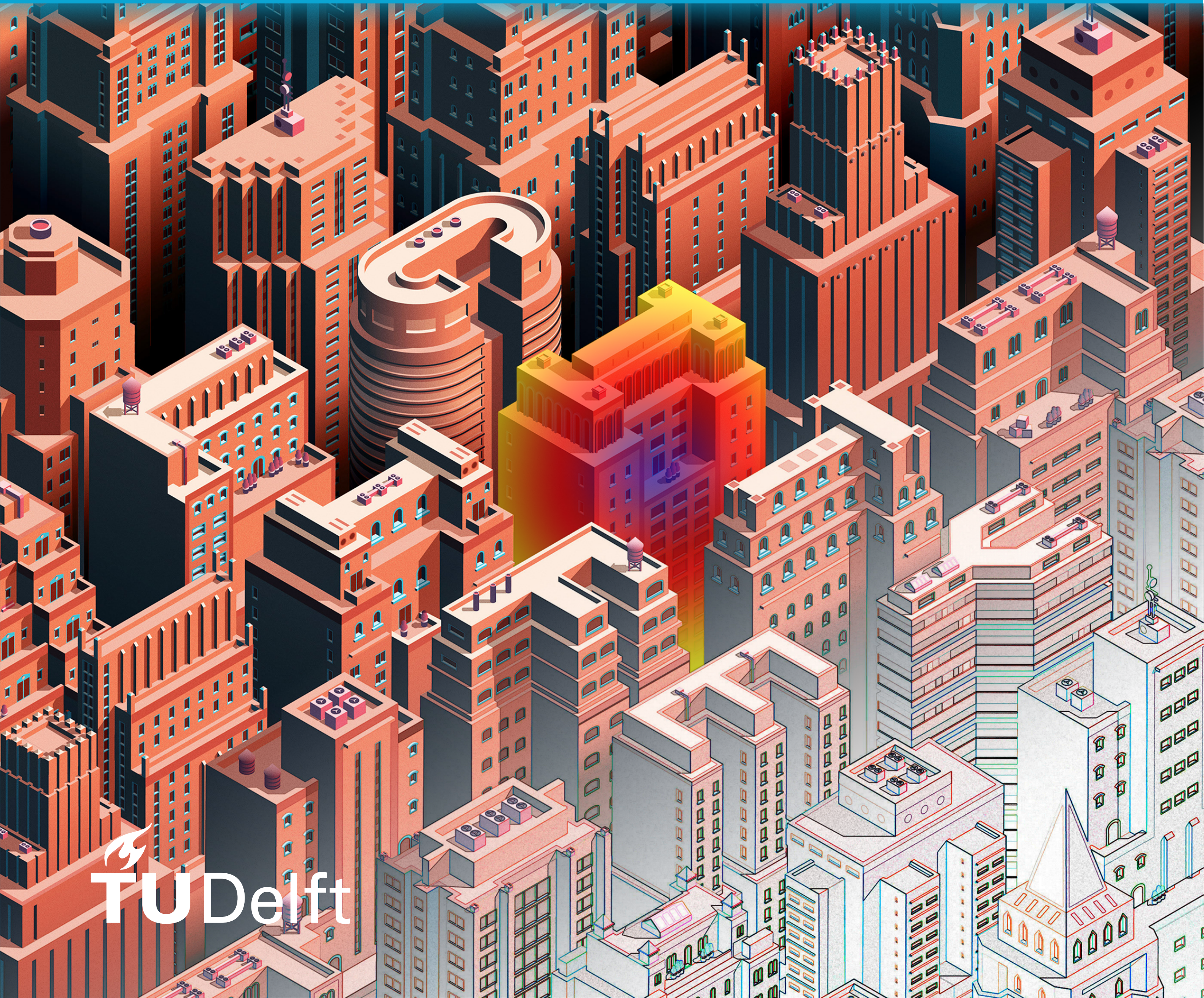
MSc thesis in Geomatics

Learning to Reconstruct Compact Building Models from Point Clouds

Zhaiyu Chen

2021



MSc thesis in Geomatics

Learning to Reconstruct Compact Building Models from Point Clouds

Zhaiyu Chen

June 2021

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of Master of Science in Geomatics

Zhaiyu Chen: *Learning to Reconstruct Compact Building Models from Point Clouds* (2021)

The work in this thesis was carried out in the:

3D geoinformation group
Delft University of Technology

Supervisors:	Dr. Liangliang Nan
	Dr. Seyran Khademi
Co-reader:	Dr. Hugo Ledoux

Abstract

Three-dimensional building models play a pivotal role in shaping the digital twin of our world. With the advance of sensing technologies, unprecedented data acquisition capabilities on capturing the built environment have surfaced, with photogrammetry and light detection and ranging being the two important sources, both of which can acquire point clouds of buildings. A point cloud is anisotropically distributed in space, which—though conveying spatial information itself—has to be converted into a surface model for a wider spectrum of usage. This conversion is often referred to as reconstruction. Despite the enhanced availability of point cloud data in the built environment, how to reconstruct high-quality building surface models remains non-trivial in remote sensing, computer vision, and computer graphics. Most reconstruction methods are dedicated to smooth surfaces represented by dense triangles, irrespective of the piecewise planarity that dominates the geometry of real-world buildings. Although some works claim the possibility of reconstructing piecewise-planar shapes from point clouds, they either struggle to comply with specific geometric constraints, or suffer from serious scalability issues. There is no versatile solution yet for building reconstruction.

In this thesis, we propose a novel framework for reconstructing compact, watertight, polygonal building models from point clouds. Our approach comprises three functional blocks: *(a)* a cell complex is generated via adaptive space partitioning that provides a polyhedral embedding as the candidate set; *(b)* an implicit field is learnt by a deep neural network that facilitates building occupancy estimation; *(c)* a Markov random field is formulated for surface extraction via combinatorial optimisation, where an efficient graph-cut solver is applied. We extensively evaluate the proposed method in comparison with state-of-the-art methods in shape reconstruction, surface approximation and geometry simplification. Experimental results reveal that, with our neural-guided strategy, high-quality building models can be obtained with significant advantages over fidelity, compactness and computational efficiency. The method shows robustness to noise and insufficient measurements due to occlusions, and generalise reasonably well from synthetic scans to real-world measurements. Moreover, our method remains generic to not only buildings, but any piecewise-planar objects.

Acknowledgements

My sincere gratitude goes to **Liangliang**, for his continuous support during the development of this thesis both scientifically and personally. I am very grateful to **Seyran** for her constructive feedback on the experiments as well as suggestions on presenting this work. I am also thankful to **Hugo** for his input to improving this thesis.

Special thanks go to **Linfu** for providing the *Shenzhen* dataset used in this thesis, and the concerning **open-source communities** that happen to ladder the research in this thesis, which would not be available anytime soon if I had to invent those wheels by myself.

I thank my **parents** for their unconditional support across six time zones, and my **friends** who have accompanied me during this journey, especially to **Qian**, whose cheers prevent me from throwing away my computer a hundred times when trapped by bugs, and whose fantastic Chinese cuisine saves me from overdosing Dutch bread.

Last but not least, I thank the **three-dimensional space** that we human beings possibly misperceive ourselves reside due to our limited cognition. This however sets the foundation of the research carried out in this thesis.

Contents

1 Introduction **1**

1.1 Background and motivation 1

1.2 Inspiration and our approach 4

1.3 Research objective and scope 6

1.4 Structure of this thesis 7

2 Related work **9**

2.1 Shape representation 9

2.2 Shape reconstruction from imagery 10

2.3 Shape reconstruction from point clouds 12

2.3.1 Primitive assembly 12

2.3.2 Surface approximation 14

2.3.3 Geometry simplification 14

2.3.4 Implicit field-based reconstruction 16

3 Methodology **19**

3.1 Overview 19

3.2 Adaptive binary space partitioning 21

3.2.1 Primitive detection 21

3.2.2 Primitive refinement 22

3.2.3 Construction of cell complex 23

3.3 Occupancy learning in function space 24

3.3.1 Signed distance function 25

3.3.2 Neural network architecture 26

3.3.3 Training with loss function 27

3.3.4 Signed distance voting 29

3.4 Surface reconstruction 29

3.4.1 Energy formulation 31

3.4.2 Surface extraction 32

4 Implementation details **35**

4.1 Datasets 35

4.2 Evaluation metrics 38

4.3 Numerical robustness 39

4.4 Libraries and software used within this thesis 40

5 Results and discussion **41**

5.1 Reconstruction results 42

5.2 Evaluation 44

5.2.1 Fidelity and complexity 44

5.2.2 Computational efficiency 50

5.2.3 Robustness to noise and incomplete input 52

5.2.4 Effect of parameter λ 52

5.3 Limitations . . . 52
5.4 Application scope . . . 54

6 Conclusions and future work 57
6.1 Research overview . . . 57
6.2 Contributions . . . 59
6.3 Future work . . . 59

A Reproducibility self-assessment 67
A.1 Marks for each of the criteria . . . 67
A.2 Self-reflection . . . 68

B Alternative cell complex formulation 69

C Alternative deep implicit field formulations 71

List of Figures

1.1 Applications of 3D city models 1
1.2 Acquisition of point clouds 2
1.3 Surface mesh of an urban scene 2
1.4 Examples of real-world point clouds 3
1.5 Examples of non-manifold topological embeddings 3
1.6 Boundary representation 4
1.7 Examples of piecewise-planar structures in the built environment 4
1.8 Facets that compose a surface: dense triangles, sparse triangles and polygons 4
1.9 Explicit versus implicit representation 5
1.10 Deep implicit field 5
1.11 Goal of our approach 6

2.1 Shape representations 9
2.2 A building represented in various levels of detail 10
2.3 Reconstruction building mass models from images of unmanned aerial vehicles 11
2.4 Automatic single-view 3D reconstruction of urban scenes 11
2.5 Single-view reconstruction with BSP-Net 12
2.6 Primitives detection with random sample consensus 13
2.7 Polygonal surface reconstruction from point clouds 13
2.8 Illustration of Poisson reconstruction 14
2.9 Edge contraction with quadric error metrics 14
2.10 Structure-aware mesh decimation 15
2.11 Variational shape approximation 15
2.12 Manhattan-world urban reconstruction 15
2.13 Building modelling with 2.5D Dual Contouring 16
2.14 LoD generation for urban scenes 16
2.15 Marching Cubes 16
2.16 Two DeepSDF instantiations 17
2.17 BSP-tree for learning convex decomposition 17

3.1 Overview of our approach 19
3.2 Intermediate formulation of our approach 20
3.3 Planar primitive detection from point clouds 21
3.4 Planar primitive refinement 23
3.5 Binary space partitioning 23
3.6 Exhaustive partitioning versus adaptive partitioning 24
3.7 Priority impacts the cell complex construction 25
3.8 Cell adjacency information is obtained while partitioning 26
3.9 Signed distance field 27
3.10 Points2Surf neural network architecture 27
3.11 Process of approximating a surface with the neural network 28
3.12 A series of 2D cross-sections of the signed distance field of a building 28
3.13 Signed distance estimation for each candidate polyhedron 29

3.14 Signed distance voting 29
3.15 Two labelling examples 30
3.16 An example of non-manifold buildings 30
3.17 Complexity term penalises zigzagging artefact 31
3.18 Cell complex interpreted as graph 32
3.19 Graph cuts as hypersurfaces 32
3.20 Different topological properties for separating source and terminal with hypersurfaces 33
3.21 Surface extraction from a labelled cell complex 33
3.22 Consistent orientation of building's surface 34

4.1 Simulation with Blensor 35
4.2 Helsinki dataset examples 36
4.3 *Helsinki full-view* versus *Helsinki no-bottom* 37
4.4 Shenzhen dataset examples 37
4.5 Exact representation by rational numbers 39
4.6 Example of degenerate geometry due to floating-point arithmetic 39

5.1 Reconstructions from *Helsinki full-view* point clouds 41
5.2 Reconstructions from *Helsinki no-bottom* point clouds 43
5.3 Complete reconstruction from insufficient scans of *Shenzhen* data 43
5.4 Reconstructions from *Shenzhen* point clouds 44
5.5 Comparison with smooth surface reconstruction 46
5.6 Comparison with piecewise-planar reconstruction methods 47
5.7 Comparison with PolyFit 47
5.8 Two-cube non-manifold example 48
5.9 Comparison with surface approximation methods 48
5.10 Adaptive partitioning versus exhaustive partitioning 49
5.11 Distribution of execution time 50
5.12 Efficiency of adaptive partitioning 50
5.13 Scalability comparison with PolyFit 51
5.14 Complex building reconstruction 51
5.15 Robustness to noise 52
5.16 Impact of parameter λ 53
5.18 An example of non-planar buildings and how it is approximated 53
5.19 Quality of planar primitives impacts reconstruction 54
5.20 Formulation of the 'caved' artefact 54
5.21 Application in building component analysis 55
5.22 Application in generic shape reconstruction 55

A.1 Reproducibility criteria to be assessed. 67

B.1 Cell complex constructed with 3D Delaunay triangulation 69
B.2 Surface extraction from a cell complex generated via hyperplane arrangement 69
B.3 Surface extraction from a cell complex generated via 3D Delaunay triangulation 69

C.1 Network architecture of implicit shape indicator 71
C.2 Adapted BSP-Net for shape reconstruction from point clouds 72
C.3 Building reconstruction from AHN point clouds 73
C.4 Reconstruction from ShapeNet point clouds 73

List of Tables

2.1 Characteristics overview of related work . . . 18

3.1 Primitive priority . . . 22

4.1 Datasets overview . . . 38
4.2 Evaluation metrics . . . 38

5.1 Error analysis on *Helsinki full-view* data . . . 42
5.2 Error analysis on *Shenzhen* data . . . 45
5.3 Evaluation of fidelity and complexity . . . 46

A.1 Reproducibility evaluation . . . 67

List of Algorithms

3.1 Refinement of planar primitives 22

Acronyms

LiDAR	light detection and ranging	1
B-rep	boundary representation	2
LoD	level of detail	3
SDF	signed distance function	4
MRF	Markov random field	5
MVS	multi-view stereo	9
SfM	structure-from-motion	10
UAVs	unmanned aerial vehicles	10
DSMs	digital surface models	10
VHR	very high-resolution	10
CNN	convolutional neural network	11
RANSAC	random sample consensus	12
QEM	quadric error metrics	14
SAMD	structure-aware mesh decimation	14
VSA	variational shape approximation	14
CSG	constructed solid geometry	17
AABB	axis-aligned bounding box	21
PCA	principal component analysis	22
BSP	Binary space partitioning	23
MLP	multi-layer perceptron	27
SMH	symmetric mean Hausdorff	38
TLS	terrestrial laser scanner	43
GUI	graphical user interface	45
BIM	building information modelling	54
RNN	recurrent neural network	59
GIS	geographic information system	60

1 Introduction

This chapter leads in with the background and motivation for building reconstruction in Section 1.1. In Section 1.2, we retrace the inspiration from which we develop our approach. The main research questions and scope are framed in Section 1.3. Finally, Section 1.4 outlines the structure of the thesis with an overview.

1.1 Background and motivation

Three-dimensional (3D) building models play a pivotal role in shaping the digital twin of our world, which, in turn, are facilitating various intelligent applications in urban planning [Herbert and Chen, 2015], solar potential analysis [Machete et al., 2018], environmental simulation [Stoter et al., 2020], etc (see Figure 1.1). Recently, with the development of augmented and virtual reality applications, the demand for high-quality building modelling is growing rapidly [Blut and Blankenbach, 2021].

