions and the time required for

	$\overline{M}(\%)$	$\overline{B}(\%)$	$\overline{O}$	$\overline{T}(s)$	$\overline{S}(\%)$
Man.	-	-	6754	4183.2	-
Seg.	91.6	74.7	6119	3565.8	-
Ours	**92.2**	**75.0**	**2992**	**2992.1**	**83.0**

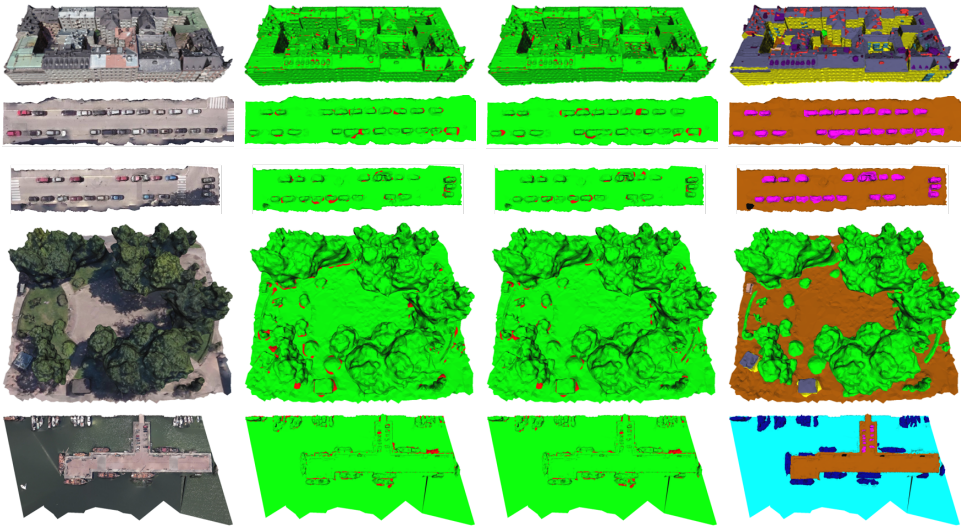
Table 5.5: Comprehensive performance evaluation of interactive mesh face annotation methods across four test scenarios: where Man. stands for manual annotation [Rouhani *et al.*, 2017], Seg. for segment-based annotation [W. Gao *et al.*, 2021]. The highest values are given in bold.

Metric	Method	Cour.	Stre.	Park.	Harb.
$\overline{mIoU}_s(\%)$	Seg	89.1	94.0	92.1	91.1
	Ours	**89.5**	**94.2**	**92.9**	**92.2**
$\overline{mBIoU}_s(\%)$	Seg.	**72.7**	**85.4**	70.6	70.3
	Man.	72.4	84.5	**71.9**	**71.0**
$\overline{mOper}_s$	Man.	18154	1589	3559	3714
	Seg.	17645	1407	2529	2894
	Ours	**13231**	**909**	**1797**	**1957**
$\overline{mTime}_s(s)$	Man.	11401.0	969.5	2146.5	2215.8
	Seg.	10441.3	757.0	1441.4	1623.7
	Ours	**9107.2**	**498.0**	**1105.9**	**1257.5**
$\overline{mSR}_s(\%)$	Ours	**66.5**	**94.9**	**85.8**	**84.8**

Table 5.6: Detailed performance evaluation of interactive mesh face annotation methods across four test scenarios: where Cour. stands for courtyard complex, Stre. for the streets with vehicles, Park. for a park with trees, Harb. for harbor with ships, Man. for manual annotation [Rouhani *et al.*, 2017], Seg. for segment-based annotation [W. Gao *et al.*, 2021]. The highest values are given in bold.

annotation compared to other methods. Specifically, our method is about 1.73 times faster than manual annotation [Rouhani *et al.*, 2017] and 1.32 times faster than segment-based methods [W. Gao *et al.*, 2021]. Notably, except in the courtyard complex scenario, our method's smart interaction usage rate exceeds 80%, indicating that minimal manual intervention is required for annotations in these scenarios. For the courtyard complex scenario, due to issues like shadows and obstructions during aerial photography, the data quality of categories such as facades and ground surfaces within the courtyard is poor and noisy, thus requiring more manual identification and annotation. We also provided a qualitative analysis, as shown in the Figure 5.16.

Evaluation of interactive texture annotation. To comprehensively evaluate our method's performance, we not only compared it with traditional purely manual annotation methods but also chose three highly representative interactive image



(a) Textured mesh (b) Segment-based (c) Ours (d) Manual

Figure 5.16: Comprehensive qualitative analysis of interactive mesh face annotation results: (b) is based on segment-based annotation [W. Gao *et al.*, 2021], (c) shows our results, and (d) shows the results based on manual annotation [Rouhani *et al.*, 2017]. In (b) and (c), the error maps are displayed, where red indicates differences from the ground truth and green indicates matches with the ground truth. (d) shows the semantic mesh, with different colors representing the various classes as described in subsection 5.6.1.

segmentation techniques for comparative analysis. These include the classic GrabCut [Rother *et al.*, 2004], the Segment anything (SAM) [Kirillov *et al.*, 2023] model-based interactive annotation, and Simpleclick [Q. Liu *et al.*, 2023], which has shown excellent performance across multiple interactive image annotation datasets. In practice, we enhanced GrabCut [Rother *et al.*, 2004] editing by incorporating tools like lasso, rectangle, circle, and polygon to increase the efficiency of foreground and background sample selection and to expand the sample set. For SAM [Kirillov *et al.*, 2023], we utilized a pre-trained Vision Transformer-Huge (ViT-H) model [Dosovitskiy *et al.*, 2020]. For Simpleclick [Q. Liu *et al.*, 2023], we employed a ViT-H model [Dosovitskiy *et al.*, 2020] trained on the COCO [T.-Y. Lin *et al.*, 2014] and LVIS [Gupta *et al.*, 2019] datasets. We selected three typical scenes from our dataset to annotate textures: facades (×2), parks (×2), and roads (×2), involving six samples in total for evaluation ($n = 6$). All test users independently used purely manual annotation, GrabCut [Rother *et al.*, 2004], SAM [Kirillov *et al.*, 2023], Simpleclick [Q. Liu *et al.*, 2023], and our method to annotate these six test samples. We used the cross-validation average of all test users' purely manual annotation results as the reference standard for quantitative analysis as shown in Table 5.7 and Table 5.8.

From Table 5.7, it can be observed that our method surpasses deep learning methods

	$\overline{M}(\%)$	$\overline{B}(\%)$	$\overline{O}$	$\overline{T}(s)$	$\overline{S}(\%)$
Man.	-	-	653	777.1	
Gra.	**88.1**	47.1	711	780.9	30.2
SAM	84.4	29.8	716	**640.6**	**71.7**
Sip.	81.4	30.9	**252**	861.3	-
Ours	87.9	**49.3**	582	663.3	40.3

Table 5.7: Comprehensive performance evaluation of interactive texture image annotation methods across six scenarios: where Man. stands for manual annotation, Gra. for GrabCut [Rother *et al.*, 2004], SAM for Segment anything [Kirillov *et al.*, 2023], and Sip. for Simpleclick [Q. Liu *et al.*, 2023]. The highest values are given in bold.

in annotation quality and outperforms traditional methods in annotation time, also comparing favorably with the shortest annotation times of SAM [Kirillov *et al.*, 2023]. Specifically, our method excels in annotation quality over deep learning methods and compares closely with the traditional GrabCut [Rother *et al.*, 2004] method. In terms of boundary annotation, our method outperforms all other methods, primarily due to our fine segmentation and template-matching methods based on structural features. Regarding the number of interactions, our method performs better than SAM [Kirillov *et al.*, 2023] and GrabCut [Rother *et al.*, 2004], but is not as good as Simpleclick [Q. Liu *et al.*, 2023]. This is primarily because Simpleclick lacks tools for manual correction, offering only functionalities for positive and negative sample clicking. Therefore, when calculating the smart interaction ratio, we exclude Simpleclick [Q. Liu *et al.*, 2023], which always has a smart interaction ratio of 100%. While our smart interaction ratio exceeds that of GrabCut [Rother *et al.*, 2004], it remains lower than that achieved by deep learning methods such as SAM [Kirillov *et al.*, 2023]. This is primarily because the simplicity of our interactions does not match the efficiency of deep learning's pure clicking approaches.

From the Table 5.8, it can be seen that although our method's $\overline{M}$ is slightly lower than GrabCut [Rother *et al.*, 2004], the $\overline{mIoU}_s$ in most scenarios is superior to GrabCut and other methods. Our method significantly outperforms other methods in boundary quality across all test scenarios. In terms of interaction count, while our method does not reach the efficiency of Simpleclick [Q. Liu *et al.*, 2023], it outperforms other methods in most scenarios. Concerning interaction time, although our method is slower than SAM [Kirillov *et al.*, 2023], it still outperforms other methods in the majority of scenarios. In terms of smart interaction ratio, although our method necessitates some manual corrections, these corrections significantly enhance annotation quality without overly consuming time. Compared to deep learning methods, which are more efficient, their shortcomings in boundary annotation accuracy mean that their post-correction workload is comparable to purely manual annotation (see Figure 5.17 and Figure 5.18).

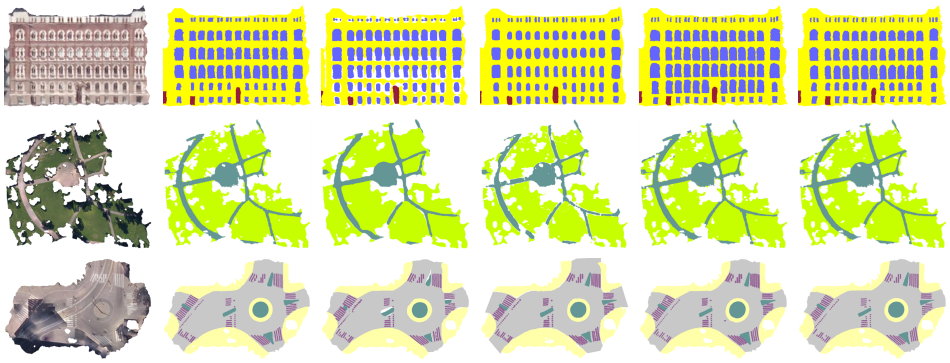
It is worth noting that for objects with regular shapes, interactive clicking may not be the optimal method of annotation. For instance, as shown in Figure 5.17 and Figure 5.18, a single click often lacks boundary precision, and multiple clicks do not significantly improve boundary accuracy. For components with repetitive structures,

Metric	Method	Fac1.	Fac2.	Par1.	Par2.	Rod1.	Rod2.
$\overline{mIoU}_s(\%)$	Gra.	80.2	87.3	94.0	90.4	91.3*	**85.4**
	SAM	**81.0**	86.4	85.3	86.4	86.5	80.7
	Sip.	73.7	77.5	87.9	86.2	84.0	79.1
	Ours	79.2	**88.7**	**95.0**	**91.2**	**91.3**	82.1
$\overline{mBIoU}_s(\%)$	Gra.	27.8	45.4	63.7	45.8	49.9	50.1
	SAM	24.7	33.9	25.8	23.3	34.1	37.0
	Sip.	19.7	26.8	38.0	34.4	31.8	34.6
	Ours	**28.1**	**50.5**	**67.8**	**48.6**	**50.5**	**50.6**
$\overline{mOper}_s$	Man.	515	124	497	297	718	1764
	Gra.	717	156	462	243	960	1729
	SAM	715	319	363	319	1020	1684
	Sip.	**297**	119	**77**	**78**	**400**	**539**
	Ours	487	**105**	497	213	720	1468
$\overline{mTime}_s(s)$	Man.	816.9	185.6	636.7	280.1	801.7	1941.4
	Gra.	920.1	242.6	494.9	270.8	836.3	1920.6
	SAM	**565.5**	**150.6**	460.9	389.5	800.6	**1476.5**
	Sip.	1128.5	338.9	**197.7**	230.8	1526.4	1745.5
	Ours	631.1	155.8	565.9	**189.1**	**767.9**	1670.3
$\overline{mSR}_s(\%)$	Gra.	7.1	11.8	59.3	66.5	9.9	26.7
	SAM	**94.7**	**92.8**	**78.8**	**78.6**	**49.2**	36.0
	Ours	20.3	21.2	51.6	75.8	32.2	**40.8**

Table 5.8: Detailed performance evaluation of interactive texture image annotation methods across six scenarios: where Man. stands for manual annotation, Gra. for GrabCut [Rother *et al.*, 2004], SAM is Segment anything [Kirillov *et al.*, 2023], and Sip. for Simpleclick [Q. Liu *et al.*, 2023], Fac1. for facade 1, Fac2. for facade 2, Par1. for park 1, Par2. for park 2, Rod1. for road 1, Rod2. for road 2. *GrabCut on road 1 is 91.29%, slightly lower than our score of 91.31%. The highest values are given in bold.

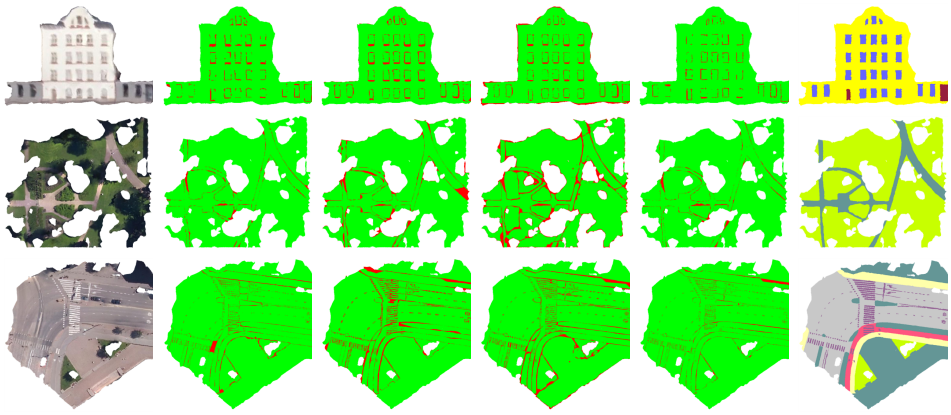
frequent clicking significantly increases the annotation burden. In practice, users often achieve high-precision annotations by simply drawing a rectangle or a polygon (like windows or doors), reducing the need for frequent interactions. Our method allows users to efficiently and accurately annotate all similar structures by manually creating a graphical template just once. In summary, we believe that if the time required for manual corrections in semi-automatic annotations equals or exceeds the time needed for fully manual annotations, then this method loses its intended utility.

Sensitivity analysis. We examined the impact of parameter adjustments and data quality on outcomes through qualitative analysis, demonstrating how these factors influence the selection results for a specific component.



(a) Input (b) GrabCut (c) Simclick (d) SAM (e) Ours (f) Manual

Figure 5.17: Comprehensive qualitative analysis of interactive texture annotation results: (b) shows the results of GrabCut [Rother *et al.*, 2004] annotation, (c) shows Simpleclick (Simclick) [Q. Liu *et al.*, 2023] annotation, (d) shows SAM [Kirillov *et al.*, 2023] annotation, and (e) shows our method's annotation. Semantic classes are labeled by colors, including facade surface ■, window ■, door ■, low vegetation ■, impervious surface ■, road ■, road marking ■, cycle lane ■, and sidewalk ■.



(a) Input (b) GrabCut (c) Simclick (d) SAM (e) Ours (f) Manual

Figure 5.18: Qualitative analysis of boundary errors in interactive texture annotation: (b) shows the error map of GrabCut [Rother *et al.*, 2004] annotation, (c) shows Simpleclick (Simclick) [Q. Liu *et al.*, 2023] error map, (d) shows SAM [Kirillov *et al.*, 2023] error map, and (e) shows our method's error map; from (b) to (e), the red color indicates differences from the ground truth, and the green color indicates matches with the ground truth; (f) displays the semantic texture mask, with different colors representing different classes as described in subsection 5.6.1.

1) Face-based analysis. In testing protrusion extraction, we examined the specific impact of the balance parameter λ^f on the outcomes. As shown in the left column

of Figure 5.19, increasing λ^f clarifies the boundary of the tree. For segment matching, we investigated the effects of the matching threshold $\epsilon^{(seg)}$ using a forest case, shown in the middle column of Figure 5.19. As $\epsilon^{(seg)}$ increases, more segments can be matched. For protrusion matching, we tested the impacts of the structural matching threshold $\epsilon^{(str)}$. In a harbor scenario, as $\epsilon^{(str)}$ increased, more boats were matched, as shown in the right column of Figure 5.19.

We further assessed how well our protrusion extraction method performs under various levels of mesh noise, simulating real-world data scenarios by introducing different intensities of Gaussian noise to the input mesh. Consistent interactive inputs and parameters were employed across varying noise levels to maintain result consistency. As shown in Figure 5.20, our approach demonstrates significant noise tolerance, accurately extracting protrusions even at $\sigma^m = 0.4m$. Besides, it is important to note that in the template matching process, the effectiveness of the results predominantly relies on the quality of protrusion extraction, given the high similarity in Gaussian noise distribution between the template and the matching data.

2) Texture-based analysis. In local region extraction, we explored how the balance parameter λ^s influences the outcomes. As shown in Figure 5.21 (a), a larger λ^s results in a smoother local region. In region matching, we discussed the impacts of the thresholds ϵ^{seed} and ϵ^{reg}. Figure 5.21 (b) and (c) show that as ϵ^{seed} and ϵ^{reg} increase, more regions can be matched. However, a large ϵ^{seed} value may lead to poor seed selection.

For most scenarios, our default parameters for the tested urban scenes yielded satisfactory results (i.e., $\lambda^f = 0.3$, $\epsilon^{(seg)} = 30$, $\epsilon^{(str)} = 20$, $\lambda^s = 0.2$, $\epsilon^{seed} = 15$, and $\epsilon^{reg} = 30$). The quantitative analysis of total interaction counts in the evaluation of interactive annotation also confirmed that users rarely need to adjust parameters. Furthermore, we believe that methods involving a few interpretable parameter adjustments typically outperform those relying solely on user clicks with deep learning methods. For instance, while methods like SAM [Kirillov *et al.*, 2023] and Simpleclick [Q. Liu *et al.*, 2023] require no parameter adjustments during use, minimal user input often does not consistently yield high-quality results. This is particularly evident in handling object boundaries and annotating unknown scenes, which may require users to make numerous additional interactive clicks, significantly increasing the annotation burden. Additionally, these models still require extensive parameter tuning during training.

We also evaluated the impact of local region extraction on handling texture image noise. Gaussian noise was utilized to mimic the noise typical of real-world images, applying consistent input positions and parameters across different noise levels. As depicted in Figure 5.22, our method effectively identifies target regions even in significant noise. Nevertheless, the extraction accuracy decreases with increased noise levels. Similarly, due to the uniform distribution of Gaussian noise in the images, the quality of the template-matching is largely influenced by the outcomes of the local region extraction.

In addition, the spatial location of user interactive inputs has a limited impact on the results of our method. For face annotation, when selecting segments, we only require

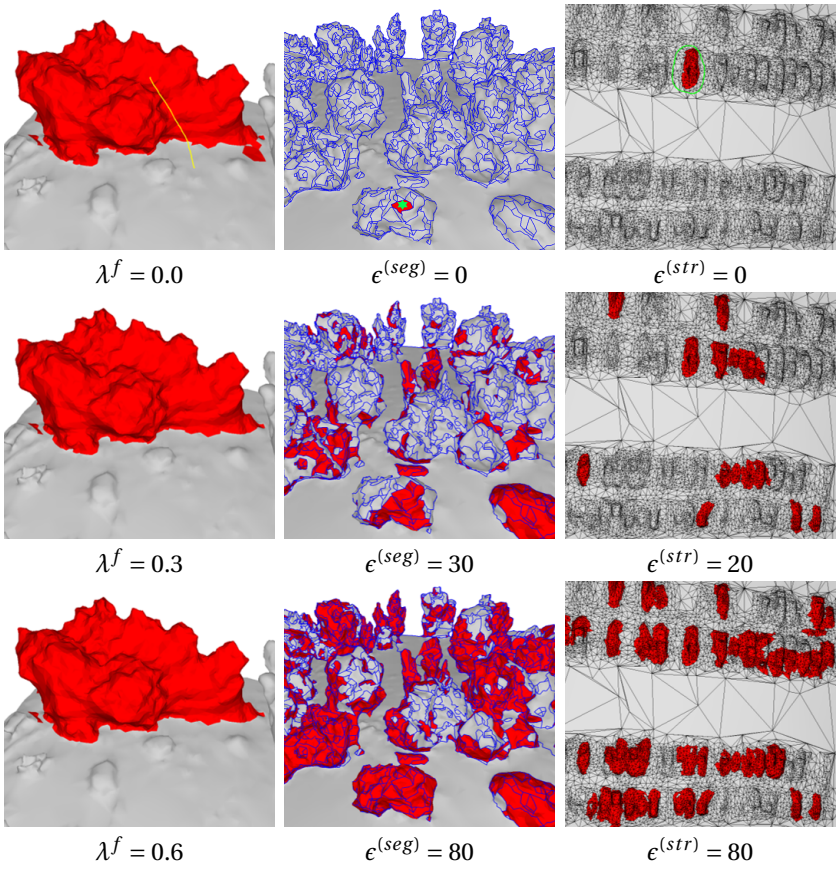


Figure 5.19: Sensitivity analysis of parameters in interactive mesh face annotation: from left to right, the images illustrate λ^f for protrusion extraction, where the yellow trajectory denotes user stroke input; $\epsilon^{(seg)}$ for segment matching, indicated by a green star representing user click input; and $\epsilon^{(str)}$ for protrusion matching, where the green trajectory signifies user lasso input.

that the user's click be within the segment, and for protrusion selection, we require that the user's lasso or stroke only encompasses the target to be extracted. For texture annotation, we require that the user ensure the click point is within the target area only.

5.6.4. Applications and limitations

Applications. For the annotation of mesh faces, our interactive annotation method is also suitable for indoor scenes, despite their relatively high complexity and diversity. In indoor scenes, items like furniture and decorations are typically considered protrusions, whereas walls, floors, and ceilings are treated as planar segments. As shown

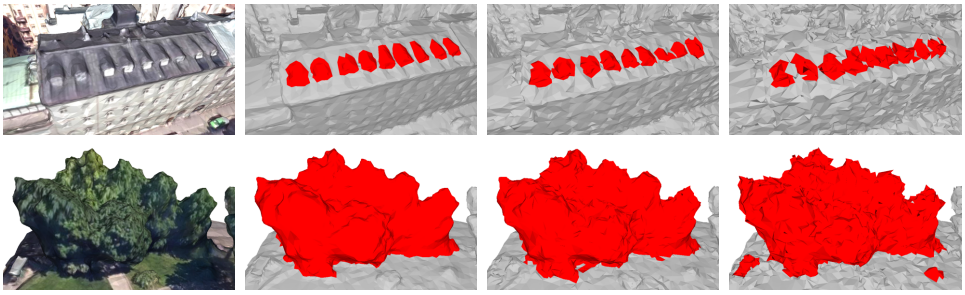


Figure 5.20: Sensitivity analysis of mesh quality in protrusion extraction: the red area indicates the selected area, and σ^m represents the standard deviation of the Gaussian noise in the mesh.

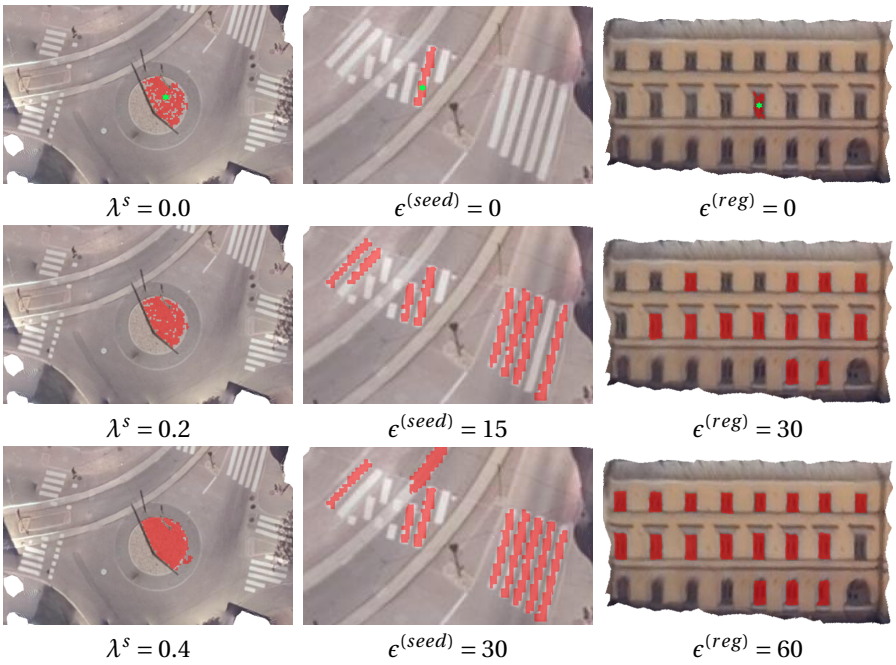


Figure 5.21: Sensitivity analysis of parameters in interactive texture annotation: from left to right, the images depict λ^s for region extraction, $\epsilon^{(seed)}$ for seed matching, and $\epsilon^{(reg)}$ for region matching, with the green star indicating user click input.

in Figure 5.23 top row, our method efficiently extracts and matches different objects in S3DIS [Armeni *et al.*, 2017] indoor meshes. Additionally, our method is equally applicable to the semantic annotation of manually reconstructed models, such as 3D building models. Given that these models typically contain minimal noise, our method

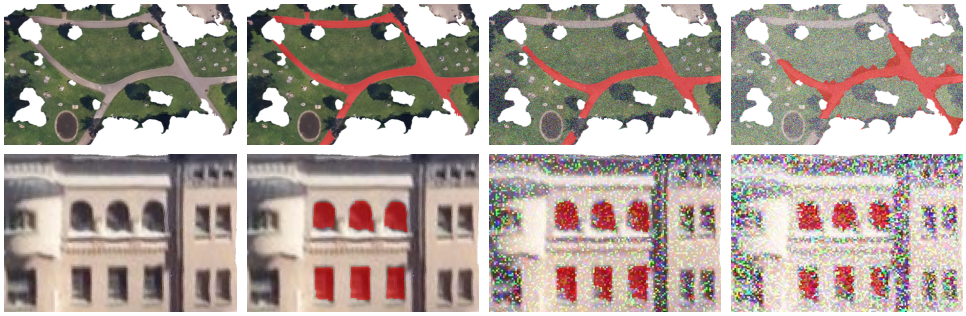


Figure 5.22: Sensitivity analysis of texture image quality in local region extraction: the red area indicates the selected area, and σ^t represents the standard deviation of the Gaussian noise in the texture image.

can achieve more accurate structure extraction and matching. As demonstrated in Figure 5.23 bottom row, our method facilitates rapid structural extraction and selection in CityGML LoD2 [Gröger and Plümer, 2012; OGC, 2012] building models.

For interactive texture annotation, our method can also be extended to semantic annotation of images. Our method is particularly suitable for scenes containing objects with high color and structural consistency. As shown in Figure 5.24, the method can effectively and rapidly extract, segment, and match multiple similar components within an image.

Limitations. Our method cannot process triangular meshes with topological errors, as these errors prevent the correct extraction of neighborhood information. Our method may also be ineffective for some natural scenes that do not fit the assumptions of planar segments and protrusions, such as mountains, riverbeds, and forests, or spaces with complex internal structures like palaces and churches. Additionally, our method's performance may suffer if the mesh resolution is too low or the results from plane-based over-segmentation are poor. This limitation arises mainly because our method relies on geometric precision and structural clarity.

As for the limitations of texture annotation, our method may struggle with textured segments or images with complex textures or cluttered backgrounds because these do not meet our basic assumptions. Additionally, the size of the superpixels can affect performance; superpixels that are too small may lead to increased processing time, while superpixels that are too large may cause under-segmentation, affecting accurate object boundary extraction or incorrectly extracting background areas. Furthermore, our method may underperform in shadowed areas or regions with minimal color differentiation.

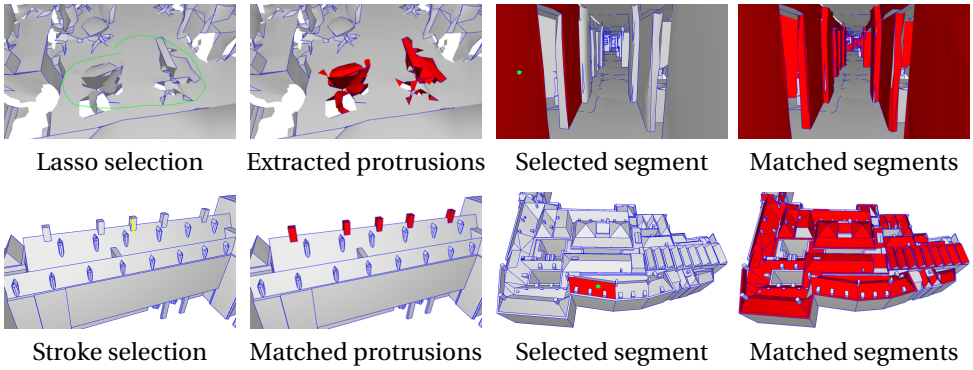
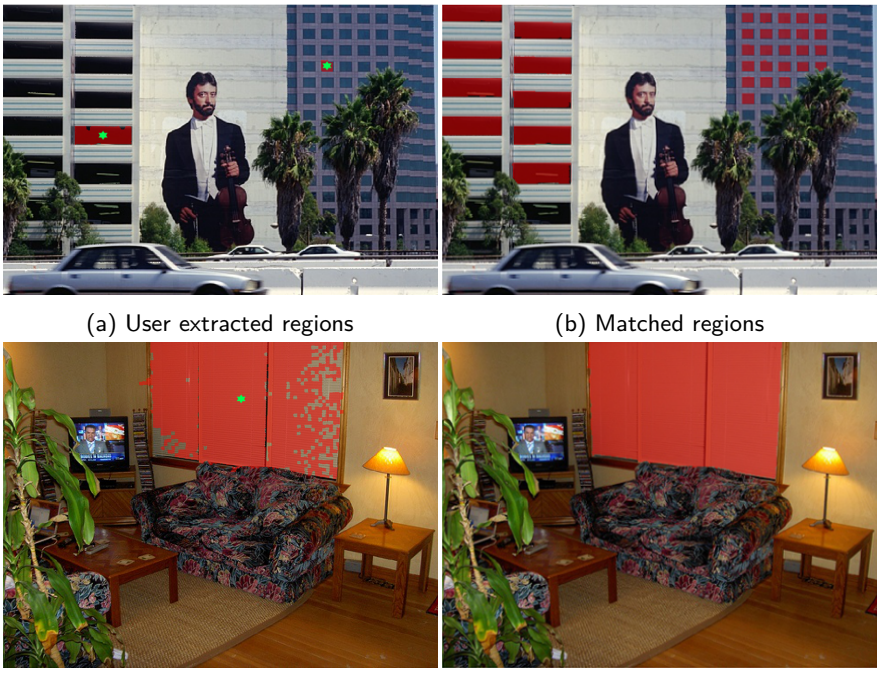


Figure 5.23: Extension of mesh face annotation applications: the first row displays the use of S3DIS meshes [Armeni *et al.*, 2017] for indoor object extraction with our method (the green trajectory indicates user lasso input) and segment-matching results (the green star indicates user click input). To address the presence of high noise in indoor scenes, the geometric sensitivity term η, as described in Equation 5.1, has been increased to 2. The second row illustrates component extraction from CityGML LoD2 [Gröger and Plümer, 2012; OGC, 2012] building models (the yellow trajectory denotes user stroke input) and their matching results (the green star indicates user click input), where the red faces indicate the selected protrusions or segments.

5.7. Conclusion

This paper has introduced a novel interactive annotation framework for semantic segmentation of large-scale urban textured meshes. Our framework, designed for minimal user interaction, enables efficient semantic labeling through two primary techniques: interactive component extraction that capitalizes on local feature differences and structure-aware matching for rapid annotation of similar structures. Experimental results confirm that our method outperforms existing frameworks regarding interaction convenience, efficiency, and accuracy. Additionally, we have developed the first part-level semantic mesh dataset for large-scale urban scenes, providing a comprehensive evaluation against state-of-the-art 3D semantic segmentation methods. Our research paves the way for further advancements in semantic understanding of urban environments.

There are several promising directions for future research. One area is the integration of pre-trained models as prior probabilities, combined with local scene features, to enhance interactive annotation. This approach can optimize annotation, especially for components with regular geometries and intricate boundaries. Another research gap is the development of end-to-end networks that can directly perform semantic segmentation on mesh triangles and their textures, eliminating the need for point cloud sampling. This would simplify the process and improve efficiency. Additionally, new feature-aware modules are needed to effectively capture local geometric density



(a) User extracted regions (b) Matched regions

(c) Local expanded region (d) Fine segmented region

Figure 5.24: Extension of texture image annotation applications: the first row demonstrates local region extraction and matching of a facade image using our method; the second row depicts region extraction from an indoor scene image using our method. The green stars indicate the user clicks, and the red areas indicate the selected pixels.

variations. Exploring these modules could lead to better handling of complex urban scenes. Addressing these gaps will significantly advance the field and improve the accuracy and efficiency of semantic segmentation for urban textured meshes.

6

Lightweight 3D city model reconstruction

From Chapter 3 to Chapter 5, we have explored methods for extracting semantic information from urban textured meshes. This chapter shifts the focus towards applying this semantic information in practical scenarios. It introduces a framework for reconstructing lightweight 3D city models using semantic textured meshes, emphasizing geometric integrity, surface continuity, and semantic retention. The framework comprises algorithms for semantic-based 3D reconstruction and surface simplification. Experiments on the SUM and SUM-Parts datasets demonstrate the framework's effectiveness in reconstructing large-scale 3D city models, with our semantic-based 3D reconstruction methods surpassing other approaches in efficiency and accuracy. The resulting lightweight city models, ranging from LoD2 to LoD3, significantly enhance semantic-based urban applications by improving automation in measurements, spatial computations, and the realism of physical simulations.

6.1. Introduction

With the extensive use of airborne LiDAR and aerial imaging technologies, we are now capable of collecting and generating large-scale urban point clouds [AHN, 2019] and textured meshes [Google, 2012; Helsinki, 2019]. To effectively apply these data in downstream applications, semantic segmentation, and lightweight reconstruction become crucial [Lafarge and Mallet, 2011; Verdie *et al.*, 2015; Xiang *et al.*, 2024]. Despite considerable research into the semantic segmentation of 3D data [Charles *et al.*, 2017; Q. Hu *et al.*, 2020; Thomas *et al.*, 2019], further utilization of this semantically enriched data, especially textured meshes with object and part-level semantic labels [W. Gao *et al.*, 2021; Rouhani *et al.*, 2017; Verdie *et al.*, 2015], remains largely underexplored. Textured meshes, due to their rich inherent semantic information, are theoretically suitable for constructing detailed, multi-level 3D city models.

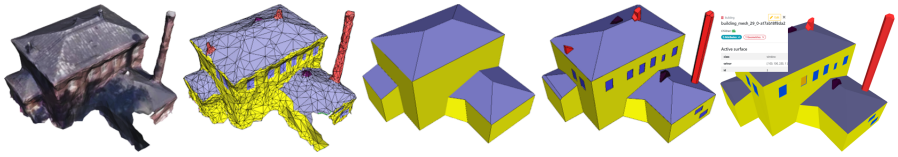


Figure 6.1: Building model reconstruction based on semantic urban texture meshes: the figure above shows, from left to right, input textured building mesh, semantically labeled mesh, reconstructed LoD2 building model, LoD3 building model, and building model displayed in CityJSON format with attributes. The semantic classes shown in the figure include roof surface ■, facade surface ■, chimney ■, roof installation ■, and window ■.

Point cloud and textured mesh data, containing numerous points or triangles, demand substantial storage and increased computational costs for data processing and analysis. To reduce the storage redundancy of these raw data and meet efficiency requirements for downstream applications, a lightweight reconstruction strategy is often adopted, representing the original data with simplified polygons [Verdie *et al.*, 2015; Xiang *et al.*, 2024]. Most current research on city model lightweight reconstruction focuses primarily on buildings [Bouzas *et al.*, 2020; Z. Chen *et al.*, 2022; J. Huang *et al.*, 2022; Peters *et al.*, 2022; Zhu *et al.*, 2017], with many details confined to the Level of Detail 2 (LoD2) standard of CityGML [Gröger and Plümer, 2012; OGC, 2012]. High-accuracy Level of Detail 3 (LoD3) reconstruction methods are scarce [Gruen *et al.*, 2020; Nan *et al.*, 2015; Pantoja-Rosero *et al.*, 2022; L. Wang *et al.*, 2024], and often require additional street-view images or TLS/MLS point clouds with part-level semantic labels to reconstruct semantic elements such as building windows and doors. However, these methods do not guarantee the watertightness and two-manifold nature of the reconstructions, which are generally limited to single buildings and pose challenges when applied in large-scale urban settings. In contrast, our method can reconstruct building models from texture meshes with semantic information ranging from LoD2 to LoD3. These models can then be converted into CityGMLL [Gröger and Plümer, 2012; OGC, 2012] or CityJSON [Ledoux *et al.*, 2019] formats suitable for various 3D urban applications (see Figure 6.1). Research

on lightweight reconstruction for other urban objects is also lacking, and existing studies fail to meet the practical application needs [Du *et al.*, 2019; Verdie *et al.*, 2015].

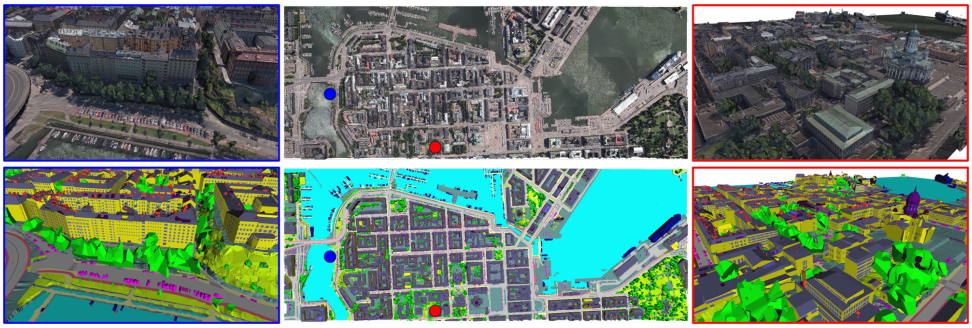


Figure 6.2: Example of a large-scale, lightweight, semantic 3D city model reconstructed from texture meshes: the top images show the texture meshes used as input data, while the bottom images show the output of our method - a semantically enhanced, lightweight city model. The semantic categories depicted in the above figure include high vegetation ■, water ■, car ■, boat ■, roof surface ■, facade surface ■, chimney ■, dormer ■, balcony ■, roof installation ■, window ■, door ■, low vegetation ■, impervious surface ■, road ■, road marking ■, cycle lane ■, and sidewalk ■.

As shown in Figure 6.2, our approach uniquely uses the urban texture mesh as input data. By exploiting the extracted semantic information, it automatically performs a lightweight reconstruction of various urban objects, achieving a large-scale, city-level 3D semantic city reconstruction from LoD2 to LoD3. We believe that lightweight 3D city models should meet the following three criteria:

- Geometric integrity: The simplified 3D solid model should be watertight and two-manifold, using as few polygons as possible while retaining the original geometric structure.
- Surface continuity: Surfaces that cannot be constructed as solid models, such as roads, should be represented as continuous and two-manifold simplified polygons.
- Semantic retention: All lightweight output results should retain the semantic information of the original data.

However, existing methods primarily focus on lightweight reconstruction of geometric solids [Bauchet and Lafarge, 2020; Nan and Wonka, 2017], often neglecting the simplification of non-solid surfaces and effective semantic information transmission. To address this issue, we propose a lightweight 3D city model reconstruction framework based on semantic textured meshes. Our specific contributions include: 1) A framework for lightweight 3D city modeling that includes semantic-based 3D reconstruction and surface simplification. 2) A semantic transmission algorithm designed to preserve the semantic labels of lightweight city models derived from the input data. 3) The first

automatically reconstructed large-scale lightweight city model dataset with multi-level detailed semantic information derived from semantically annotated urban textured meshes. 4) An in-depth discussion of potential applications for large-scale 3D city models with rich semantic information, offering new perspectives for future research directions.

6.2. Related work

With extensive research into lightweight city model reconstruction, this section focuses on fully automatic reconstruction methods using 3D data (point clouds and meshes), excluding image-based, generative, manual, and semi-automatic approaches. Fully automatic lightweight reconstruction methods fall into four main categories: primitive fitting, spatial partitioning, mesh simplification, and deep learning-based approaches.

The method of primitive fitting employs predefined geometric models (such as planes, cubes, cones, spheres, or roof primitives) to fit the input 3D data using a cost function [Z. He *et al.*, 2021; J. Jiang *et al.*, 2023; Lafarge and Mallet, 2011; Nan *et al.*, 2010; Nan and Wonka, 2017; Xiong *et al.*, 2015; W. Zhang *et al.*, 2021]. It efficiently reconstructs watertight and manifold models, ideally suited for buildings or trees and applicable to large-scale reconstructions. However, its generalization ability is limited due to the reliance on predefined geometric shapes, making it unsuitable for complex geometries or tasks demanding high geometric precision.

The spatial partitioning approach can be further divided into 2D and 3D spatial partitioning. 2D methods often handle aerial point clouds or meshes by projecting them onto a 2D plane and constructing 2D grids or planar partitions, and then extruding these into 2.5D models [J. Han *et al.*, 2021; Kelly *et al.*, 2017; Peters *et al.*, 2022; Sui *et al.*, 2016; Zhu *et al.*, 2018; Zhu *et al.*, 2017]. 3D methods utilize planar primitives or profiles for partitioning, ideal for constructing models with complex geometries and ensuring their watertightness via topological constraints [Bauchet and Lafarge, 2020; Chauve *et al.*, 2010; Haala *et al.*, 2007; Kada and Wichmann, 2013; Verdie *et al.*, 2015]. However, as the complexity of the input data increases, the time required to solve the cost function for these methods may become excessively long, making them unsuitable for large-scale, high-precision 3D reconstructions.

The mesh simplification method targets triangular meshes, reducing mesh facets by merging or reconstructing uniformly distributed triangles [Bouzas *et al.*, 2020; X. Gao *et al.*, 2022; M. Li and Nan, 2021; Y. Liu *et al.*, 2022a; Salinas *et al.*, 2015]. Although effective in simplifying models, this method requires topologically correct and continuous input meshes. However, it may lose important geometric details due to excessive smoothing, making it unsuitable for most large-scale urban scenes. Furthermore, this method performs poorly in preserving unique architectural features, such as chimneys and decorative details, limiting its application in high-fidelity modeling.

Deep learning-based methods heavily rely on large-scale manually reconstructed

models as training data, learning implicit fields' predictions [Z. Chen *et al.*, 2022; Z. Chen *et al.*, 2023] or the geometric structures (such as vertices and edges) and their topological relationships [S. Huang *et al.*, 2024; L. Li *et al.*, 2022; Y. Liu *et al.*, 2024]. Although their generated models depend on the accuracy of the predictions, these methods have poor generalization abilities and perform poorly in processing complex models or unseen scenes, and cannot guarantee both the two-manifold nature and watertightness of the models.

Despite the advantages of these methods, they are generally unsuitable for reconstructing large-scale, lightweight city models with comprehensive and high-level semantic details, as most are only suitable for building models. In contrast, our proposed framework utilizes the semantic information of textured meshes to reconstruct various types of urban objects and their parts, producing lightweight large-scale city models that are two-manifold, watertight (except semantic surfaces), highly detailed, and rich in semantic information.

6.3. Overview

Our goal is to utilize semantic information to simplify urban textured meshes, aiming to replace the numerous triangles with a smaller number of simplified polygons. Notably, polygonization is mainly suitable for 3D objects that can be divided into planar segments, such as buildings and vegetation, or surfaces with regular boundaries, like roads, but not for irregular surfaces, such as terrain, grassland, or water. Our framework processes textured meshes enriched with semantic information by reconstructing or simplifying each semantic category individually, then merging them and accordingly transferring the semantic labels. We divide semantic-based lightweight reconstruction into two main types: semantic-based 3D reconstruction and surface simplification.

Semantic-based 3D reconstruction aims to convert 3D objects, composed of point clouds sampled from the meshes, into boundary representations (see from Figure 6.3 and Figure 6.4). This is mainly used for objects requiring watertightness, such as buildings and their semantic parts, trees, and vehicles. Using semantic information, we identify and extract connected components, not instances, of the same semantic class within the urban textured mesh. We then conduct individual semantic-based 3D reconstructions on these connected components using the PolyFit algorithm [Nan and Wonka, 2017] to achieve polygonization. For complex objects with multiple semantic parts, each is independently reconstructed and eventually merged with others, maintaining the semantic labels of the original mesh. The advantage of using semantic information lies in reducing the number of variables handled in surface extraction, particularly the number of planar segments. Additionally, 3D reconstruction methods that use uniform parameters for planar partition often overlook small or geometrically minor components, such as windows or chimneys, resulting in ineffective reconstructions [Bauchet and Lafarge, 2020; Nan and Wonka, 2017]. By using different parameters for planar partition across various semantic classes, the reconstruction

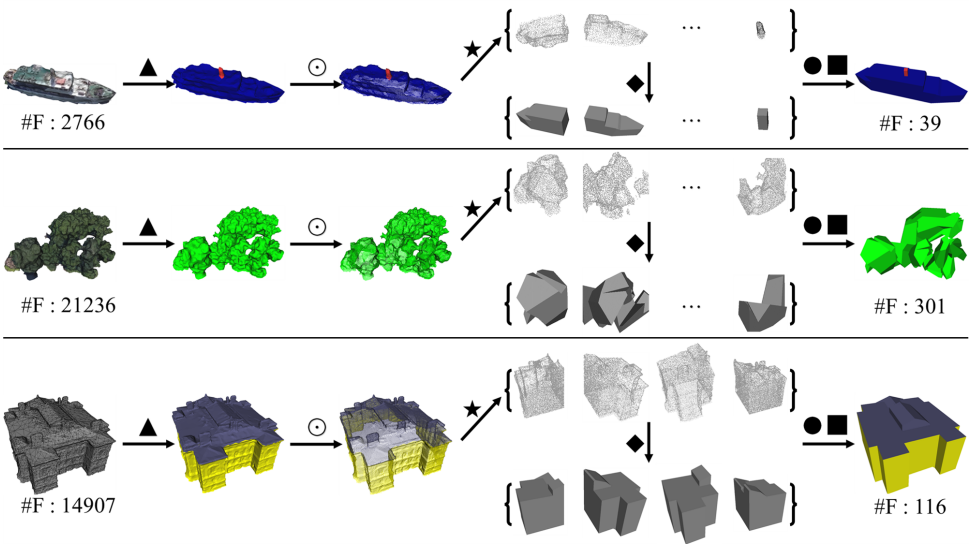


Figure 6.3: LoD2 model reconstruction pipeline: first, we semantically label (▲) the triangular faces of the input textured mesh, followed by surface point cloud sampling (⊙); we then spatially partition (★) the sampled point cloud and reconstruct (◆) each segmented sub-cloud; finally, we use Boolean operations (●) to fuse the results and add semantic information (■) to the lightweight model. #F denotes the number of faces in the current model.

results can be more complete and accurate.

The goal of surface simplification is to geometrically simplify objects with non-closed continuous surfaces, using fewer polygons or triangles, mainly applicable to terrain, water, grasslands, and roads (see Figure 6.5). Based on the semantic information, we employ two simplification strategies: planar surface simplification and triangular mesh simplification. Planar surface simplification is suitable for objects that can be represented by continuous planar surfaces with regular boundaries, such as roads and cycle lanes. Triangular mesh simplification, on the other hand, is suitable for irregular surfaces better represented by a mesh of triangles, such as terrain, water, or grasslands. Using semantic information, we extract and simplify connected surfaces within the same semantic class. Compared to traditional methods that primarily focus on terrain mesh simplification [K. Kumar *et al.*, 2018; Ledoux *et al.*, 2023], the use of semantic information allows for the adoption of appropriate simplification methods for different types of ground surfaces, thus achieving further simplification of urban meshes.

It should be noted that textured meshes are not ideally suited for direct use in lightweight 3D city model reconstructions due to potential issues like mesh tiling and topological errors. In the post-process of constructing large-scale urban meshes, splitting the mesh into tiles for easier manipulation often disrupts the topological continuity, posing challenges in reassembling tiles with preserved texture and topology. Furthermore,

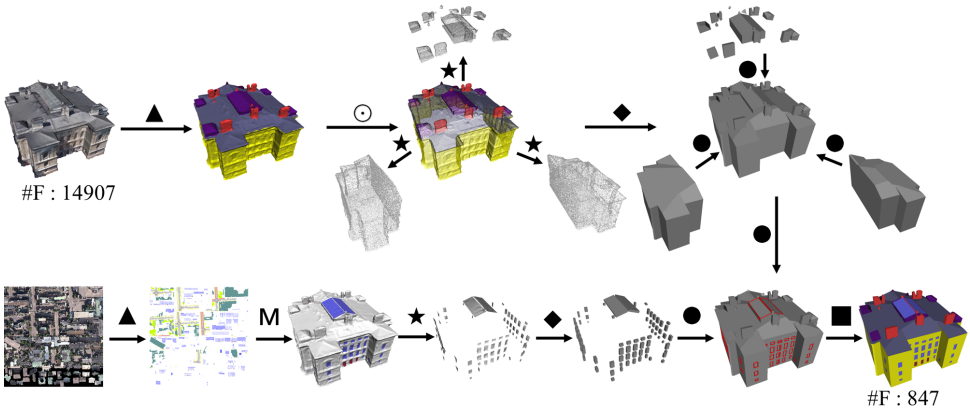


Figure 6.4: LoD3 building model reconstruction pipeline: initially, we semantically label (▲) the triangular faces of the input mesh and its mapped (M) texture images, followed by surface point cloud sampling (⊙); we then divide (★) the point cloud into different components based on spatial and semantic criteria and reconstruct (◆) each component; finally, we merge these components using Boolean operations (●), transferring the semantic labels (■) to the reconstructed building model. #F denotes the number of faces in the current model.

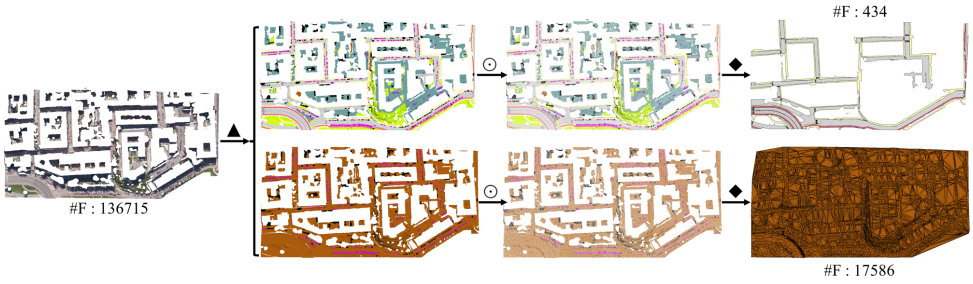
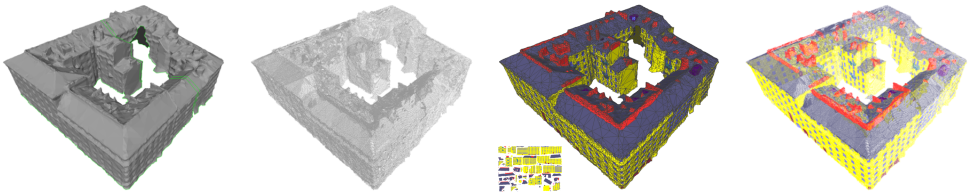


Figure 6.5: Surface simplification pipeline: first, we semantically label (▲) the triangular faces of the input mesh and the mapped texture pixels, followed by surface point cloud sampling (⊙); we then project regular features such as roads onto a plane to extract polygons (◆) and simplify (◆) the terrain through irregular surface triangulation. #F denotes the number of faces in the current surface.

such meshes may exhibit topological errors [Attene *et al.*, 2013], such as non-manifold vertices and self-intersecting faces, which impede accurate model reconstruction. In contrast, point cloud-based methods, which avoid these topology-related issues, are most suitable for reconstructing the geometric structure of lightweight city models (see Figure 6.6a and Figure 6.6b).

Despite this, we believe that urban meshes are ideal carriers for semantic information due to their continuous surfaces made of triangles that can approximate any object's



(a) Mesh tiles (b) Sampled points (c) Semantic meshes (d) Semantic points

Figure 6.6: Mesh vs. point cloud: (a) shows the mesh tiles, with the green line indicating the mesh boundary; (b) shows the point cloud after sampling the mesh surface; (c) shows the semantic texture mesh, where the semantic information is stored on the mesh faces and the texture masks, respectively. The binary storage of the data in the figure occupies a total space of about 1.3 Mb; (d) denotes the semantic dense point cloud, each point of which corresponds to a semantic label. The binary storage of the data in the figure occupies a total space of about 69.3 Mb.

geometry and their textured images, which contain rich semantic details. Compared to point clouds and images with semantic labels, meshes that store semantic information through face labels and texture masks require the least storage space and are easier to handle (see Figure 6.6c and Figure 6.6d). Obtaining ground-truth semantic labels is simpler with textured meshes than with point clouds and images [W. Gao *et al.*, 2021], and they can be freely converted between these formats. Overall, textured meshes are best suited for reconstructing semantic details of large-scale city models, offering both the convenience of semantic annotation and flexibility in format conversion, making them ideal for lightweight semantic 3D city model reconstruction.

6.4. Semantic-based 3D reconstruction

Our objective is to perform 3D model reconstruction from semantic textured meshes, aiming to create watertight, 2-manifold polygonized models with semantic labels. Our method is divided into four main steps: 1) semantic subdivision, 2) 3D model reconstruction, 3) texture parts reconstruction, and 4) integration.

1) Semantic subdivision. In this step, we first subdivide the input textured meshes into 3D mesh objects and 2D texture parts using semantic information. For 3D mesh objects, we further subdivide them into main structures, such as the building's main body, and 3D semantic parts, including chimneys and dormers. After subdivision, the connection regions may be missing. We then fill any holes in their connection regions to ensure geometric integrity (see Figure 6.7). Next, we randomly sample the mesh surfaces to generate semantic point clouds for subsequent 3D model reconstruction.

For texture parts such as the doors and windows of buildings, we directly extract them from the semantic textured masks and transform them into 3D point clouds using

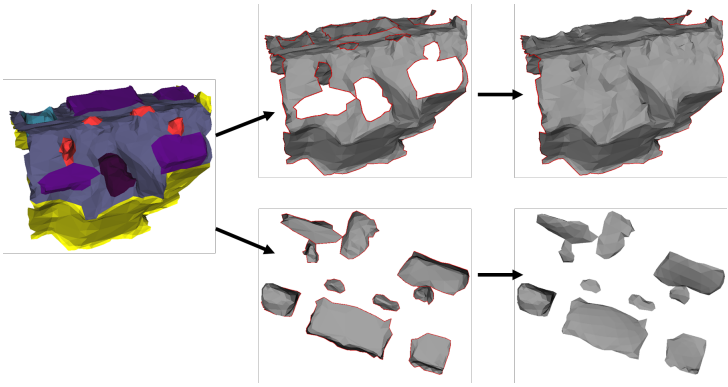


Figure 6.7: Semantic subdivision and hole filling: based on semantic labels, the mesh is divided into a main structure and 3D semantic parts. The holes in the connected regions are filled after the division, and the red lines indicate the hole boundaries.

texture coordinates. This ensures that the geometric characteristics of texture parts remain consistent during their reconstruction.

It is important to note that directly simplifying the mesh could introduce complex problems due to potential topological errors, including non-manifold vertices or edges, duplicate vertices, and gaps caused by tiling (see Figure 6.6). Therefore, we opt for point cloud-based 3D reconstruction, a strategy that avoids problems associated with topological errors and enhances the robustness of the reconstruction process.

2) 3D model reconstruction. This step concentrates on reconstructing solid models from sampled point clouds of 3D mesh objects created in the initial phase. It produces watertight polygonal models that conform to the 2-manifold requirement. Initially, we perform over-segmentation of the main structures and their 3D semantic part point clouds to identify various planar segments. After regularizing these segments [Verdie *et al.*, 2015], we apply the PolyFit algorithm [Nan and Wonka, 2017] for surface extraction.

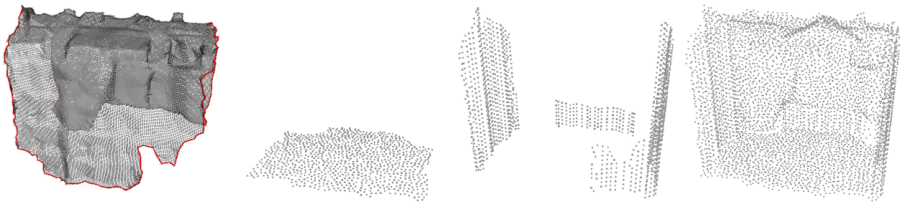


Figure 6.8: Completion of the mesh-sampled point cloud: from left to right are the incomplete point cloud, the projected ground point cloud, the mesh boundary interpolated vertical surface point cloud, and the complete downsampled point cloud.

To enhance the efficiency and completeness of the reconstruction process, we perform additional processing on the main structures and their 3D semantic parts. The main issues with the 3D main structure point clouds may include missing point clouds and excessive planar segments. Missing point clouds, especially large-scale structural absences caused by obstructions like missing ground and facade of buildings due to occlusion or mesh tiling, can result in the inability to reconstruct complete models. As depicted in Figure 6.8, we complete the ground by projecting the point clouds to the lowest horizontal plane. Vertical absences are addressed by downward vertical interpolation of the mesh boundaries. Compared to the vertical surface completion of aerial point clouds in City3D [J. Huang *et al.*, 2022], which involves extracting polylines from projected points and extruding them using a height map, our method is more intuitive and direct. This advantage is due to the topological properties of the mesh, which make it easier to extract mesh boundaries. The process of our approach involves calculating the elevation difference $\Delta z_i = z_i - z_{\min}$ for each point (x_i, y_i, z_i) along the mesh border. Interpolation is performed only when Δz_i exceeds the minimum threshold e_{min}, which is set to 1.0 meter for all experiments in this paper. The number of interpolated points n_i is determined by $n_i = \left\lfloor \frac{\Delta z_i}{\delta} \right\rfloor$, where δ is the constant vertical interval, set to 0.2 meters across all experiments. Each interpolated point q_{ij} is given by $q_{ij} = (x_i, y_i, z_i - j \cdot \delta)$ for $j = 0, 1, \ldots, n_i - 1$. This sequence generates a set of vertically interpolated points Q, effectively filling the vertical gaps along the mesh boundary.

For geometrically complex objects like courtyard complexes and trees, an excessive number of planar segments can significantly reduce the computational efficiency during PolyFit surface extraction [Nan and Wonka, 2017]. We employ a divide-and-conquer strategy inspired by Bauchet and Lafarge, (2020), using spatial subdivision to divide the main structure S_{main} into different sub-point clouds S_{sub}. This approach effectively reduces the number of planar segments required for each individual reconstruction. Subsequently, each sub-point cloud is processed with the PolyFit algorithm [Nan and Wonka, 2017] and integrated into a coherent whole model through mesh union operations (as illustrated in Figure 6.3, Figure 6.4, and Algorithm 6.1).

For point clouds representing 3D semantic parts S_{parts} of the main structure, as well as for sub-point clouds derived from the main structure, their reduced scale and fewer inliers may lead to a failure in detecting sufficient planar segments, which can hinder effective 3D reconstruction. Therefore, we adopt an adaptive reconstruction approach (depicted in Algorithm 6.1), initially using predefined over-segmentation parameters for trial reconstructions. If the initial reconstruction does not meet the preset volume ratio compared to the original mesh, we adjust the over-segmentation parameters and retry. This process is repeated until the conditions are met or the maximum number of attempts is reached. This method allows us to adjust the reconstruction strategy while maintaining the original geometric shape, ensuring the model M_{main} closely matches the true structure of the original data.

Algorithm 6.1 details the divide-and-conquer algorithm for 3D model reconstruction. Lines 1 to 2 represent the divide phase, encompassing semantic and spatial subdivisions.

Algorithm 6.1 Divide and Conquer with Adaptive Reconstruction

1: **Input:** $S_{semantic\_point\_cloud}$
2: **Output:** M_{main}, M_{parts}
3: $\{S_{main}, S_{parts}\} \leftarrow$ semantic_subdivision($S_{semantic\_point\_cloud}$) ▷ Start divide
4: $S_{sub} \leftarrow$ spatial_subdivision(S_{main})
5: $M_{parts} \leftarrow \emptyset$, $M_{sub} \leftarrow \emptyset$ ▷ Start adaptive reconstruction
6: $N_{attempts} \leftarrow 3$, $r_{min} \leftarrow 50$, $d_{max} \leftarrow 0.5$, $\theta_{max} \leftarrow 25.0$
7: $\Delta_r \leftarrow r_{min}/N_{attempts}$
8: $\Delta_d \leftarrow d_{max}/N_{attempts}$
9: $\Delta_\theta \leftarrow \theta_{max}/N_{attempts}$
10: **for** $p \in \{S_{parts}, S_{sub}\}$ **do**
11: $to\_stop \leftarrow$ false
12: $V_{\min} \leftarrow \alpha_{vol} \times$ volume(optimal_bounding_box(p))
13: **while** not to_stop **do**
14: planar_segments_detection($p, r_{min}, d_{max}, \theta_{max}$)
15: $m_i \leftarrow$ polyfit(p)
16: **if** is_closed(m_i) $\wedge$ volume(m_i) $\geq V_{\min}$ **then**
17: M_{parts}.append(m_i) or M_{sub}.append(m_i)
18: $to\_stop \leftarrow$ true
19: **else**
20: **if** is_empty(m_i) **then**
21: $r_{min} \leftarrow r_{min} - \Delta_r$
22: $d_{max} \leftarrow d_{max} - \Delta_d$
23: $\theta_{max} \leftarrow \theta_{max} - \Delta_\theta$
24: **else**
25: $r_{min} \leftarrow r_{min} + \Delta_r$
26: $d_{max} \leftarrow d_{max} + \Delta_d$
27: $\theta_{max} \leftarrow \theta_{max} + \Delta_\theta$
28: **end if**
29: **end if**
30: **end while**
31: **end for**
32: $M_{main} \leftarrow M_{sub}[0]$ ▷ Start conquer
33: **for** $i = 1$ to size(M_{sub}) **do**
34: union($M_{main}, M_{sub}[i]$)
35: **end for**

Lines 3 to 29 elaborate on our proposed adaptive reconstruction algorithm. In this phase, M_{parts} and M_{sub} are the collections of models reconstructed from S_{parts} and S_{sub}, respectively. The parameters r_{min}, d_{max}, and θ_{max} signify the minimum number of points per planar segment, the maximum distance from points to a plane, and the maximum angle between point normals and the plane, respectively. $N_{attempts}$ specifies the number of attempts at reconstruction, while Δ_r, Δ_d, and Δ_θ are the update values for the number of points, distance, and angle with each attempt. $V_{\min}$ denotes

the minimum acceptable volume for the reconstructed models, which is derived from multiplying a predefined volume ratio $\alpha_{vol} = 0.3$ by the volume of the optimal bounding box of the input point cloud. Lines 30 to 33 depict the conquer phase, where M_{main} corresponds to the model reconstructed from S_{main}. In subsequent processes, M_{parts} will be integrated through a Boolean operation binary tree. Note that we use the default settings of the PolyFit [Nan and Wonka, 2017] parameters as stated in the paper, and we do not update them during adaptive reconstruction to ensure robust results. The default values for these parameters are the same across all experiments in this study.