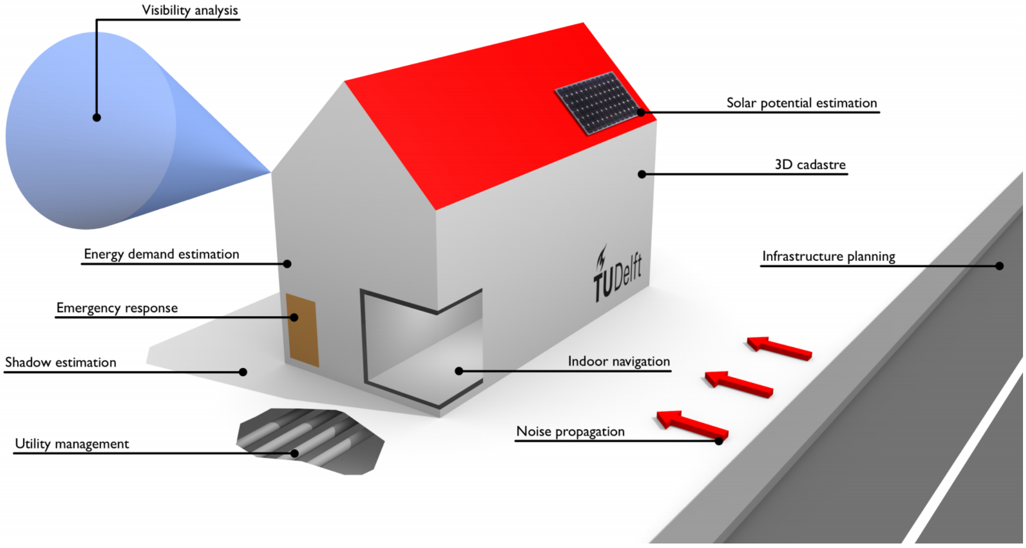
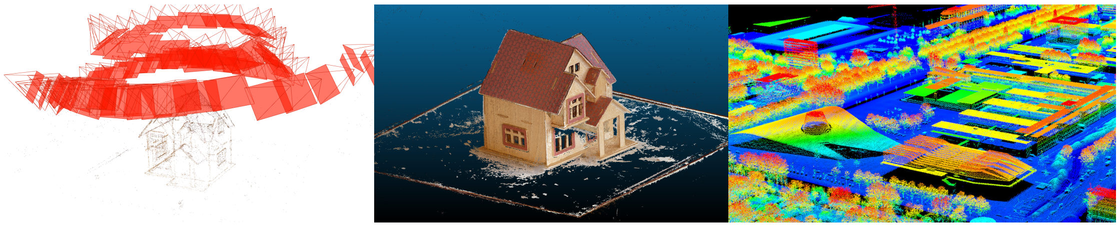


Figure 1.1: Applications of 3D city models [Biljecki et al., 2015]

The advance of sensing technologies has empowered unprecedented data acquisition capabilities on capturing the built environment, with photogrammetry and light detection and ranging (LiDAR) being the two important sources, both of which can acquire point clouds of buildings, as shown in Figure 1.2. A point cloud is anisotropically distributed in space, which—though conveying spatial information itself—has to be converted into a surface mesh for a wider spectrum of usage. The latter consists of facets, which themselves are represented by edges and vertices, as illustrated in Figure 1.3. This conversion is often referred to as *reconstruction*.

Despite the enhanced availability of point cloud data in the built environment, how to reconstruct high-quality building surface models remains non-trivial concerning the current state of automation in this field. Specifically, the reconstruction problem is conceived challenging in terms of two factors summarised as follows:



(a) Multi-view images (b) Photogrammetric point cloud (c) LiDAR point cloud[1]

Figure 1.2: Acquisition of point clouds. A photogrammetric point cloud (b) can be obtained from multi-view images (a). LiDAR can directly acquire a point cloud (c).

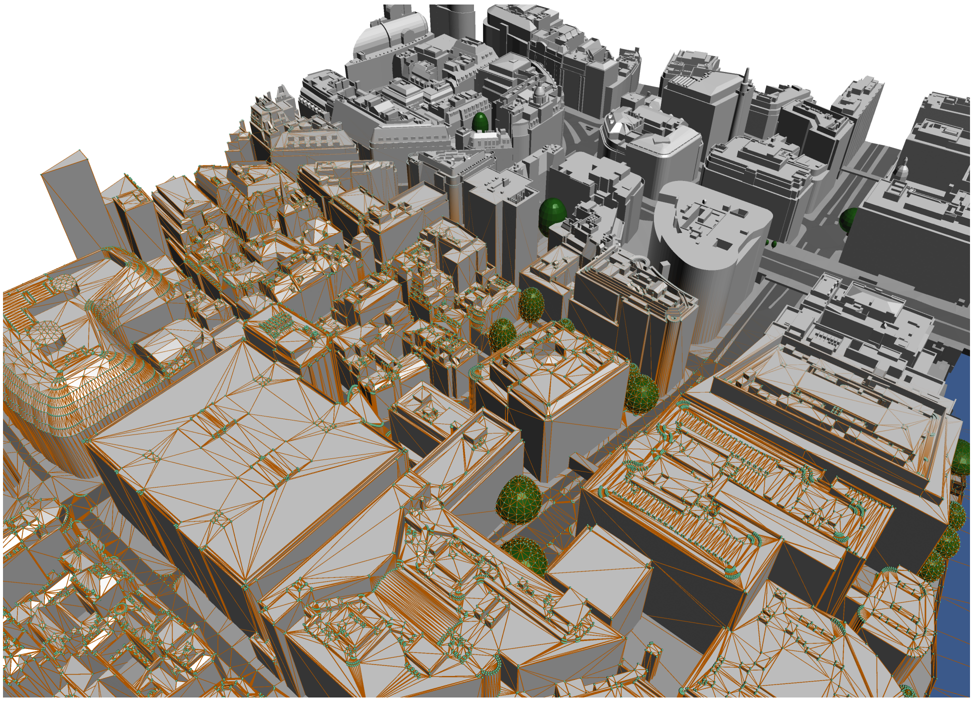


Figure 1.3: Surface mesh of an urban scene. A surface mesh consists of facets, which themselves consist of edges (coloured brown) and vertices (coloured green).

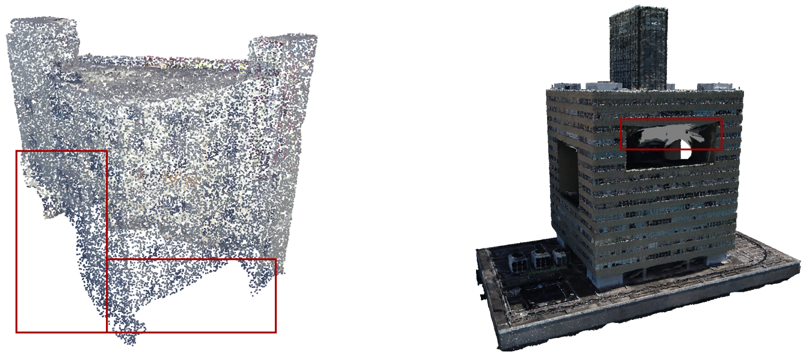
- **Geometric information** is inevitably degraded or even lost during data acquisition. For example, noises are always concomitant with measurements; occlusion, low-contrast or non-reflective surfaces all possibly result in unreliability in the data. Figure 1.4 shows examples of real-world point clouds with geometric defects.

- **Topological constraints** need to be satisfied for the reconstructed building's geometry. Figure 1.5 presents possible invalid topological embeddings that defect a 3D building model. As required is a boundary representation (B-rep) that expresses a geometry of dimension n with its boundary of dimension $n - 1$ (see Figure 1.6), how to guarantee the reconstructed building surface is *watertight* is of particular importance.

To address this long-standing problem in remote sensing, computer vision and graphics, various data- or model-driven approaches or their combination have been proposed. However, there is no versatile solution yet to the fully automatic reconstruction. Existing methods either fail to incorporate high-level shape information, or struggle to comply with certain geometric constraints effectively and efficiently.

Human-created structures follow certain patterns, one of which—distinct from the natural ones—is *piecewise pla-*

[1] https://www.tudelft.nl/bk/onderzoek/projecten/geoinformation-technology-governance



(a) (b)

Figure 1.4: Examples of real-world point clouds from incomplete (a) and occluded (b) scans

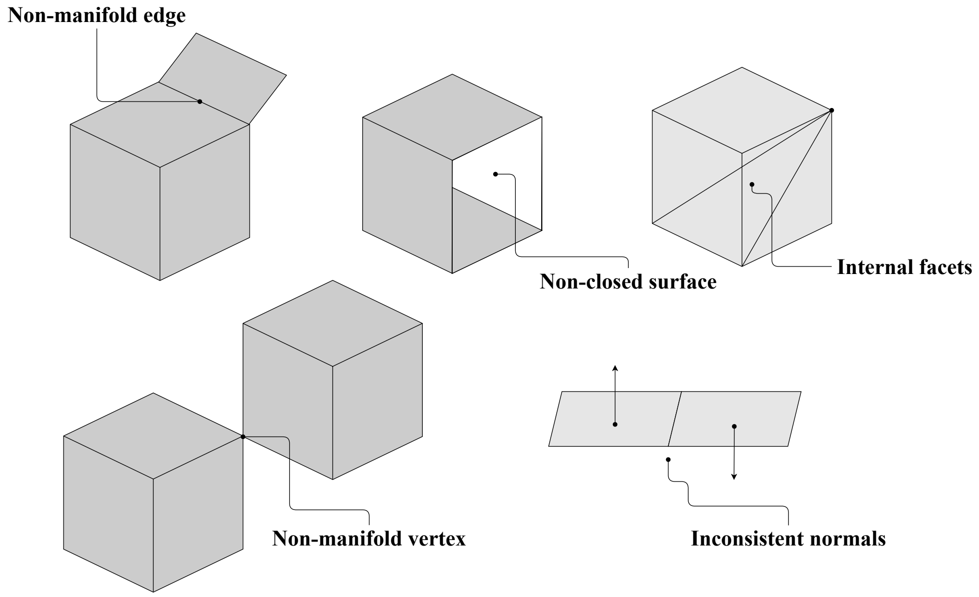


Figure 1.5: Examples of non-manifold topological embeddings

narity. Piecewise-planar surfaces are ubiquitous in the built environment (see Figure 1.7). Being an efficient representation for computing, storing and rendering, it describes a geometry with non-uniformity to represent large flat regions and sharp features with sparse sets of parameters. Figure 1.8 illustrates a building described as a smooth surface and piecewise-planar ones, where an arbitrary-sided polygonal surface exhibits the highest compactness with respect to the number of facets that compose the surface. This non-uniformity and irregularity in the polygonal form, however, hinder its creation from remote sensing measurement as significant abstraction is to impose and the problem is *de-facto* ill-posed.

Most reconstruction methods are dedicated to smooth surfaces represented as dense triangles (see Figure 1.8a), irrespective of piecewise planarity that exhibits in the built environment. Simplification is therefore required as a follow-up procedure to convert the smooth surface into a compact one [Garland and Heckbert, 1997; Cohen-Steiner et al., 2004; Salinas et al., 2015; Bouzas et al., 2020]. Although some works claim the possibility of reconstructing piecewise-planar shapes directly from point clouds, they suffer from serious scalability issues [Boulch et al., 2014; Mura et al., 2016; Nan and Wonka, 2017]. In this work, we aim at efficiently reconstructing compact building surfaces directly from point clouds. Notice that 3D city buildings, in particular, can be abstracted with various level of detail (LoD) via B-rep. In this research, however, a generic reconstruction approach is dedicated that does not limit its application to specific LoD, but applies to any piecewise-planar objects.

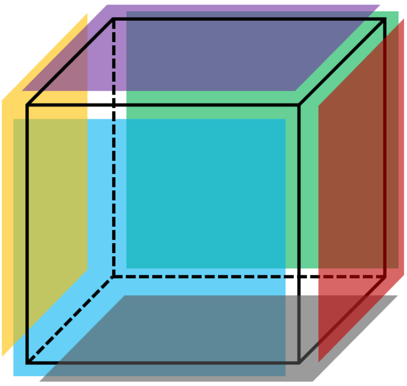
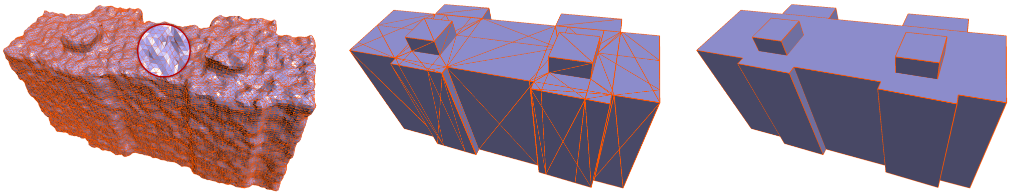


Figure 1.6: Boundary representation: a cube can be expressed by its six bounding facets.

Figure 1.7: Examples of piecewise-planar structures in the built environment

1.2 Inspiration and our approach

3D shapes are not confined to as explicit representations (e.g., point cloud, surface mesh, voxels), but can be encoded implicitly in a function space (see Figure 1.9). A signed distance function (SDF), for instance, can describe an implicit field, where the surface of a shape is implicitly interpreted as zero-set of the SDF. The learnable indicator function of the SDF takes as input a query point and yields an indication on whether the point belongs to the shape (see Figure 1.10). The explicit geometry is then often extracted from the field via computational-expensive iso-surfacing [Mescheder et al., 2019]. Compared with explicit expressions that are heterogeneously distributed, this homogeneous functional representation is particularly favourable for geometric machine learning. Especially recently, the scheme for *learning in the function space* has shown its competence in 3D geometric modelling [Park et al., 2019].



(a) Dense triangles

(b) Sparse triangles

(c) Sparse polygons

Figure 1.8: Facets that compose a surface: dense triangles (326,234 facets), sparse triangles (198 facets) and polygons (61 facets). Representing such piecewise-planar shape with arbitrary-sided polygons (c) is most compact.

Existing deep implicit fields-based reconstruction methods primarily aim at smooth surfaces [Park et al., 2019; Mescheder et al., 2019; Chen and Zhang, 2019; Erler et al., 2020], and thus are unsuitable for addressing compact building models. A few deliver limited piecewise-planar approximation while still lacking reasonable generalisation

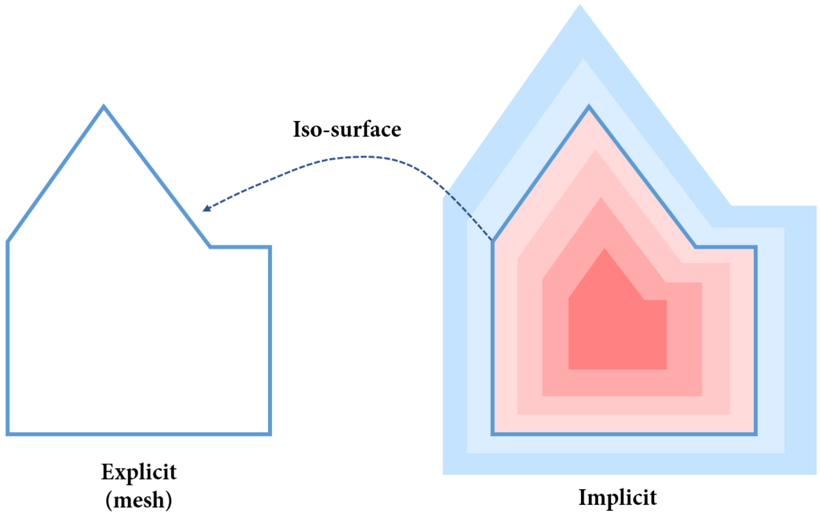


Figure 1.9: Explicit and implicit representation. A surface mesh can be extracted from the zero-set of the implicit field via iso-surfacing.

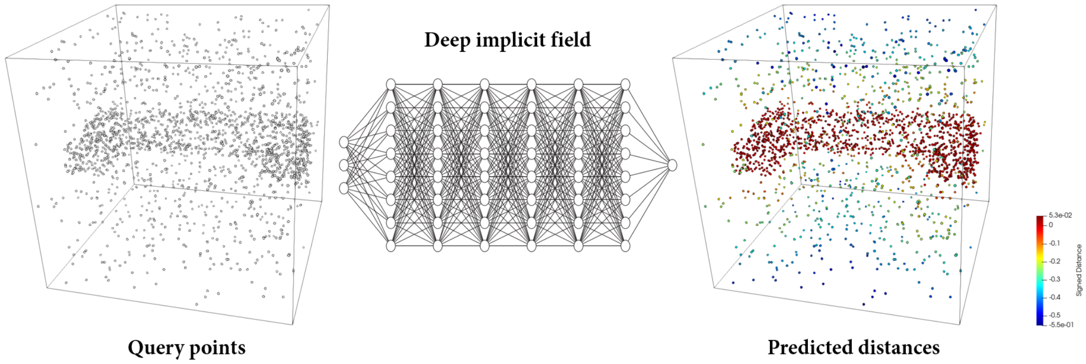


Figure 1.10: Deep implicit field. Given query points sampled in space, the neural network learns to estimate their distance (coloured) to the surface of the shape.

capability to unseen data during training [Chen et al., 2020; Deng et al., 2020]. None of these methods has tackled the reconstruction problem with reasonable generalisation capability to real-world point clouds meanwhile with piecewise planarity preserved. Nevertheless, the learnable implicit functions employed by these methods underpin strong priors exhibited in the training data, by enabling expression of automatically learnt deep features that traditional methods with their hand-crafted features fall short of. Inspired by the recent advance in learning in the function space, we further explore the possibility of this learning-based approach for piecewise-planar building reconstruction with deep implicit fields.

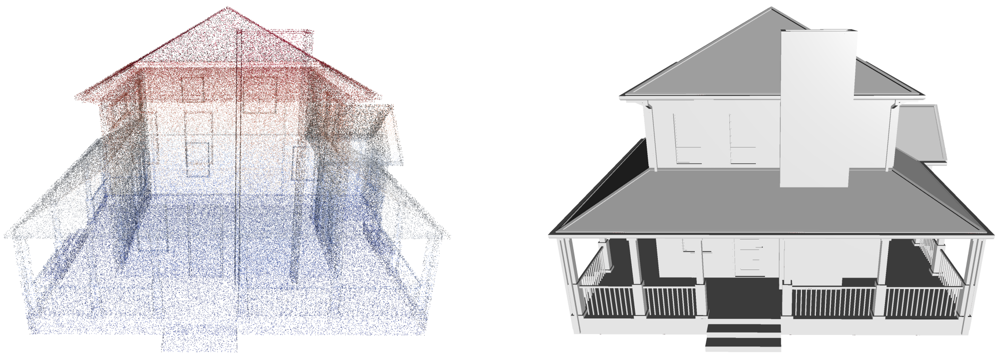
In this thesis, we propose a novel framework for reconstructing compact, watertight, polygonal building surface meshes from point clouds by incorporating implicitly encoded function space with explicitly constructed geometry. The explicit geometry provides a polyhedral embedding as the candidate set, from which extraction of the building's surface is neural-guided by a learnt deep implicit field. We formulate a Markov random field (MRF) to regularise surface complexity, and solve this combinatorial optimisation problem using an efficient graph-cut solver.

We extensively evaluate the proposed method in comparison with state-of-the-art methods in shape reconstruction, surface approximation and geometry simplification. Experimental results reveal that our adaptive strategy drastically reduces redundant candidate polyhedra while respecting the spatial layout. Moreover, the learnt implicit field shows robustness to noise and insufficient measurements, and generalises reasonably well from synthetic scans to real-world measurements. The graph-cut solver is capable of extracting complex building surfaces efficiently. With our approach, high-quality building models can be obtained with significant advantages over fidelity, compactness and computational efficiency.

1.3 Research objective and scope

The goal of this research is to develop a learning-based approach for piecewise-planar 3D building model reconstruction from point clouds, as depicted in Figure 1.11. The main research question to be addressed is *how can deep implicit fields be used for compact building model reconstruction*, which further branches into the following research sub-questions:

- How to incorporate a neural network architecture that leverages an implicit field?
- How to guarantee the reconstructed surface is compact and watertight?
- To what extent does the proposed method generalise across different point clouds?
- How sensitive is the method regarding contaminated (e.g., noisy) and incomplete point clouds?
- What are the (dis)advantages of the proposed method compared with state-of-the-art methods in terms of geometry accuracy and topology validity?



(a) Point cloud

(b) Watertight surface mesh

Figure 1.11: Goal of our approach: reconstruction of compact building surface meshes (b) from point clouds (a) with deep implicit fields. Points coloured with height.

The output building model should be expressed in B-rep, representing the surface with a sparse set of polygons. The model is watertight—but not necessarily manifold—such that volumetric information shall inhabit. Except for piecewise planarity, no geometric assumption is imposed; the method thus remains generic to any piecewise-planar objects besides buildings. Moreover, though deep learning is involved therefore the learnt priors may favour characteristics of the training data, the proposed method should reasonably generalise to real-world point clouds of any kind.

Due to the variety of neural network architectures for deep implicit field learning, it is thereby beyond the scope of this thesis to prove which architecture outclasses the others. Instead, we select with justification one recent architecture [Erler et al., 2020] with demonstrated generalisation capability for learning building-specific SDF, and highlight the advantages and disadvantages of our approach—to the best of our knowledge *the first to exploit deep implicit fields for urban building reconstruction*—so as to further evaluate the potential of deep implicit fields for future urban applications.

1.4 Structure of this thesis

The rest of the thesis is organised in chapters summarised as follows:

Chapter 2 describes and analyses the related work on approaching building surfaces, from both imagery and point clouds. It concludes by listing some of the shortcomings of current approaches.

Chapter 3 presents the proposed building reconstruction method that serves as the basis of this thesis. Our method consists of three components namely *adaptive binary space partitioning*, *occupancy learning in function space* and *surface extraction*.

Chapter 4 reveals the implementation details including how the experiments are framed, how the reconstruction results are evaluated, and how the numerical computations are performed for robustness concern.

Chapter 5 shows the experimental results. Extensive comparison is made against the state-of-the-art methods for shape reconstruction, surface approximation and geometry simplification, from which the characteristics of the proposed method are discussed. It concludes with potential applications of our approach.

Chapter 6 concludes this thesis by revisiting the research questions and listing the contributions of the research. It also provides recommendations for future work as well as an outlook on how deep implicit fields can be utilised in urban modelling.

Appendix A assesses the reproducibility of the research conducted within the thesis.

Appendix B describes an alternative cell complex formulation for generic shape reconstruction.

Appendix C describes alternative deep implicit field formulations and corresponding experimental results.

2 Related work

In this chapter, previous related work on reconstruction is described and briefly analysed. The chapter begins with minimal background on shape representation in Section 2.1. We then relate methods for imagery-based reconstruction in Section 2.2, and, mainly, reconstruction methods from point clouds in Section 2.3. The latter are summarised as involving primitive detection, assembly of the primitives, and alternative approaches. We also cover the recent implicit field-based reconstruction that part of this thesis is based on.

2.1 Shape representation

A shape's geometry can be expressed in various formats: the underlying representations can be broadly categorised as grid models, point sets, surface models and functional representation, as shown in Figure 2.1.

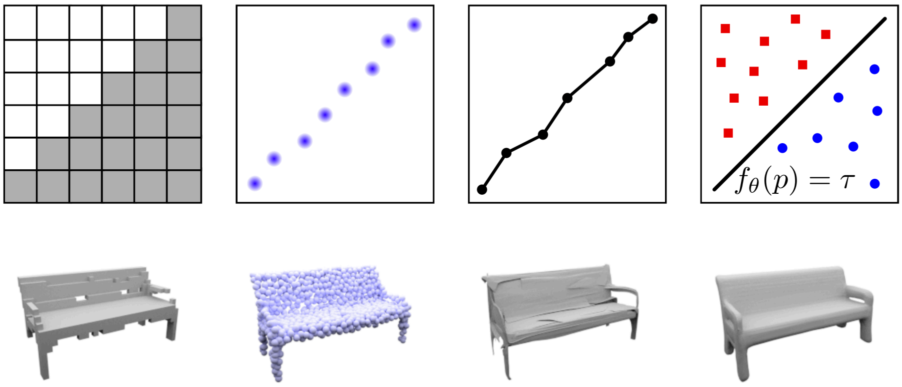


Figure 2.1: Shape representations. From left to right: voxels, point cloud, surface mesh and implicit function [Mescheder et al., 2019].

Among these representations, a point cloud is the most primitive one that can be obtained by multi-view stereo (MVS) from overlapping images with correspondences, or directly from LiDAR measurements. Though preserving the geometric detail, possibly with additional information such as colours or intensity, a point set's incapacity for representing the shape results from being anisotropically scattered in space therefore neither connectivity nor volumetric information shall inhabit.

Voxels are equal-sized units embedded in 3D grid space, as a generalisation of pixels in the 3D domain. This uniform representation allows both preservation of volumetric information and inherent parallelisation of operations on the voxels. However, this uniformity burdens the memory requirements when the grid resolution (i.e., density of voxels) is high; the memory consumption grows cubically with the resolution. A low resolution, instead, hinders its expression of geometric details. Though adaptive data structures such as octree [Meagher, 1982] and recent work on multi-resolution shape reconstruction [Häne et al., 2017; Wang et al., 2018a] are possible to reduce the memory consumption, voxel representations are still limited to comparably small grid resolution, e.g., 256^3. Moreover, sampling errors are introduced irreversibly with the process of voxelisation unless the sampling interval is less than a threshold described by the Nyquist-Shannon sampling theorem [Shannon, 1949].

Mesh representation describes the surface of a shape with vertices, edges and faces. A triangle mesh comprises triangles and is supported by most renderers and other graphics applications. A polygonal mesh relaxes the constraint on the number of sides and can thus be regarded as a generalised form of a triangle mesh. Representing a shape with such arbitrary-sided polygons further reduces the redundancy, as shown in Figure 1.8. Nowadays many renderers support quadrangles and higher-sided polygons in addition to triangles. Urban building models, as one particular mesh category, can be abstracted in various predefined LoD as shown in Figure 2.2. LoD0 delineates the footprint of a building in 2D. LoD1 coarsely represents the building with a prismatic volume usually extruded from LoD1 model. LoD2 models the roof with simplified piecewise-planar approximation. LoD3 provides further architectural detail with roof superstructures, windows and doors. LoD4 complement LoD3 with a description of indoor features. In this thesis, we aim at a general reconstruction approach that does not limit its application to specific LoD, but applies to any piecewise-planar objects.

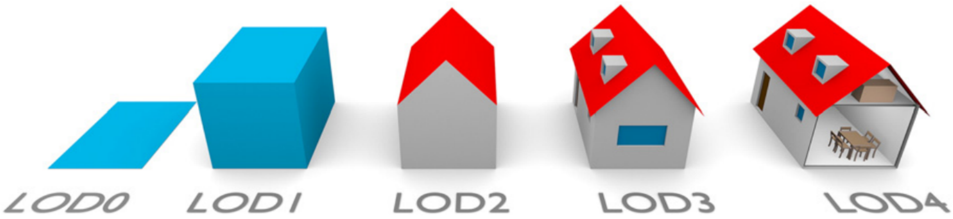


Figure 2.2: A building represented in various LoD [Biljecki et al., 2016]

Besides the aforementioned explicit representations, 3D shapes can be encoded implicitly in a function space, as shown in Figure 2.1. An implicit field is defined by such a continuous function, where the explicit geometry is then often extracted from the field via iso-surfacing such as marching cubes [Lorensen and Cline, 1987]. Recently, modelling a shape as a learnable indicator function has been prevalent for geometric machine learning [Park et al., 2019; Chen and Zhang, 2019; Mescheder et al., 2019]. Inspired by the recent advance in learning in the function space, we further explore the possibility of urban building reconstruction empowered by deep implicit fields.

2.2 Shape reconstruction from imagery

Images are embedded in 2D space, yet are arguably the foremost source for machine perception of 3D scenes. Most imagery-based reconstruction solutions extract point clouds from multi-view images using structure-from-motion (SfM) and MVS algorithms, sequentially, from which the 3D geometry can be obtained. Calakli et al. [2012] propose a framework for surface reconstruction from multi-view aerial images, which combines probabilistic volumetric modelling with implicit surface estimation. Xu et al. [2016] propose an interactive system to generate both 3D models and motion parameters; the latter can be directly animated through kinematic simulation. We also refer to Schöning and Heidemann [2015] for a comprehensive evaluation of software solutions for 3D reconstruction from multi-view images.

Multi-view images are an important source of accurate and timely updated 3D urban building models. Li et al. [2016a] propose an automatic reconstruction method for large-scale urban scenes from images acquired by unmanned aerial vehicles (UAVs), which comprises an object-level segmentation algorithm and a roof extraction algorithm based on a regularised MRF formulation, as shown in Figure 2.3. Rupnik et al. [2018] address 3D reconstruction of digital surface models from high-resolution multi-view satellite images by fusing the calculated depths maps with dense image matching. Recent development in very high-resolution (VHR) satellite images has facilitated automatic 3D building model reconstruction. Partovi et al. [2019] propose an automatic multistage method for 3D building model reconstruction from VHR stereo satellite images, which combines digital surface models (DSMs) and multispectral information from satellite sensors. Verdie et al. [2015] propose a method that reconstructs 3D urban scenes in multiple LoD. Furthermore, Yu et al. [2021] propose a fully automatic 3D building reconstruction method targeting LoD1 building models from aerial imagery.

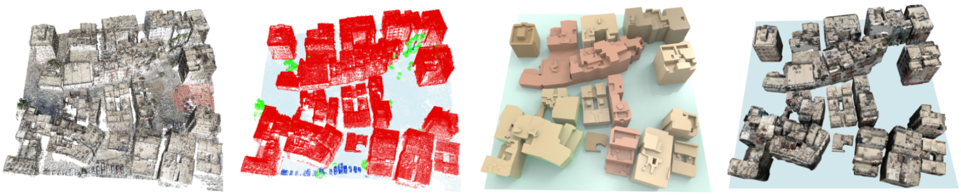


Figure 2.3: Reconstruction building mass models from images of UAVs. From left to right: point cloud, object-level segmentation, reconstructed building models and textured models [Li et al., 2016a].

While conventional approaches can leverage stereo correspondence for multi-views, the reconstruction from a single image is essentially ill-posed. Nonetheless, single-view reconstruction can be approached in a compromised manner. Barinova et al. [2008] propose an efficient method to recover 3D models of urban scenes, as shown in Figure 2.4. The variants include Koutsourakis et al. [2009] which incorporates shape grammars for urban environments. However, the 3D geometry inferred from these two methods is subject to topological ambiguity and is only applicable to specific use cases, e.g., where a limited perspective suffice. Recently, the single-view reconstruction problem has been approached with strong shape priors that can be learnt by deep neural networks. Wang et al. [2018b]; Pan et al. [2019] both address the problem by deforming an initial mesh template guided by the learnt visual prior from the image. Mesh R-CNN [Gkioxari et al., 2019], alternatively, extends the versatile Mask R-CNN architecture [He et al., 2017] with a branch dedicated for mesh reconstruction with possibly varying topological structures.