3) Texture parts reconstruction. This step aims to reconstruct 3D models from the point clouds of textured parts, such as windows or doors in the texture masks, and consists of two main processes: regularization and extrusion. Texture parts regularization comprises three processing modules: global regularization, boundary regularization, and local regularization. Global regularization groups each part and matches it with corresponding main structure planar segments, aligning the parts segments with the main structure plane (see Algorithm 6.2).

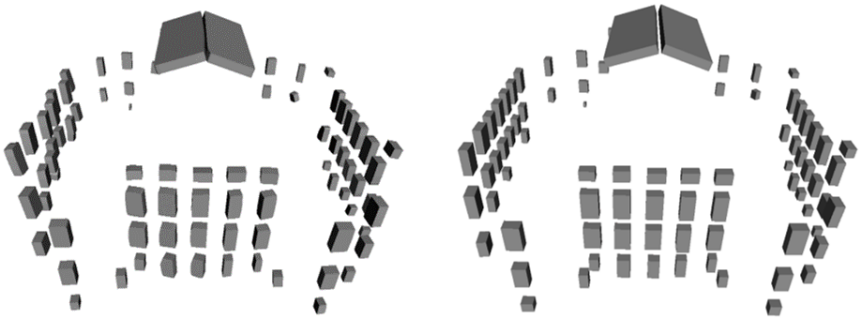
Algorithm 6.2 Texture Part Global Regularization

Input: $\{plane_i\}_{i=1...N_{tex}}$
Output: $\{plane_i^{'}\}_{i=1...N_{tex}}$
$d_{\max} \leftarrow 1.0$, $\theta_{\max} \leftarrow 25.0$
for $i = 0$ to N_{tex} **do**
    $count_{max} = 0$
    for $j = 0$ to N_{main} **do**
        if $\text{dist}(C_i, plane_j) < d_{\max}$ **and** $\cos^{-1}(\mathbf{n}_i \cdot \mathbf{n}_j) < \theta_{\max}$ **then**
            $count \leftarrow 0$
            for p in P_i **do**
                $p_{proj} \leftarrow \text{projection}(p, plane_j)$
                if $\text{point\_in\_polygon}(p_{proj}, polygon_j)$ **then**
                    $count \leftarrow count + 1$
                end if
            end for
            if $count > count_{max}$ **then**
                $count_{max} \leftarrow count$
                $plane_{match} \leftarrow plane_j$
            end if
        end if
    end for
    $plane_i^{'} = plane_{match}$
end for

In Algorithm 6.2, $\{plane_i\}$ and $\{plane_i^{'}\}$ are the input and output planes of the texture part point cloud. $d_{\max}$ and $\theta_{\max}$ respectively denote the maximum distance from a

point to a plane and the maximum angle between their normal vectors. N_{tex} represents the number of texture part point clouds, while N_{main} indicates the number of planar segments in the main structure. C_i and $\mathbf{n}_j$ correspond to the center point of a texture part and the normal vector of its fitted plane, respectively. $plane_j$ and $\mathbf{n}_j$ represent the plane and its normal vector for the j^{th} planar segment of the main structure, and $plane_{match}$ denotes the plane of the main structure that matches with texture part i. The default values for these parameters remain consistent throughout all experiments in this study.

Boundary regularization uses a convex hull algorithm [Bykat, 1978] to identify and extract boundaries of part point clouds. These boundaries are then simplified using a Euclidean distance-based cost function and subjected to multi-angle and offset regularization to ensure regularity and accuracy [Bauchet and Lafarge, 2018]. Local regularization uniformly applies angle and offset treatments to all closed boundaries within the same global regularization group, ensuring parallelism and consistent offsets [Bauchet and Lafarge, 2018]. We illustrate the differences between unregularized and regularized texture part models in Figure 6.9.



(a) Before regularization (b) After regularization

Figure 6.9: Texture parts regularization

The extrusion stage follows the regularization processes, first using the extracted boundaries to construct a planar mesh using constrained Delaunay triangulation [D.-T. Lee and Schachter, 1980]. The planar mesh is then extruded along the normal direction of the textured parts and in the opposite direction, forming 3D structures. The length of the extrusion is determined by the semantic class (e.g., 10 cm for the window).

4) Integration. This step involves integrating solid models with their parts and semantics to create a polygonal surface model that preserves the original semantic information. It ensures the final model is watertight and adheres to the two-manifold requirement.

Solid model integration aims to merge the reconstructed 3D main structure models, 3D semantic part models, and the texture part extruded models into a unified polygon surface model. To achieve this, we construct a Boolean operation [Q. Zhou *et al.*, 2016] binary tree for solid geometric integration, as shown in Figure 6.10. The 3D parts are classified as internal or external, depending on their relationship with the 3D main structure. We embed the texture part model by subtracting their extruded models from the corresponding 3D semantic part models. Subsequently, internal 3D semantic parts are subtracted from the main structure to reveal embedded surfaces. Finally, a Boolean union merges the 3D main structure with its part models into a coherent surface model.

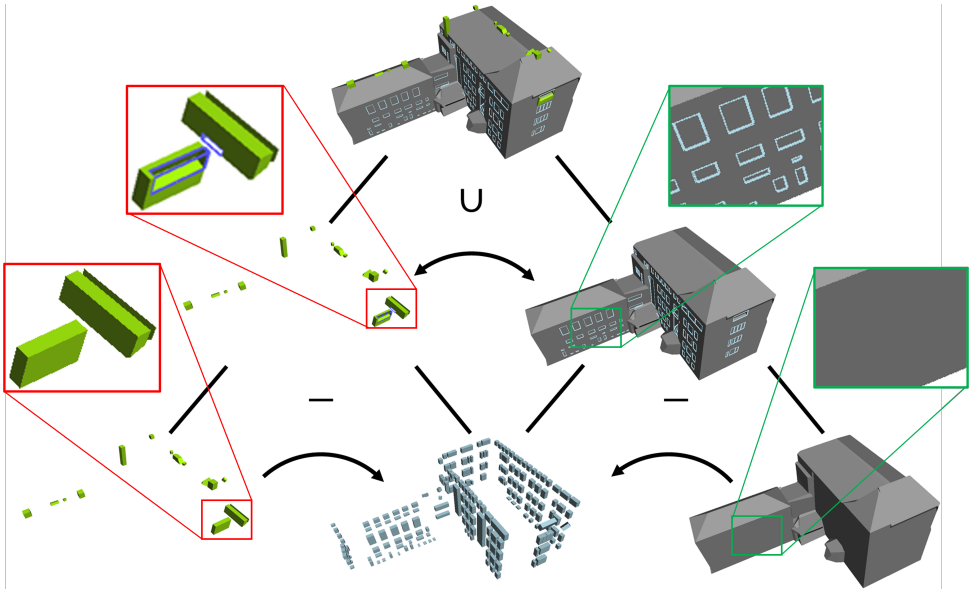
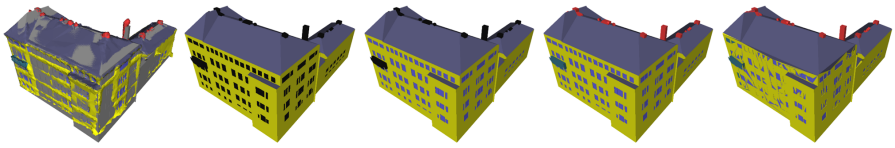


Figure 6.10: Boolean binary tree for solid model integration: we first use 3D main structure models and 3D semantic part models, respectively, to embed geometric information of texture part extruded models into their corresponding models using Boolean subtraction operation; then we perform Boolean union of the obtained results to obtain the final watertight surface model.

Semantic integration aims to effectively transfer the labels of the original textured mesh to the integrated geometric model. This process begins with a semantic initialization of the main structure using a semantically down-sampled point cloud. The solid model is initially partitioned into different planar segments through over-segmentation. Subsequently, the semantic point cloud is assigned to the corresponding planar segments. Similar to the method outlined in Algorithm 6.2, we coarsely match each point from the semantic point cloud to the planar segments by considering both the point-to-plane distance and the angle between the normals of the points and the planar segments. For points belonging to a specific planar segment, we use a voting

mechanism to decide the final labels.



(a) (b) (c) (d) (e)

Figure 6.11: Example of semantic integration: where (a) shows the result of the mixing between the semantic point cloud (in color) and the reconstructed model (in grey), which can be seen to have varying degrees of geometrical deviation from the original point cloud to the reconstructed model; (b) shows the result of the semantic initialization of the 3D main structure; (c) shows the result after integrating the texture part models; (d) shows the result after integrating the 3D semantic part models; (e) shows the results after semantic label voting using a traditional KNN approach.

Given the potential geometric deviations between the reconstructed model and the original mesh (or the sampled point cloud), especially in 3D parts and texture parts, semantic transfer based on nearest neighbors might lead to significant errors. To reduce these errors, we propose a Boolean operation-based semantic transfer optimization strategy (see Figure 6.11). This strategy establishes a correspondence between each face in the merged model and part models from previous steps, effectively assigning semantic labels (see Figure 6.11). After transferring semantic labels to the merged model, we create a fully semantically informed solid model.

6.5. Semantic-based surface simplification

The objective of this step is to use semantic information to geometrically simplify surfaces that cannot be utilized for 3D model reconstruction. This involves planar surface simplification and triangular mesh simplification. To ensure the accuracy of the simplification process and avoid the effects of topological errors and detail reconstruction errors, we use semantic point clouds obtained through mesh sampling as the input data for surface simplification.

Planar surface simplification is primarily applied to surfaces with regular boundaries, including roads, sidewalks, cycle lanes, road markings, etc. We first identify and extract these point clouds' connected components based on semantics. Due to occlusion, these surfaces may contain holes or missing parts. For instance, parts of the road occupied by cars will have gaps after the cars are removed. To address this issue, we first establish a semantic hierarchy of containment, where the road is the parent class and cars are the child class. When constructing the surface of the semantic parent class, all adjacent connected child class point clouds are projected onto the plane of the parent class to fill the missing parts. Next, for each type of connected component, we

acquire planar segments via over-segmentation and then project the point clouds onto the corresponding planes. We then use the Alpha Shape algorithm [Bernardini and Bajaj, 1997] to extract boundaries and construct planar polygons (see Figure 6.5).

Triangular mesh simplification targets surfaces with irregular boundaries or uneven surfaces, such as terrain, water, and grasslands. In this process, we first simplify the sampling points uniformly. Then, utilizing the Delaunay triangulation [D.-T. Lee and Schachter, 1980] method, we construct a triangular mesh, which sets the stage for subsequent simplification steps. After obtaining the initial triangular mesh, the Lindstrom-Turk cost [Lindstrom and Turk, 1998; Lindstrom and Turk, 1999] and position strategy are applied to further refine the mesh, optimizing its quality and performance (see Figure 6.5).

Through these two simplification methods, we can effectively handle various types of urban mesh surfaces, whether regular or irregular, achieving geometric simplification through appropriate strategies.

6.6. Experiments

6.6.1. Dataset and evaluation metrics

We selected the SUM dataset [W. Gao *et al.*, 2021] to test our methods, which is currently the most extensive known semantic textured mesh dataset (see Chapter 3). Semantic labels in this dataset are primarily based on mesh faces and include categories such as terrain, buildings, water bodies, tall vegetation, vehicles, and boats. Additionally, we developed SUM-Parts (see Chapter 5), an extension of SUM, which includes both mesh face and texture labels like terrain, high vegetation, water, cars, boats, walls, roof surfaces, facade surfaces, chimneys, dormer windows, balconies, roof installations, windows, doors, low vegetation, impervious surfaces, roads, road markings, cycle lanes, and sidewalks. We conducted tests over an area of 1.5 km^2 that overlaps with SUM and SUM-Parts, using our proposed framework to perform lightweight reconstructions for all semantic categories.

The lightweight reconstruction results were evaluated based on four criteria: compression degree, efficiency, geometric accuracy, and semantic accuracy. The compression ratio is measured by the difference in the number of facets before and after reconstruction. The efficiency is quantified by the time (in seconds) required to complete the task. Geometric accuracy is evaluated using symmetric mean Hausdorff distance (H-dis), Chamfer distance (C-dis), and root-mean-square error (RMSE). Note that the evaluation metric of geometric accuracy is based on point clouds sampled from the reconstructed model surface (with a random sampling density of 10 points per m^2). Semantic accuracy is quantified by the mean intersection over Union (mIoU) and mean accuracy (mAcc) based on the manually annotated reconstructed models.

6.6.2. Implementation details

Our test environment utilizes a computer equipped with an Intel Core i7-7700HQ CPU (4 cores, 2.8 GHz) and 32GB of RAM. We build our software using C++ and open-source libraries such as Boost, CGAL [The CGAL Project, 2024], and Easy3D [Nan, 2021a]. For PolyFit surface extraction, we employ the Gurobi 11.0 solver.

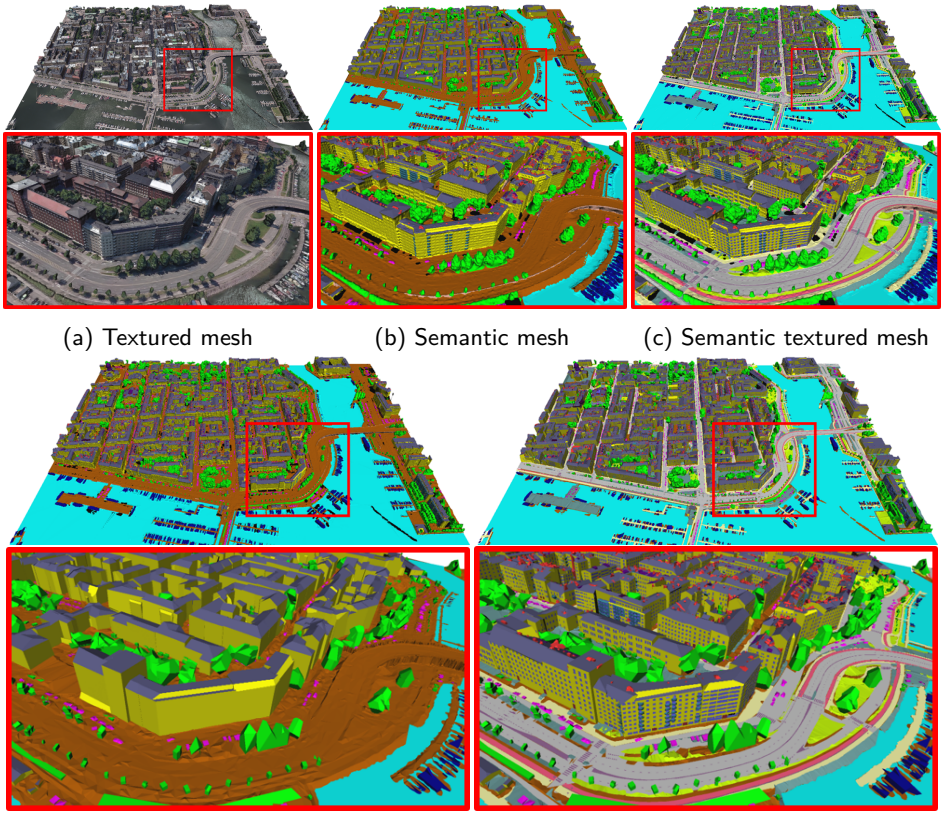
6.6.3. Results

We conducted quantitative (see Table 6.1) and qualitative (see Figure 6.12) evaluations of our method on the SUM [W. Gao *et al.*, 2021] and SUM-Parts datasets. For the SUM dataset, we performed LoD2 model reconstructions of buildings, vegetation, cars, and ships. Meanwhile, we applied mesh simplification to the terrain and water. For the SUM-Parts dataset, we performed LoD3 model reconstructions of buildings and planar surface simplification of roads and their markings, cycle lanes, sidewalks, low vegetation, and impervious surfaces, producing LoD3 surface models. The approach to processing terrain, water, vegetation, cars, and ships was consistent with that on the SUM dataset.

	B-LoD2.	B-LoD3.	H-veg.	Car	Boat	Road	L-veg.	Imp.	Water	Terr.
C. / A. (km^2)	147	147	682	1547	273	0.32	0.08	0.24	0.46	1.58
Fdi-N. (k)	2711	2711	1223	168	177	873	873	873	103	873
Fdo-N. (k)	37	802	57	38	6	54	75	162	5	175
$\overline{C-dis.}$	1.18	0.97	0.74	0.49	0.92	9.98	4.25	3.60	0.31	4.44
$\overline{H-dis.}$	1.76	1.32	0.90	1.17	1.18	17.37	8.40	6.88	0.32	8.43
$\overline{RMS.}$ (%)	4.70	4.15	6.10	6.90	5.25	2.78	0.03	0.16	0.03	2.90
Time-P. (h)	4.6	18.2	3.52	0.56	0.19	-	-	-	-	-
Time-A. (h)	14.53	69.63	6.82	0.98	0.28	2.57	0.22	0.76	0.15	0.17

Table 6.1: Quantitative analysis of lightweight reconstruction in the SUM and SUM-Parts datasets: where C. represents the total number of connected components for category counting in the middle column; A. indicates the total surface area in square kilometers for category counting in the right column; Fdi-N. denotes the total number of input triangular faces in thousands (k); Fdo-N. represents the total number of output polygonal faces, also in thousands (k); $\overline{C-dis.}$ refers to the average Chamfer distance; $\overline{H-dis.}$ refers to the average Hausdorff distance; $\overline{RMS.}$ is the average RMSE, calculated as a percentage of the optimal bounding box length per object or surface; Tim-P. indicates total time spent on parallel processing using a quad-core CPU, while Tim-A. denotes cumulative reconstruction time per object, both in hours (h); B-LoD2 and B-LoD3 designate LoD2 and LoD3 building models, respectively; H-veg. signifies high vegetation; L-veg. denotes low vegetation; Imp. stands for impervious surfaces; Terr. refers to the terrain.

Due to the inclusion of only semantic segmentation and not instance segmentation labels in SUM and SUM-Parts, the input data for Table 6.1 middle column, which includes buildings, high vegetation, cars, and boats, primarily consists of connected components. Compared to instance segmentation as input data, connected



(a) Textured mesh (b) Semantic mesh (c) Semantic textured mesh

(d) SUM results (e) SUM-Parts results

Figure 6.12: Lightweight 3D semantic city model reconstruction results: where (a) shows the input textured meshes; (b) shows the semantic mesh with face labels; (c) shows the semantic mesh with both face and texture labels; (d) shows the SUM reconstructed models at LoD2; (e) shows the SUM-Parts reconstructed models with LoD3 building and road models. The semantic classes shown in the figure include high vegetation ■, water ■, car ■, boat ■, roof surface ■, facade surface ■, chimney ■, dormer ■, balcony ■, roof installation ■, window ■, door ■, low vegetation ■, impervious surface ■, road ■, road marking ■, cycle lane ■, and sidewalk ■.

components typically include larger-scale and more geometrically complex objects such as building blocks and interconnected groups of trees, which pose significant challenges to reconstruction algorithms in terms of efficiency.

From the middle column of Table 6.1, it is evident that our algorithm reduces the number of faces by approximately 70% to 98% (Fdi-N. indicates the number of input faces, and Fdo-N. indicates the number of output faces, both in thousands). The highest compression is observed in LoD2 buildings, high vegetation, and boats, whereas the compression is lower for LoD3 building models with rich semantic details and

vehicles with lower geometric precision. Furthermore, from the comparison of LoD2 and LoD3 results, we find that semantic information enhances the geometric accuracy of reconstructed models, largely thanks to our proposed adaptive reconstruction strategy. Regarding reconstruction efficiency, we have calculated the total time (Time-P.) spent (based on parallel processing on our quad-core CPU) and the cumulative time (Time-A.) for individual object reconstruction. It is apparent that although buildings are the least numerous, their complex geometric structures result in longer processing times compared to other categories.

For the surface reconstruction detailed in the right column of Table 6.1, all categories except water and terrain utilize texture-level semantic labels. Therefore, we use terrain meshes and their corresponding texture images to extract roads, low vegetation, and impervious surfaces. Our algorithm significantly reduces the face numbers by approximately 80% to 95%. In terms of geometric accuracy, except for water, these surfaces display notably high average Hausdorff and Chamfer distances, particularly roads. This primarily results from our approach of using a uniform plane projection to maintain surface coherence and extract polygons. Moreover, surface simplification is notably more time-efficient compared to object reconstruction.

6.6.4. Comparisions

To conduct a more detailed evaluation of our proposed semantic-based 3D reconstruction method, we compared it against other methods in terms of geometric accuracy and semantic accuracy.

Geometry-based comparisons. In terms of geometric accuracy, we compared three lightweight model reconstruction methods2.5D Contour [Q.-Y. Zhou and U. Neumann, 2010], PolyFit (with Gurobi 11.0 solver) [Nan and Wonka, 2017], and KSR [Bauchet and Lafarge, 2020]. Additionally, we evaluated three mesh simplification methods: Spred [Salinas *et al.*, 2015], MeshPoly [Bouzas *et al.*, 2020], and LowPoly [X. Gao *et al.*, 2022]. Since the competing methods do not process semantic information, we ensured a fair comparison by utilizing only the spatial subdivision aspect of our reconstruction pipeline. We selected in total 12 test samples from the SUM dataset across four semantic categories: buildings, trees, cars, and ships. To ensure the integrity of the reconstruction results, we completed all missing parts in the test data, including point cloud and mesh completion. We optimized the parameters of each method to ensure the best performance. We set a processing time limit of three hours; if no results are produced within this period, we categorize it as a failure to produce results. Notably, we also attempted to compare our approach with City3D [J. Huang *et al.*, 2022] using extracted roof surfaces. However, the complexity of our building components and the extensive number of vertical planes introduced by City3D prevented the program from delivering results within a reasonable timeframe.

We present the quantitative and qualitative analysis of these methods in Table 6.2

	Mod-N.	Fdo-N.	$\overline{C-dis.}$	$\overline{H-dis.}$	$\overline{RMS.}$(%)	2-mani.	Wat.	Sem.	$\overline{Tim.}$(s)
PolyFit	3	23	0.88	1.27	4.51	3	3	No	4.1
MeshPoly	10	232	2.44	3.61	1.95	1	3	No	1810.0
LowPoly	11	248	1.20	1.38	2.56	10	1	No	705.5
2.5D Con.	12	1912	0.99	1.22	3.50	6	0	No	**2.4**
KSR	12	**488**	0.90	1.23	2.22	1	**12**	No	1170.2
Spred	12	1749	**0.31**	**0.34**	**0.69**	2	0	No	150.1
Ours	12	497	1.01	1.38	2.34	**12**	**12**	**Yes**	316.2

Table 6.2: Quantitative comparison of lightweight city modeling methods: this includes PolyFit [Nan and Wonka, 2017], MeshPoly [Bouzas *et al.*, 2020], LowPoly [X. Gao *et al.*, 2022], 2.5D Contour [Q.-Y. Zhou and U. Neumann, 2010], KSR [Bauchet and Lafarge, 2020], and Spred [Salinas *et al.*, 2015]. Mod-N. indicates the number of models successfully reconstructed; Fdo-N. is the average number of output polygonal faces across all samples; $\overline{C-dis.}$ and $\overline{H-dis.}$ represent the average Chamfer and Hausdorff distances, respectively; $\overline{RMS.}$ is the average RMSE, calculated as a percentage of the optimal bounding box length per object or surface; 2-Mani shows the number of models meeting the 2-manifolds condition; Wat. indicates watertight models; Sem. notes the inclusion of semantic information; $\overline{Tim.}$ lists the average reconstruction time per object in seconds. The average input model has approximately 33,552 faces and 140 planes.

and Figure 6.13 to Figure 6.15, respectively. For qualitative analysis, we classified the results into three categories based on complexity: simple (less than 50 detected planes), moderate (50 to 150 detected planes), and complex (more than 150 detected planes), with RMSE represented by color scales. Table Table 6.2 shows that PolyFit [Nan and Wonka, 2017], MeshPoly [Bouzas *et al.*, 2020], and LowPoly [X. Gao *et al.*, 2022] were unable to reconstruct all results within the time limit due to the large number of input faces. Figure Figure 6.13 indicates that PolyFit [Nan and Wonka, 2017] and MeshPoly [Bouzas *et al.*, 2020] failed to produce correct results when the number of extracted planes was insufficient. Additionally, despite multiple parameter adjustments, MeshPoly [Bouzas *et al.*, 2020] consistently failed to produce complete results for our test data, likely due to mesh tile discontinuities. Among all methods, Spred [Salinas *et al.*, 2015] achieved the highest geometric accuracy for simplified models but also resulted in high output face numbers. The 2.5D Contour [Q.-Y. Zhou and U. Neumann, 2010] method is the fastest but retained the most faces and could not ensure model watertightness. KSR [Bauchet and Lafarge, 2020] maintained high geometric accuracy with the fewest faces, but most models did not satisfy the 2-manifolds condition. In contrast, our method ensured manifold and watertight models, preserved semantic information, and achieved model simplification and geometric accuracy comparable to KSR [Bauchet and Lafarge, 2020], while being far more time-efficient. Figures Figure 6.13 to Figure 6.15 show that both our method and KSR [Bauchet and Lafarge, 2020] produced similar results, capable of reconstructing relatively complete and regularized models.

We further explored the reconstruction performance for buildings in Table 6.3 and Figure 6.16. Table 6.3 shows that our LoD2 method achieves similar output plane

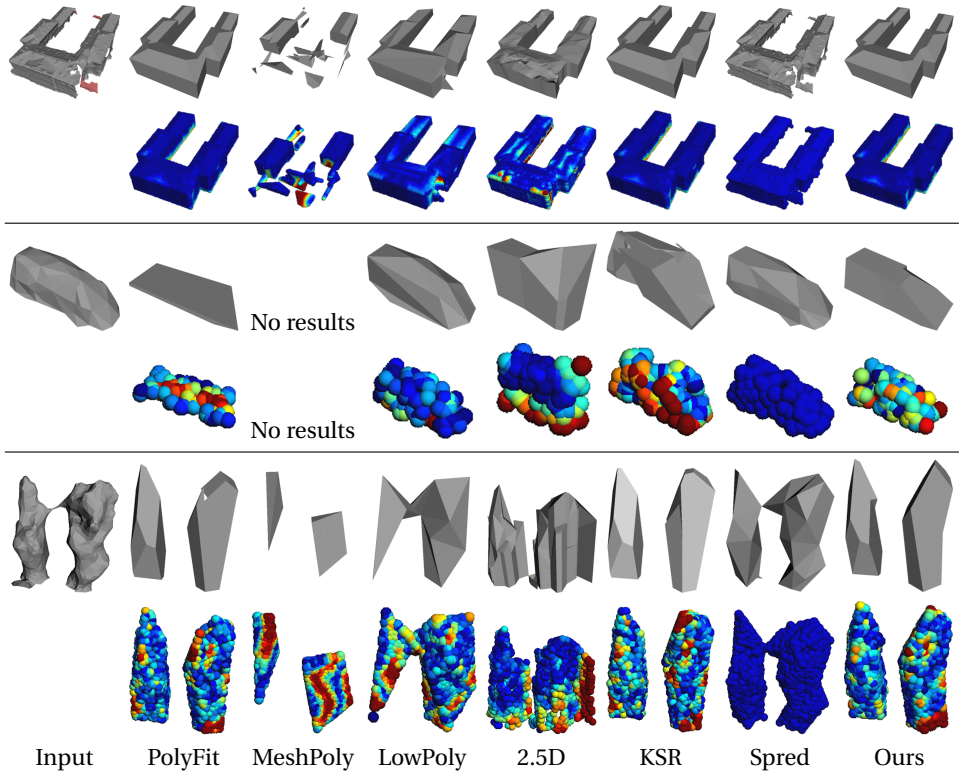


Figure 6.13: Comparative qualitative analysis of different model reconstruction methods for simple cases (less than 50 detected planes), including PolyFit [Nan and Wonka, 2017], MeshPoly [Bouzas *et al.*, 2020], LowPoly [X. Gao *et al.*, 2022], 2.5D Contour [Q.-Y. Zhou and U. Neumann, 2010], KSR [Bauchet and Lafarge, 2020], Spred [Salinas *et al.*, 2015], and ours. We performed point cloud sampling on the models reconstructed by each method and calculated their RMSE (%) using the length of the optimal bounding box as a baseline. RMSE values are visually represented by a gradient from blue to red, indicating a range from 0% to 10%. From top to bottom, the image shows a building, a car, and trees.

counts and geometric accuracy as KSR [Bauchet and Lafarge, 2020], but with greater time efficiency. Introducing semantic information has significantly improved geometric accuracy while also increasing both the output face count and processing time. Figure 6.16 demonstrates that our approach effectively reconstructs both the geometric structure and semantic information of models.

Semantic-based comparisons. To evaluate the accuracy of automatic semantic labeling, we compared our method with traditional K-Nearest Neighbors (KNN)-based semantic voting techniques. We manually annotated three building models

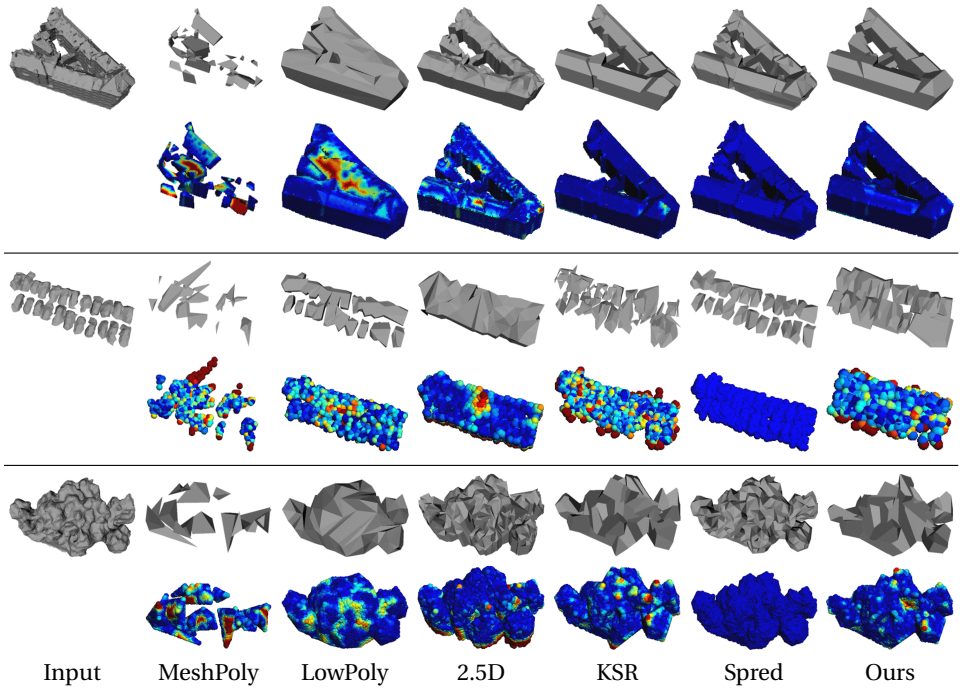


Figure 6.14: Comparative qualitative analysis of different model reconstruction methods for moderate cases (50 to 150 detected planes), including MeshPoly [Bouzas *et al.*, 2020], LowPoly [X. Gao *et al.*, 2022], 2.5D Contour [Q.-Y. Zhou and U. Neumann, 2010], KSR [Bauchet and Lafarge, 2020], Spred [Salinas *et al.*, 2015] and ours. We performed point cloud sampling on the models reconstructed by each method and calculated their RMSE (%) using the length of the optimal bounding box as a baseline. RMSE values are visually represented by a gradient from blue to red, indicating a range from 0% to 10%. From top to bottom, the image shows a building block, cars, and a tree.

reconstructed from LoD2 to LoD3 using our methods (see Figure 6.16) and used these as ground truth. As indicated in Table 6.4, our method greatly outperformed traditional KNN approaches in preserving the semantic accuracy of objects and their components. Consequently, users spend less time correcting semantic errors in our reconstructed models for further applications than with KNN methods.

6.6.5. Limitations

Similar to vectorization methods used in 2D mapping, our semantic-based 3D reconstruction and surface simplification methods also rely on accurate ground truth semantic information for fully automatic lightweight reconstruction. When using semantic labels predicted by learning-based methods, uncorrected errors in these

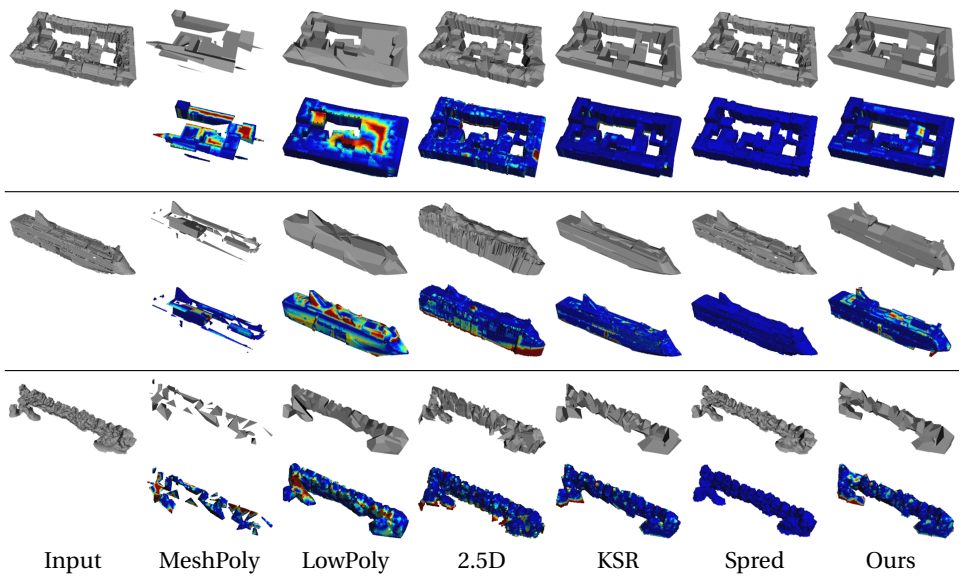


Figure 6.15: Comparative qualitative analysis of different model reconstruction methods for complex cases (more than 150 detected planes), including MeshPoly [Bouzas *et al.*, 2020], LowPoly [X. Gao *et al.*, 2022], 2.5D Contour [Q.-Y. Zhou and U. Neumann, 2010], KSR [Bauchet and Lafarge, 2020], Spred [Salinas *et al.*, 2015] and ours. We performed point cloud sampling on the models reconstructed by each method and calculated their RMSE (%) using the length of the optimal bounding box as a baseline. RMSE values are visually represented by a gradient from blue to red, indicating a range from 0% to 10%. From top to bottom, the image shows a building block, a cruise ship, and trees.

	Build-1. (#F: 11946, #P: 94)			Build-2. (#F: 33483, #P: 110)			Build-3. (#F: 71550, #P: 226)		
	Fdo-N.	H-dis.	Time (s)	Fdo-N.	H-dis.	Time (s)	Fdo-N.	H-dis.	Time (s)
KSR	55	2.22	5.0	201	1.40	136.0	447	1.70	4205.5
Spred	3364	0.29	5.2	514	0.41	10.2	2516	0.35	39.5
LoD2	62	2.23	17.4	222	1.60	99.5	659	1.85	282.8
LoD3	3356	1.58	57.4	9273	0.95	253.6	22808	1.02	4258.7

Table 6.3: Comparison of building model reconstruction results, including KSR [Bauchet and Lafarge, 2020] with better overall performance, Spred [Salinas *et al.*, 2015] with the highest geometric accuracy, and our LoD2 and semantic-based LoD3 reconstruction methods. #F indicates the number of input faces, #P denotes the number of input planes, Fdo-N. reflects the total number of output polygon faces, H-dis measures the Hausdorff distance, and Time quantifies the reconstruction time.

labels often fail to meet our input requirements. Additionally, our 3D reconstruction involves surface extraction based on planar segments generated through over-segmentation. Poor quality over-segmentation directly reduces the accuracy and

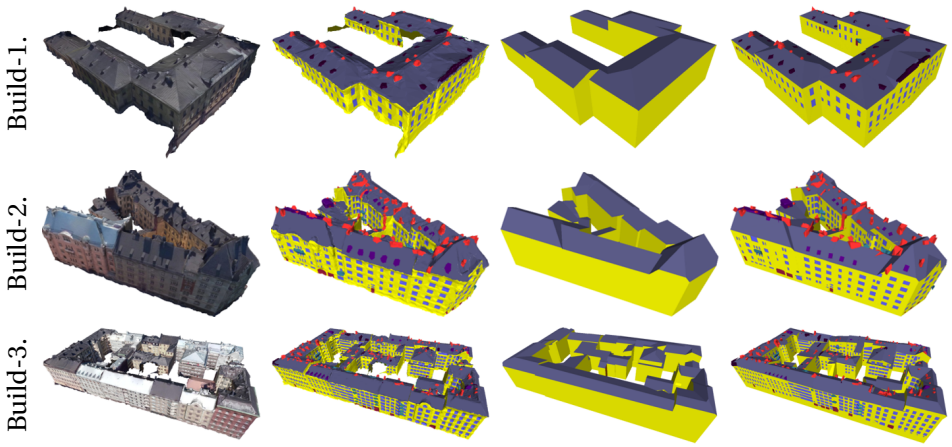


Figure 6.16: Building model reconstruction results of our methods, showing from left to right the input texture mesh, the manually labeled semantic mesh, the LoD2 reconstruction results, and the LoD3 reconstruction results. The semantic classes shown in the figure include roof surface ■, facade surface ■, chimney ■, roof installation ■, dormer ■, balcony ■, window ■, and door ■.

	Build-1.		Build-2.		Build-3.		Average	
	mIoU (%)	mAcc (%)	mIoU (%)	mAcc (%)	mIoU (%)	mAcc (%)	mIoU (%)	mAcc (%)
KNN LoD2	52.64	96.14	51.59	90.01	49.71	90.04	51.31	92.06
Ours LoD2	**100.00**	**100.00**	**94.10**	**98.90**	**93.51**	**99.21**	**95.87**	**99.37**
KNN LoD3	54.29	79.88	47.70	68.20	52.12	73.66	51.37	73.91
Ours LoD3	**99.95**	**99.96**	**99.01**	**99.52**	**89.61**	**99.52**	**96.19**	**99.66**

Table 6.4: Quantitative analysis of automatic semantic labeling results for reconstructed building models, ranging from LoD2 to LoD3

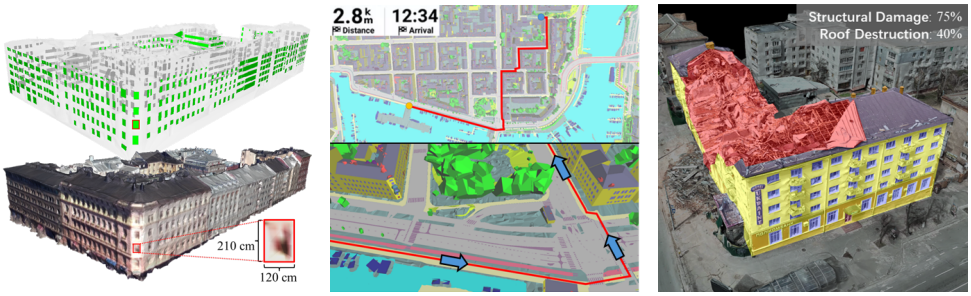
quality of the reconstruction, affecting the final integration of geometric models and semantic integration.

In semantic-based surface simplification, objects such as roads, sidewalks, and cycle lanes are projected onto the same horizontal plane for simplification. This implies that our methods work well in relatively flat areas but struggle in hilly regions. Currently, we cannot individually reconstruct them based on local planes and establish their topological relationships. Furthermore, our current method treats road markings as a floating layer instead of embedding them geometrically into the roads, potentially impacting subsequent applications.

6.7. Potential applications

We aim to explore how semantic information at different levels of detail can be applied to scenarios involving 3D city models. We believe that although textured meshes with semantic information can support a range of applications, such as displaying urban object attributes or performing semantic-based automatic measurements like areas of roofs and windows, and widths and lengths of roads, this approach has some limitations. Since semantic information in triangular meshes and texture images attaches to the model surface in different formstriangular facets and texture masksthis structural complexity increases the burden of reading, writing, and computing for various applications. In contrast, lightweight semantic city models show more advantages in data reading, writing, storage, and computational processing. To better meet the storage and format requirements of different applications, we recommend first converting the reconstructed lightweight models into CityGML [Gröger and Plümer, 2012; OGC, 2012] or CityJSON [Ledoux *et al.*, 2019] formats, which not only enhances processing efficiency but also improves model usability and interoperability.

We believe that lightweight 3D city models enhanced with semantic data have wide-ranging applications, as illustrated in Figure 6.17. We categorize applications of semantic-based city models into four types: automatic geometric measurements, interactive spatial computations, spatial analysis based on external data, and environment simulation based on physical engines.



(a) (b) (c)

Figure 6.17: Examples of applications for semantic-based 3D city models: (a) shows automated measurements and statistics of building windows using semantic information; (b) depicts high-precision pedestrian navigation using a 3D semantic map, including detailed path planning via recognized sidewalks and crosswalks; (c) illustrates damage assessment of buildings using 3D semantic information.

Automatic geometric measurements. These applications automatically measure geometric parameters of city model elements, like building height (number of stories), area, volume, and the areas and volumes of features such as doors, windows, roads, and vegetation. With semantic information, geometric calculations can shift from manual

to fully automated processes, thus enhancing efficiency and accuracy. The level of detail in the semantic information directly influences automation, with precise information crucial for measuring dimensions such as facade area and window area [Catita *et al.*, 2014; Panagiotidou *et al.*, 2021].

Interactive spatial computations. These are spatial computations within 3D city models based on user interactions. Examples include calculating the shortest paths for pedestrians [Sun *et al.*, 2019], vehicles [S. H. Park *et al.*, 2019], robots [Pütz *et al.*, 2021b], or drones [Y. Liu *et al.*, 2022b] from user-specified starting points, displaying urban landscapes from specific building windows [Kent and Schiavon, 2020], and interactive editing and design using selected semantic objects [Arikan *et al.*, 2013; . Demir *et al.*, 2016]. Semantic information outlines the 3D spaces or surfaces used for these computations, with higher levels of detail enabling a more precise partition of the space.

Spatial analysis based on external data. 3D semantic city models serve as base maps, merged with external data for spatial analysis and property display. For instance, combining sunlight data to estimate potential electricity from rooftop solar panels requires knowing the rooftop area and distribution of roof superstructures [Sánchez-Aparicio *et al.*, 2021]; using wind monitoring data to estimate the range of air pollution generated by buildings necessitates locating the positions and sizes of chimneys [Cichowicz and Dobrzaski, 2021]; analyzing the noise pollution range of highways needs information on road materials and traffic volume [Puyana-Romero *et al.*, 2020]. Semantic information provides the necessary geometric properties and spatial positioning for these external data.

Environment simulation based on physical engines. Utilizes the geometric structure and semantic information of 3D city models to simulate physical state changes in the environment. With precise semantic information, physical engines can realistically simulate the responses of buildings to forces, such as impacts, destruction, or collapse [Ferworn *et al.*, 2013]. In disaster simulations, semantic information aids in more accurately simulating the physical effects of earthquakes [R. Sinha *et al.*, 2012] or floods [Rong *et al.*, 2020; Schröter *et al.*, 2018] on different types of ground surfaces. In autonomous driving simulations, semantic information helps vehicles understand and predict environmental elements such as traffic signs and road markings to make appropriate driving decisions [Schwab *et al.*, 2020]. In CFD wind simulations, the detailed modeling of building facades and openings allows for a nuanced analysis of wind flow patterns around and through structures, which is crucial for assessing wind loads and ventilation effectiveness [García-Sánchez *et al.*, 2021; Paen *et al.*, 2022]. Finer semantic details significantly enhance the realism and accuracy of physical simulations.

These applications highlight the extensive potential of semantic-based 3D city models to address real-world challenges, underscoring the significance of detailed semantic levels

in boosting the efficiency and precision of these applications.

6.8. Conclusions

This chapter addresses critical challenges in reconstructing lightweight 3D city models from large-scale urban textured meshes, focusing on semantic-based approaches. Our framework efficiently creates lightweight, geometrically precise, and semantically enriched city models. By leveraging semantic information, it significantly enhances the efficiency and accuracy of 3D model reconstruction for objects and parts, while ensuring precise semantic labels for both solid models and simplified surfaces. Experimental evaluations on the SUM and SUM-Parts datasets demonstrate significant enhancements in data compression, processing speed, and accuracy over competing methods. The resulting large-scale city models can be applied to various practical applications, ranging from automatic geometric measurements to environmental simulations. Future research will refine these methods and further integrate a broader range of semantic categories into urban analysis tools and simulations, thereby enhancing the real-world applicability of semantic 3D city models.

7

Conclusions and future work

This chapter outlines the contributions and key findings of the thesis on the semantic understanding of urban scenes from textured meshes. It delves into the creation of a large-scale semantic urban mesh dataset, the development of an interactive 3D annotation tool, and the design of a deep learning framework for semantic segmentation tailored to urban meshes. This research not only addresses the challenges in efficient semantic annotation and segmentation of urban textured meshes but also integrates these advancements into practical applications like lightweight city modeling. Furthermore, this work pioneers the application of semantic information in urban mesh data, setting a foundation for future research to expand on semantic segmentation methods and their application in real-world city modeling scenarios.

7.1. Contributions and key findings

Large-scale urban textured mesh datasets are rich in semantic information. However, research on efficiently extracting and applying this information is still scarce. To address this research gap, this thesis comprehensively has explored the semantic segmentation of urban textured meshes, including creating benchmark data, semantic annotation, semantic segmentation, and the practical application of semantic information. This thesis has detailed the strategies, technical methods, and practical applications used. Specifically, this study has reconstructed a large-scale urban textured mesh dataset at the object and part levels for the first time to meet the demand for extensive training data required by learning-based 3D semantic segmentation methods, significantly enhancing the usability of urban mesh data.

Additionally, the interactive 3D annotation tool we have developed addresses the deficiencies in efficient semantic annotation methods for urban textured meshes, achieving efficient and accurate semantic annotation of mesh faces and textures. This thesis not only proposes the first deep learning-based semantic segmentation framework specifically designed for urban meshes but also integrates semantic information into practical applications such as lightweight city modeling, providing innovative solutions for understanding and modeling 3D urban scenes. In subsequent sections, I will elaborate in detail on the core contributions to the research objectives and the key findings regarding research questions based on the relevant literature and the content of each chapter of this thesis.

I. Semantic urban mesh benchmark dataset

- **Contributions:** Developed a comprehensive benchmark dataset of large-scale semantic urban meshes featuring both object-level and part-level semantics. This dataset is critical for evaluating and advancing state-of-the-art 3D semantic segmentation methods.

- **Key findings:**

    1. *Which semantic objects and parts can be precisely identified by humans and computers from the triangular facets and texture images in urban textured meshes?*

       We believe that the semantic objects or parts that can be recognized by humans or machines in an urban textured mesh largely depend on its quality. Specifically, the higher the quality of the mesh, the more recognizable the semantic classes. The quality of the mesh mainly involves two aspects: geometry and color. The geometric quality is primarily determined by the quality of the multi-view images (including color, resolution, coverage, and viewing angles) and the robustness of the reconstruction algorithm; the color quality depends not only on the quality of the multi-view images but also on the effectiveness of the texture generation algorithm.

The Helsinki mesh [Helsinki, 2019], highlighted in this thesis, employs the UltraCam Osprey Prime II five-lens imaging system [VexcelImaging, 2016]. It features a ground sampling distance of 7.5 cm, with 60% side coverage and 80% length coverage [KIGA-digi, 2019]. At the mesh face level, identifiable semantic objects encompass terrain, high vegetation, buildings, cars, boats, and water, with parts such as roof surfaces, facade surfaces, chimneys, dormers, balconies, roof installations, and walls. At the texture pixel level, recognizable objects include low vegetation, roads, cycle lanes, and sidewalks alongside parts like windows, doors, and road markings. Unidentifiable objects at the mesh face level, such as utility poles, flagpoles, awnings, streetlights, trash bins, and fences, are mainly due to insufficient resolution and algorithmic constraints that fail to reconstruct their geometric structures. Semantic classes that remain unidentifiable at the texture pixel level (such as window frames, door handles, small billboards, manhole covers, and building materials.) are primarily limited by image resolution.

2. *What 3D semantic segmentation methods are suitable for performance evaluation in urban semantic meshes?*

   We find that both mesh-based and point cloud-based 3D semantic segmentation methods can be used to evaluate the semantic segmentation performance of urban meshes. This is mainly attributed to the highly consistent 3D spatial characteristics of large-scale urban meshes and point clouds. Specifically, whether using manually extracted features or deep learning methods, it is necessary to integrate contextual information from their respective neighborhoods. On mesh surfaces, calculating geodesic distances instead of Euclidean distances to gather contextual information has a significant impact on the accuracy required for small-scale indoor scenes [Z. Hu *et al.*, 2021; Schult *et al.*, 2020]; however, this difference is not significant in large-scale urban meshes. This is because urban meshes often suffer from oversmoothing and low precision [W. Gao *et al.*, 2021], and when objects are close in Euclidean distance, their geodesic distances are also often close, such as trees near buildings whose surfaces are often connected.

   Additionally, some mesh segmentation methods suitable for indoor or small object meshes are not applicable to large-scale urban meshes. These methods involve convolutions and pooling on the faces [S.-M. Hu *et al.*, 2022], edges [Hanocka *et al.*, 2019], or vertices [Schult *et al.*, 2020] of the mesh, which consume substantial memory and require the topology of the input mesh to be flawless. To address this challenge, this thesis proposes over-segmenting the mesh, using planar and non-planar segments to replace large numbers of triangular faces, thereby significantly reducing the computational load during the feature extraction phase.

   Point cloud-based 3D semantic segmentation methods involve sampling

from mesh surfaces as inputs [W. Gao *et al.*, 2021; W. Gao *et al.*, 2023]. Point cloud processing is flexible [Charles *et al.*, 2017; Deng *et al.*, 2023; Q. Hu *et al.*, 2020; Qian *et al.*, 2022; Thomas *et al.*, 2019], allowing arbitrary splitting, downsampling and multi-scale neighborhood searches, effectively avoiding issues caused by mesh topology errors. However, these methods are less effective at recognizing small-scale object parts (such as windows and road markings), and overly small sub-point clouds (from splitting) may result in the loss of critical contextual information.

Overall, although these methods are applicable to the semantic segmentation tasks of urban meshes, their performance still needs further improvement.

II. Urban textured mesh annotation framework

- **Contributions:** Designed and developed an interactive 3D annotation tool that facilitates semi-automatic semantic labeling of mesh facets and textures. This tool enables the creation of extensively labeled semantic urban meshes and assists in correcting prediction errors from learning-based methods.

- **Key findings:**

  1. *How can an interactive annotation user interface be designed for urban textured meshes?*

     We believe that designing an interactive user interface for annotating urban textured meshes should adhere to three critical core principles: freedom in viewpoint selection, diversity in the selection of objects and methods, and varied rendering and display modes.

     Firstly, the functionality of freely choosing viewpoints is intended to allow users to view any part of the 3D model. This is achieved by enabling users to freely choose different viewpoints in a 3D scene, for which we have developed operations such as translating, zooming, rotating, and setting new observation pivots to enhance users' ability to explore space freely.

     Secondly, the functionality of selecting objects primarily serves the annotation task. We provide a variety of selectable elements and their combinations, including triangular faces, segments, texture pixels, and superpixels. To accommodate different selection needs, we have designed various 2D and 3D selection methods such as brush, lasso, stroke, basic shapes, polygons, expansion and reduce, select all and deselect, and semantically based selection, enabling users to adopt the most suitable selection strategy according to different scenarios.

     Finally, the design of various rendering and display modes aims to assist users in effectively viewing and understanding the information provided by the semantic textured mesh. This includes texture rendering (displaying the

true colors of objects), flat shading (showing the semantic colors of faces), texture mask rendering (displaying semantic colors based on textures), various translucent blending renderings (overlaying different types of semantic and color information), and Gouraud shading [Gouraud, 1971] of faces (assisting in the selection of texture segments); display modes cover showing triangle edges (selecting triangular faces), showing segment boundaries (selecting segments), and displaying texture segments (selecting texture pixels).

These core design principles establish an efficient, flexible, and user-friendly interactive environment that significantly improves the labeling efficiency and accuracy of urban textured meshes.

2. *How can we develop a semi-automatic annotation framework that leverages existing annotated data to predict and correct unannotated regions?*

   We have designed a semi-automatic urban textured mesh annotation framework, where the core mechanism involves using labeled data to predict unlabeled data. Errors in the predictions are then corrected using annotation tools, and the corrected data are fed back into the model as newly labeled data. This cycle continues until all data are annotated. This method offers the advantage of significantly reducing the time required to correct prediction errors compared to manually re-annotating the entire dataset. This approach places higher demands on semantic segmentation, requiring the training of high-accuracy models with only limited labeled data. Traditional deep learning methods often rely on large amounts of training data or on transfer learning, which is difficult to achieve in the initial stages of annotating urban textured meshes. Therefore, we employed a combination of handcrafted features based on planar segments and the random forest algorithm, achieving the goal of automatically classifying unlabeled mesh faces with minimal training data and obtaining high-accuracy prediction results. Additionally, we introduced a feature discrepancy analysis mechanism by selecting data rich in semantic information from the prediction results for label correction to further enhance the model's predictive accuracy. Experiments have shown that, compared to full manual annotation, our method significantly reduces annotation time.

3. *How can we achieve an efficient selection of 3D objects and parts in urban textured meshes with minimal user interaction?*

   We propose that objects with similar structures in urban meshes typically share the same semantic labels. Building on this assumption, we designed a structure-aware matching selection strategy. Moreover, we've incorporated interactive approaches for selecting planar segments and protrusions, facilitating fast and efficient selection of 3D objects and parts within the mesh. To ease user interaction, we initially apply plane-based

over-segmentation to the mesh, utilizing these planar segments for fast selection and structural matching. To ensure precise object boundary selection, we calculate local features from the selected area provided during a single user interaction, extracting protrusions based on triangular faces. These protrusions are then used as templates to match with other similar structures, enabling rapid structural selection with minimal user interactions. Experimental results indicate that this approach not only considerably reduces user interactions but also significantly decreases the time required for labeling, thus boosting overall productivity.

4. *How can regions of interest be efficiently selected in the mesh textures?*

   To achieve efficient annotation of mesh texture images, we have developed an interactive region selection strategy based on planar texture segments. Our strategy is based on a core assumption: regions in the texture image with similar colors usually have the same semantic labels. Based on this assumption, we designed a selection method that expands a local area via user clicks, allowing users to quickly and precisely define regions of interest. Additionally, we calculated the structural features of these areas, which are used to match and identify other areas in the image with similar structures. Experimental tests have shown that our method significantly outperforms traditional annotation methods in terms of efficiency and surpasses existing deep learning techniques in accuracy. This strategy not only enhances the speed of annotation but also improves the accuracy of the results, demonstrating its practical value in the field of 2D semantic annotation.

III. Semantic segmentation of large-scale urban textured meshes

- **Contributions:** Developed a fully automatic semantic mesh segmentation algorithm tailored for large-scale urban scenes. This algorithm demonstrates superior generalization capabilities compared to existing methods and is sufficient enough to be applied to other datasets, bridging domain gaps effectively.

- **Key findings:**

  1. *How can a deep learning framework be developed for semantic segmentation of large-scale urban meshes?*

     When applying deep learning to semantic segmentation of large-scale urban textured meshes, we encounter challenges like managing numerous irregular triangular faces and complex mesh structures. Traditional convolutional neural networks and multilayer perceptron-based networks often struggle with memory limitations and high data throughput when processing large-scale mesh datasets. Unlike point clouds, manipulating mesh data through downsampling, upsampling, or arbitrary splitting can affect its topological structure and texture mapping. Furthermore, potential topological defects

might hinder effective feature extraction.

To tackle these challenges, we developed a planarity-sensible graph neural network tailored for face-level semantic segmentation of meshes. This method significantly reduces computational demands by over-segmenting the mesh into planar and non-planar segments, effectively maintaining both local and global mesh features. Additionally, we innovatively established short-range and long-range segment connections that effectively substitute for the mesh's complex topological connections, enhancing the network's ability to capture contextual features.

Our method has been experimentally validated to effectively handle large-scale urban mesh data, greatly improving the efficiency and accuracy of semantic segmentation. This advancement not only optimizes the data processing workflow but also opens new avenues for deep learning applications in urban textured meshes.

2. *How can the generalizability of deep learning be enhanced for 3D semantic segmentation tasks?*

   Compared to traditional machine learning methods, deep learning often performs poorly in terms of generalization [W. Gao *et al.*, 2023; Q. Hu *et al.*, 2022]. Although adding a large variety of training data can enhance the generalization capability of deep learning models, this approach is not suitable for all scenarios, especially when training data is scarce. To address this issue, we adopted an innovative strategy: integrating robust generalizable handcrafted features with deep learning features to enhance the network's generalization ability. A series of experiments have confirmed that our method significantly outperforms traditional machine learning and purely deep learning-based methods in terms of generalization. This integration strategy not only improves the adaptability of the model but also provides an effective solution for complex application scenarios with insufficient data.

3. *How to obtain sufficient contextual information for semantic segmentation?*

   Contextual features are vital in semantic segmentation, greatly affecting its performance. Common methods for gathering contextual information include global pooling, multi-scale feature extraction, attention mechanisms, graph neural networks, and recurrent neural networks. Nonetheless, the application and study of these approaches in urban mesh semantic segmentation remain limited.

   To address this research gap, we have developed a novel approach by over-segmenting the textured mesh and using the segments to extract local contextual features. Furthermore, we have constructed a graph based on the geometric relationships between segments to capture global contextual

information. Graph neural networks [Y. Li *et al.*, 2017] are used to integrate these local and global contextual features, enhancing the model's capability to comprehensively understand and process complex urban mesh data.

Following extensive experimental testing, our method outperforms existing methods in object-level urban mesh semantic segmentation tasks. This success not only boosts the accuracy of semantic segmentation but also offers a more effective approach for analyzing urban meshes.

IV. Applications of the semantic urban meshes

- **Contributions:** Established a semantic-based framework for lightweight city model reconstruction from textured meshes and explored its integration into various practical applications. This framework enhances the utility of lightweight semantic city models across multiple real-world applications, demonstrating significant practical benefits.
- **Key findings:**

  1. *How can semantic information from textured meshes be utilized for lightweight reconstruction of 3D city models?*

     Existing methods for lightweight 3D city modeling often overlook the crucial role of semantic information in model reconstruction. We believe that incorporating semantic information not only enhances the efficiency of model reconstruction but also significantly improves the geometric precision and detail representation of the reconstructed results. Based on this concept, we propose a novel semantic-based 3D reconstruction and surface simplification framework based on semantic textured meshes.

     We start by separately reconstructing objects and their parts using semantic labels, then combine them into a sealed polygon model. In addition, we perform targeted simplifications on surfaces of different semantic types aimed at maintaining the geometric and semantic properties of the surface while reducing data volume. We have also developed an efficient semantic label transfer strategy that preserves rich semantic information throughout the lightweight construction process of the city model.

     After experimental validation, our method has proven effective in creating large-scale, semantically detailed 3D city models ranging from LoD2 to LoD3. This approach not only optimizes processing speed and storage requirements but also enhances the practicality and application value of the models.

  2. *What applications can be developed using urban meshes and lightweight models equipped with object and part-level semantics?*

     We observed that urban meshes carrying rich objects and part-level

semantics are relatively limited in practical applications. This is primarily due to the complex nature of semantic information in triangular meshes and texture images, which increases the difficulty of reading, writing, and computing in application development. In contrast, we believe that lightweight 3D semantic city models have broader application potential, particularly when converted to CityGML [Gröger and Plümer, 2012; OGC, 2012; OGC, 2021] or CityJSON [Ledoux *et al.*, 2019] formats. These models are especially useful in the following four areas: automatic geometric measurements, interactive spatial computations, spatial analysis based on external data, and environment simulation based on physical engines.

The effectiveness of these applications is significantly influenced by the level of semantic detail. Specifically, the richness of semantic information can enhance the automation level of applications, improve the precision of space partition, improve the accuracy of spatial positioning and geometric information, and increase the realism and precision of physical simulations. By integrating these detailed semantics, lightweight 3D city models not only simplify processing workflows but also provide more accurate and practical analysis tools, offering strong technical support for urban planning and management.

7.2. Reflections

This thesis primarily explores methods for acquiring semantic information from textured meshes and their applications. By thoroughly analyzing the content in Section 7.1, we have reached a key conclusion: using urban textured meshes as input, along with efficient feature extraction methods, can effectively achieve interactive semantic annotation and automatic semantic segmentation of these meshes. Extracting and applying semantic information provides strong support for large-scale 3D city modeling.

In addition to these results, this paper will also delve into and reflect on some key issues and challenges encountered in the research, which I outline as follows:

Semantic 3D urban annotation and segmentation. For 3D semantic segmentation of urban scenes, in addition to the textured meshes used in this research, LiDAR point clouds, MVS dense point clouds, and multi-view images can also be used as data sources.

LiDAR point clouds can be categorized based on the data collection method into Terrestrial Laser Scanning (TLS), Mobile Laser Scanning (MLS), and Aerial Laser Scanning (ALS) point clouds. Although point cloud data have advantages in precision, they face more challenges in annotation and semantic segmentation compared to textured meshes, including:

- Lack of color information: LiDAR point clouds typically lack color information, which is crucial for distinguishing certain semantic categories like billboards versus building surfaces or sidewalks versus parking lots. Although color can be added through image projection, this process increases the costs of data collection and processing.

- Resolution and data integrity issues: LiDAR point clouds have lower resolution, with a limited number of points per unit area, which may result in small-scale objects being represented by only a few points and, thus, difficult to accurately capture certain geometric structures and, boundaries of these objects. Additionally, issues like occlusion, signal absorption, and reflection may lead to large gaps [Fei *et al.*, 2022], such as missing building facades in ALS point clouds. While resolution and data completeness can be improved through point cloud completion networks, increasing scanning frequency, or integrating other point clouds, this increases computational demands.