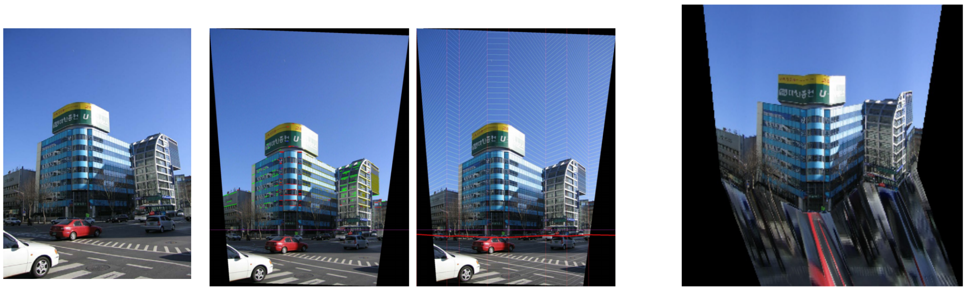


Figure 2.4: Fast automatic single-view 3D reconstruction of urban scenes. From left to right: source image, processed image, different positions of ground-vertical border along the vertical axis, reconstructed scene [Barinova et al., 2008].

Particularly, piecewise-planar structures can be recovered from a single image. A convolutional neural network (CNN) can be utilised to learn to directly estimate a set of plane parameters with a corresponding segmentation map from a single RGB image [Liu et al., 2018; Yang and Zhou, 2018; Liu et al., 2019]. However, only a fixed number of planes with predefined order can be detected by this approach. To address this limitation, Yu et al. [2019] propose to train a CNN to map pixels to a feature space where planes are obtained by grouping the feature vectors in planar regions with a mean shift clustering algorithm. Furthermore, the pairwise inter-plane relations are exploited for piecewise-planar surface reconstruction by Qian and Furukawa [2020]. BSP-Net is claimed the first to achieve structured single-view reconstruction by learning the convex decomposition [Chen et al., 2020], which can produce compact meshes (see Figure 2.5). Similarly, Deng et al. [2020] propose to learn a piecewise approximation of the geometry with a set of convexes. In the context of building reconstruction, however, none of the aforementioned single-view reconstruction methods is capable to deliver quality building models, due to the lack of geometric information that can inhabit one single image.

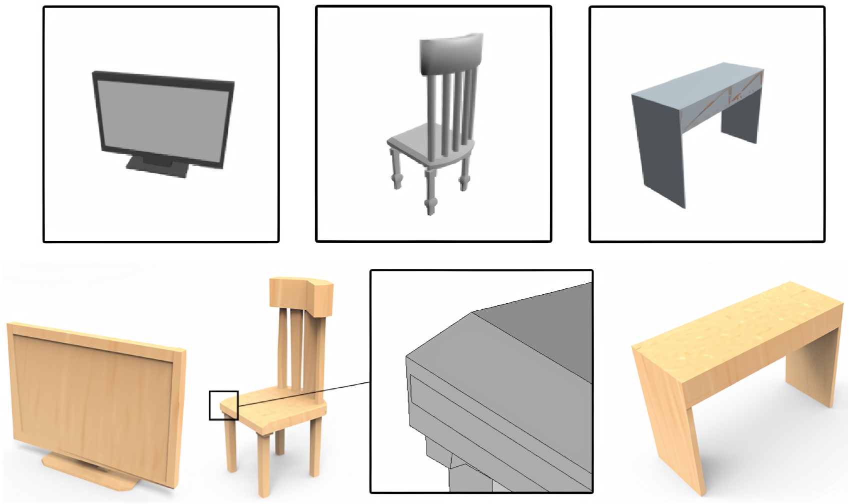


Figure 2.5: Single-view compact mesh reconstruction with BSP-Net. Top: input images. Bottom: reconstructed meshes [Chen et al., 2020].

2.3 Shape reconstruction from point clouds

A point cloud is anisotropically distributed in 3D space. To convert it into a compact surface mesh, significant abstraction is required. In this section, we describe bottom-up approaches that first detect the planar primitives then assembly them, as well as alternative methods by surface approximation and geometry simplification. Finally, recent work on implicit field learning is introduced.

2.3.1 Primitive assembly

High-quality instances of primitives (e.g., plane, cylinder, sphere) are building blocks for bottom-up shape reconstruction. In order to form a piecewise-planar surface, planar primitives can be detected by region growing [Vo et al., 2015] or random sample consensus (RANSAC) [Schnabel et al., 2007]. Region growing segments the point set into subsets by selectively accumulating neighbouring points around initial seeds, in an iterated manner. This relies on a careful selection of the initial seeds and the similarity threshold value which determines whether a point should be appended to the subset, and therefore may fail on corrupted data. RANSAC, instead, can produce reliable plane estimation from samples of the point set. More advanced methods have been proposed which leverage the geometric relationship between the primitives such as parallelism or orthogonality [Monszpart et al., 2015; Oesau et al., 2016]. In our work, we directly use an efficient RANSAC algorithm [Schnabel et al., 2007] for planar primitive detection. Figure 2.6 shows examples of the detected primitives by RANSAC. Reconstruction can then be addressed by properly assembling the detected high-level primitives where two main families of methods exist.

Connectivity-based methods address the primitive assembly by extracting proper geometric primitives from an adjacency graph built on planar shapes [Chen and Chen, 2008; Schindler et al., 2011; Van Kreveld et al., 2011]. Though the graph analysis can be efficiently executed, these methods are sensitive to the quality of the adjacency graph: incorrect connectivity contaminated by linkage errors is prone to an incomplete reconstruction. Arikan et al. [2013] propose an interactive solution enabling the user to complete the surface through an optimisation-based snapping. Labatut et al. [2009]; Lafarge and Alliez [2013] propose a mixed strategy where the confident areas are represented by polygonal shapes and the challenging ones by dense triangles. However, these two solutions either suffer from laborious human interventions for complex scenes, or lack the required compactness for building models, respectively.

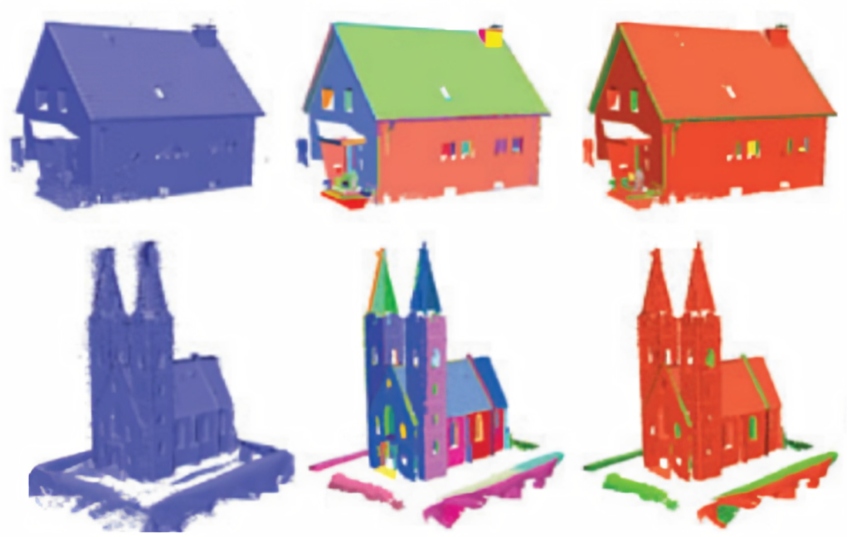


Figure 2.6: Primitives detection with RANSAC. First column: point clouds. Second column: Detected shapes coloured randomly. Last column: Shapes coloured by type [Schnabel et al., 2007].

Slicing-based methods show stronger robustness to challenging data with the divide-and-conquer strategy [Chauve et al., 2010; Mura et al., 2016; Nan and Wonka, 2017]. They partition the 3D space with supporting planes of the detected primitives into polyhedral cells, which themselves consist of polygonal facets. The reconstruction problem is therefore transformed into a labelling problem where the polyhedral cells are labelled as either inside or outside the shape, or equivalently with labelling other primitives. The main limitation of slicing-based methods is the scalability of their data structure. The pairwise intersection of supporting planes—known as a plane arrangement—results in an over-complex tessellation, which is computationally expensive to compute, commonly via a binary tree updated upon each plane's insertion [Murali and Funkhouser, 1997]. When many planar primitives contribute to the intersection, the resulting tessellation may hamper the surface extraction. Moreover, since many anisotropic cells are generated regardless of their spatial hierarchy, the resulting surface is inclined to geometric artefacts. For instance, PolyFit [Nan and Wonka, 2017] formulates polygonal surface reconstruction as an integer programming problem, with hard constraints guaranteeing the generated surface is watertight and manifold. However, it suffers heavily from the scalability issue thus can only process simple models with limited complexity. In contrast, we do not exhaustively partition the 3D space with pairwise intersections, but with a spatially adaptive strategy to drastically reduce the algorithmic complexity.

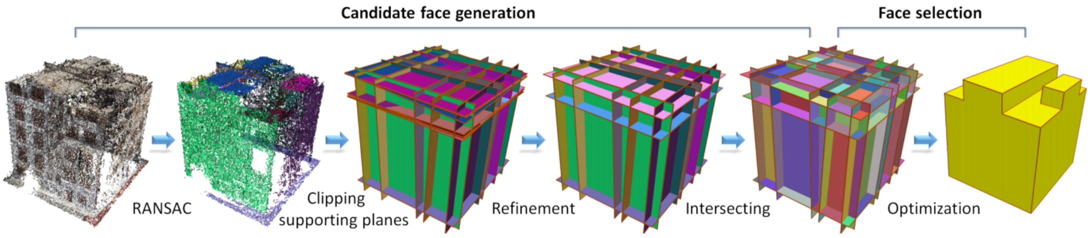


Figure 2.7: Polygonal surface reconstruction from point clouds [Nan and Wonka, 2017]

Recently, several works extend PolyFit to further exploit the inter-relation of the primitives. For example, Li and Wu [2021] develop a method that incorporates the relations into procedural modelling for building reconstruction in CityGML format. Xie et al. [2021] propose to combine the rule-based and the hypothesis-based strategies for efficient building reconstruction. Distinct from these works on urban building reconstruction which use hand-crafted features, our method exploits automatically learnt deep features in the function space.

2.3.2 Surface approximation

An alternative approach to producing compact polygonal surfaces is to simplify an existing dense smooth surface, which we refer to as surface approximation. The smooth surface can be approached from point clouds with Poisson reconstruction [Kazhdan et al., 2006; Kazhdan and Hoppe, 2013], which addresses the problem by establishing an indicator function whose gradient approximates the vector field, underpinned by the points with their oriented normals. The output surface is acquired by extracting an iso-surface of this function (see Figure 2.8). We refer to Berger et al. [2017] for a survey of smooth surface reconstruction algorithms.

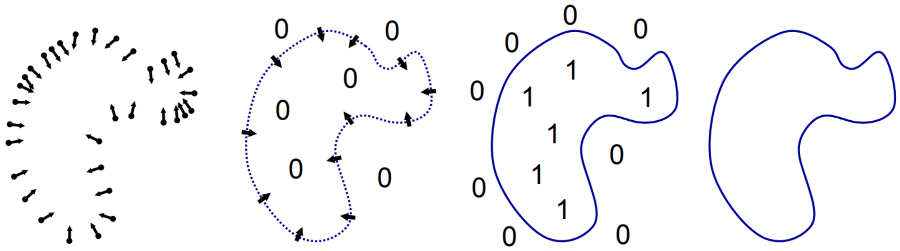
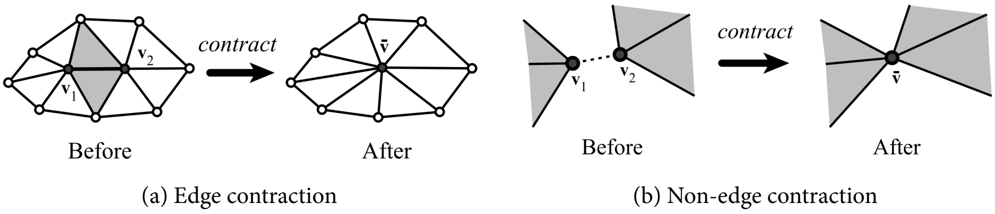


Figure 2.8: Illustration of Poisson reconstruction. From left to right: oriented points, indicator gradient, indicator function and surface [Kazhdan et al., 2006].

Dense triangles on a smooth surface can then be simplified into concise polygons via various approaches. Garland and Heckbert [1997] propose to iteratively contract vertex pairs under quadric error metrics (QEM), such that the number of facets is reduced to a specified number, as illustrated in Figure 2.9. To preserve the piecewise-planar structure during contraction, Salinas et al. [2015] propose structure-aware mesh decimation (SAMD) which incorporates the planar proxies detected from pre-processing analysis into an adjacency graph, then approximate the mesh as well as the proxies, as shown in Figure 2.10. However, these contraction operators are inclined to more triangles than desired for piecewise-planar building representation. Alternatively, the variational shape approximation (VSA) proposed by Cohen-Steiner et al. [2004] approaches the approximation through repeatedly error-driven partitioning, as shown in Figure 2.11. These methods demand the input smooth surface being accurate in both geometry and topology for a faithful surface approximation. However, the requirement is rarely satisfied for real-world measurements. Instead, our reconstruction method can directly output a concise polygonal mesh without approximating an intermediate.



(a) Edge contraction

(b) Non-edge contraction

Figure 2.9: Edge contraction with quadric error metrics (QEM). Both edge (a) and non-edge (b) vertex pairs can be contracted [Garland and Heckbert, 1997].