- Large data volume: Compared to textured meshes, LiDAR point clouds contain significantly more data, but this does not necessarily mean they contain more information. The larger data volume implies longer times needed for reading, writing, computing, and rendering in annotation and semantic segmentation tasks, as well as more storage space being occupied.

MVS dense point clouds and point clouds sampled from mesh surfaces, while slightly less precise, contain color information and have higher resolution. However, their data volumes are similarly large, which increases the storage and computational burden during semantic segmentation and data annotation.

Multi-view images are the foundational input data for constructing textured meshes. By annotating these images and performing semantic segmentation, along with considering the camera's intrinsic and extrinsic parameters, semantic labels can be effectively projected onto the textured mesh, thus creating 3D models with semantic information [Blaha *et al.*, 2017; Riemenschneider *et al.*, 2014; Romanoni *et al.*, 2017; J. C. Valentin *et al.*, 2013; Y. Zhou *et al.*, 2018]. The main advantage is that annotating two-dimensional images is relatively simple, and existing image semantic segmentation techniques are very mature, effectively avoiding precision loss and error impacts caused by 3D reconstruction algorithms. However, this method still faces several challenges:

- Repetitiveness of image annotations: The same object may need to be repeatedly annotated in images from different perspectives, reducing overall annotation efficiency.

- Label consistency issues: Annotations of the same object in images from different perspectives may be inconsistent, especially around the object's edges, which can lead to significant errors in projected annotations.

- Resolution and coverage challenges: Aerial oblique photography, often used to build large-scale urban scene 3D models, offers high resolution and wide coverage

but complicates semantic segmentation and annotation of small-scale objects.

- Geometric distortion issues: Due to perspective, objects may appear geometrically distorted in images, like windows appearing as severely compressed parallelograms, increasing annotation difficulty and potentially reducing semantic segmentation accuracy.
- Color limitations: Variations in lighting conditions across different perspectives make distinguishing consistently colored objects in images more challenging.

Although these problems can be alleviated by designing new algorithms and improving data collection and processing methods, these solutions are often complex and costly. In contrast, direct annotation and semantic segmentation of textured meshes are not only more intuitive but also more convenient. Textured meshes show irreplaceable advantages in data storage, reading, writing, rendering, annotating, and semantic segmentation.

Taking the SUM dataset proposed in Chapter 3 of this thesis as an example, as of August 2024, it has been cited by 42 related studies. These references span several areas, including semantic or instance segmentation methods for urban meshes [J. Chen *et al.*, 2022b; X. Liu *et al.*, 2024; Yoo *et al.*, 2023], lightweight generation and reconstruction of 3D city models [W. Han *et al.*, 2024; Pang and Biljecki, 2022; S. Wang *et al.*, 2023], and the extension of semantic 3D urban datasets [Gabara and Sawicki, 2023; T. Jiang *et al.*, 2024; J. Luo *et al.*, 2022]. The growing body of research underscores the rising importance of textured meshes in understanding 3D urban environments.

In conclusion, although various types of data can be used for the 3D semantic segmentation of urban scenes, each has its unique advantages and challenges. A deep understanding and appropriate handling of these data sources are crucial for improving the accuracy and efficiency of semantic segmentation.

Large-scale lightweight 3D city modeling. We examine various strategies for creating large-scale lightweight 3D city models, analyzing their strengths and weaknesses from the perspectives of reconstruction methods and data sources.

From the perspective of reconstruction methods, this thesis primarily uses sampled point clouds from mesh surfaces as input, employing lightweight reconstruction to build solid models and simplify surfaces. We have found that in practical applications, texture meshes are not entirely suitable as direct input data for lightweight 3D city model reconstruction. This is mainly due to the potential adverse effects of mesh tiling and topological errors. In the post-process of constructing large-scale urban texture meshes, it is common to split the mesh into tiles for easier storage, reading, writing, and rendering. This approach may result in the same building being split into different tiles, thereby losing topological connections. Reassembling these tiles while preserving texture information and recovering topological connections poses a significant challenge. Additionally, such meshes may contain topological errors [Attene

et al., 2013] like non-manifold vertices or edges, duplicate points, degenerate faces, self-intersecting faces, and holes, which directly impact the ability to accurately obtain topological information, thereby hindering the correct reconstruction of solid models and effective surface simplification. In contrast, lightweight reconstruction methods based on point clouds can entirely avoid the problems caused by mesh topological errors. Our approach starts with point cloud data sampled from the mesh, which is more flexible and robust, making it particularly suitable for handling large-scale urban scene data and ensuring that the final 3D models are both lightweight and accurate. In addition, mesh textures are utilized to identify and reconstruct semantic parts, such as windows and doors, which lack distinct geometric features in low-resolution meshes.

In the context of data sources for 3D city modeling, apart from the semantic texture meshes used in this thesis, LiDAR point clouds, SfM (Structure from Motion) sparse point clouds, MVS (Multi-View Stereo) dense point clouds, remote sensing satellite images, aerial images, and street view images can also serve as input data. LiDAR point clouds are favored for their high precision and large-area coverage of urban scenes, but they lack color information and suffer from data gaps. Typically, aerial laser scanning (ALS) can only reconstruct 2.5D LoD2 building models [Bauchet and Lafarge, 2020; J. Huang *et al.*, 2022]. MLS and TLS point clouds are more suited to reconstructing building facades, vegetation, and roads [Du *et al.*, 2019; Z. Li *et al.*, 2017; W. Liu *et al.*, 2023; Wysocki *et al.*, 2022]. A more detailed model reconstruction usually requires integrating additional image data or combining multiple sources of point clouds. SfM sparse point clouds and MVS dense point clouds both are reconstructed from multi-view images. Their advantage is that they contain color information; however, SfM point clouds are too sparse for high-precision and detailed model reconstruction [Pantoja-Rosero *et al.*, 2022]. MVS dense point clouds, although detailed, also suffer from data gaps, such as in building facades, and require substantial computational resources [M. Li *et al.*, 2016b].

Remote sensing satellite images excel in wide coverage and high temporal resolution but are generally only suitable for reconstructing city models with lower precision and detail, such as LoD1 or LoD2 building models [Gui *et al.*, 2022; Lussange *et al.*, 2023]. Aerial images (including UAV images) offer higher resolution than satellite images and are commonly used for constructing large-scale urban texture mesh models. When building lightweight city models, it is often necessary to create MVS point clouds to extract 3D features [M. Li *et al.*, 2016a] or use true ortho-photos to extract planar structures (like for roofs) [Schuegraf *et al.*, 2023] and use elevation data to generate 2.5D models. Directly extracting and recovering key 3D features (like 3D vertices and edges or topological relationships) from 2D images is relatively difficult, and research in this area is still limited. Street view images are typically employed to individually reconstruct building facades or supplement data for evolving LoD2 into more detailed LoD3 models [Nan *et al.*, 2015; Ogawa *et al.*, 2024; C. Zhang *et al.*, 2021]. Their high resolution is ideal for creating highly detailed models. However, they fall short of capturing complete urban data, such as rooftops and interiors of courtyard-style buildings.

Compared to other input data for 3D city modeling, semantic textured meshes offer the greatest amount of information with the least data volume and can be flexibly converted

into point cloud or image data, making them highly suitable for constructing detailed, large-scale, lightweight semantic 3D city models.

7.3. Future prospects

This section examines the current state and future directions for the semantic understanding of urban texture meshes, focusing on segmentation and interactive annotation, as well as dataset extension, given the research limitations identified in this thesis.

Urban textured mesh segmentation. Urban textured mesh segmentation primarily includes object-level and part-level semantic segmentation [Rouhani *et al.*, 2017; Verdie *et al.*, 2015], as well as instance segmentation [J. Chen *et al.*, 2022a; M. Dong *et al.*, 2024]. This thesis has focused on semantic segmentation, leaving instance segmentation unexplored. For semantic segmentation, our proposed deep learning framework for urban meshes is primarily limited to handling object and part-level semantic labels of mesh faces and does not include semantic segmentation of the mesh textures. In practice, we evaluate 3D semantic segmentation methods based on point clouds sampled from meshes, which generally perform poorly. For instance, the most effective method [Deng *et al.*, 2023] reached only a 47.9% mean IoU across SUM-Parts' 19 classes. These methods particularly fail with imbalanced classes and regular objects like building doors, windows, and road markings, often missing precise boundaries due to the inherent information loss in point cloud sampling. Directly utilizing texture meshes could circumvent these limitations.

Currently, to the best of our knowledge, there is no deep learning method suitable for large-scale urban scenes that applies to both meshes and textures, marking a potential future research direction. Moreover, instance segmentation is equally important in interpreting textured meshes, but current research is limited [J. Chen *et al.*, 2022a; M. Dong *et al.*, 2024], mainly focusing on buildings and rarely covering other urban features like trees, vehicles, and building part-level instances.

In summary, future research needs to develop higher precision, end-to-end 3D semantic and instance segmentation methods that meet industrial production requirements for automatic segmentation and interpretation of urban textured meshes.

Interactive urban mesh annotation and extended datasets. This thesis primarily explores interactive semantic annotation methods for mesh faces and textures but does not cover instance annotation methods. Compared to semantic annotation, instance annotation of texture meshes poses greater challenges primarily because the mesh represents a continuous surface where instance boundaries are often unclear. Therefore, developing efficient interactive instance segmentation methods for texture meshes

and constructing corresponding benchmark datasets are important future research directions.

For 2D texture image annotation, the predominant method currently involves combining models trained on large-scale data sets, such as SAM [Kirillov *et al.*, 2023], with user interaction inputs to achieve precise segmentation of areas of interest. Although these methods are fast and accurate [X. Chen *et al.*, 2022; Kirillov *et al.*, 2023; Q. Liu *et al.*, 2023], they often perform poorly in unseen scenarios and at delineating regular object boundaries, and precise boundaries are difficult to achieve through repeated interactions. Future research could explore incorporating prior information or regularization to improve the accuracy of urban texture image annotation.

For 3D interactive annotation, the methods proposed in this thesis are primarily unsupervised and based on local features, not yet fully utilizing the characteristics of labeled data. As the volume of semantic annotation data for urban meshes continues to grow, future research should more fully leverage these data to aid interactive 3D annotation of unknown data.

Additionally, apart from the object and part-level semantic labels proposed in this thesis, semantic annotation of urban textured meshes could be extended to include labels for man-made object materials and detailed categories of vegetation. Labels for materials of man-made objects, such as building materials and road materials, are crucial for applications such as physical environment simulations and energy estimations. Detailed labels for specific types of vegetation, like distinguishing poplars from willows or large-leaved lindens, are important for applications in simulating ecological environments and urban greening projects. Additionally, I believe that enhancing the reconstruction quality of textured meshes is essential for better semantic segmentation in urban scenes. This can be achieved by using drones closer to the ground, designing flight paths with higher photo overlap rates, using higher-resolution cameras, integrating multispectral or LiDAR sensors, and improving the precision and robustness of texture mesh reconstruction algorithms.

Bibliography

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P. and Süsstrunk, S. (2012). 'SLIC Superpixels Compared to State-of-the-Art Superpixel Methods'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11, pp. 2274–2282. DOI: 10.1109/TPAMI.2012.120.

Acuna, D., Ling, H., Kar, A. and Fidler, S. (2018). 'Efficient Interactive Annotation of Segmentation Datasets with Polygon-RNN++'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 859–868. DOI: 10.1109/CVPR.2018.00096.

Agugiaro, G., Benner, J., Cipriano, P. and Nouvel, R. (2018). 'The Energy Application Domain Extension for CityGML: enhancing interoperability for urban energy simulations'. In: *Open Geospatial Data, Software and Standards* 3.1, p. 2. ISSN: 2363-7501. DOI: 10.1186/s40965-018-0042-y.

AHN (2019). *Actueel Hoogtebestand Nederland (AHN)*. https://www.ahn.nl/. Accessed: 2021-04-16.

Alam, N., Coors, V. and Zlatanova, S. (2013). 'Detecting shadow for direct radiation using CityGML models for photovoltaic potentiality analysis'. In: *Urban and Regional Data Management*. Taylor & Francis, pp. 191–210. ISBN: 9780429228001. DOI: 10.1201/b14914-23.

Amirebrahimi, S., Rajabifard, A., Mendis, P. and Ngo, T. (2016). 'A framework for a microscale flood damage assessment and visualization for a building using BIM–GIS integration'. In: *International Journal of Digital Earth* 9.4, pp. 363–386. DOI: 10.1080/17538947.2015.1034201.

Apollonio, F. I., Gaiani, M. and Benedetti, B. (2012). '3D reality-based artefact models for the management of archaeological sites using 3D Gis: a framework starting from the case study of the Pompeii Archaeological area'. In: *Journal of Archaeological Science* 39.5, pp. 1271–1287. ISSN: 0305-4403. DOI: 10.1016/j.jas.2011.12.034.

Arikan, M., Schwärzler, M., Flöry, S., Wimmer, M. and Maierhofer, S. (2013). 'O-snap: Optimization-based snapping for modeling architecture'. In: *ACM Trans. Graph.* 32.1. ISSN: 0730-0301. DOI: 10.1145/2421636.2421642.

Armeni, I., Sax, S., Zamir, A. R. and Savarese, S. (2017). *Joint 2D-3D-Semantic Data for Indoor Scene Understanding*. URL: https://arxiv.org/abs/1702.01105.

Aschwanden, G., Haegler, S., Halatsch, J., Jeker, R., Schmitt, G. and Van Gool, L. (2009). 'Evaluation of 3D city models using automatic placed urban agents'. In: *9th International Conference on Construction Applications of Virtual Reality*, pp. 165–176. URL: https://lirias.kuleuven.be/1669545&lang=en.

Attene, M., Campen, M. and Kobbelt, L. (2013). 'Polygon mesh repairing: An application perspective'. In: *ACM Comput. Surv.* 45.2. ISSN: 0360-0300. DOI: 10.1145/2431211.2431214.

Authority, S. L. (2021). *Virtual Singapore*. https://www.nrf.gov.sg/programmes/virtual-singapore. Accessed: 2022-11-25.

Autodesk (2000). *Autodesk Revit*. https://www.autodesk.com/.

Badshah, N. and Chen, K. (2010). 'Image selective segmentation under geometrical constraints using an active contour approach'. In: *Communications in Computational Physics* 7.4, p. 759. DOI: 10.4208/cicp.2009.09.026.

Bauchet, J.-P. and Lafarge, F. (2018). 'KIPPI: KInetic Polygonal Partitioning of Images'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3146–3154. DOI: 10.1109/CVPR.2018.00332.

Bauchet, J.-P. and Lafarge, F. (2020). 'Kinetic Shape Reconstruction'. In: *ACM Trans. Graph.* 39.5. ISSN: 0730-0301. DOI: 10.1145/3376918.

Bay, H., Tuytelaars, T. and Van Gool, L. (2006). 'SURF: Speeded Up Robust Features'. In: *Computer Vision – ECCV 2006*. Ed. by Leonardis, A., Bischof, H. and Pinz, A. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 404–417. DOI: 10.1007/11744023_32.

Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C. and Gall, J. (2019). 'SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences'. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9296–9306. DOI: 10.1109/ICCV.2019.00939.

Ben-Shabat, Y., Avraham, T., Lindenbaum, M. and Fischer, A. (2018). 'Graph based over-segmentation methods for 3D point clouds'. In: *Computer Vision and Image Understanding* 174, pp. 12–23. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2018.06.004.

Benner, J., Geiger, A. and Leinemann, K. (2005). 'Flexible generation of semantic 3D building models'. In: *Proc of the 1st Intern. Workshop on Next Generation 3D City Models, Gröger/Kolbe (Eds.), Bonn*, pp. 17–22. URL: https://api.semanticscholar.org/CorpusID:16065625.

BENTLEY (2016). *ContextCapture*. https://www.bentley.com/zh/products/product-line/reality-modeling-software/contextcapture. Accessed: 2021-01-16.

Bentley (2015). *Acute3D*. https://www.acute3d.com/. URL: https://www.acute3d.com/.

Bernardini, F. and Bajaj, C. L. (1997). 'Sampling and reconstructing manifolds using alpha-shapes'. In: *Canadian Conference on Computational Geometry*. URL: https://api.semanticscholar.org/CorpusID:3327349.

Besuievsky, G., Barroso, S., Beckers, B. and Patow, G. (2014). 'A Configurable LoD for Procedural Urban Models intended for Daylight Simulation'. In: *Eurographics Workshop on Urban Data Modelling and Visualisation*. Ed. by Besuievsky, G. and Tourre, V. The Eurographics Association. ISBN: 978-3-905674-49-1. DOI: /10.2312/udmv.20141073.

Besuievsky, G., Beckers, B. and Patow, G. (2018). 'Skyline-based geometric simplification for urban solar analysis'. In: *Graphical Models* 95, pp. 42–50. ISSN: 1524-0703. DOI: 10.1016/j.gmod.2017.06.002.

Biljecki, F., Heuvelink, G. B., Ledoux, H. and Stoter, J. (2015a). 'Propagation of positional error in 3D GIS: estimation of the solar irradiation of building roofs'. In: *International Journal of Geographical Information Science* 29.12, pp. 2269–2294. DOI: 10.1080/13658816.2015.1073292.

Biljecki, F., Ledoux, H. and Stoter, J. (2014). 'Redefining the Level of Detail for 3D models'. In: *GIM International* 28, pp. 21–23. URL: https://filipbiljecki.com/publications/2014_gim_lod.pdf.

Biljecki, F., Ledoux, H. and Stoter, J. (2017). 'Does a finer level of detail of a 3D city model bring an improvement for estimating shadows?' In: *Advances in 3D geoinformation*, pp. 31–47. DOI: 10.1007/978-3-319-25691-7_2.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S. and Çöltekin, A. (2015b). 'Applications of 3D City Models: State of the Art Review'. In: *ISPRS International Journal of Geo-Information* 4.4, pp. 2842–2889. ISSN: 2220-9964. DOI: 10.3390/ijgi4042842.

Bischof, L. and Adams, R. (1994). 'Seeded Region Growing'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16.06, pp. 641–647. ISSN: 1939-3539. DOI: 10.1109/34.295913.

Blaha, M., Rothermel, M., Oswald, M. R., Sattler, T., Richard, A., Wegner, J. D., Pollefeys, M. and Schindler, K. (2017). 'Semantically Informed Multiview Surface Refinement'. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 3839–3847. DOI: 10.1109/ICCV.2017.412.

Blake, A., Rother, C., Brown, M., Perez, P. and Torr, P. (2004). 'Interactive image segmentation using an adaptive GMMRF model'. In: *Computer Vision-ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part I 8*. Springer, pp. 428–441. DOI: 10.1007/978-3-540-24670-1_33.

Bouzas, V., Ledoux, H. and Nan, L. (2020). 'Structure-aware Building Mesh Polygonization'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 167, pp. 432–442. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2020.07.010.

Boykov, Y., Veksler, O. and Zabih, R. (2001). 'Fast approximate energy minimization via graph cuts'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23.11, pp. 1222–1239. DOI: 10.1109/34.969114.

Boykov, Y. and Jolly, M.-P. (2001). 'Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images'. In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. Vol. 1, 105–112 vol.1. DOI: 10.1109/ICCV.2001.937505.

Bradley, D. and Roth, G. (2007). 'Adaptive thresholding using the integral image'. In: *Journal of graphics tools* 12.2, pp. 13–21. DOI: 10.1080/2151237X.2007.10129236.

Brédif, M. (2013). 'Image-Based Rendering of LOD1 3D City Models for traffic-augmented Immersive Street-view Navigation'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* II-3/W3, pp. 7–11. DOI: 10.5194/isprsannals-II-3-W3-7-2013.

Bresson, X., Esedolu, S., Vandergheynst, P., Thiran, J.-P. and Osher, S. (2007). 'Fast global minimization of the active contour/snake model'. In: *Journal of Mathematical Imaging and vision* 28.2, pp. 151–167. DOI: 10.1007/s10851-007-0002-0.

Briechle, K. and Hanebeck, U. D. (2001). 'Template matching using fast normalized cross correlation'. In: *Optical pattern recognition XII*. Vol. 4387. SPIE, pp. 95–102. DOI: 10.1117/12.421129.

Brostow, G. J., Fauqueur, J. and Cipolla, R. (2009). 'Semantic object classes in video: A high-definition ground truth database'. In: *Pattern Recognition Letters* 30.2. Video-based Object and Event Analysis, pp. 88–97. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2008.04.005.

Bykat, A. (1978). 'Convex hull of a finite set of points in two dimensions'. In: *Information Processing Letters* 7.6, pp. 296–298. ISSN: 0020-0190. DOI: 10.1016/0020-0190(78)90021-2.

Can, G., Mantegazza, D., Abbate, G., Chappuis, S. and Giusti, A. (2021). 'Semantic segmentation on Swiss3DCities: A benchmark study on aerial photogrammetric 3D pointcloud dataset'. In: *Pattern Recognition Letters* 150, pp. 108–114. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2021.06.004.

Cappelle, C., El Najjar, M. E., Charpillet, F. and Pomorski, D. (2012). 'Virtual 3D city model for navigation in urban areas'. In: *Journal of Intelligent & Robotic Systems* 66.3, pp. 377–399. DOI: 10.1007/s10846-011-9594-0.

Castrejón, L., Kundu, K., Urtasun, R. and Fidler, S. (2017). 'Annotating Object Instances with a Polygon-RNN'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4485–4493. DOI: 10.1109/CVPR.2017.477.

Catita, C., Redweik, P., Pereira, J. and Brito, M. (2014). 'Extending solar potential analysis in buildings to vertical facades'. In: *Computers & Geosciences* 66, pp. 1–12. ISSN: 0098-3004. DOI: 10.1016/j.cageo.2014.01.002.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Niebner, M., Savva, M., Song, S., Zeng, A. and Zhang, Y. (2017). 'Matterport3D: Learning from RGB-D Data in Indoor Environments'. In: *2017 International Conference on 3D Vision (3DV)*, pp. 667–676. DOI: 10.1109/3DV.2017.00081.

Chao, Q., Bi, H., Li, W., Mao, T., Wang, Z., Lin, M. C. and Deng, Z. (2020). 'A Survey on Visual Traffic Simulation: Models, Evaluations, and Applications in Autonomous Driving'. In: *Computer Graphics Forum* 39.1, pp. 287–308. DOI: 10.1111/cgf.13803.

Charles, R. Q., Su, H., Kaichun, M. and Guibas, L. J. (2017). 'PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 77–85. DOI: 10.1109/CVPR.2017.16.

Chauve, A.-L., Labatut, P. and Pons, J.-P. (2010). 'Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data'. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1261–1268. DOI: 10.1109/CVPR.2010.5539824.

Chen, J., Xu, Y., Lu, S., Liang, R. and Nan, L. (2022a). '3-D Instance Segmentation of MVS Buildings'. English. In: *IEEE Transactions on Geoscience and Remote Sensing* 60. ISSN: 0196-2892. DOI: 10.1109/TGRS.2022.3183567.

Chen, J., Xu, Y., Lu, S., Liang, R. and Nan, L. (2022b). '3-D Instance Segmentation of MVS Buildings'. In: *IEEE Transactions on Geoscience and Remote Sensing* 60, pp. 1–14. DOI: 10.1109/TGRS.2022.3183567.

Chen, M., Hu, Q., Yu, Z., THOMAS, H., Feng, A., Hou, Y., McCullough, K., Ren, F. and Soibelman, L. (2022). 'STPLS3D: A Large-Scale Synthetic and Real Aerial Photogrammetry 3D Point Cloud Dataset'. In: *33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022*. BMVA Press. URL: https://bmvc2022.mpi-inf.mpg.de/0429.pdf.

Chen, X., Zhao, Z., Zhang, Y., Duan, M., Qi, D. and Zhao, H. (2022). 'FocalClick: Towards Practical Interactive Image Segmentation'. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1290–1299. DOI: 10.1109/CVPR52688.2022.00136.

Chen, Y., Gao, W., Widyaningrum, E., Zheng, M. and Zhou, K. (2018). 'Building Classification of VHR Airborne Stereo Images Using Fully Convolutional Networks and Free Training Samples'. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-4, pp. 87–92. DOI: 10.5194/isprs-archives-XLII-4-87-2018.

Chen, Z., Ledoux, H., Khademi, S. and Nan, L. (2022). 'Reconstructing compact building models from point clouds using deep implicit fields'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 194, pp. 58–73. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2022.09.017.

Chen, Z., Shi, Y., Nan, L., Xiong, Z. and Zhu, X. X. (2023). 'PolyGNN: Polyhedron-based Graph Neural Network for 3D Building Reconstruction from Point Clouds'. In: *ArXiv* abs/2307.08636. URL: https://api.semanticscholar.org/CorpusID:259937801.

Cheng, B., Girshick, R., Dollár, P., Berg, A. C. and Kirillov, A. (2021). 'Boundary IoU: Improving Object-Centric Image Segmentation Evaluation'. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15329–15337. DOI: 10.1109/CVPR46437.2021.01508.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014). 'Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation'. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Moschitti, A., Pang, B. and Daelemans, W. Doha, Qatar: Association for Computational Linguistics, pp. 1724–1734. DOI: 10.3115/v1/D14-1179.

Chun, W., Ge, C., Yanyan, L. and Horne, M. (2008). 'Virtual-Reality Based Integrated Traffic Simulation for Urban Planning'. In: *2008 International Conference on Computer Science and Software Engineering*. Vol. 2, pp. 1137–1140. DOI: 10.1109/CSSE.2008.1074.

Cichowicz, R. and Dobrzaski, M. (2021). '3D Spatial Analysis of Particulate Matter (PM10, PM2.5 and PM1.0) and Gaseous Pollutants (H2S, SO2 and VOC) in Urban Areas Surrounding a Large Heat and Power Plant'. In: *Energies* 14.14. ISSN: 1996-1073. URL: https://www.mdpi.com/1996-1073/14/14/4070.

Cignoni, P., Rocchini, C. and Scopigno, R. (1998). 'Metro: Measuring Error on Simplified Surfaces'. In: *Computer Graphics Forum* 17.2, pp. 167–174. DOI: 10.1111/1467-8659.00236.

Cohen-Steiner, D., Alliez, P. and Desbrun, M. (2004). 'Variational shape approximation'. In: *ACM Trans. Graph.* 23.3, pp. 905–914. ISSN: 0730-0301. DOI: 10.1145/1015706.1015817.

Cook, R. L. (1986). 'Stochastic sampling in computer graphics'. In: *ACM Trans. Graph.* 5.1, pp. 51–72. ISSN: 0730-0301. DOI: 10.1145/7529.8927.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S. and Schiele, B. (2016). 'The Cityscapes Dataset for Semantic Urban Scene Understanding'. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 3213–3223. DOI: 10.1109/CVPR.2016.350.

Corongiu, M., Tucci, G., Santoro, E. and Kourounioti, O. (2018). 'Data Integration of Different Domains in Geo-Information Management: A Railway Infrastructure Case Study'. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-4, pp. 121–127. DOI: 10.5194/isprs-archives-XLII-4-121-2018.

Corsini, M., Cignoni, P. and Scopigno, R. (2012). 'Efficient and Flexible Sampling with Blue Noise Properties of Triangular Meshes'. In: *IEEE Transactions on Visualization and Computer Graphics* 18.6, pp. 914–924. DOI: 10.1109/TVCG.2012.34.

Czyska, K. and Rubinowicz, P. (2014). 'Application of 3D virtual city models in urban analyses of tall buildings: today practice and future challenges'. In: *Architecturae et Artibus* 6.1, pp. 9–13. URL: https://api.semanticscholar.org/CorpusID:29570051.

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T. and NieSSner, M. (2017). 'ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2432–2443. DOI: 10.1109/CVPR.2017.261.

Demantké, J., Mallet, C., David, N. and Vallet, B. (2011). 'Dimensionality-Based Scale Selection in 3D LiDAR Point Clouds'. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XXXVIII-5/W12, pp. 97–102. DOI: 10.5194/isprsarchives-XXXVIII-5-W12-97-2011.

Demir, I., Aliaga, D. G. and Benes, B. (2014). 'Proceduralization of Buildings at City Scale'. In: *2014 2nd International Conference on 3D Vision*. Vol. 1, pp. 456–463. DOI: 10.1109/3DV.2014.31.

Demir,., Aliaga, D. G. and Benes, B. (2016). 'Proceduralization for Editing 3D Architectural Models'. In: *2016 Fourth International Conference on 3D Vision (3DV)*, pp. 194–202. DOI: 10.1109/3DV.2016.28.

Deng, X., Zhang, W., Ding, Q. and Zhang, X. (2023). 'PointVector: A Vector Representation In Point Cloud Analysis'. In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9455–9465. DOI: 10.1109/CVPR52729.2023.00912.

DeTone, D., Malisiewicz, T. and Rabinovich, A. (2018). 'SuperPoint: Self-Supervised Interest Point Detection and Description'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 337–33712. DOI: 10.1109/CVPRW.2018.00060.

Dong, M., Xie, K. and Huang, H. (2024). 'Coupled Study on Instance Segmentation and Structural Reconstruction of Buildings in 3D Urban Scenes'. In: *SCIENTIA SINICA Informationis* 54.2, pp. 281–300. DOI: 10.1360/SSI-2023-0221.

Dong, X., Shen, J., Shao, L. and Van Gool, L. (2016). 'Sub-Markov Random Walk for Image Segmentation'. In: *IEEE Transactions on Image Processing* 25.2, pp. 516–527. DOI: 10.1109/TIP.2015.2505184.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S. *et al.* (2020). 'An image is worth 16x16 words: Transformers for image recognition at scale'. In: *arXiv preprint arXiv:2010.11929*. URL: https://arxiv.org/abs/2010.11929.

Doyle, S., Dodge, M. and Smith, A. (1998). 'The potential of Web-based mapping and virtual reality technologies for modelling urban environments'. In: *Computers, Environment and Urban Systems* 22.2, pp. 137–155. ISSN: 0198-9715. DOI: 10.1016/S0198-9715(98)00014-3.

Du, S., Lindenbergh, R., Ledoux, H., Stoter, J. and Nan, L. (2019). 'AdTree: Accurate, Detailed, and Automatic Modelling of Laser-Scanned Trees'. In: *Remote Sensing* 11.18. ISSN: 2072-4292. DOI: 10.3390/rs11182074.

Duane, C. B. (1971). 'Close-range camera calibration'. In: *Photogramm. Eng* 37.8, pp. 855–866. URL: https://www.isprs.org/proceedings/xxvi/part5/30_XXVI-part5.pdf.

Eicker, U., Nouvel, R., Duminil, E. and Coors, V. (2014). 'Assessing Passive and Active Solar Energy Resources in Cities Using 3D City Models'. In: *Energy Procedia* 57. 2013 ISES Solar World Congress, pp. 896–905. ISSN: 1876-6102. DOI: 10.1016/j.egypro.2014.10.299.

Fan, M. and Lee, T. C. (2015). 'Variants of seeded region growing'. In: *IET Image Processing* 9.6, pp. 478–485. DOI: 10.1049/iet-ipr.2014.0490.

Faugeras, O. and Keriven, R. (1998). 'Variational principles, surface evolution, PDEs, level set methods, and the stereo problem'. In: *IEEE Transactions on Image Processing* 7.3, pp. 336–344. DOI: 10.1109/83.661183.

Fei, B., Yang, W., Chen, W.-M., Li, Z., Li, Y., Ma, T., Hu, X. and Ma, L. (2022). 'Comprehensive Review of Deep Learning-Based 3D Point Cloud Completion Processing and Analysis'. In: *IEEE Transactions on Intelligent Transportation Systems* 23.12, pp. 22862–22883. DOI: 10.1109/TITS.2022.3195555.