2.3.3 Geometry simplification

By imposing stronger geometric assumptions, the reconstruction is further regularised towards model-driven where the best-fitted model is selected from a model library. The Manhattan-world assumption [Coughlan and Yuille, 2000] restricts the orientation of facets in only three orthogonal directions and represent the 3D scene with axis-aligned non-uniform polycubes [Ikehata et al., 2015; Li et al., 2016b]. Figure 2.12 demonstrates how a Manhattan-world urban building can be reconstructed from a point cloud. This simplification drastically reduces the geomet-

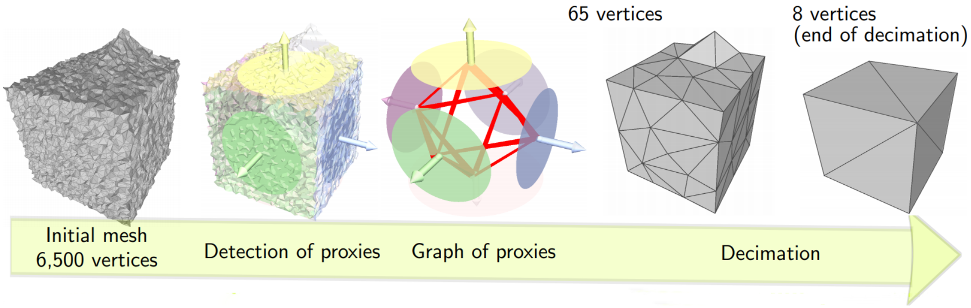


Figure 2.10: Structure-aware mesh decimation (SAMD) [Salinas et al., 2015]

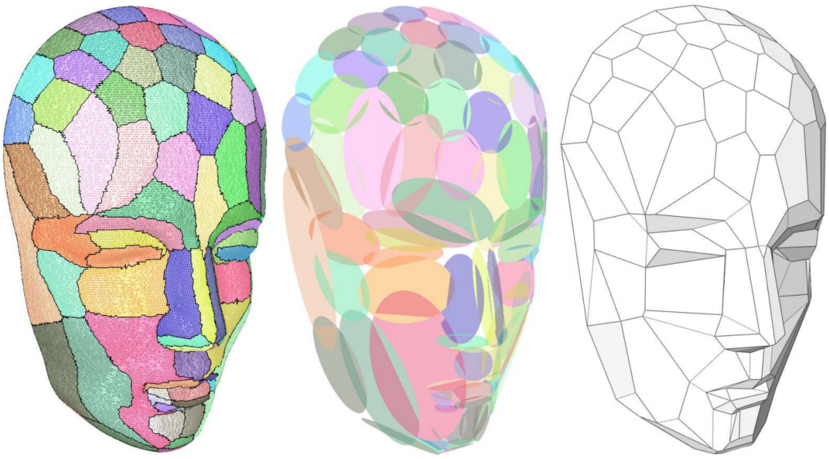


Figure 2.11: Variational shape approximation (VSA). From left to right: error-driven partitioning, geometric proxies and approximated polygonal mesh [Cohen-Steiner et al., 2004].

ric complexity and the solution space to explore. Another common assumption is restricting the output surface to specific disk-topologies. The 2.5D view-dependent representation [Zhou and Neumann, 2010], for instance, can generate arbitrarily shaped roofs with vertical walls connecting them, from airborne LiDAR measurements, as illustrated in Figure 2.13. Furthermore, Verdie et al. [2015] propose an approach for reconstructing urban scenes with various LoD configurations through classification of point cloud, abstraction of proxies and reconstruction, as shown in Figure 2.14. The assumptions can maintain the uniformity of the reconstruction and thus efficient to implement. However, they only apply to specific domains as a limited variety of the layout hypothesis astricts the generalisation of these methods. Our reconstruction method, instead, imposes no geometric assumption except for piecewise planarity, thus remaining generic.

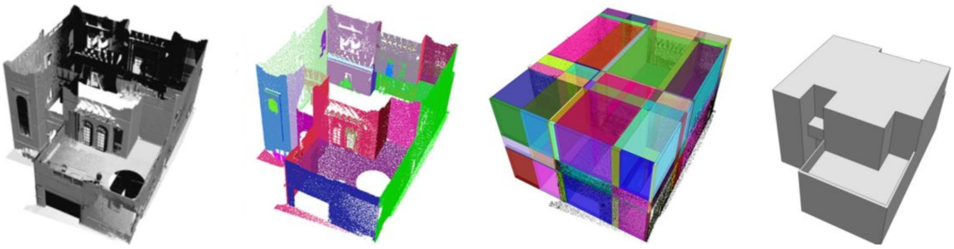


Figure 2.12: Manhattan-world urban reconstruction. From left to right: input point cloud, plane primitives, candidate boxes, 3D model [Li et al., 2016b].

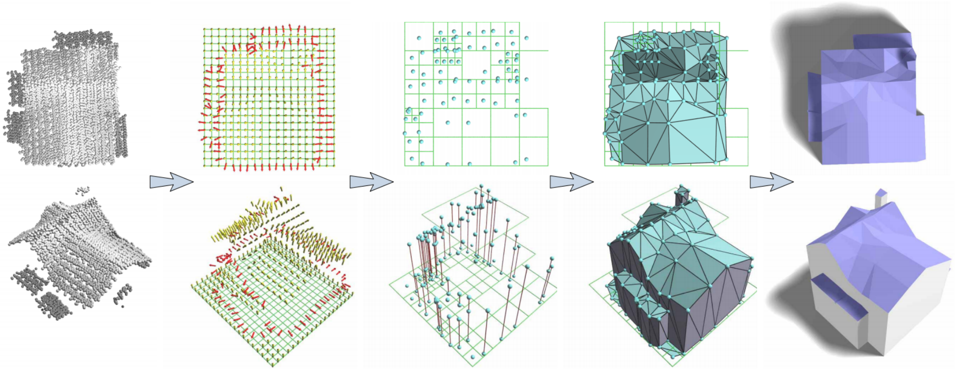


Figure 2.13: Building modelling with 2.5D Dual Contouring. From left to right: input point cloud, 2D grid with surface and boundary, hyper-points, reconstructed mesh model, final model with boundaries snapped to principal directions [Zhou and Neumann, 2010].

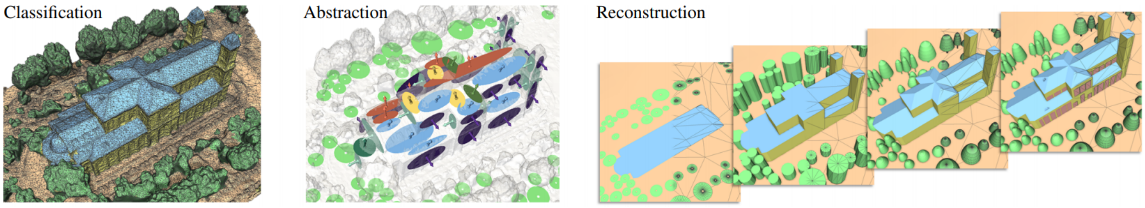


Figure 2.14: LoD generation for urban scenes. From left to right: classification of point cloud, abstraction of icons and proxies according to the LoD configuration, and reconstruction of four LoDs [Verdie et al., 2015].

2.3.4 Implicit field-based reconstruction

Recent advance in deep implicit fields has revealed their potential for 3D reconstruction, most of which aims at reconstructing a smooth surface. The crux of implicit field learning is to learn a robust mapping from the input (e.g., a point cloud) to a continuous scalar field, as shown in Figure 1.9. The surface of the shape can then be extracted via iso-surfacing techniques such as Marching Cubes [Lorensen and Cline, 1987] as illustrated in Figure 2.15. However, iso-surfacing is computationally expensive and inevitably introduces discretization error. This discretization may even result in non-watertight surfaces.

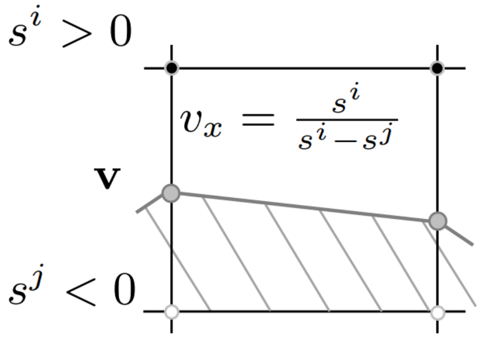
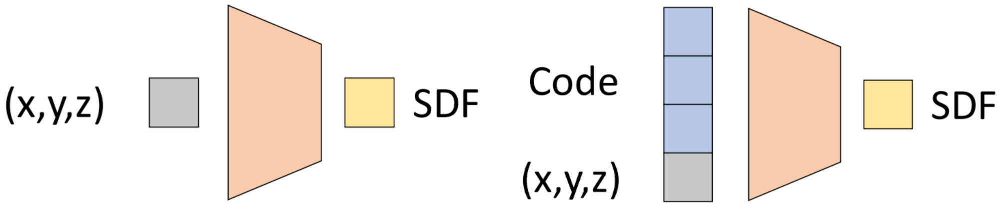


Figure 2.15: Marching Cubes. The position v_x of a vertex v is determined along an edge via linear interpolation, in between $s^i > 0$ and $s^i < 0$ [Remelli et al., 2020].

Various neural network architectures utilising the implicit representation have emerged since the trailblazing success of DeepSDF [Park et al., 2019], which introduces two instantiations as shown in Figure 2.16. The single-shape instantiation encodes the shape information in the neural network itself while the coded shape instantiation takes a code vector containing the shape information that is concatenated with the query point's coordinates. Although both produce the SDF estimation for an input query, training a neural network for each shape is of limited usage[1]. Instead, the coded shape instantiation finds its applications in differentiable rendering [Liu et al., 2020], generative modelling [Chen and Zhang, 2019] and surface reconstruction [Atzmon and Lipman, 2020]. Modelling the surface as an indicator function transforms the reconstruction as a binary classification problem, whose precision is proportional to the network complexity and thus can, in theory, approach to infinity if computation capacity allows. However, like Poisson reconstruction which also formulates an indicator function, iso-surfacing is required as post-processing at inference time. This not only adds execution overhead but fail to guarantee the topological validity of the reconstructed surface.



(a) Single shape instantiation (b) Coded shape instantiation

Figure 2.16: Two DeepSDF instantiations. The single-shape instantiation encodes the shape information in the neural network itself while the coded shape instantiation takes a code vector containing the shape information that is concatenated with the query point's coordinates [Park et al., 2019].

By incorporating constructed solid geometry (CSG), Chen et al. [2020] introduce an end-to-end neural network, BSP-Net, to reconstruct a shape from a set of convexes obtained via binary space partitioning, as shown in Figure 2.17. Similarly, Deng et al. [2020] propose an architecture to represent a low dimensional family of convexes. These two methods both learn to divide and conquer the 3D space with implicit fields. However, the inputs to these two neural networks are either images or voxels, instead of point clouds that this thesis aims to address. Notice that in Appendix C we describe an adapted network architecture with point clouds as input based on BSP-Net.

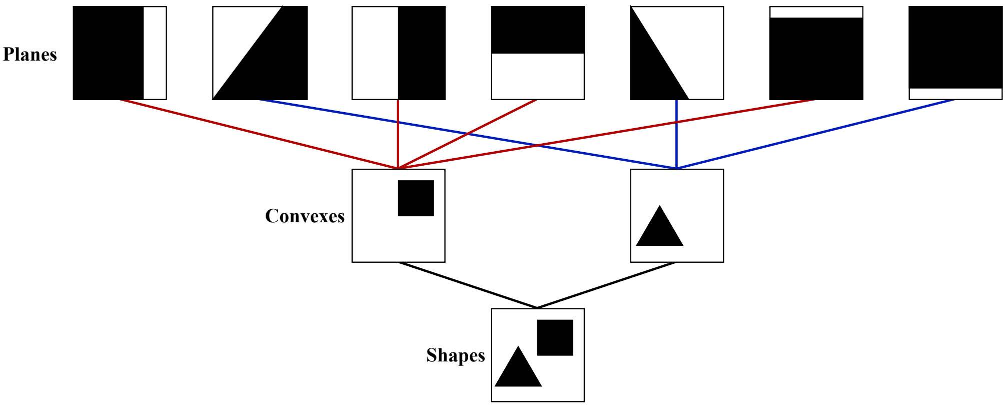


Figure 2.17: BSP-tree for learning convex decomposition. Adapted from Chen et al. [2020].

The single latent feature vector used by the aforementioned methods implies strong priors dependent on the training data. While this allows plausible surface reconstruction even with highly contaminated data, it significantly limits the generalisation ability of these methods. With one feature vector encoding the whole shape, the feature space inevitably overfits the shapes in the training set, and may fail completely given a shape of unseen categories. To mitigate the notorious generalisation incapacity, Erler et al. [2020] propose Points2Surf architecture that estimates an SDF with both local and global feature vectors. The separate feature vectors capture both the detailed local

[1] With one known exception of geometry compression by Davies et al. [2020].

geometry and the course global shape information, the aggregation of which drastically enhances the generalisation capability of the SDF learning. Similarly, the local deep implicit functions proposed by Genova et al. [2020] decompose the implicit field into a structured set of implicit functions, which achieves accurate surface reconstruction with improved efficiency. Since this thesis aims at a generalised reconstruction across point clouds of various kinds, incorporating local geometry into the neural network is vital, we thereby adopt the Points2Surf architecture with demonstrated generalisation capability.

Table 2.1 summarises the highly related work that we further compare our method with, including Poisson reconstruction [Kazhdan et al., 2006], PolyFit [Nan and Wonka, 2017], QEM [Garland and Heckbert, 1997], SAMD [Salinas et al., 2015], VSA [Cohen-Steiner et al., 2004], Manhattan-world urban reconstruction [Li et al., 2016b] and 2.5D Dual Contouring [Zhou and Neumann, 2010]. Verdicts are attached on whether each method can produce compact and watertight surfaces, whether it applies to generic 3D objects, and whether it is efficient. Among these competitors, PolyFit is the only one capable of reconstructing compact, watertight, generic surfaces from point clouds and thus is considered the closest to ours. Yet our method satisfies all four requirements.

Related work	**Characteristics**				
Name	**Category**	**Compact**	**Watertight**	**Generic**	**Efficient**
Poisson [Kazhdan et al., 2006]	RC	✗	✗	✓	✓
Points2Surf [Erler et al., 2020]	RC	✗	✗	✓	✗
PolyFit [Nan and Wonka, 2017]	RC	✓	✓	✓	✗
QEM [Garland and Heckbert, 1997]	AP	✓	✗	✓	✗
SAMD [Salinas et al., 2015]	AP	✓	✗	✓	✗
VSA [Cohen-Steiner et al., 2004]	AP	✓	✗	✓	✗
Manhattan-world [Li et al., 2016b]	SP	✓	✓	✗	✓
2.5D DC [Zhou and Neumann, 2010]	SP	✗	✓	✗	✓
Ours	RC	✓	✓	✓	✓

Table 2.1: Characteristics overview of related work. RC, AP and SP stand for surface reconstruction, surface approximation and geometry simplification, respectively.

3 Methodology

This chapter presents the method proposed in this thesis. Section 3.1 conveys an overview where the high-level components and their interactions are introduced. The follow-up sections describe the respective components in detail. Specifically, in Section 3.2 we propose adaptive binary space partitioning for candidate polyhedra construction. Section 3.3 describes how an implicit field is learnt that facilitates shape-conditioned building interior/exterior classification. Finally, in Section 3.4, we formulate an MRF for surface extraction, and solve the optimisation problem using an efficient graph-cut solver.