Ferworn, A., Herman, S., Tran, J., Ufkes, A. and Mcdonald, R. (2013). 'Disaster scene reconstruction: modeling and simulating urban building collapse rubble within a game engine'. In: *Proceedings of the 2013 Summer Computer Simulation Conference*. SCSC '13. Toronto, Ontario, Canada: Society for Modeling & Simulation International. ISBN: 9781627482769. DOI: 10.5555/2557696.2557717.

Foundation, B. (2002). *Blender*. https://www.blender.org/. Accessed: 2021-01-16.

Frueh, C., Sammon, R. and Zakhor, A. (2004). 'Automated texture mapping of 3D city models with oblique aerial imagery'. In: *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.* Pp. 396–403. DOI: 10.1109/TDPVT.2004.1335266.

Fu, H., Jia, R., Gao, L., Gong, M., Zhao, B., Maybank, S. and Tao, D. (2021). '3D-FUTURE: 3D Furniture Shape with TextURE'. In: *Int. J. Comput. Vision* 129.12, pp. 3313–3337. ISSN: 0920-5691. DOI: 10.1007/s11263-021-01534-z.

Furukawa, Y. and Hernández, C. (2015). 'Multi-View Stereo: A Tutorial'. In: *Foundations and Trendső in Computer Graphics and Vision* 9.1-2, pp. 1–148. DOI: 10.1561/0600000052.

Furukawa, Y. and Ponce, J. (2010). 'Accurate, Dense, and Robust Multiview Stereopsis'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.8, pp. 1362–1376. DOI: 10.1109/TPAMI.2009.161.

Gabara, G. and Sawicki, P. (2023). 'CRBeDaSet: A Benchmark Dataset for High Accuracy Close Range 3D Object Reconstruction'. In: *Remote Sensing* 15.4. ISSN: 2072-4292. DOI: 10.3390/rs15041116.

Gao, L., Yang, J., Wu, T., Yuan, Y.-J., Fu, H., Lai, Y.-K. and Zhang, H. (2019). 'SDM-NET: deep generative network for structured deformable mesh'. In: *ACM Trans. Graph.* 38.6. ISSN: 0730-0301. DOI: 10.1145/3355089.3356488.

Gao, Q., Shen, X. and Niu, W. (2020). 'Large-Scale Synthetic Urban Dataset for Aerial Scene Understanding'. In: *IEEE Access* 8, pp. 42131–42140. DOI: 10.1109/ACCESS.2020.2976686.

Gao, W., Peters, R., Ledoux, H. and Stoter, J. (2024a). 'Filling holes in LoD2 building models'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* X-4/W5-2024, pp. 171–177. DOI: 10.5194/isprs-annals-X-4-W5-2024-171-2024.

Gao, W., Peters, R. and Stoter, J. (2024b). 'Building-PCC: Building Point Cloud Completion Benchmarks'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* X-4/W5-2024, pp. 179–186. DOI: 10.5194/isprs-annals-X-4-W5-2024-179-2024.

Gao, W., Nan, L., Boom, B. and Ledoux, H. (2021). 'SUM: A benchmark dataset of Semantic Urban Meshes'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 179, pp. 108–120. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2021.07.008.

Gao, W., Nan, L., Boom, B. and Ledoux, H. (2023). 'PSSNet: Planarity-sensible Semantic Segmentation of large-scale urban meshes'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 196, pp. 32–44. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2022.12.020.

Gao, W., Peters, R. and Stoter, J. (2024c). 'Unsupervised Roofline Extraction from True Orthophotos for LoD2 Building Model Reconstruction'. In: *Recent Advances in 3D Geoinformation Science*. Ed. by Kolbe, T. H., Donaubauer, A. and Beil, C. Cham: Springer Nature Switzerland, pp. 425–436. ISBN: 978-3-031-43699-4. DOI: 10.1007/978-3-031-43699-4_27.

Gao, X., Wu, K. and Pan, Z. (2022). 'Low-poly Mesh Generation for Building Models'. In: *ACM SIGGRAPH 2022 Conference Proceedings*. SIGGRAPH '22. Vancouver, BC, Canada: Association for Computing Machinery. ISBN: 9781450393379. DOI: 10.1145/3528233.3530716.

Garcia-Dorado, I., Aliaga, D. G., Bhalachandran, S., Schmid, P. and Niyogi, D. (2017). 'Fast Weather Simulation for Inverse Procedural Design of 3D Urban Models'. In: *ACM Trans. Graph.* 36.4. ISSN: 0730-0301. DOI: 10.1145/3072959.2999534.

García-Sánchez, C., Philips, D. and Gorlé, C. (2014). 'Quantifying inflow uncertainties for CFD simulations of the flow in downtown Oklahoma City'. In: *Building and Environment* 78, pp. 118–129. DOI: 10.1016/j.buildenv.2014.04.013.

García-Sánchez, C., Vitalis, S., Paden, I. and Stoter, J. (2021). 'The Impact of Level of Detail in 3D City Models for CFD-Based Wind Flow Simulations'. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLVI-4/W4-2021, pp. 67–72. DOI: 10.5194/isprs-archives-XLVI-4-W4-2021-67-2021.

Geiger, A., Lauer, M., Wojek, C., Stiller, C. and Urtasun, R. (2014). '3D Traffic Scene Understanding From Movable Platforms'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.5, pp. 1012–1025. DOI: 10.1109/TPAMI.2013.185.

Geurts, P., Ernst, D. and Wehenkel, L. (2006). 'Extremely randomized trees'. In: *Machine learning* 63.1, pp. 3–42. DOI: 10.1007/s10994-006-6226-1.

Girardeau-Montaut, D. (2016). *CloudCompare*. https://www.danielgm.net/cc/. Accessed: 2021-01-16.

Glander, T. and Döllner, J. (2009). 'Abstract representations for interactive visualization of virtual 3D city models'. In: *Computers, Environment and Urban Systems* 33.5. Geoinformation Generalisation and Multiple Representation, pp. 375–387. ISSN: 0198-9715. DOI: 10.1016/j.compenvurbsys.2009.07.003.

Gooch, B., Ashikhmin, M., Reinhard, E. and Shirley, P. (2001). 'Color Transfer between Images'. In: *IEEE Computer Graphics and Applications* 21.05, pp. 34–41. ISSN: 1558-1756. DOI: 10.1109/38.946629.

Google (2012). *3D Imagery in Google Earth*. https://earth.google.com/web/. Accessed: 2021-01-16.

Gouraud, H. (1971). 'Continuous Shading of Curved Surfaces'. In: *IEEE Transactions on Computers* C-20.6, pp. 623–629. DOI: 10.1109/T-C.1971.223313.

Grady, L. (2006). 'Random Walks for Image Segmentation'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.11, pp. 1768–1783. DOI: 10.1109/TPAMI.2006.233.

Graham, B., Engelcke, M. and Maaten, L. v. d. (2018). '3D Semantic Segmentation with Submanifold Sparse Convolutional Networks'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9224–9232. DOI: 10.1109/CVPR.2018.00961.

Gröger, G. and Plümer, L. (2012). 'CityGML Interoperable semantic 3D city models'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 71, pp. 12–33. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2012.04.004.

Gruen, A., Schrotter, G., Schubiger, S., Qin, R., Xiong, B., Xiao, C., Li, J. and Ling, X. (2020). *An Operable System for LoD3 Model Generation Using Multi-Source Data and User-Friendly Interactive Editing Virtual Singapore R&D Program -Towards Automatic Acquisition of 3D City Models for Virtual Singapore*. Tech. rep. ETH Zurich. DOI: 10.13140/RG.2.2.30667.21281.

Gui, S., Qin, R. and Tang, Y. (2022). 'Sat2LOD2: A Software for Automated LOD-2 Building Reconstruction from Satellite-Derived Orthophoto and Digital Surface Model'. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information*

Sciences XLIII-B2-2022, pp. 379–386. DOI: 10.5194/isprs-archives-XLIII-B2-2022-379-2022.

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L. and Bennamoun, M. (2021). 'Deep Learning for 3D Point Clouds: A Survey'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.12, pp. 4338–4364. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2020.3005434.

Gupta, A., Dollár, P. and Girshick, R. (2019). 'LVIS: A Dataset for Large Vocabulary Instance Segmentation'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5351–5359. DOI: 10.1109/CVPR.2019.00550.

Haala, N., Becker, S. and Kada, M. (2007). 'Cell Decomposition for Building Model Generation at Different Scales'. In: *2007 Urban Remote Sensing Joint Event*, pp. 1–6. DOI: 10.1109/URS.2007.371856.

Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K. and Pollefeys, M. (2017). 'SEMANTIC3D.NET: A New Large-Scale Point Cloud Classification Benchmark'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-1/W1, pp. 91–98. DOI: 10.5194/isprs-annals-IV-1-W1-91-2017.

Hackel, T., Wegner, J. D. and Schindler, K. (2016). 'Fast Semantic Segmentation of 3D Point Clouds with Strongly Varying Density'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* III-3, pp. 177–184. DOI: 10.5194/isprs-annals-III-3-177-2016.

Han, J., Zhu, L., Gao, X., Hu, Z., Zhou, L., Liu, H. and Shen, S. (2021). 'Urban Scene LOD Vectorized Modeling From Photogrammetry Meshes'. In: *IEEE Transactions on Image Processing* 30, pp. 7458–7471. DOI: 10.1109/TIP.2021.3106811.

Han, W., Wen, C., Chok, L., Tan, Y. L., Chan, S. L., Zhao, H. and Feng, C. (2024). 'Autoencoding tree for city generation and applications'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 208, pp. 176–189. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2024.01.010.

Hanocka, R., Hertz, A., Fish, N., Giryes, R., Fleishman, S. and Cohen-Or, D. (2019). 'MeshCNN: a network with an edge'. In: *ACM Trans. Graph.* 38.4. ISSN: 0730-0301. DOI: 10.1145/3306346.3322959.

He, K., Zhang, X., Ren, S. and Sun, J. (2016). 'Deep Residual Learning for Image Recognition'. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

He, Z., Wang, Y. and Cheng, Z. (2021). 'Manhattan-world urban building reconstruction by fitting cubes'. In: *Computer Graphics Forum* 40.7, pp. 289–300. DOI: 10.1111/cgf.14421.

Helsinki, C. (2019). *Helsinki's 3D city models*. https://www.hel.fi/helsinki/en/administration/information/general/3d. Accessed: 2020-11-25.

Hu, Q., Yang, B., Khalid, S., Xiao, W., Trigoni, N. and Markham, A. (2021). 'Towards Semantic Segmentation of Urban-Scale 3D Point Clouds: A Dataset, Benchmarks and Challenges'. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4975–4985. DOI: 10.1109/CVPR46437.2021.00494.

Hu, Q., Yang, B., Khalid, S., Xiao, W., Trigoni, N. and Markham, A. (2022). 'Sensaturban: Learning semantics from urban-scale photogrammetric point clouds'. In:

International Journal of Computer Vision 130.2, pp. 316–343. DOI: 10.1007/s11263-021-01554-9.

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N. and Markham, A. (2020). 'RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds'. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11105–11114. DOI: 10.1109/CVPR42600.2020.01112.

Hu, S.-M., Liu, Z.-N., Guo, M.-H., Cai, J.-X., Huang, J., Mu, T.-J. and Martin, R. R. (2022). 'Subdivision-based Mesh Convolution Networks'. In: *ACM Trans. Graph.* 41.3. ISSN: 0730-0301. DOI: 10.1145/3506694.

Hu, Z., Bai, X., Shang, J., Zhang, R., Dong, J., Wang, X., Sun, G., Fu, H. and Tai, C.-L. (2021). 'VMNet: Voxel-Mesh Network for Geodesic-Aware 3D Semantic Segmentation'. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15468–15478. DOI: 10.1109/ICCV48922.2021.01520.

Huang, P.-H., Matzen, K., Kopf, J., Ahuja, N. and Huang, J.-B. (2018). 'DeepMVS: Learning Multi-view Stereopsis'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2821–2830. DOI: 10.1109/CVPR.2018.00298.

Huang, J., Stoter, J., Peters, R. and Nan, L. (2022). 'City3D: Large-Scale Building Reconstruction from Airborne LiDAR Point Clouds'. In: *Remote Sensing* 14.9. ISSN: 2072-4292. DOI: 10.3390/rs14092254.

Huang, J., Zhang, H., Yi, L., Funkhouser, T., NieSSner, M. and Guibas, L. J. (2019). 'TextureNet: Consistent Local Parametrizations for Learning From High-Resolution Signals on Meshes'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4435–4444. DOI: 10.1109/CVPR.2019.00457.

Huang, S., Wang, R., Guo, B. and Yang, H. (2024). 'PBWR: Parametric-Building-Wireframe Reconstruction from Aerial LiDAR Point Clouds'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27778–27787. URL: https://arxiv.org/abs/2311.12062.

Hui, L., Yuan, J., Cheng, M., Xie, J., Zhang, X. and Yang, J. (2021). 'Superpoint Network for Point Cloud Oversegmentation'. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5490–5499. DOI: 10.1109/ICCV48922.2021.00546.

Ibrahim, M., Akhtar, N., Wise, M. and Mian, A. (2021). 'Annotation Tool and Urban Dataset for 3D Point Cloud Semantic Segmentation'. In: *IEEE Access* 9, pp. 35984–35996. DOI: 10.1109/ACCESS.2021.3062547.

Ioffe, S. and Szegedy, C. (2015). 'Batch normalization: accelerating deep network training by reducing internal covariate shift'. In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, pp. 448–456. URL: https://arxiv.org/abs/1502.03167.

Jain, S., Singh, R. and Seth, S. (2000). 'Design flood estimation using GIS supported GIUHApproach'. In: *Water resources management* 14, pp. 369–376. DOI: 10.1023/A:1011147623014.

Jang, W.-D. and Kim, C.-S. (2019). 'Interactive Image Segmentation via Backpropagating Refinement Scheme'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5292–5301. DOI: 10.1109/CVPR.2019.00544.

Jaromczyk, J. and Toussaint, G. (1992). 'Relative neighborhood graphs and their relatives'. In: *Proceedings of the IEEE* 80.9, pp. 1502–1517. DOI: 10.1109/5.163414.

Jayaraman, P. K., Zheng, J., Chen, Y. and Tan, C. B. (2021). 'Interactive Labeling for Generation of CityGML Building Models from Meshes'. In: *Intelligent Scene Modeling and Human-Computer Interaction*. Ed. by Thalmann, N. M., Zhang, J. J., Ramanathan, M. and Thalmann, D. Cham: Springer International Publishing, pp. 147–163. ISBN: 978-3-030-71002-6. DOI: 10.1007/978-3-030-71002-6_9.

Jiang, J., Zhao, M., Xin, S., Yang, Y., Wang, H., Jia, X. and Yan, D.-M. (2023). 'Structure-Aware Surface Reconstruction via Primitive Assembly'. In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14125–14134. DOI: 10.1109/ICCV51070.2023.01303.

Jiang, T., Li, S., Zhang, Q., Wang, G., Zhang, Z., Zeng, F., An, P., Jin, X., Liu, S. and Wang, Y. (2024). 'RailPC: A large-scale railway point cloud semantic segmentation dataset'. In: *CAAI Transactions on Intelligence Technology*. DOI: https://doi.org/10.1049/cit2.12349.

Kada, M. and Wichmann, A. (2013). 'Feature-Driven 3D Building Modeling using Planar Halfspaces'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* II-3/W3, pp. 37–42. DOI: 10.5194/isprsannals-II-3-W3-37-2013.

Katal, A., Mortezazadeh, M., Wang, L. (and Yu, H. (2022). 'Urban building energy and microclimate modeling From 3D city generation to dynamic simulations'. In: *Energy* 251, p. 123817. ISSN: 0360-5442. DOI: 10.1016/j.energy.2022.123817.

Kazak, J., Luca, F. and Partanen, J. (2022). 'Wind Comfort Analysis and Design of Small Scale Elements for Improving Urban Space Livability. A case study in Tallinn, Estonia'. In: eCAADe, pp. 247–256. ISBN: 9789491207334. URL: https://papers.cumincad.org/data/works/att/ecaade2022_55.pdf.

Kazhdan, M., Bolitho, M. and Hoppe, H. (2006). 'Poisson Surface Reconstruction'. In: *Symposium on Geometry Processing*. Ed. by Sheffer, A. and Polthier, K. The Eurographics Association. ISBN: 3-905673-24-X. DOI: /10.2312/SGP/SGP06/061-070.

Kelly, T., Femiani, J., Wonka, P. and Mitra, N. J. (2017). 'BigSUR: large-scale structured urban reconstruction'. In: *ACM Trans. Graph.* 36.6. ISSN: 0730-0301. DOI: 10.1145/3130800.3130823.

Kent, M. and Schiavon, S. (2020). 'Evaluation of the effect of landscape distance seen in window views on visual satisfaction'. In: *Building and Environment* 183, p. 107160. ISSN: 0360-1323. DOI: 10.1016/j.buildenv.2020.107160.

KIGA-digi (2019). *The Kalasatama Digital Twins Project - The final report of the KIRA-digi pilot project*. https://www.hel.fi/hel2/tietokeskus/data/helsinki/kaupunginkanslia/3D-malli/Helsinki3D_Kalasatama_Digital_Twins_020519.pdf. Accessed: 2020-11-25.

Kim, T. H., Lee, K. M. and Lee, S. U. (2008). 'Generative image segmentation using random walks with restart'. In: *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III 10*. Springer, pp. 264–275. DOI: 10.1007/978-3-540-88690-7_20.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P. and Girshick, R. (2023). 'Segment Anything'. In: *2023*

IEEE/CVF International Conference on Computer Vision (ICCV), pp. 3992–4003. DOI: 10.1109/ICCV51070.2023.00371.

Kolbe, T. H. and Donaubauer, A. (2021). 'Semantic 3D City Modeling and BIM'. In: *Urban Informatics*. Ed. by Shi, W., Goodchild, M. F., Batty, M., Kwan, M.-P. and Zhang, A. Singapore: Springer Singapore, pp. 609–636. ISBN: 978-981-15-8983-6. DOI: 10.1007/978-981-15-8983-6_34.

Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J. D. and Ledoux, H. (2021). 'The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and Multi-View-Stereo'. In: *ISPRS Open Journal of Photogrammetry and Remote Sensing* 1, p. 100001. ISSN: 2667-3932. DOI: 10.1016/j.ophoto.2021.100001.

Kontogianni, T., Celikkan, E., Tang, S. and Schindler, K. (2023). 'Interactive Object Segmentation in 3D Point Clouds'. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2891–2897. DOI: 10.1109/ICRA48891.2023.10160904.

Koutsoudis, A., Arnaoutoglou, F. and Chamzas, C. (2007). 'On 3D reconstruction of the old city of Xanthi. A minimum budget approach to virtual touring based on photogrammetry'. In: *Journal of Cultural Heritage* 8.1, pp. 26–31. ISSN: 1296-2074. DOI: 10.1016/j.culher.2006.08.003.

Krehl, A., Siedentop, S., Taubenböck, H. and Wurm, M. (2016). 'A Comprehensive View on Urban Spatial Structure: Urban Density Patterns of German City Regions'. In: *ISPRS International Journal of Geo-Information* 5.6. ISSN: 2220-9964. DOI: 10.3390/ijgi5060076.

Kumar, K., Labetski, A., Ledoux, H. and Stoter, J. (2019). 'An Improved LOD Framework for the Terrains in 3D City Models'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-4/W8, pp. 75–82. DOI: 10.5194/isprs-annals-IV-4-W8-75-2019.

Kumar, K., Ledoux, H. and Stoter, J. (2018). 'Compactly representing massive terrain models as TINs in CityGML'. In: *Transactions in GIS* 22.5, pp. 1152–1178. DOI: 10.1111/tgis.12456.

Kumar, R. and Petovello, M. G. (2014). 'A novel GNSS positioning technique for improved accuracy in urban canyon scenarios using 3D city model'. In: *Proceedings of the 27th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2014)*, pp. 2139–2148. URL: https://www.ion.org/publications/abstract.cfm?articleID=12508.

Kuru, A. and Yüzer, M. A. (2021). 'Urban growth prediction with parcel based 3D urban growth model (PURGOM)'. In: *MethodsX* 8, p. 101302. ISSN: 2215-0161. DOI: 10.1016/j.mex.2021.101302.

Lafarge, F. and Mallet, C. (2011). *Modeling urban landscapes from point clouds: a generic approach*. Technical Report RR-7612. Inria. URL: https://inria.hal.science/inria-00590897.

Lafarge, F. and Mallet, C. (2012). 'Creating large-scale city models from 3D-point clouds: a robust approach with hybrid representation'. In: *International journal of computer vision* 99.1, pp. 69–85. DOI: 10.1007/s11263-012-0517-8.

Landrieu, L. and Boussaha, M. (2019). 'Point Cloud Oversegmentation With Graph-Structured Deep Metric Learning'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7432–7441. DOI: 10.1109/CVPR.2019.00762.

Landrieu, L. and Simonovsky, M. (2018). 'Large-Scale Point Cloud Semantic Segmentation with Superpoint Graphs'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4558–4567. DOI: 10.1109/CVPR.2018.00479.

Lang, I., Xu, F., Decatur, D., Babu, S. and Hanocka, R. (2024). *iSeg: Interactive 3D Segmentation via Interactive Attention*. URL: https://arxiv.org/abs/2404.03219.

Laupheimer, D., Shams Eddin, M. H. and Haala, N. (2020). 'On the Association of LiDAR Point Clouds and Textured Meshes for Multi-Modal Semantic Segmentation'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* V-2-2020, pp. 509–516. DOI: 10.5194/isprs-annals-V-2-2020-509-2020.

Ledoux, H., Arroyo Ohori, K., Kumar, K., Dukai, B., Labetski, A. and Vitalis, S. (2019). 'CityJSON: A compact and easy-to-use encoding of the CityGML data model'. In: *Open Geospatial Data, Software and Standards* 4.1, pp. 1–12. DOI: 10.1186/s40965-019-0064-0.

Ledoux, H., Biljecki, F., Dukai, B., Kumar, K., Peters, R., Stoter, J. and Commandeur, T. (2021). '3dfier: automatic reconstruction of 3D city models'. In: *Journal of Open Source Software* 6.57, p. 2866. DOI: 10.21105/joss.02866.

Ledoux, H., Ohori, K. A., Peters, R. and Pronk, M. (2023). *Computational modelling of terrains*. Self-published. DOI: 10.5281/zenodo.3992107.

Lee, D.-T. and Schachter, B. J. (1980). 'Two algorithms for constructing a Delaunay triangulation'. In: *International Journal of Computer & Information Sciences* 9.3, pp. 219–242. DOI: 10.1007/BF00977785.

Lee, D., Cha, G. and Park, S. (2016). 'A study on data visualization of embedded sensors for building energy monitoring using BIM'. In: *International Journal of Precision Engineering and Manufacturing* 17.6, pp. 807–814. ISSN: 2005-4602. DOI: 10.1007/s12541-016-0099-4.

Lee, K. M., Myeong, H. and Song, G. (2018). 'SeedNet: Automatic Seed Generation with Deep Reinforcement Learning for Robust Interactive Segmentation'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1760–1768. DOI: 10.1109/CVPR.2018.00189.

Lei, H., Akhtar, N. and Mian, A. (2021). 'Picasso: A CUDA-based Library for Deep Learning over 3D Meshes'. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13849–13859. DOI: 10.1109/CVPR46437.2021.01364.

Leszek, K. (2015). 'Environmental and Urban Spatial Analysis Based on a 3D City Model'. In: *Computational Science and Its Applications – ICCSA 2015*. Cham: Springer International Publishing, pp. 633–645. ISBN: 978-3-319-21470-2. DOI: 10.1007/978-3-319-21470-2_46.

Lévy, B., Petitjean, S., Ray, N. and Maillot, J. (2002). 'Least squares conformal maps for automatic texture atlas generation'. In: *ACM Trans. Graph.* 21.3, pp. 362–371. ISSN: 0730-0301. DOI: 10.1145/566654.566590.

Li, L., Song, N., Sun, F., Liu, X., Wang, R., Yao, J. and Cao, S. (2022). 'Point2Roof: End-to-end 3D building roof modeling from airborne LiDAR point clouds'. In: *ISPRS Journal of*

Photogrammetry and Remote Sensing 193, pp. 17–28. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2022.08.027.

Li, M. and Nan, L. (2021). 'Feature-preserving 3D mesh simplification for urban buildings'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 173, pp. 135–150. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2021.01.006.

Li, M., Nan, L., Smith, N. and Wonka, P. (2016a). 'Reconstructing building mass models from UAV images'. In: *Computers & Graphics* 54. Special Issue on CAD/Graphics 2015, pp. 84–93. ISSN: 0097-8493. DOI: 10.1016/j.cag.2015.07.004.

Li, M., Wonka, P. and Nan, L. (2016b). 'Manhattan-world urban reconstruction from point clouds'. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*. Springer, pp. 54–69. DOI: 10.1007/978-3-319-46493-0_4.

Li, S., Luo, Z., Zhen, M., Yao, Y., Shen, T., Fang, T. and Quan, L. (2019). 'Cross-Atlas Convolution for Parameterization Invariant Learning on Textured Mesh Surface'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6136–6145. DOI: 10.1109/CVPR.2019.00630.

Li, W., Wolberg, G. and Zokai, S. (2011). 'Lightweight 3D Modeling of Urban Buildings from Range Data'. In: *2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pp. 124–131. DOI: 10.1109/3DIMPVT.2011.23.

Li, X., Li, C., Tong, Z., Lim, A., Yuan, J., Wu, Y., Tang, J. and Huang, R. (2020). 'Campus3D: A Photogrammetry Point Cloud Benchmark for Hierarchical Understanding of Outdoor Scene'. In: *Proceedings of the 28th ACM International Conference on Multimedia*. MM '20. Seattle, WA, USA: Association for Computing Machinery, pp. 238–246. ISBN: 9781450379885. DOI: 10.1145/3394171.3413661.

Li, Y., Tarlow, D., Brockschmidt, M. and Zemel, R. (2017). *Gated Graph Sequence Neural Networks*. URL: https://arxiv.org/abs/1511.05493.

Li, Z., Zhang, L., Mathiopoulos, P. T., Liu, F., Zhang, L., Li, S. and Liu, H. (2017). 'A hierarchical methodology for urban facade parsing from TLS point clouds'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 123, pp. 75–93. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2016.11.008.

Liao, Y., Donné, S. and Geiger, A. (2018). 'Deep Marching Cubes: Learning Explicit Surface Representations'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2916–2925. DOI: 10.1109/CVPR.2018.00308.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C. L. (2014). 'Microsoft coco: Common objects in context'. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48.

Lin, Y., Wang, C., Zhai, D., Li, W. and Li, J. (2018). 'Toward better boundary preserved supervoxel segmentation for 3D point clouds'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 143. ISPRS Journal of Photogrammetry and Remote Sensing Theme Issue Point Cloud Processing, pp. 39–47. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2018.05.004.

Lin, Z., Zhang, Z., Chen, L.-Z., Cheng, M.-M. and Lu, S.-P. (2020). 'Interactive Image Segmentation With First Click Attention'. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13336–13345. DOI: 10.1109/CVPR42600.2020.01335.

Lindstrom, P. and Turk, G. (1998). 'Fast and memory efficient polygonal simplification'. In: *Proceedings Visualization '98 (Cat. No.98CB36276)*, pp. 279–286. DOI: 10.1109/VISUAL.1998.745314.

Lindstrom, P. and Turk, G. (1999). 'Evaluation of memoryless simplification'. In: *IEEE Transactions on Visualization and Computer Graphics* 5.2, pp. 98–115. DOI: 10.1109/2945.773803.

Ling, H., Gao, J., Kar, A., Chen, W. and Fidler, S. (2019). 'Fast Interactive Object Annotation With Curve-GCN'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5252–5261. DOI: 10.1109/CVPR.2019.00540.

Liu, K. and Boehm, J. (2014). 'A New Framework For Interactive Segmentation of Point Clouds'. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XL-5, pp. 357–362. DOI: 10.5194/isprsarchives-XL-5-357-2014.

Liu, L., Zhang, L., Ma, J., Zhang, L., Zhang, X., Xiao, Z. and Yang, L. (2010). 'An improved line-of-sight method for visibility analysis in 3D complex landscapes'. In: *Science China Information Sciences* 53, pp. 2185–2194. DOI: 10.1007/s11432-010-4090-x.

Liu, M.-Y., Tuzel, O., Ramalingam, S. and Chellappa, R. (2011). 'Entropy rate superpixel segmentation'. In: *CVPR 2011*, pp. 2097–2104. DOI: 10.1109/CVPR.2011.5995323.

Liu, Q., Xu, Z., Bertasius, G. and Niethammer, M. (2023). 'SimpleClick: Interactive Image Segmentation with Simple Vision Transformers'. In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 22233–22243. DOI: 10.1109/ICCV51070.2023.02037.

Liu, W., Zang, Y., Xiong, Z., Bian, X., Wen, C., Lu, X., Wang, C., Marcato, J., Gonçalves, W. N. and Li, J. (2023). '3D building model generation from MLS point cloud and 3D mesh using multi-source data fusion'. In: *International Journal of Applied Earth Observation and Geoinformation* 116, p. 103171. ISSN: 1569-8432. DOI: 10.1016/j.jag.2022.103171.

Liu, X., Liu, Z., Zhang, Y., Gao, Z. and Tan, Y. (2024). 'UMeshSegNet: Semantic Segmentation of 3D Mesh Generated from UAV Photogrammetry *'. In: *2024 IEEE 18th International Conference on Control & Automation (ICCA)*, pp. 388–393. DOI: 10.1109/ICCA62789.2024.10591843.

Liu, Y., Gevers, T. and Li, X. (2015). 'Estimation of Sunlight Direction Using 3D Object Models'. In: *IEEE Transactions on Image Processing* 24.3, pp. 932–942. DOI: 10.1109/TIP.2014.2378032.

Liu, Y., Guo, B., Wang, S., Liu, S., Peng, Z. and Li, D. (2022a). 'Urban Building Mesh Polygonization Based on Plane-Guided Segmentation, Topology Correction and Corner Point Clump Optimization'. In: *Remote Sensing* 14.17. ISSN: 2072-4292. URL: https://www.mdpi.com/2072-4292/14/17/4300.

Liu, Y., Lin, L., Hu, Y., Xie, K., Fu, C.-W., Zhang, H. and Huang, H. (2022b). 'Learning Reconstructability for Drone Aerial Path Planning'. In: *ACM Trans. Graph.* 41.6. ISSN: 0730-0301. DOI: 10.1145/3550454.3555433.

Liu, Y., Obukhov, A., Wegner, J. D. and Schindler, K. (2024). 'Point2Building: Reconstructing buildings from airborne LiDAR point clouds'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 215, pp. 351–368. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2024.07.012.

Lorensen, W. E. and Cline, H. E. (1987). 'Marching cubes: A high resolution 3D surface construction algorithm'. In: *SIGGRAPH Comput. Graph.* 21.4, pp. 163–169. ISSN: 0097-8930. DOI: 10.1145/37402.37422.

Lothe, P., Bourgeois, S., Royer, E., Dhome, M. and Naudet-Collette, S. (2010). 'Real-time vehicle global localisation with a single camera in dense urban areas: Exploitation of coarse 3D city models'. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 863–870. DOI: 10.1109/CVPR.2010.5540127.

Lowe, D. (1999). 'Object recognition from local scale-invariant features'. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2, 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.

Luan, F., Paris, S., Shechtman, E. and Bala, K. (2017). 'Deep Photo Style Transfer'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 6997–7005. DOI: 10.1109/CVPR.2017.740.

Luo, J., Zhao, T., Cao, L. and Biljecki, F. (2022). 'Semantic Riverscapes: Perception and evaluation of linear landscapes from oblique imagery using computer vision'. In: *Landscape and Urban Planning* 228, p. 104569. ISSN: 0169-2046. DOI: https://doi.org/10.1016/j.landurbplan.2022.104569.

Luo, L., Zhu, J., Fu, L., Pirasteh, S., Li, W., Han, X. and Guo, Y. (2021). 'A suitability visualisation method for flood fusion 3D scene guided by disaster information'. In: *International Journal of Image and Data Fusion* 12.4, pp. 301–318. DOI: 10.1080/19479832.2021.1961315.

Luo, M. R., Cui, G. and Rigg, B. (2001). 'The development of the CIE 2000 colour-difference formula: CIEDE2000'. In: *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur* 26.5, pp. 340–350. DOI: 10.1002/col.1049.

Lussange, J., Yu, M., Tarabalka, Y. and Lafarge, F. (2023). *3D detection of roof sections from a single satellite image and application to LOD2-building reconstruction*. URL: https://arxiv.org/abs/2307.05409.

Ma, J., Bae, S. W. and Choi, S. (2012). '3D medial axis point approximation using nearest neighbors and the normal field'. In: *The Visual Computer* 28.1, pp. 7–19. DOI: 10.1007/s00371-011-0594-7.

Madrazo, L., Sicilia, Á. and Gamboa, G. (2012). 'SEMANCO: Semantic tools for carbon reduction in urban planning'. English. In: *eWork and eBusiness in Architecture, Engineering and Construction - Proceedings of the European Conference on Product and Process Modelling 2012, ECPPM 2012*, pp. 899–908. ISBN: 9780415621281. URL: https://api.semanticscholar.org/CorpusID:132429984.

Maninis, K., Caelles, S., Pont-Tuset, J. and Gool, L. V. (2018). 'Deep Extreme Cut: From Extreme Points to Object Segmentation'. In: *2018 IEEE/CVF Conference on Computer*

Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, pp. 616–625. DOI: 10.1109/CVPR.2018.00071.

Mao, B. and Ban, Y. (2011). 'Online visualization of 3D city model using CityGML and X3DOM'. In: *Cartographica: The International Journal for Geographic Information and Geovisualization* 46.2, pp. 109–114. DOI: 10.3138/carto.46.2.109.

Maragkogiannis, K., Kolokotsa, D., Maravelakis, E. and Konstantaras, A. (2014). 'Combining terrestrial laser scanning and computational fluid dynamics for the study of the urban thermal environment'. In: *Sustainable Cities and Society* 13, pp. 207–216. ISSN: 2210-6707. DOI: 10.1016/j.scs.2013.12.002.

Martinovi, A., Knopp, J., Riemenschneider, H. and Van Gool, L. (2015). '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)*, pp. 4456–4465. DOI: 10.1109/CVPR.2015.7299075.

Matrone, F., Lingua, A., Pierdicca, R., Malinverni, E. S., Paolanti, M., Grilli, E., Remondino, F., Murtiyoso, A. and Landes, T. (2020). 'A Benchmark for Large-Scale Heritage Point Cloud Semantic Segmentation'. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLIII-B2-2020, pp. 1419–1426. DOI: 10.5194/isprs-archives-XLIII-B2-2020-1419-2020.

McKinnon, T. and Hoff, P. (2017). 'Comparing RGB-based vegetation indices with NDVI for drone based agricultural sensing'. In: *Agribotix. Com* 21.17, pp. 1–8. URL: https://api.semanticscholar.org/CorpusID:53450160.

Melzer, T. (2007). 'Non-parametric segmentation of ALS point clouds using mean shift'. In: *Journal of Applied Geodesy* 1.3, pp. 159–170. DOI: doi:10.1515/jag.2007.018.

Michael Bleyer, C. R. and Rother, C. (2011). 'PatchMatch Stereo - Stereo Matching with Slanted Support Windows'. In: *Proceedings of the British Machine Vision Conference*. BMVA Press, pp. 14.1–14.11. ISBN: 1-901725-43-X. DOI: 10.5244/C.25.14.

Miksik, O., Vineet, V., Lidegaard, M., Prasaath, R., NieSSner, M., Golodetz, S., Hicks, S. L., Pérez, P., Izadi, S. and Torr, P. H. (2015). 'The Semantic Paintbrush: Interactive 3D Mapping and Recognition in Large Outdoor Spaces'. In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI '15. Seoul, Republic of Korea: Association for Computing Machinery, pp. 3317–3326. ISBN: 9781450331456. DOI: 10.1145/2702123.2702222.

Mortensen, E. N. and Barrett, W. A. (1998). 'Interactive Segmentation with Intelligent Scissors'. In: *Graphical Models and Image Processing* 60.5, pp. 349–384. ISSN: 1077-3169. DOI: 10.1006/gmip.1998.0480.

Munoz, D., Bagnell, J. A., Vandapel, N. and Hebert, M. (2009). 'Contextual classification with functional Max-Margin Markov Networks'. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 975–982. DOI: 10.1109/CVPR.2009.5206590.

Nair, V. and Hinton, G. E. (2010). 'Rectified linear units improve restricted boltzmann machines'. In: *Proceedings of the 27th International Conference on Machine Learning*. ICML'10. Haifa, Israel: Omnipress, pp. 807–814. ISBN: 9781605589077. DOI: 10.5555/3104322.3104425.

Nan, L. (2021a). 'Easy3D: a lightweight, easy-to-use, and efficient C++ library for processing and rendering 3D data'. In: *Journal of Open Source Software* 6.64, p. 3255. DOI: 10.21105/joss.03255.

Nan, L. (2021b). 'Easy3D: a lightweight, easy-to-use, and efficient C++ library for processing and rendering 3D data'. In: *Journal of Open Source Software* 6.64, p. 3255. DOI: 10.21105/joss.03255.

Nan, L., Jiang, C., Ghanem, B. and Wonka, P. (2015). 'Template Assembly for Detailed Urban Reconstruction'. In: *Computer Graphics Forum* 34.2, pp. 217–228. DOI: 10.1111/cgf.12554.

Nan, L., Sharf, A., Zhang, H., Cohen-Or, D. and Chen, B. (2010). 'SmartBoxes for interactive urban reconstruction'. In: *ACM Trans. Graph.* 29.4. ISSN: 0730-0301. DOI: 10.1145/1778765.1778830.

Nan, L. and Wonka, P. (2017). 'PolyFit: Polygonal Surface Reconstruction from Point Clouds'. In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2372–2380. DOI: 10.1109/ICCV.2017.258.

Nan, L., Xie, K. and Sharf, A. (2012). 'A search-classify approach for cluttered indoor scene understanding'. In: *ACM Trans. Graph.* 31.6. ISSN: 0730-0301. DOI: 10.1145/2366145.2366156.

Neumann, A. and Neumann, L. (2005). 'Color Style Transfer Techniques using Hue, Lightness and Saturation Histogram Matching'. In: *Computational Aesthetics in Graphics, Visualization and Imaging*. Ed. by Neumann, L., Sbert, M., Gooch, B. and Purgathofer, W. The Eurographics Association. ISBN: 3-905673-27-4. DOI: /10.2312/COMPAESTH/COMPAESTH05/111-122.

Nguyen, D. T., Hua, B.-S., Yu, L.-F. and Yeung, S.-K. (2018). 'A Robust 3D-2D Interactive Tool for Scene Segmentation and Annotation'. In: *IEEE Transactions on Visualization and Computer Graphics* 24.12, pp. 3005–3018. DOI: 10.1109/TVCG.2017.2772238.

Nguyen, T. N. A., Cai, J., Zhang, J. and Zheng, J. (2012). 'Robust Interactive Image Segmentation Using Convex Active Contours'. In: *IEEE Transactions on Image Processing* 21.8, pp. 3734–3743. DOI: 10.1109/TIP.2012.2191566.

Niemeyer, J., Rottensteiner, F. and Soergel, U. (2014). 'Contextual classification of lidar data and building object detection in urban areas'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 87, pp. 152–165. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2013.11.001.

Ning, J., Zhang, L., Zhang, D. and Wu, C. (2010). 'Interactive image segmentation by maximal similarity based region merging'. In: *Pattern Recognition* 43.2. Interactive Imaging and Vision, pp. 445–456. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2009.03.004.

Ogawa, Y., Nakamura, R., Sato, G., Maeda, H. and Sekimoto, Y. (2024). 'End-to-End Framework for the Automatic Matching of Omnidirectional Street Images and Building Data and the Creation of 3D Building Models'. In: *Remote Sensing* 16.11. ISSN: 2072-4292. DOI: 10.3390/rs16111858.

OGC (2012). *OGC City Geography Markup Language (CityGML) Encoding Standard*. Open Geospatial Consortium inc. Document 12-019, version 2.0.0.

OGC (2021). *OGC City Geography Markup Language (CityGML) Part 1: Conceptual Model Standard*. Open Geospatial Consortium inc. Document 20-010, version 3.0.0, available at https://docs.ogc.org/is/20-010/20-010.html.

Over, M., Schilling, A., Neubauer, S. and Zipf, A. (2010). 'Generating web-based 3D City Models from OpenStreetMap: The current situation in Germany'. In: *Computers,*

Environment and urban systems 34.6, pp. 496–507. DOI: 10.1007/978-3-319-52522-8_9.

Paen, I., García-Sánchez, C. and Ledoux, H. (2022). 'Towards automatic reconstruction of 3D city models tailored for urban flow simulations'. In: *Frontiers in Built Environment*, p. 141. DOI: 10.3389/fbuil.2022.899332.

Padsala, R. and Coors, V. (2015). 'Conceptualizing, Managing and Developing: A Web Based 3D City Information Model for Urban Energy Demand Simulation'. In: *Eurographics Workshop on Urban Data Modelling and Visualisation*. Ed. by Biljecki, F. and Tourre, V. The Eurographics Association. ISBN: 978-3-905674-80-4. DOI: 10.2312/udmv.20151347.

Pan, S. J. and Yang, Q. (2010). 'A Survey on Transfer Learning'. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10, pp. 1345–1359. DOI: 10.1109/TKDE.2009.191.

Panagiotidou, M., Brito, M. C., Hamza, K., Jasieniak, J. J. and Zhou, J. (2021). 'Prospects of photovoltaic rooftops, walls and windows at a city to building scale'. In: *Solar Energy* 230, pp. 675–687. ISSN: 0038-092X. DOI: 10.1016/j.solener.2021.10.060.

Pang, H. E. and Biljecki, F. (2022). '3D building reconstruction from single street view images using deep learning'. In: *International Journal of Applied Earth Observation and Geoinformation* 112, p. 102859. ISSN: 1569-8432. DOI: https://doi.org/10.1016/j.jag.2022.102859.

Pantoja-Rosero, B., Achanta, R., Kozinski, M., Fua, P., Perez-Cruz, F. and Beyer, K. (2022). 'Generating LOD3 building models from structure-from-motion and semantic segmentation'. In: *Automation in Construction* 141, p. 104430. DOI: 10.1016/j.autcon.2022.104430.

Papon, J., Abramov, A., Schoeler, M. and Wörgötter, F. (2013). 'Voxel Cloud Connectivity Segmentation - Supervoxels for Point Clouds'. In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2027–2034. DOI: 10.1109/CVPR.2013.264.

Park, J. J., Florence, P., Straub, J., Newcombe, R. and Lovegrove, S. (2019). 'DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 165–174. DOI: 10.1109/CVPR.2019.00025.

Park, S. H., Jang, Y.-H., Geem, Z. W. and Lee, S.-H. (2019). 'CityGML-Based Road Information Model for Route Optimization of Snow-Removal Vehicle'. In: *ISPRS International Journal of Geo-Information* 8.12. ISSN: 2220-9964. URL: https://www.mdpi.com/2220-9964/8/12/588.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. and Chintala, S. (2019). 'PyTorch: an imperative style, high-performance deep learning library'. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc. DOI: 10.5555/3454287.3455008.

Patil, A., Malla, S., Gang, H. and Chen, Y.-T. (2019). 'The H3D Dataset for Full-Surround 3D Multi-Object Detection and Tracking in Crowded Urban Scenes'. In: *2019 International Conference on Robotics and Automation (ICRA)*. Montreal, QC, Canada: IEEE Press, pp. 9552–9557. DOI: 10.1109/ICRA.2019.8793925.

Pearson, K. (1901). 'LIII. On lines and planes of closest fit to systems of points in space'. In: *Philosophical Magazine Series 1* 2.11, pp. 559–572. DOI: 10.1080/14786440109462720.

Pédrinis, F., Morel, M. and Gesquière, G. (2015). 'Change Detection of Cities'. In: *3D Geoinformation Science: The Selected Papers of the 3D GeoInfo 2014*. Ed. by Breunig, M., Al-Doori, M., Butwilowski, E., Kuper, P. V., Benner, J. and Haefele, K. H. Cham: Springer International Publishing, pp. 123–139. ISBN: 978-3-319-12181-9. DOI: 10.1007/978-3-319-12181-9_8.

Peters, R. (2018). 'Geographical point cloud modelling with the 3D medial axis transform'. PhD thesis. Technische Universiteit Delft. DOI: 10.4233/f3a5f5af-ea54-40ba-8702-e193a087f243.

Peters, R., Dukai, B., Vitalis, S., van Liempt, J. and Stoter, J. (2022). 'Automated 3D Reconstruction of LoD2 and LoD1 Models for All 10 Million Buildings of the Netherlands'. English. In: *Photogrammetric Engineering and Remote Sensing* 88.3, pp. 165–170. ISSN: 0099-1112. DOI: 10.14358/PERS.21-00032R2.

Peters, R. and Ledoux, H. (2016). 'Robust approximation of the Medial Axis Transform of LiDAR point clouds as a tool for visualisation'. In: *Computers & Geosciences* 90, pp. 123–133. ISSN: 0098-3004. DOI: 10.1016/j.cageo.2016.02.019.

Peters, R., Dukai, B., Gao, W. and Stoter, J. (2023). '3D BAG-Geactualiseerd op basis van AHN4'. In: *Geo-Info* 2023.2, pp. 30–35. URL: https://repository.tudelft.nl/record/uuid:daf27e62-9a93-4b92-8646-c7cd16f572ce.

Peyraud, S., Bétaille, D., Renault, S., Ortiz, M., Mougel, F., Meizel, D. and Peyret, F. (2013). 'About Non-Line-Of-Sight Satellite Detection and Exclusion in a 3D Map-Aided Localization Algorithm'. In: *Sensors* 13.1, pp. 829–847. ISSN: 1424-8220. DOI: 10.3390/s130100829.

Pons, J.-P., Keriven, R. and Faugeras, O. (2007). 'Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score'. In: *International Journal of Computer Vision* 72.2, p. 179. DOI: 10.1007/s11263-006-8671-5.

Purnomo, B., Cohen, J. D. and Kumar, S. (2004). 'Seamless texture atlases'. In: *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*. SGP '04. Nice, France: Association for Computing Machinery, pp. 65–74. ISBN: 3905673134. DOI: 10.1145/1057432.1057441.

Pütz, S., Wiemann, T. and Hertzberg, J. (2021a). 'The Mesh Tools Package Introducing Annotated 3D Triangle Maps in ROS'. In: *Robotics and Autonomous Systems* 138, p. 103688. ISSN: 0921-8890. DOI: 10.1016/j.robot.2020.103688.

Pütz, S., Wiemann, T., Piening, M. K. and Hertzberg, J. (2021b). 'Continuous Shortest Path Vector Field Navigation on 3D Triangular Meshes for Mobile Robots'. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2256–2263. DOI: 10.1109/ICRA48506.2021.9560981.

Puyana-Romero, V., Cueto, J. L. and Gey, R. (2020). 'A 3D GIS tool for the detection of noise hot-spots from major roads'. In: *Transportation Research Part D: Transport and Environment* 84, p. 102376. ISSN: 1361-9209. DOI: 10.1016/j.trd.2020.102376.

Qi, C. R., Yi, L., Su, H. and Guibas, L. J. (2017). 'PointNet++: deep hierarchical feature learning on point sets in a metric space'. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California,

USA: Curran Associates Inc., pp. 5105–5114. ISBN: 9781510860964. DOI: 10.5555/3295222.3295263.

Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H., Elhoseiny, M. and Ghanem, B. (2022). 'PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies'. In: *Advances in Neural Information Processing Systems*. Ed. by Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K. and Oh, A. Vol. 35. Curran Associates, Inc., pp. 23192–23204. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/9318763d049edf9a1f2779b2a59911d3-Paper-Conference.pdf.

Qin, R. (2014). 'Change detection on LOD 2 building models with very high resolution spaceborne stereo imagery'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 96, pp. 179–192. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2014.07.007.

Rajchl, M., Lee, M. C. H., Oktay, O., Kamnitsas, K., Passerat-Palmbach, J., Bai, W., Damodaram, M., Rutherford, M. A., Hajnal, J. V., Kainz, B. and Rueckert, D. (2017). 'DeepCut: Object Segmentation From Bounding Box Annotations Using Convolutional Neural Networks'. In: *IEEE Transactions on Medical Imaging* 36.2, pp. 674–683. DOI: 10.1109/TMI.2016.2621185.

Ramadan, H., Lachqar, C. and Tairi, H. (2020). 'A survey of recent interactive image segmentation methods'. In: *Computational visual media* 6.4, pp. 355–384. DOI: 10.1007/s41095-020-0177-5.

Ran, C. (2011). 'The development of 3D city model and its applications in urban planning'. In: *2011 19th International Conference on Geoinformatics*. IEEE. DOI: 10.1109/geoinformatics.2011.5981007.

Rhinocentre (1992). *Rhinoceros 3D*. https://www.rhinocentre.com/.

Riemenschneider, H., Bódis-Szomorú, A., Weissenberg, J. and Van Gool, L. (2014). 'Learning where to classify in multi-view semantic segmentation'. In: *European Conference on Computer Vision*. Springer, pp. 516–532. DOI: 10.1007/978-3-319-10602-1_34.

Romanoni, A., Ciccone, M., Visin, F. and Matteucci, M. (2017). 'Multi-view Stereo with Single-View Semantic Mesh Refinement'. In: *2017 IEEE International Conference on Computer Vision Workshop (ICCVW)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 706–715. DOI: 10.1109/ICCVW.2017.89.

Romanoni, A. and Matteucci, M. (2018). 'A Data-Driven Prior on Facet Orientation for Semantic Mesh Labeling'. In: *2018 International Conference on 3D Vision (3DV)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 662–671. DOI: 10.1109/3DV.2018.00081.

Rong, Y., Zhang, T., Zheng, Y., Hu, C., Peng, L. and Feng, P. (2020). 'Three-dimensional urban flood inundation simulation based on digital aerial photogrammetry'. In: *Journal of Hydrology* 584, p. 124308. ISSN: 0022-1694. DOI: 10.1016/j.jhydrol.2019.124308.

Ropinski, T., Steinicke, F. and Hinrichs, K. (2005). 'A constrained road-based VR navigation technique for travelling in 3D city models'. In: *Proceedings of the 2005 International Conference on Augmented Tele-Existence*. ICAT '05. Christchurch, New Zealand: Association for Computing Machinery, pp. 228–235. ISBN: 0473106574. DOI: 10.1145/1152399.1152441.

Ross, L. (2011). 'Virtual 3D City Models in Urban Land Management - Technologies and Applications'. PhD thesis. Technische Universität Berlin. DOI: 10.14279/depositonce-2744.

Rother, C., Kolmogorov, V. and Blake, A. (2004). '"GrabCut": interactive foreground extraction using iterated graph cuts'. In: *ACM Trans. Graph.* 23.3, pp. 309–314. ISSN: 0730-0301. DOI: 10.1145/1015706.1015720.

Rouhani, M., Lafarge, F. and Alliez, P. (2017). 'Semantic segmentation of 3D textured meshes for urban scene analysis'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 123, pp. 124–139. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2016.12.001.

Roynard, X., Deschaud, J.-E. and Goulette, F. (2018). 'Paris-Lille-3D: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification'. In: *The International Journal of Robotics Research* 37.6, pp. 545–557. DOI: 10.1177/0278364918767506.

Salinas, D., Lafarge, F. and Alliez, P. (2015). 'Structure-Aware Mesh Decimation'. In: *Computer Graphics Forum* 34.6, pp. 211–227. DOI: 10.1111/cgf.12531.

San José, R., Pérez, J. L. and González, R. (2012). 'Advances in 3D visualization of air quality data'. In: *Usage, Usability, and Utility of 3D City Models–European COST Action TU0801*, p. 02002. DOI: 10.1051/3u3d/201202002.

Sánchez-Aparicio, M., Martín-Jiménez, J., Del Pozo, S., González-González, E. and Lagüela, S. (2021). 'Ener3DMap-SolarWeb roofs: A geospatial web-based platform to compute photovoltaic potential'. In: *Renewable and Sustainable Energy Reviews* 135, p. 110203. ISSN: 1364-0321. DOI: 10.1016/j.rser.2020.110203.

Santana, J. M., Wendel, J., Trujillo, A., Suárez, J. P., Simons, A. and Koch, A. (2017). 'Multimodal location based services semantic 3D city data as virtual and augmented reality'. In: *Progress in location-based services 2016*. Springer, pp. 329–353. DOI: 10.1007/978-3-319-47289-8_17.

Santos, T., Gomes, N., Freire, S., Brito, M., Santos, L. and Tenedório, J. (2014). 'Applications of solar mapping in the urban environment'. In: *Applied Geography* 51, pp. 48–57. ISSN: 0143-6228. DOI: 10.1016/j.apgeog.2014.03.008.

Saran, S., Wate, P., Srivastav, S. and Krishna Murthy, Y. (2015). 'CityGML at semantic level for urban energy conservation strategies'. In: *Annals of GIS* 21.1, pp. 27–41. DOI: 10.1080/19475683.2014.992370.

Sarlin, P.-E., DeTone, D., Malisiewicz, T. and Rabinovich, A. (2020). 'SuperGlue: Learning Feature Matching With Graph Neural Networks'. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4937–4946. DOI: 10.1109/CVPR42600.2020.00499.

Schmitz, P., Suder, S., Schuster, K. and Kobbelt, L. (2022). 'Interactive Segmentation of Textured Point Clouds'. In: *Vision, Modeling, and Visualization*. Ed. by Bender, J., Botsch, M. and Keim, D. A. The Eurographics Association. ISBN: 978-3-03868-189-2. DOI: 10.2312/vmv.20221200.

Schnabel, R., Wahl, R. and Klein, R. (2007). 'Efficient RANSAC for Point-Cloud Shape Detection'. In: *Computer Graphics Forum* 26.2, pp. 214–226. DOI: 10.1111/j.1467-8659.2007.01016.x.

Schönberger, J. L. and Frahm, J.-M. (2016). 'Structure-from-Motion Revisited'. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4104–4113. DOI: 10.1109/CVPR.2016.445.

Schröter, K., Lüdtke, S., Redweik, R., Meier, J., Bochow, M., Ross, L., Nagel, C. and Kreibich, H. (2018). 'Flood loss estimation using 3D city models and remote sensing data'. In: *Environmental Modelling & Software* 105, pp. 118–131. ISSN: 1364-8152. DOI: 10.1016/j.envsoft.2018.03.032.

Schuegraf, P., Fuentes Reyes, M., Xu, Y. and Bittner, K. (2023). 'Roof3D: A Real and Synthetic Data Collection for Individual Building Roof Plane and Building Sections Detection'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* X-1/W1-2023, pp. 971–979. DOI: 10.5194/isprs-annals-X-1-W1-2023-971-2023.

Schult, J., Engelmann, F., Kontogianni, T. and Leibe, B. (2020). 'DualConvMesh-Net: Joint Geodesic and Euclidean Convolutions on 3D Meshes'. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8609–8619. DOI: 10.1109/CVPR42600.2020.00864.

Schwab, B., Beil, C. and Kolbe, T. H. (2020). 'Spatio-Semantic Road Space Modeling for VehiclePedestrian Simulation to Test Automated Driving Systems'. In: *Sustainability* 12.9. ISSN: 2071-1050. DOI: 10.3390/su12093799.

Se, S. and Jasiobedzki, P. (2006). 'Photo-realistic 3D model reconstruction'. In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* Pp. 3076–3082. DOI: 10.1109/ROBOT.2006.1642169.

Selvaraju, P., Nabail, M., Loizou, M., Maslioukova, M., Averkiou, M., Andreou, A., Chaudhuri, S. and Kalogerakis, E. (2021). 'BuildingNet: Learning to Label 3D Buildings'. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10377–10387. DOI: 10.1109/ICCV48922.2021.01023.

Serna, A., Marcotegui, B., Goulette, F. and Deschaud, J.-E. (2014). 'Paris-rue-Madame database: a 3D mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods'. In: *4th International Conference on Pattern Recognition, Applications and Methods ICPRAM 2014*. Angers, France. URL: https://hal.science/hal-00963812.

Sharkawi, K. H. and Rahman, A. A. (2014). 'Towards an Efficient City Inventory Management System for Urban Authorities in Developing Countries–The Case of 3D Change Detection'. In: *The Electronic Journal of Information Systems in Developing Countries* 60.1, pp. 1–13. DOI: 10.1002/j.1681-4835.2014.tb00424.x.

Sheffer, A., Lévy, B., Mogilnitsky, M. and Bogomyakov, A. (2005). 'ABF++: fast and robust angle based flattening'. In: *ACM Trans. Graph.* 24.2, pp. 311–330. ISSN: 0730-0301. DOI: 10.1145/1061347.1061354.

Sherbrooke, E., Patrikalakis, N. and Brisson, E. (1996). 'An algorithm for the medial axis transform of 3D polyhedral solids'. In: *IEEE Transactions on Visualization and Computer Graphics* 2.1, pp. 44–61. DOI: 10.1109/2945.489386.

Simonovsky, M. and Komodakis, N. (2017). 'Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29–38. DOI: 10.1109/CVPR.2017.11.

Sindram, M. and Kolbe, T. H. (2014). 'Modeling of urban planning actions by complex transactions on semantic 3D city models'. In: *Proceedings of the International Environmental Modelling and Software Society (iEMSs)*. URL: https://mediatum.ub.tum.de/doc/1224665/file.pdf.

Sinha, R., Sapre, A., Patil, A., Singhvi, A., Sathe, M. and Rathi, V. (2012). 'Earthquake disaster simulation in immersive 3d environment'. In: *15th World conference on earthquake engineering*, pp. 24–28. URL: https://www.iitk.ac.in/nicee/wcee/article/WCEE2012_3044.pdf.

Sinha, S. N., Steedly, D., Szeliski, R., Agrawala, M. and Pollefeys, M. (2008). 'Interactive 3D architectural modeling from unordered photo collections'. In: *ACM SIGGRAPH Asia 2008 Papers*. SIGGRAPH Asia '08. Singapore: Association for Computing Machinery. ISBN: 9781450318310. DOI: 10.1145/1457515.1409112.

Snavely, N., Seitz, S. M. and Szeliski, R. (2008). 'Modeling the world from internet photo collections'. In: *International journal of computer vision* 80, pp. 189–210. DOI: 10.1007/s11263-007-0107-3.

Snavely, N., Seitz, S. M. and Szeliski, R. (2006). 'Photo tourism: exploring photo collections in 3D'. In: *ACM siggraph 2006 papers*. Vol. 25. 3. New York, NY, USA: Association for Computing Machinery, pp. 835–846. DOI: 10.1145/1141911.1141964.

Sofia, H., Anas, E. and Faïz, O. (2020). 'Mobile Mapping, Machine Learning and Digital Twin for Road Infrastructure Monitoring and Maintenance: Case Study of Mohammed VI Bridge in Morocco'. In: *2020 IEEE International conference of Moroccan Geomatics (Morgeo)*, pp. 1–6. DOI: 10.1109/Morgeo49228.2020.9121882.

Sofiiuk, K., Petrov, I., Barinova, O. and Konushin, A. (2020). 'F-BRS: Rethinking Backpropagating Refinement for Interactive Segmentation'. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8620–8629. DOI: 10.1109/CVPR42600.2020.00865.

Sofiiuk, K., Petrov, I. A. and Konushin, A. (2022). 'Reviving Iterative Training with Mask Guidance for Interactive Segmentation'. In: *2022 IEEE International Conference on Image Processing (ICIP)*, pp. 3141–3145. DOI: 10.1109/ICIP46576.2022.9897365.

Soheilian, B., Tournaire, O., Paparoditis, N., Vallet, B. and Papelard, J.-P. (2013). 'Generation of an integrated 3D city model with visual landmarks for autonomous navigation in dense urban areas'. In: *2013 IEEE Intelligent Vehicles Symposium (IV)*, pp. 304–309. DOI: 10.1109/IVS.2013.6629486.

Song, S., Lichtenberg, S. P. and Xiao, J. (2015). 'SUN RGB-D: A RGB-D scene understanding benchmark suite'. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 567–576. DOI: 10.1109/CVPR.2015.7298655.

Steinlechner, H., Rainer, B., Schwärzler, M., Haaser, G., Szabo, A., Maierhofer, S. and Wimmer, M. (2019). 'Adaptive pointcloud segmentation for assisted interactions'. In: *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. I3D '19. Montreal, Quebec, Canada: Association for Computing Machinery. ISBN: 9781450363105. DOI: 10.1145/3306131.3317023.

Sui, W., Wang, L., Fan, B., Xiao, H., Wu, H. and Pan, C. (2016). 'Layer-Wise Floorplan Extraction for Automatic Urban Building Reconstruction'. In: *IEEE Transactions on*

Visualization and Computer Graphics 22.3, pp. 1261–1277. DOI: 10.1109/TVCG.2015.2505296.

Sun, G., Webster, C. and Zhang, X. (2019). 'Connecting the city: A three-dimensional pedestrian network of Hong Kong (RTPI Research Excellence Commended Award)'. In: *Environment and Planning B Urban Analytics and City Science*, pp. 1–16. DOI: 10.1177/2399808319847204.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. (2016). 'Rethinking the Inception Architecture for Computer Vision'. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 2818–2826. DOI: 10.1109/CVPR.2016.308.

Li-Ta, H., Yanlei, G. and Shunsuke, K. (2015). 'NLOS Correction/Exclusion for GNSS Measurement Using RAIM and City Building Models'. In: *Sensors* 15.7, pp. 17329–17349. DOI: 10.3390/s150717329.

Tan, W., Qin, N., Ma, L., Li, Y., Du, J., Cai, G., Yang, K. and Li, J. (2020). 'Toronto-3D: A Large-scale Mobile LiDAR Dataset for Semantic Segmentation of Urban Roadways'. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 797–806. DOI: 10.1109/CVPRW50498.2020.00109.

Tang, M., Gorelick, L., Veksler, O. and Boykov, Y. (2013). 'GrabCut in One Cut'. In: *2013 IEEE International Conference on Computer Vision*, pp. 1769–1776. DOI: 10.1109/ICCV.2013.222.

The CGAL Project (2024). *CGAL User and Reference Manual*. 5.6.1. CGAL Editorial Board. URL: https://doc.cgal.org/5.6.1/Manual/packages.html.

Thomas, H., Goulette, F., Deschaud, J.-E., Marcotegui, B. and LeGall, Y. (2018). 'Semantic Classification of 3D Point Clouds with Multiscale Spherical Neighborhoods'. In: *2018 International Conference on 3D Vision (3DV)*, pp. 390–398. DOI: 10.1109/3DV.2018.00052.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F. and Guibas, L. (2019). 'KPConv: Flexible and Deformable Convolution for Point Clouds'. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6410–6419. DOI: 10.1109/ICCV.2019.00651.