3.1 Overview

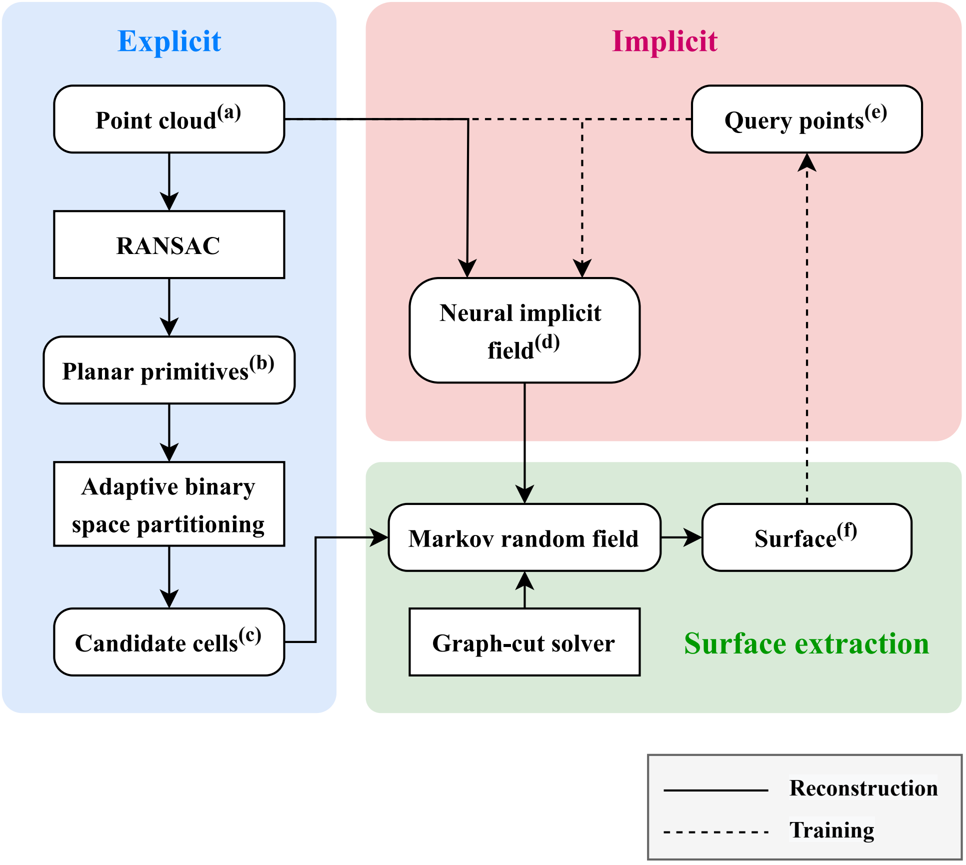
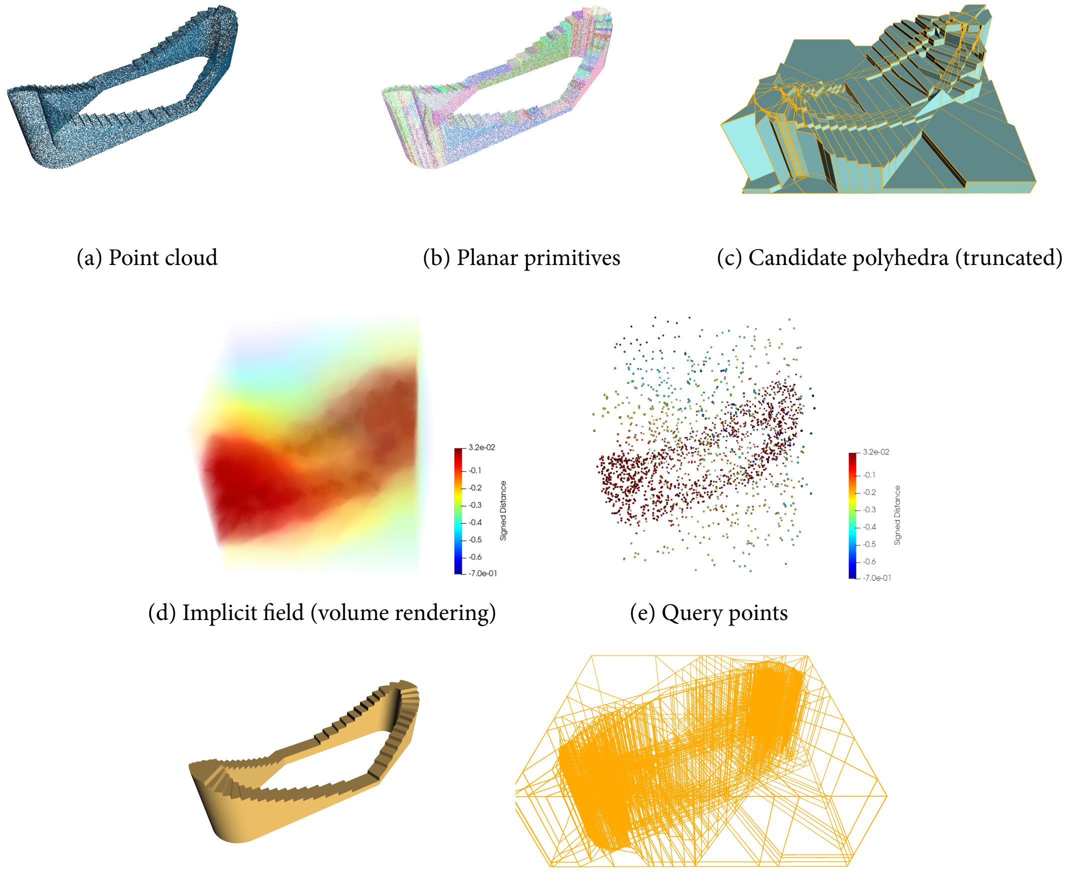


Figure 3.1: Overview of our approach. The approach comprises three functional blocks: within the explicit block (coloured blue), candidate cells are generated from a point cloud; within the implicit block (coloured red), a neural implicit field is learnt that indicates spatial occupancy of the building instance; within the surface extraction block (coloured green), an MRF is formulated for surface extraction where an efficient graph-cut solver is applied. Indices correspond to those in Figure 3.2.

We propose a novel bottom-up method for building reconstruction from point clouds utilising the learnt implicit representation as an occupancy indicator for explicitly constructed cell complexes. The indicator can be intuitively interpreted as a shape-conditioned binary classifier for which the decision boundary is the surface of the building itself [Park et al., 2019]. An MRF is formulated for surface extraction where an efficient graph-cut solver is applied. Figure 3.1 reveals the method conceptually. The proposed method comprises three main components summarised as follows:



(a) Point cloud (b) Planar primitives (c) Candidate polyhedra (truncated)

(d) Implicit field (volume rendering) (e) Query points

(f) Surface (g) Candidate wireframe

Figure 3.2: Intermediate formulation of our approach. Planar primitives (b) are detected from the input point cloud (a). The primitives tessellate the ambient space into candidate polyhedra (d, truncated with a clipping plane for clearer visualisation), with wireframe (g). An implicit field (d, volume rendering) is learnt by a neural network, from which the signed distance of query points (e) can be obtained. The surface (f) is extracted from the candidate polyhedra with the distance queries. Indices correspond to those in Figure 3.1.

- **Adaptive binary space partitioning** tessellates the ambient 3D space to generates a cell complex that complies with planar primitives detected from the point cloud. The partitioning is spatially adaptive therefore being efficient and respective of the building's geometry. The non-overlapping polyhedra in the complex serve as candidates that constitute the final surface.
- **Occupancy learning in function space** utilises a deep neural network to learn a shape-conditioned implicit field that characterises the input point cloud. The implicit field describes the spatial occupancy of the building given any query point in space.
- **Surface extraction with graph-cut optimisation** takes as input the learnt implicit function as an occupancy

indicator, and output the B-rep of the building's surface with complexity regularisation. We formulate surface extraction as an MRF optimisation problem where an efficient graph-cut solver is applied.

Figure 3.2 further illustrates the intermediate formulation corresponding to Figure 3.1. Indicated by the learnt implicit field, each candidate polyhedron is classified as being inside or outside the building. The building's surface can be extracted from the corresponding cell pairs where one is classified as inside and the other one as outside. With this *hypothesizing-and-selection* strategy, the candidate polyhedral embedding is guaranteed to be watertight, leading to the inherited building model being watertight.

3.2 Adaptive binary space partitioning

In this section, we introduce *adaptive binary space partitioning* that generates a cell complex from an input point cloud. Planar primitives are first detected and refined, the hyperplanes of which recursively subdivide the ambient space into a convex set. The generated cell complex consists of high-quality polytopes that serve as candidate cells of the building's shape. With our adaptive strategy, the partition is efficient yet respects the building's geometric structure.

3.2.1 Primitive detection

Planar primitives present in the point cloud are detected using an efficient RANSAC algorithm [Schnabel et al., 2007]. These primitives not only characterise the piece-wise planarity of the shape, but also approximate non-planar surfaces with consecutive segments, as shown in Figure 3.3. To preserve the buildings' structure from point clouds that lack measurements from certain perspectives (e.g., the ground of a building is often invisible to the LiDAR scanner), six facets of the axis-aligned bounding box (AABB) are appended as extra primitives.

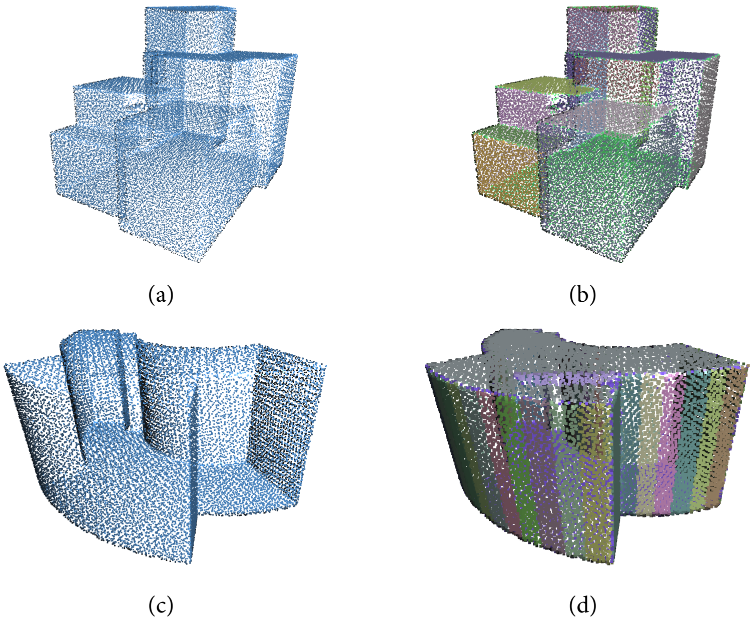


Figure 3.3: Planar primitive detection from point clouds of piecewise-planar (a) and curve surface (c). The curve surface can be approximated by consecutively segmented primitives (d). (b) and (d) are coloured randomly per primitive.

RANSAC can be configured for its rejection rate of low-quality planar primitives. In this step, however, we favour a relatively large number of primitives being detected, to capture as much geometric detail of a building as possible.

Nevertheless, not all detected primitives appear in the ultimate reconstruction results because of the subsequent measures to mitigate noisy input and inappropriate detection.

3.2.2 Primitive refinement

Due to possible contamination that exists in the input point cloud, the detected primitives in Section 3.2.1 may not faithfully reflect planar structures of the shape. Therefore, we refine the primitives by iteratively merging them under specific proximity conditions, with re-estimated plane parameters using principal component analysis (PCA), as detailed in Algorithm 3.1. Meanwhile, we assign a priority to each primitive based on the criteria described in Table 3.1. The high priority on vertical primitives ensures walls are present in the final reconstruction even if points are missing on the building facades. The area-based priority favours larger primitives such that the unnecessary partitions can be minimised. Section 3.2.3 further elaborates the superiority of the chosen priority order.

Algorithm 3.1: Refinement of planar primitives ($\mathcal{S}$)

Input: Raw planar segments $\mathcal{S}$, angle tolerance θ and distance tolerance ε
Output: Refined planar segments $\widetilde{\mathcal{S}}$

```
$\mathcal{Q} \leftarrow$ initialise priority queue;
for $(i, j) \in \mathcal{S}$ do
    $\alpha_{i,j} \leftarrow$ compute angle between $\mathcal{S}_i$ and $\mathcal{S}_j$;
    $\mathcal{Q} \leftarrow$ push $(\alpha_{i,j}, i, j)$ ordered by $\alpha_{i,j}$;
while $\mathcal{Q}$ not empty do
    $(\alpha_{i,j}, i, j) \leftarrow$ pop from $\mathcal{Q}$ with the smallest $\alpha_{i,j}$;
    $d_{i,j} \leftarrow$ compute distance between $\mathcal{S}_i$ and $\mathcal{S}_j$;
    if $\alpha_{i,j} < \theta$ and $d_{i,j} < \varepsilon$ then
        $m \leftarrow$ merge $\mathcal{S}_i$ and $\mathcal{S}_j$ with new plane parameters by PCA;
        $\alpha_{m,\mathbf{n}} \leftarrow$ compute angle between $m$ and every plane in $\mathcal{Q}$;
        $\mathcal{Q} \leftarrow$ push $(\alpha_{m,n}, m, \mathbf{n})$ ordered by $\alpha_{m,\mathbf{n}}$;
    else
        // no more possible coplanar pairs can exist in the priority queue
        $\widetilde{\mathcal{S}} \leftarrow$ extract planar segments from $\mathcal{Q}$;
        break;
    return $\widetilde{\mathcal{S}}$
```

Type	Priority	Rule
Perpendicularity	Highest (binary)	$slope > threshold(0.9)$
Size	By area	$area/max(area)$
$Overall\ priority = priority(perpendicularity) * priority(size)$		

Table 3.1: Primitive priority. Perpendicular and large-area primitives are assigned higher priorities.

Figure 3.4 shows how the normals of refined primitives are of higher regularization than the original ones. Primitives with proximate positions and normal directions are clustered into one, since they high likely belong to the same plane but are separated due to noise. Notice for extremely noisy scenes where many false-positive planes are detected by RANSAC, the reduction of the number of planes can be significant.

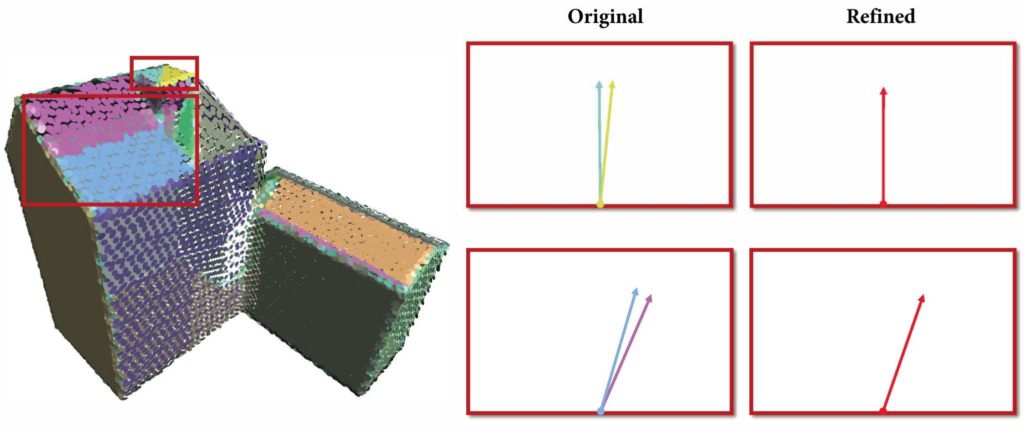


Figure 3.4: Planar primitive refinement. Normal angles are exaggerated.

3.2.3 Construction of cell complex

A topological space can be decomposed into non-intersecting polytopes, each with simple topology and glued along their boundaries into a cell complex. This explicit space partitioning inherently preserves volumetric information, with which the reconstruction problem is then transformed into a proper selection of cells from the complex. Underpinned by this *hypothesizing-and-selection* strategy, we construct a cell complex of candidate polyhedra C from the refined primitives.

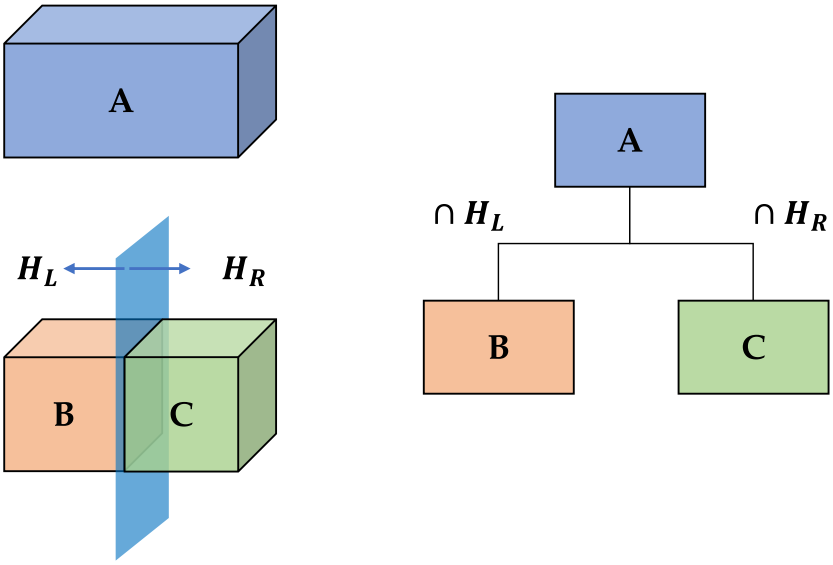
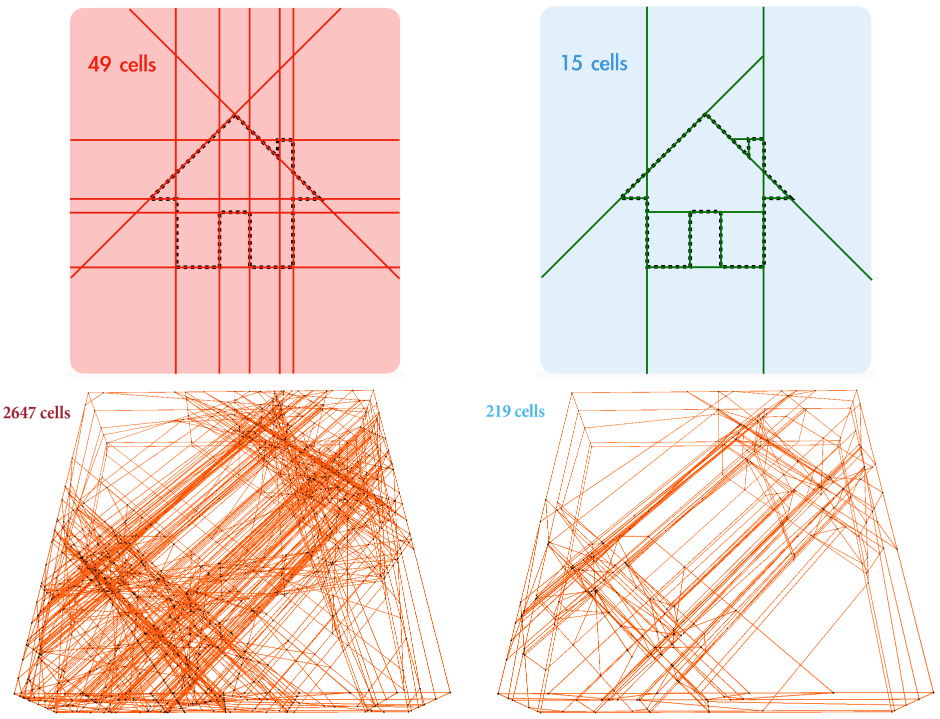


Figure 3.5: Binary space partitioning. The supporting plane of a primitive partitions a parent polyhedron into two children. A binary tree structure is dynamically updated upon insertion of each primitive.

Binary space partitioning (BSP) recursively subdivides a given space into two convex sets using hyperplanes as partitions. Traditionally, the initial BSP tree has one root corresponding to the bounding box of the point cloud. Each time a primitive P_i is inserted, two infinite half-space P_iL and P_iR are generated to partition the parent cell into two children with the Boolean intersection operator, as shown in Figure 3.5. The pairwise intersections of these infinitely-extended hyperplanes, however, burden the creation of candidate sets (e.g., facets or convexes) with unnecessary complexity (Figure 3.6a). To avoid redundant partitioning, instead, we propose an efficient adaptive binary space partitioning that minimises the intersections while respecting the spatial layout which primitives present.

Upon insertion of a primitive, with our adaptive strategy, only the polyhedra that are spatially correlated with the primitive are to be partitioned, as shown in Figure 3.6b. We describe this correlation by intersection test of the AABB between the primitive and the existing polyhedra in the BSP tree, which underlines parallelisation. Compared with exhaustive partitioning, our strategy can minimise redundant partitions that would otherwise generate meaningless candidates, which in turn results in both computational overhead and incompact building surfaces.



(a) Exhaustive

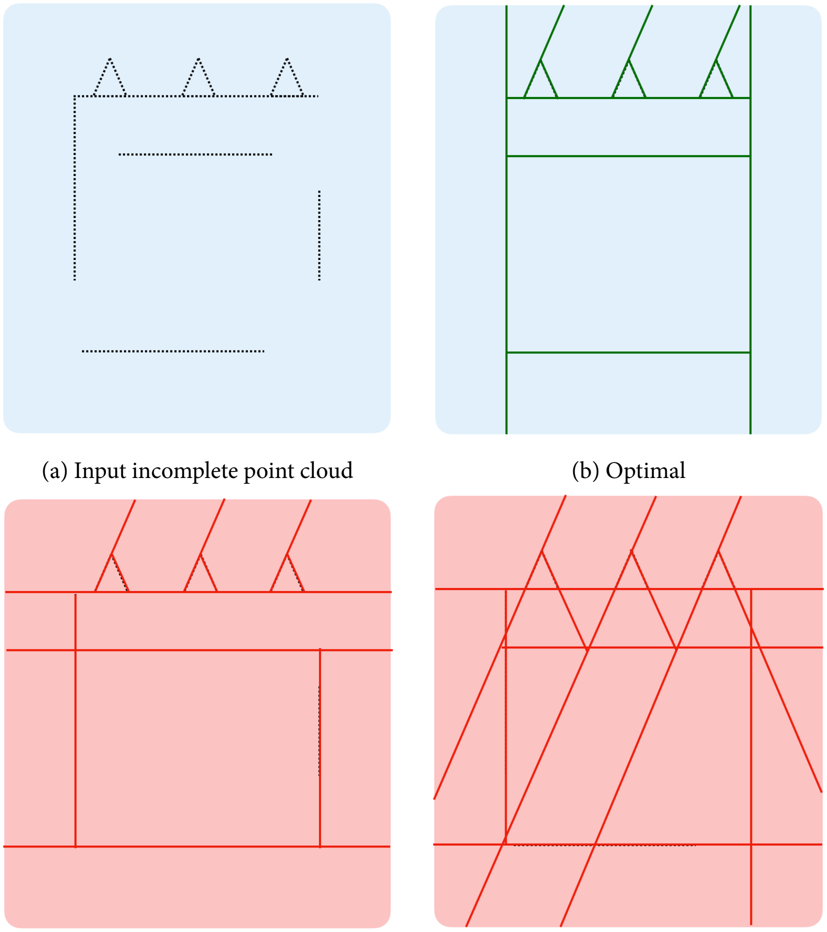
(b) Adaptive

Figure 3.6: Exhaustive partitioning versus adaptive partitioning. Top: 2D illustration. Bottom: 3D wireframe with intersections. Adaptive partitioning minimises redundant intersections while generates fewer yet more meaningful polyhedral candidates.

As illustrated in Figure 3.7, the priority assigned in Table 3.1 preserves the building facades, which may otherwise be underrepresented in the cell complex due to deficient measurements (see Figure 3.7c). Higher priority on larger primitives intuitively resembles an octree structure [Meagher, 1982] where the constructed BSP tree has a minimal number of nodes (see Figure 3.7b). During partitioning, the BSP tree structure, as well as the adjacency among the cells, is incrementally obtained, as shown in Figure 3.8. Notice that the prioritised partition not only improves the performance, but also avoids redundant cells that hinder the follow-up surface reconstruction from the complex.

3.3 Occupancy learning in function space

The constructed cell complex C from adaptive binary space partitioning serves as a candidate set where a subcomplex L of C is to be extracted such that its occupancy O_L is predicted as inside the shape by the learnt implicit function. To address this occupancy classification problem, in this section, we train a deep neural network that learns an implicit function representing the signed distance from each polyhedron to the surface. The learnt mapping can fully exploit the shape prior that resides in the training data, making it flexible for encoding various forms of point clouds. Meanwhile, the aggregation of local information empowers the functional mapping with strong generalisation performance.


(a) Input incomplete point cloud

(b) Optimal

(c) Missing facade

(d) Random

Figure 3.7: Priority impacts the cell complex construction. The optimal priority in Table 3.1 (b) yields a simpler cell complex while keeping the complete facade structure of a building.

3.3.1 Signed distance function

An SDF is learnt by a neural network such that, for any given query point $\mathbf{x}$ in 3D space, it outputs the distance s between the point to its closest surface whose sign signifies whether the point lies inside or outside of the watertight surface:

$$SDF(\mathbf{x}) = s : \mathbf{x} \in \mathbb{R}^3, s \in \mathbb{R}. \tag{3.1}$$

The SDF depicts a continuous field in 3D space. The surface is thus implicitly defined by the iso-surface where $SDF(\cdot) = 0$. The SDF for a unit sphere centred at the origin, for example, can be expressed as

$$SDF(x, y, z) = \sqrt{x^2 + y^2 + z^2} - 1. \tag{3.2}$$

where (x, y, z) are the coordinates of the query point. Figure 3.9 further illustrates an example of the signed distance field defined by Equation 3.1 where the building's surface lies on the interface between the samples with positive distances and those with negative distances. We acquire the SDF from each building's point cloud with deep learning, and employ the estimated signed distance as a confidence indicator for determining whether each polyhedron in the cell complex belongs to the building or not.

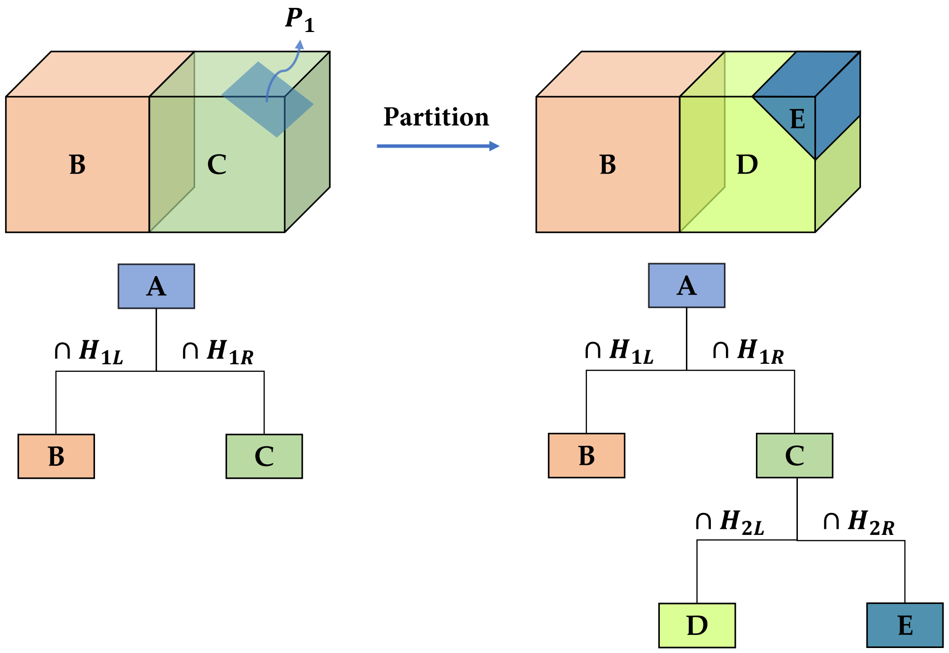


Figure 3.8: Cell adjacency is obtained while partitioning. A binary tree structure is dynamically and locally updated upon insertion of each primitive.

3.3.2 Neural network architecture

We train a neural network that takes a point cloud as input and outputs an estimation of its SDF:

$$f_S(\mathbf{x}) \approx \tilde{f}_P(\mathbf{x}) = s_\theta(\mathbf{x} \mid \mathbf{z}), \text{ with } \mathbf{z} = e_\varphi(P), \tag{3.3}$$

where $\mathbf{z}$ is a latent representation of building surface S that is encoded from the input point cloud P by an encoder e, and s represents a neural network. Encoder e and neural network s are parameterised by θ and φ, respectively. Following the neural network formulation by Erler et al. [2020], we aspire to learn a reliable mapping from the input point cloud to its signed distance field. Since the parameters θ and φ are learnt via supervised learning, the neural network should derive strong priors that are specific to the training data. However, it is required to generalise across various types of point clouds.

We decompose the SDF into two separate components: the absolute distance f^d and its sign f^s, based on the Points2Surf architecture [Erler et al., 2020]. The estimated absolute distance $\tilde{f}^d_P(\mathbf{x})$ can be determined from only the neighbourhood of the query point:

$$\tilde{f}^d_P(\mathbf{x}) = s^d_\theta\left(\mathbf{x} \mid \mathbf{z}^d_x\right), \text{ with } \mathbf{z}^d_{\mathbf{x}} = e^d_\varphi\left(\mathbf{p}^d_{\mathbf{x}}\right) \tag{3.4}$$

where $\mathbf{p}^d_{\mathbf{x}} \in P$ is a sampling of the neighbouring points around query point $\mathbf{x}$. Estimating the absolute distance locally instead of deriving from the entire shape forces the network to rely on local encoding $\mathbf{z}^d_{\mathbf{x}}$ for more accurate distance estimation around $\mathbf{x}$.

For estimating the sign $\tilde{f}^s_P(\mathbf{x})$ at $\mathbf{x}$, local sampling does not suffice because the occupancy information cannot be reliably estimated from the local neighbourhood only. Therefore, a global sub-sample P is taken as input:

$$\tilde{f}^s_P(\mathbf{x}) = \operatorname{sgn}\left(\tilde{g}^s_P(\mathbf{x})\right) = \operatorname{sgn}\left(s^s_\theta\left(\mathbf{x} \mid \mathbf{z}^s_{\mathbf{x}}\right)\right), \text{ with } \mathbf{z}^s_{\mathbf{x}} = e^s_\psi\left(\mathbf{p}^s_{\mathbf{x}}\right) \tag{3.5}$$

where $\mathbf{p}^s_{\mathbf{x}} \in P$ is a uniform subsample of the input point cloud, ψ parameterises the encoder, and $\tilde{g}^s_P(\mathbf{x})$ are logits expressing the confidence that $\mathbf{x}$ has a positive distance to the surface. The two latent descriptions $\mathbf{z}^s_{\mathbf{x}}$ and $\mathbf{z}^d_{\mathbf{x}}$ share

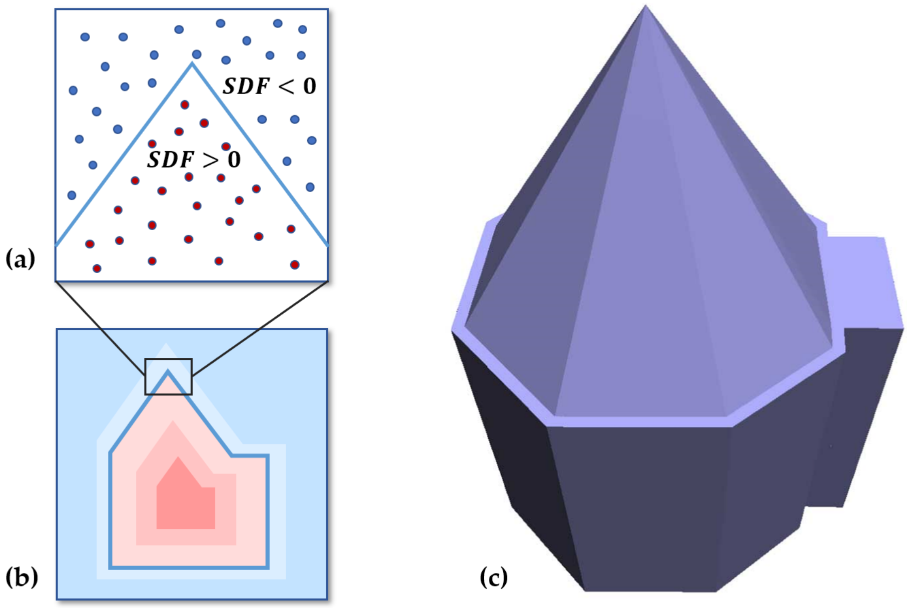


Figure 3.9: Signed distance field. (a) signed distance for sampled points; (b) cross-section of the signed distance field; (c) 3D surface where *SDF* = 0.

information in between, resulting in the formulation for signed distance learning:

$$\left(\tilde{f}^d_P(\mathbf{x}), \tilde{g}^s_P(\mathbf{x})\right) = s_\theta\left(\mathbf{x} \mid \mathbf{z}^d_\mathbf{x}, \mathbf{z}^s_\mathbf{x}\right), \text{ with } \mathbf{z}^d_\mathbf{x} = e^d_\varphi\left(\mathbf{p}^d_\mathbf{x}\right) \text{ and } \mathbf{z}^s_\mathbf{x} = e^s_\psi\left(\mathbf{p}^s_\mathbf{x}\right) \tag{3.6}$$

Figure 3.10 shows an overview of the neural network architecture for SDF estimation. The two encoders e^d_ψ and e^s_ψ are both implemented with PointNets [Qi et al., 2017], with identical architecture but asynchronous parameters. Each point's feature vector is computed using multiple multi-layer perceptron (MLP) layers. The point features are then aggregated into latent vectors $\mathbf{z}^d_x = e^d_\psi(\mathbf{p}^d_\mathbf{x})$ and $\mathbf{z}^s_\mathbf{x} = e^s_\psi(\mathbf{p}^d_\mathbf{x})$ with channel-wise maximum. The decoder s_θ is implemented with MLP as well that takes as input the concatenated feature vectors $\mathbf{z}^d_\mathbf{x}$ and $\mathbf{z}^s_\mathbf{x}$ and outputs both the the absolute distance $\tilde{f}^d(\mathbf{x})$ and sign logits $\tilde{g}^s(\mathbf{x})$. For a detailed description of the architecture, we refer to Erler et al. [2020].