Tian, P., Hua, X., Tao, W. and Zhang, M. (2022). 'Robust Extraction of 3D Line Segment Features from Unorganized Building Point Clouds'. In: *Remote Sensing* 14.14. ISSN: 2072-4292. DOI: 10.3390/rs14143279.

Trimble (2000). *SketchUp*. https://www.sketchup.com/.

Tsai, Y., Shen, X., Lin, Z., Sunkavalli, K., Lu, X. and Yang, M. (2017). 'Deep Image Harmonization'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 2799–2807. DOI: 10.1109/CVPR.2017.299.

Uznir, U., Francois, A. and Alias Abdul, R. (2013). 'Unified data model of urban air pollution dispersion and 3D spatial city model: Groundwork assessment towards sustainable urban development for Malaysia'. In: *Journal of Environmental Protection* 2013. DOI: 10.4236/jep.2013.47081.

Valentin, J. C., Sengupta, S., Warrell, J., Shahrokni, A. and Torr, P. S. (2013). 'Mesh Based Semantic Modelling for Indoor and Outdoor Scenes'. In: *2013 IEEE Conference*

on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, pp. 2067–2074. DOI: 10.1109/CVPR.2013.269.

Valentin, J., Vineet, V., Cheng, M.-M., Kim, D., Shotton, J., Kohli, P., NieSSner, M., Criminisi, A., Izadi, S. and Torr, P. (2015). 'SemanticPaint: Interactive 3D Labeling and Learning at your Fingertips'. In: *ACM Trans. Graph.* 34.5. ISSN: 0730-0301. DOI: 10.1145/2751556.

Vallet, B., Brédif, M., Serna, A., Marcotegui, B. and Paparoditis, N. (2015). 'TerraMobilita/iQmulus urban point cloud analysis benchmark'. In: *Computers & Graphics* 49, pp. 126–133. ISSN: 0097-8493. DOI: 10.1016/j.cag.2015.03.004.

Vanegas, C. A., Garcia-Dorado, I., Aliaga, D. G., Benes, B. and Waddell, P. (2012). 'Inverse design of urban procedural models'. In: *ACM Trans. Graph.* 31.6. ISSN: 0730-0301. DOI: 10.1145/2366145.2366187.

Vanhoey, K., de Oliveira, C. E. P., Riemenschneider, H., Bódis-Szomorú, A., Manén, S., Paudel, D. P., Gygli, M., Kobyshev, N., Kroeger, T., Dai, D. and Van Gool, L. (2017). 'VarCity - the Video: The Struggles and Triumphs of Leveraging Fundamental Research Results in a Graphics Video Production'. In: *ACM SIGGRAPH 2017 Talks*. SIGGRAPH '17. Los Angeles, California: Association for Computing Machinery. ISBN: 9781450350082. DOI: 10.1145/3084363.3085085.

Varney, N., Asari, V. K. and Graehling, Q. (2020). 'DALES: A Large-scale Aerial LiDAR Data Set for Semantic Segmentation'. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 717–726. DOI: 10.1109/CVPRW50498.2020.00101.

Veksler, O. (2008). 'Star shape prior for graph-cut image segmentation'. In: *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III 10*. Springer, pp. 454–467. DOI: 10.1007/978-3-540-88690-7_34.

Verdie, Y., Lafarge, F. and Alliez, P. (2015). 'LOD Generation for Urban Scenes'. In: *ACM Transactions on Graphics* 34.3, pp. 1–14. DOI: 10.1145/2732527.

Verykokou, S., Doulamis, A., Athanasiou, G., Ioannidis, C. and Amditis, A. (2016). 'UAV-based 3D modelling of disaster scenes for Urban Search and Rescue'. In: *2016 IEEE International Conference on Imaging Systems and Techniques (IST)*, pp. 106–111. DOI: 10.1109/IST.2016.7738206.

VexcelImaging (2016). *Photogrammetry meets oblique: UltraCam Osprey Prime II combining two cameras in one photogrammetric housing*. https://www.vexcel-imaging.com/wp-content/uploads/2016/05/Brochure_OspreyPrimeII.pdf. Accessed: 2024-06-02.

Vijayanarasimhan, S., Ricco, S., Schmid, C., Sukthankar, R. and Fragkiadaki, K. (2017). *SfM-Net: Learning of Structure and Motion from Video*. DOI: 10.48550/arXiv.1704.07804.

Vosselman, G., Coenen, M. and Rottensteiner, F. (2017). 'Contextual segment-based classification of airborne laser scanner data'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 128, pp. 354–371. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2017.03.010.

Vu, H.-H., Labatut, P., Pons, J.-P. and Keriven, R. (2012). 'High Accuracy and Visibility-Consistent Dense Multiview Stereo'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.5, pp. 889–901. DOI: 10.1109/TPAMI.2011.172.

Wagener, N., Beckmann, J. and Eckstein, L. (2022). 'Efficient Creation of 3D-Virtual Environments for Driving Simulators'. In: *2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, pp. 1–6. DOI: 10.1109/ICECCME55909.2022.9988421.

Walk, S. (2014). *ETHZ random forest*. https://prs.igp.ethz.ch/research/Source_code_and_datasets.html. Accessed: 2020-11-25.

Wang, C., Hou, J., Miller, D., Brown, I. and Jiang, Y. (2019). 'Flood risk management in sponge cities: The role of integrated simulation and 3D visualization'. In: *International Journal of Disaster Risk Reduction* 39, p. 101139. ISSN: 2212-4209. DOI: 10.1016/j.ijdrr.2019.101139.

Wang, L., Groves, P. D. and Ziebart, M. K. (2012). 'Multi-constellation GNSS performance evaluation for urban canyons using large virtual reality city models'. In: *The Journal of Navigation* 65.3, pp. 459–476. DOI: 10.1017/S0373463312000082.

Wang, L., Groves, P. D. and Ziebart, M. K. (2013). 'GNSS shadow matching: Improving urban positioning accuracy using a 3D city model with optimized visibility scoring scheme'. In: *NAVIGATION: Journal of the Institute of Navigation* 60.3, pp. 195–207. DOI: 10.1002/navi.38.

Wang, L., Hu, H., Shang, Q., Zeng, H. and Zhu, Q. (2024). 'StructuredMesh: 3-D Structured Optimization of Façade Components on Photogrammetric Mesh Models Using Binary Integer Programming'. In: *IEEE Transactions on Geoscience and Remote Sensing* 62, pp. 1–12. DOI: 10.1109/TGRS.2023.3348492.

Wang, S., Liu, X., Zhang, Y., Li, J., Zou, S., Wu, J., Tao, C., Liu, Q. and Cai, G. (2023). 'Semantic-guided 3D building reconstruction from triangle meshes'. In: *International Journal of Applied Earth Observation and Geoinformation* 119, p. 103324. ISSN: 1569-8432. DOI: https://doi.org/10.1016/j.jag.2023.103324.

Wei, X., Zhang, Y., Li, Z., Fu, Y. and Xue, X. (2020). 'DeepSFM: Structure from Motion via Deep Bundle Adjustment'. In: *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 2328, 2020, Proceedings, Part I*. Glasgow, United Kingdom: Springer-Verlag, pp. 230–247. ISBN: 978-3-030-58451-1. DOI: 10.1007/978-3-030-58452-8_14.

Weinmann, M., Schmidt, A., Mallet, C., Hinz, S., Rottensteiner, F. and Jutzi, B. (2015). 'Contextual Classification of Point Cloud Data by Exploiting Individual 3D Neighbourhoods'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* II-3/W4, pp. 271–278. DOI: 10.5194/isprsannals-II-3-W4-271-2015.

Weinmann, M., Jutzi, B. and Mallet, C. (2013). 'Feature relevance assessment for the semantic interpretation of 3D point cloud data'. In: *ISPRS Workshop Laser Scanning 2013. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. II-5/W2*, pp. 313–318. DOI: 10.5194/isprsannals-II-5-W2-313-2013.

West, K. F., Webb, B. N., Lersch, J. R., Pothier, S., Triscari, J. M. and Iverson, A. E. (2004). 'Context-driven automated target detection in 3D data'. In: *Automatic Target*

Recognition XIV. Ed. by Sadjadi, F. A. Vol. 5426. International Society for Optics and Photonics. SPIE, pp. 133–143. DOI: 10.1117/12.542536.

Willenborg, B., Sindram, M. and Kolbe, T. H. (2018). 'Applications of 3D city models for a better understanding of the built environment'. In: *Trends in Spatial Analysis and Modelling: Decision-Support and Planning Strategies*, pp. 167–191. DOI: 10.1007/978-3-319-52522-8_9.

Wong, K. and Ellul, C. (2016). 'Using Geometry-Based Metrics as Part of Fitness-for-Purpose Evaluations of 3D City Models'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-2/W1, pp. 129–136. DOI: 10.5194/isprs-annals-IV-2-W1-129-2016.

Wu, C. (2013). 'Towards Linear-Time Incremental Structure from Motion'. In: *2013 International Conference on 3D Vision - 3DV 2013*, pp. 127–134. DOI: 10.1109/3DV.2013.25.

Wu, C., Agarwal, S., Curless, B. and Seitz, S. M. (2011). 'Multicore bundle adjustment'. In: *CVPR 2011*, pp. 3057–3064. DOI: 10.1109/CVPR.2011.5995552.

Wu, S.-C., Wald, J., Tateno, K., Navab, N. and Tombari, F. (2021). 'SceneGraphFusion: Incremental 3D Scene Graph Prediction from RGB-D Sequences'. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7511–7521. DOI: 10.1109/CVPR46437.2021.00743.

Wu, X., Jiang, L., Wang, P.-S., Liu, Z., Liu, X., Qiao, Y., Ouyang, W., He, T. and Zhao, H. (2024). 'Point Transformer V3: Simpler, Faster, Stronger'. In: *CVPR*. URL: https://github.com/Pointcept/PointTransformerV3.

Wysocki, O. (2020). 'Semantic-based Geometry Refinement of 3D City Models for Testing Automated Driving'. MA thesis. Technische Universität München. URL: https://mediatum.ub.tum.de/doc/1580077.

Wysocki, O., Hoegner, L. and Stilla, U. (2022). 'Refinement of semantic 3D building models by reconstructing underpasses from MLS point clouds'. In: *International Journal of Applied Earth Observation and Geoinformation* 111, p. 102841. ISSN: 1569-8432. DOI: 10.1016/j.jag.2022.102841.

Xia, T., Lin, J., Li, Y., Feng, J., Hui, P., Sun, F., Guo, D. and Jin, D. (2021). '3DGCN: 3-Dimensional Dynamic Graph Convolutional Network for Citywide Crowd Flow Prediction'. In: *ACM Trans. Knowl. Discov. Data* 15.6. ISSN: 1556-4681. DOI: 10.1145/3451394.

Xiang, X., Jiang, H., Yu, Y., Shen, D., Zhen, J., Bao, H., Zhou, X. and Zhang, G. (2024). 'Efficient High-Quality Vectorized Modeling of Large-Scale Scenes'. In: *International Journal of Computer Vision*. ISSN: 1573-1405. DOI: 10.1007/s11263-024-02059-x.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M. and Luo, P. (2024). 'SegFormer: simple and efficient design for semantic segmentation with transformers'. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc. ISBN: 9781713845393. DOI: 10.5555/3540261.3541185.

Xiong, B., Jancosek, M., Oude Elberink, S. and Vosselman, G. (2015). 'Flexible building primitives for 3D building modeling'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 101, pp. 275–290. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2015.01.002.

Xiong, B., Oude Elberink, S. and Vosselman, G. (2014). 'A graph edit dictionary for correcting errors in roof topology graphs reconstructed from point clouds'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 93, pp. 227–242. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2014.01.007.

Xu, N., Price, B., Cohen, S., Yang, J. and Huang, T. (2017). 'Deep GrabCut for Object Selection'. In: *Proceedings of the British Machine Vision Conference (BMVC)*. Ed. by Tae-Kyun Kim Stefanos Zafeiriou, G. B. and Mikolajczyk, K. BMVA Press, pp. 182.1–182.12. ISBN: 1-901725-60-X. DOI: 10.5244/C.31.182.

Yaagoubi, R., Yarmani, M. E., Kamel, A. and Khemiri, W. (2015). 'HybVOR: A Voronoi-Based 3D GIS Approach for Camera Surveillance Network Placement'. In: *ISPRS International Journal of Geo-Information* 4.2, pp. 754–782. ISSN: 2220-9964. DOI: 10.3390/ijgi4020754.

Yang, G., Xue, F., Zhang, Q., Xie, K., Fu, C.-W. and Huang, H. (2023). 'UrbanBIS: a Large-scale Benchmark for Fine-grained Urban Building Instance Segmentation'. In: *ACM SIGGRAPH 2023 Conference Proceedings*. SIGGRAPH '23. Los Angeles, CA, USA: Association for Computing Machinery. DOI: 10.1145/3588432.3591508.

Yang, M., Yu, K., Zhang, C., Li, Z. and Yang, K. (2018). 'DenseASPP for Semantic Segmentation in Street Scenes'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3684–3692. DOI: 10.1109/CVPR.2018.00388.

Yang, P. P.-J., Putra, S. Y. and Li, W. (2007). 'Viewsphere: a GIS-based 3D visibility analysis for urban design evaluation'. In: *Environment and Planning B: Planning and Design* 34.6, pp. 971–992. DOI: 10.1068/b32142.

Yao, Y., Luo, Z., Li, S., Fang, T. and Quan, L. (2018). 'MVSNet: Depth Inference for Unstructured Multi-view Stereo'. In: *Computer Vision – ECCV 2018*. Ed. by Ferrari, V., Hebert, M., Sminchisescu, C. and Weiss, Y. Cham: Springer International Publishing, pp. 785–801. ISBN: 978-3-030-01237-3. DOI: 10.1007/978-3-030-01237-3_47.

Yasumoto, S., Jones, A., Yano, K. and Nakaya, T. (2012). 'Virtual city models for assessing environmental equity of access to sunlight: a case study of Kyoto, Japan'. In: *International Journal of Geographical Information Science* 26.1, pp. 1–13. DOI: 10.1080/13658816.2011.570268.

Ye, Z., Xu, Y., Huang, R., Tong, X., Li, X., Liu, X., Luan, K., Hoegner, L. and Stilla, U. (2020). 'LASDU: A Large-Scale Aerial LiDAR Dataset for Semantic Labeling in Dense Urban Areas'. In: *ISPRS International Journal of Geo-Information* 9.7. ISSN: 2220-9964. DOI: 10.3390/ijgi9070450.

Yichuan, D., Jack, C. P. C. and Chimay, A. (2016). 'A framework for 3D traffic noise mapping using data from BIM and GIS integration'. In: *Structure and Infrastructure Engineering* 12.10, pp. 1267–1280. DOI: 10.1080/15732479.2015.1110603.

Yoo, S., Jeong, Y., Jameela, M. and Sohn, G. (2023). 'Human Vision Based 3D Point Cloud Semantic Segmentation of Large-Scale Outdoor Scenes'. In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE Computer Society, pp. 6577–6586. DOI: 10.1109/CVPRW59228.2023.00699.

Yue, Y., Mahadevan, S., Schult, J., Engelmann, F., Leibe, B., Schindler, K. and Kontogianni, T. (2024). 'AGILE3D: Attention Guided Interactive Multi-object 3D Segmentation'. In: *International Conference on Learning Representations (ICLR)*. URL: https://ywyue.github.io/AGILE3D/.

Zhang, C., Fan, H. and Kong, G. (2021). 'VGI3D: an interactive and low-cost solution for 3D building modelling from street-level VGI images'. In: *Journal of Geovisualization and Spatial Analysis* 5.2, p. 18. DOI: 10.1007/s41651-021-00086-7.

Zhang, L., Han, C., Zhang, L., Zhang, X. and Li, J. (2014). 'Web-based visualization of large 3D urban building models'. In: *International Journal of Digital Earth* 7.1, pp. 53–67. DOI: 10.1080/17538947.2012.667159.

Zhang, M., Li, T., Yu, Y., Li, Y., Hui, P. and Zheng, Y. (2022). 'Urban Anomaly Analytics: Description, Detection, and Prediction'. In: *IEEE Transactions on Big Data* 8.3, pp. 809–826. DOI: 10.1109/TBDATA.2020.2991008.

Zhang, Q. and Chan, A. B. (2020). '3d crowd counting via multi-view fusion with 3d gaussian kernels'. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 07, pp. 12837–12844. DOI: 10.1609/aaai.v34i07.6980.

Zhang, W., Li, Z. and Shan, J. (2021). 'Optimal Model Fitting for Building Reconstruction From Point Clouds'. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 14, pp. 9636–9650. DOI: 10.1109/JSTARS.2021.3110429.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A. and Torralba, A. (2016). 'Learning Deep Features for Discriminative Localization'. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 2921–2929. DOI: 10.1109/CVPR.2016.319.

Zhou, J., Long, C., Xie, Y., Wang, J., Li, B., Wang, H., Chen, Z. and Dong, Z. (2024). *VEnvision3D: A Synthetic Perception Dataset for 3D Multi-Task Model Research*. URL: https://arxiv.org/abs/2402.19059.

Zhou, Q.-Y. and Neumann, U. (2010). '2.5D Dual Contouring: A Robust Approach to Creating Building Models from Aerial LiDAR Point Clouds'. In: *Computer Vision – ECCV 2010*. Ed. by Daniilidis, K., Maragos, P. and Paragios, N. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 115–128. ISBN: 978-3-642-15558-1. DOI: 10.1007/978-3-642-15558-1_9.

Zhou, Q., Grinspun, E., Zorin, D. and Jacobson, A. (2016). 'Mesh arrangements for solid geometry'. In: *ACM Trans. Graph.* 35.4. ISSN: 0730-0301. DOI: 10.1145/2897824.2925901.

Zhou, Y., Shen, S. and Hu, Z. (2018). 'Fine-Level Semantic Labeling of Large-Scale 3D Model by Active Learning'. In: *2018 International Conference on 3D Vision (3DV)*, pp. 523–532. DOI: 10.1109/3DV.2018.00066.

Zhu, L., Shen, S., Gao, X. and Hu, Z. (2018). 'Large scale urban scene modeling from MVS meshes'. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 614–629. DOI: 10.1007/978-3-030-01252-6_38.

Zhu, L., Shen, S., Hu, L. and Hu, Z. (2017). 'Variational Building Modeling from Urban MVS Meshes'. In: *2017 International Conference on 3D Vision (3DV)*, pp. 318–326. DOI: 10.1109/3DV.2017.00044.

Zimmer, W., Rangesh, A. and Trivedi, M. (2019). '3D BAT: A Semi-Automatic, Web-based 3D Annotation Toolbox for Full-Surround, Multi-Modal Data Streams'. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. Paris, France: IEEE Press, pp. 1816–1821. DOI: 10.1109/IVS.2019.8814071.

Zolanvari, S., Ruano, S., Rana, A., Cummins, A., da Silva, R. E., Rahbar, M. and Smolic, A. (2019). 'DublinCity: Annotated LiDAR Point Cloud and its Applications'. In: *BMVC*

30th British Machine Vision Conference. URL: https://api.semanticscholar.org/CorpusID:202540484.

Summary

The thesis explores the semantic understanding of urban textured meshes derived from photogrammetric methods. It primarily addresses three aspects with regard to urban textured meshes: 1) semantic annotation and the creation of benchmark datasets, 2) semantic segmentation, and 3) the automation of lightweight 3D city modeling using semantic information.

The first focus of the thesis is the development of a benchmark dataset to evaluate the performance of advanced 3D semantic segmentation methods in urban settings. An interactive 3D annotation framework has been proposed to assign ground truth labels to the urban meshes' triangle faces and texture pixels. This framework achieves efficient and accurate semi-automatic annotation through segment classification and structure-aware interactive selection. In the center of Helsinki, Finland, object-level annotations were made over approximately 4 km^2 (including buildings, vegetation, and vehicles, etc.), and part-level annotations over about 2.5 km^2 (including building parts like doors, windows, and road markings, etc.). The design of the annotation tools improves user operation and enables quick annotation of large scenes, while the resulting datasets allow researchers to refine their deep learning models for urban analysis.

Another research focus is on mesh segmentation algorithms. A novel semantic mesh segmentation algorithm has been introduced for large-scale urban environments, employing plane-sensitive over-segmentation combined with graph-based methods for contextual data integration. This approach, which utilizes graph convolutional networks for classification, significantly improves performance over traditional techniques based on our proposed benchmark datasets.

Finally, leveraging this semantic information, a pipeline for reconstructing lightweight 3D city models has been designed. This facilitates the automated reconstruction of CityGML-based LoD2 and LoD3 city models, ensuring high fidelity in geometric detail and semantic accuracy. The reconstructed large-scale, lightweight, and semantic city models significantly broaden applications in urban spatial intelligence, including automatic geometric measurements, interactive spatial computations, spatial analysis based on external data, and environment simulation using physical engines.

This thesis enhances the practicality of 3D data in real-world applications by utilizing semantic parsing of urban textured meshes to generate lightweight 3D urban semantic models, greatly enriching their usability. It also lays a solid foundation for future progress in understanding, modeling, and analyzing 3D urban scenes.

Samenvatting

Dit proefschrift onderzoekt het semantisch begrip van stedelijke getextureerde meshes die zijn afgeleid van fotogrammetrische methoden. Het richt zich voornamelijk op drie aspecten met betrekking tot stedelijke getextureerde meshes: 1) semantische annotatie en het creëren van benchmark datasets, 2) semantische segmentatie, en 3) de automatisering van compacte 3D-stadsmodellering met gebruik van semantische informatie.

De eerste focus van de thesis is de ontwikkeling van een benchmarkdataset om de prestaties van geavanceerde 3D semantische segmentatiemethoden in stedelijke omgevingen te evalueren. Er wordt een interactieve 3D-annotatiemethodiek voorgesteld om grondwaarheidslabels toe te wijzen aan de driehoekige vlakken en textuurpixels van de stedelijke meshes. Deze methodiek bereikt efficiënte en nauwkeurige semi-automatische annotatie door middel van segmentclassificatie en structuurbewuste interactieve selectie. Voor het centrum van Helsinki, Finland, zijn objectniveau-annotaties gemaakt over ongeveer 4 km^2 (inclusief gebouwen, vegetatie en voertuigen, enz.), en deel-niveau-annotaties over ongeveer 2,5 km^2 (inclusief bouwdelen zoals deuren, ramen en wegmarkeringen, enz.). Het ontwerp van de annotatiehulpmiddelen vergemakkelijkt de bediening door gebruikers, maakt uitgebreide stedelijke scène-annotaties mogelijk, en is daarmee essentieel voor het verfijnen van deep learning modellen gericht op stedelijke analyse.

De tweede onderzoeksfocus ligt op meshsegmentatie-algoritmen. Een nieuw semantisch mesh segmentatie-algoritme wordt geïntroduceerd voor grootschalige stedelijke omgevingen, gebruikmakend van vlakgevoelige over-segmentatie gecombineerd met op grafen gebaseerde methoden voor contextuele data-integratie. Deze aanpak, die gebruik maakt van graafconvolutienetwerken voor classificatie, verbetert de prestaties aanzienlijk ten opzichte van traditionele technieken, gebaseerd op onze voorgestelde benchmark datasets.

Tot slot is, door gebruik te maken van deze semantische informatie, een pijplijn ontworpen voor het reconstrueren van compacte 3D-stadsmodellen. Dit vergemakkelijkt de geautomatiseerde reconstructie van CityGML-gebaseerde LoD2 en LoD3 stadsmodellen, waarbij een hoge getrouwheid in geometrische details en semantische nauwkeurigheid wordt gewaarborgd. De gereconstrueerde grootschalige, lichtgewicht en semantische stadsmodellen verbreden de toepassingen in stedelijke ruimtelijke intelligentie aanzienlijk, inclusief automatische geometrische metingen, interactieve ruimtelijke berekeningen, ruimtelijke analyses op basis van externe

gegevens, en omgevingssimulaties met behulp van fysieke motoren.

Deze thesis verbetert de praktische bruikbaarheid van 3D-gegevens in toepassingen op de fysieke wereld door semantische verwerking van stedelijke getextureerde meshes in te zetten om compacte 3D stedelijke semantische modellen te genereren, waardoor hun bruikbaarheid en relevantie aanzienlijk toeneemt. Het legt ook een solide basis voor toekomstige vooruitgang in het begrijpen, modelleren en analyseren van stedelijke 3D scènes.

Curriculum vitae

Driven by a passion for urban reality mapping and scene understanding, Weixiao Gao pursued a rigorous academic journey in Geoinformation Science. He majored in Geographic Information Systems during his undergraduate and first Master's degree. In 2015, he obtained a Master's degree in Geographic Information Science from The Chinese University of Hong Kong. Subsequently, seeking to deepen his expertise, he pursued another Master's degree in Geodesy and Geographic Information Science at Technische Universität Berlin in Germany. He specialized in three areas, including photogrammetry and remote sensing, geographic information science, and spatial geodesy and navigation, graduating successfully in 2018.

After obtaining two Master's degrees, Weixiao joined the 3D Geoinformation research group at the Delft University of Technology to commence his doctoral studies in collaboration with the company Cyclomedia. During his PhD research, he successfully tackled numerous challenges associated with the interpretation of urban textured meshes, including the lack of benchmark datasets, annotation tools, and supervised semantic segmentation methods. He established a reconstruction pipeline from urban textured meshes to lightweight semantic 3D city models. In addition, he published several academic papers and developed multiple open-source software projects.

List of publications

- Gao, W., Peters, R. and Stoter, J. (2024b). 'Building-PCC: Building Point Cloud Completion Benchmarks'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* X-4/W5-2024, pp. 179–186. DOI: 10.5194/isprs-annals-X-4-W5-2024-179-2024

- Gao, W., Peters, R., Ledoux, H. and Stoter, J. (2024a). 'Filling holes in LoD2 building models'. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* X-4/W5-2024, pp. 171–177. DOI: 10.5194/isprs-annals-X-4-W5-2024-171-2024 (**Best paper award**)

- Gao, W., Peters, R. and Stoter, J. (2024c). 'Unsupervised Roofline Extraction from True Orthophotos for LoD2 Building Model Reconstruction'. In: *Recent Advances in 3D Geoinformation Science*. Ed. by Kolbe, T. H., Donaubauer, A. and Beil, C. Cham: Springer Nature Switzerland, pp. 425–436. ISBN: 978-3-031-43699-4. DOI: 10.1007/978-3-031-43699-4_27

- Peters, R., Dukai, B., Gao, W. and Stoter, J. (2023). '3D BAG-Geactualiseerd op basis van AHN4'. In: *Geo-Info* 2023.2, pp. 30–35. URL: https://repository.tudelft.nl/record/uuid:daf27e62-9a93-4b92-8646-c7cd16f572ce

- Gao, W., Nan, L., Boom, B. and Ledoux, H. (2023). 'PSSNet: Planarity-sensible Semantic Segmentation of large-scale urban meshes'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 196, pp. 32–44. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2022.12.020

- Gao, W., Nan, L., Boom, B. and Ledoux, H. (2021). 'SUM: A benchmark dataset of Semantic Urban Meshes'. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 179, pp. 108–120. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2021.07.008

- Chen, Y., Gao, W., Widyaningrum, E., Zheng, M. and Zhou, K. (2018). 'Building Classification of VHR Airborne Stereo Images Using Fully Convolutional Networks and Free Training Samples'. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-4, pp. 87–92. DOI: 10.5194/isprs-archives-XLII-4-87-2018 (**Best poster award**)

Acknowledgements

As I approach the completion of my PhD thesis, my heart is brimming with profound gratitude. This significant academic journey has been marked indelibly by the support and assistance of many, each of whom has left a lasting imprint on me. At this moment, I wish to express my deepest appreciation to all who have been part of this transformative experience.

First and foremost, I extend my deepest respect and gratitude to my supervisors, Hugo Ledoux and Liangliang Nan. Throughout my PhD studies, you have not only provided meticulous academic guidance but also stood by me during challenging times with unwavering support and encouragement. Your passion for scholarship and steadfast commitment have deeply inspired me, continually driving me to pursue academic excellence. Additionally, I owe a special thanks to the leader of our research group, Jantien Stoter, whose thoughtful guidance in both academic and personal matters significantly eased the pressures I faced. I also extend my sincere appreciation to the members of the defense committee, Florent Lafarge, Sander Oude Elberink, Norbert Haala, Steffen Nijhuis, and Georg Vrachliotis, who meticulously reviewed my thesis and provided invaluable feedback. To our dedicated secretaries, Martine de Jong-Lansbergen and Margo van der Helm, and our diligent office manager, Nadine Hobeika, your efficiency and thoroughness have enabled us to dedicate ourselves fully to our research endeavors.

I must also express my gratitude to Jinjin Yan for your warm reception when I first arrived in the Netherlands, and to Ravi Peters for your continuous support during my PhD research and for the opportunities to join the 3D BAG project. I am grateful to Giorgio Agugiaro for enabling me to venture into new realms of postdoctoral research. Your support has been pivotal to my professional growth. I am equally thankful to my colleagues Anna Labetski, Stelios Vitalis, Kavisha, Tom Commandeur, Teng Wu, Balázs Dukai, Ken Arroyo Ohori, Francesca Noardo, Abdoulaye Diakité, and Ziqian Ni for your support in my doctoral journey and the enriching social interactions that provided much-needed relaxation and rejuvenation.

A heartfelt thank you goes to my juniors Shenglan Du, Huang Jin, Zexin Yang, Nail Ibrahimli, Camilo León-Sánchez, Ivan Pađen, Zhaiyu Chen, and Xiaoxin Mi. Conversations and camaraderie in our coffee corner not only lightened the burdens of research but also sparked creative insights. The gatherings during the pandemic provided both comfort and strength during uncertain times.

I also want to express my gratitude to all my other colleagues in the 3D Geoinformation group. Working and interacting with you in such a relaxed and enjoyable work atmosphere has been truly comforting.

I am grateful to the company Cyclomedia Technology Inc. and my mentors, Arjen Swart, Bas Boom, Thijs van Lankveld, and Bart Beers, for the practical support that has enriched my academic perspective and deepened my understanding of both theoretical and practical applications.

Thank you to my beloved partner, Yunzhou Han, for your meticulous care, your silent companionship during my moments of despair, and your understanding during my busiest times when I couldn't be by your side. We have shared countless joyful moments together, savoring delicious food and experiencing beautiful landscapes. Your understanding and love are the driving forces behind my progress, and I am deeply grateful for your unwavering support and companionship. I also want to thank my daughter, Ruoxi Gao, your every smile is my greatest motivation. To my family, particularly my parents, Jungui Gao, and Xiangyu Wang, and my in-laws Wenbin Han and Weiting Liu, your unconditional love and steadfast support have been my backbone.

Furthermore, I thank my friends Ling Chang, Wufan Zhao, Lanye Zhou, Chengbo Yang, Shizhe Zhang, Kaili Mao, Sitong Luo, Kaixuan Zhou, and Mingxue Zheng for every moment of care and interaction, each of which has propelled me forward.

I will eternally treasure the memories created with all of you during this phase of my life, which has become an invaluable part of my journey. Wherever the future takes me, I am committed to perpetuating the spirit of support and assistance, helping others who dare to dream big.

Once again, a heartfelt thank you to everyone who has been a part of my journey.

With deepest gratitude,

Weixiao Gao
Delft, 31st August 2024

ISBN 978-94-6366-942-9