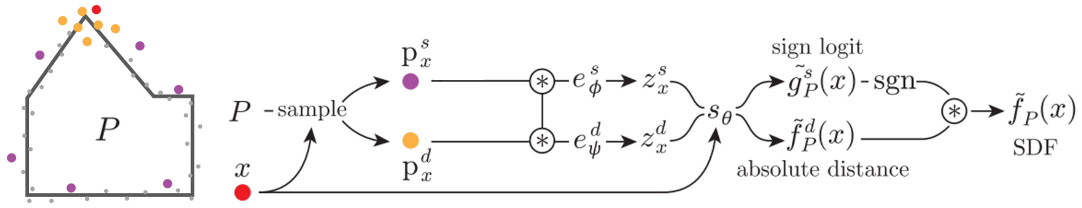


Figure 3.10: Points2Surf neural network architecture. Given a query point (in red), both its neighbouring points (in yellow) and a global subsample (in purple) are fed as input, which are encoded into two feature vectors. The network outputs the estimated SDF as a combination of absolute distance and the sign. Adapted from Erler et al. [2020].

3.3.3 Training with loss function

We train the neural network to estimate separately the absolute distance from the query point $\mathbf{x}$ to the building surface S, and to classify whether $\mathbf{x}$ is inside S or outside S. The building's surfaces are available during training.

Therefore, we pre-sample query points with known signed distance, and use these query points as training examples. We use an L_2-based loss function for the absolute distance:

$$\mathcal{L}^d(\mathbf{x},P,S) = \left\|\tanh\left(\left|\tilde{f}^d_P(\mathbf{x})\right|\right) - \tanh(|d(\mathbf{x},S)|)\right\|_2^2 \tag{3.7}$$

where $d(\mathbf{x},S)$ is the distance from $\mathbf{x}$ to the reference building surface S. The tanh function distributes more weight to the query points with smaller absolute distances, since they are critical for accurate reconstruction. For the sign classification branch, the binary cross-entropy H is used as a loss:

$$\mathcal{L}^s(\mathbf{x},P,S) = H\left(\sigma\left(\tilde{g}^s_P(\mathbf{x})\right), [f_S(\mathbf{x}) > 0]\right) \tag{3.8}$$

where σ is the sigmoid function: $\sigma(x) = \frac{1}{1+e^{-x}}$, and $[f_S(\mathbf{x}) > 0]$ is an indicator function which equals 1 if $\mathbf{x}$ lies outside the surface S, and 0 otherwise. During the training, we jointly minimise these two losses across all buildings and the attached query points in our training set:

$$\sum_{(P,S)\in\mathcal{S}} \sum_{\mathbf{x}\in\mathcal{X}_S} \mathcal{L}^d(\mathbf{x},P,S) + \mathcal{L}^s(\mathbf{x},P,S) \tag{3.9}$$

where $\mathcal{S}$ represents a set including buildings' surfaces S and their point clouds P, and X_S is a set of pre-sampled query points.

Figure 3.11 reveals how the neural network learns to approximate the surfaces of two buildings in a synthetic training dataset. The implicit field gradually approximates the ground truth geometry with iterations. To clarify, the iso-surfacing for the visualisation is performed with marching cubes [Lorensen and Cline, 1987] without constraining the piecewise planarity.

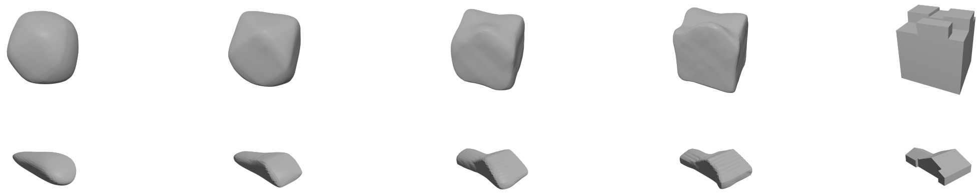


Figure 3.11: Process of approximating a surface with the neural network. Iteration step increases from left to right with the last column depicting the ground truth geometry.

After the neural network is trained, for any given point cloud, it can generate its signed distance field. Since the field is a continuous representation, we can query the distance from any point in space to the surface of the building. Figure 3.12 illustrates a series of cross-sections of the signed distance field as predicted by the neural network.

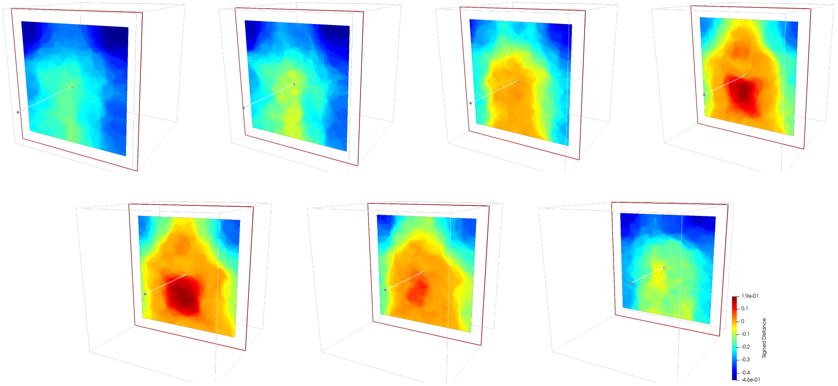


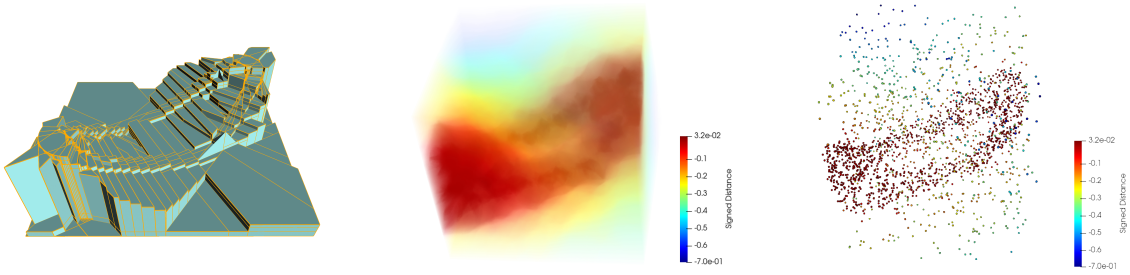
Figure 3.12: A series of 2D cross-sections of a signed distance field of the building in Figure 3.9

3.3.4 Signed distance voting

The continuous SDF describes a scalar field, where we now assign each candidate polyhedron in the complex an aggregated signed distance through voting (see Figure 3.13):

$$\bar{SD}^P = \frac{1}{p}\sum_{i\in P} SD_i^P \tag{3.10}$$

where SD_i^p is the inferred signed distance of the i-th query point in the polyhedron P, and p is the number of query points in P.



(a) Candidate polyhedra (b) Implicit field (c) Query points with distances

Figure 3.13: Signed distance estimation for each candidate polyhedron

As shown in Figure 3.14, the signed distance value of each polyhedron is generated by averaging the votes of SDF values from the samples in the polyhedron., which all represents the confidence that the polyhedron belongs to the shape. With more representatives joining the voting, the polyhedral signed distance can be more accurately reflected. However, too many representatives may burden the inference computation. Since the SDF depicts a continuous field, no abrupt changes exist given a high-quality candidate polyhedral embedding. Therefore, we experimented with different settings and eventually choose only the centre of each polyhedron as a representative vote.

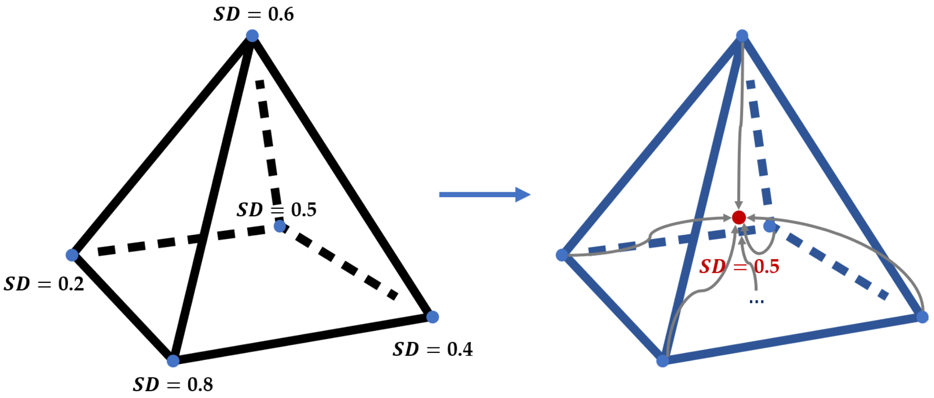
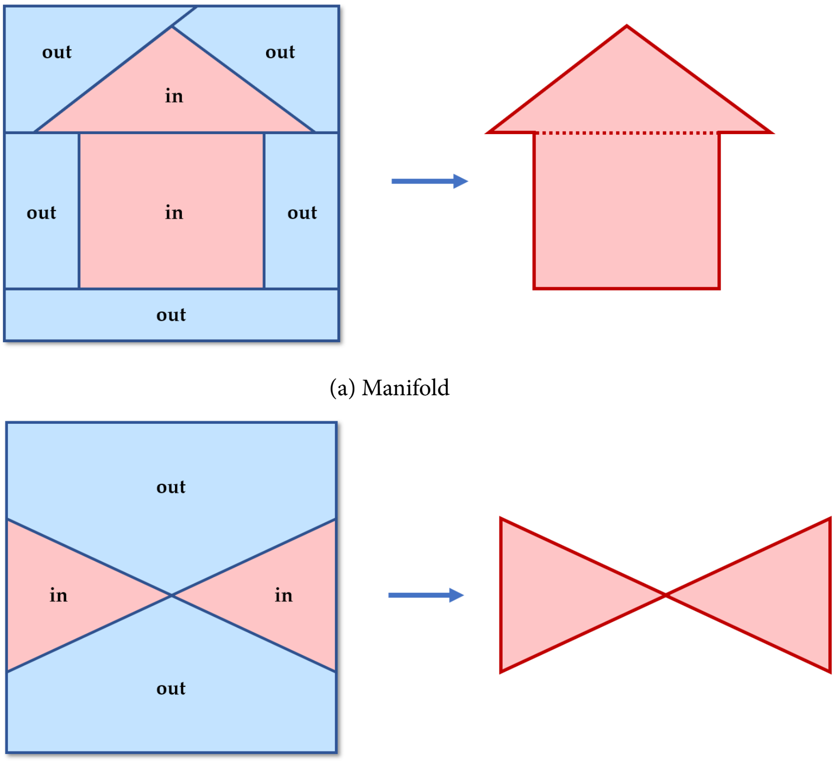


Figure 3.14: Signed distance voting. The signed distance of a candidate polyhedron can be expressed by averaging the votes from representatives inside the polyhedron.

3.4 Surface reconstruction

With the cell complex constructed via adaptive binary space partitioning, and the indicator function learnt by the neural network, the surface reconstruction can be addressed as a combinatorial binary labelling problem, where an inside-outside label is to be assigned to each polyhedron in the complex. The surface exists between every pair of inside polyhedron and outside polyhedron, as shown in Figure 3.15a. As our adaptive binary space partitioning produces a valid polyhedral embedding, the surface is inherently guaranteed to be watertight. Notice that, however, the output surface may be non-manifold. As shown in Figure 3.15b, two inside polyhedra can be connected along one non-manifold edge or at one non-manifold vertex. We argue this non-manifold abstraction can be seen in real-world building structures (see Figure 3.16) and therefore should be allowed in reconstruction.



(a) Manifold

(b) Non-manifold

Figure 3.15: Two labelling examples. Both are valid polyhedral embeddings from our reconstruction.

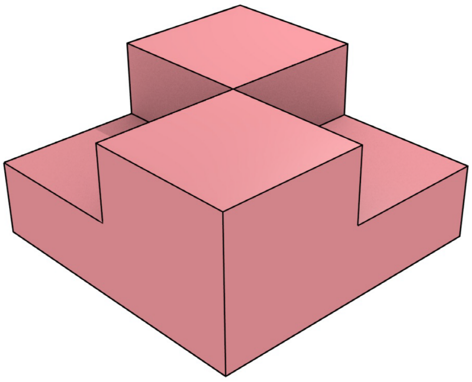


Figure 3.16: An example of non-manifold buildings. The 2D surface is not homeomorphic to a plane [Ohori, 2016]. Non-manifold structure can exist in real-world buildings therefore should be allowed in reconstruction.

A naive reconstruction can be achieved by extracting the surface between each pair of polyhedra where one is classified as inside while the other one as outside. However, this solution does not take into consideration the regularisation needed for compact building models, thus may result in overly complex geometry. In this section, we propose an efficient surface extraction method based on an MRF formulation, and solve the optimisation problem using an efficient graph-cut solver. Our method allows adjustable complexity configurations on the reconstructed building surfaces.

3.4.1 Energy formulation

We formulate reconstruction as a binary labelling problem. Given the cell complex C we denote the binary label to be assigned to each polyhedron in C by $x_i = \{in, out\}$. The quality of the reconstruction is measured with the two-term energy denoted as

$$E(x) = D(x) + \lambda V(x) \tag{3.11}$$

where $D(x)$ and $V(x)$ measures the fidelity and complexity of the reconstructed building surface, respectively. λ is a parameter that weights the regularisation imposed by $V(x)$. The optimal surface is then obtained where $E(x)$ is minimised.

The fidelity term $D(x)$ measures the confidence that the subcomplex L belongs to the building's shape, and is indicated by the neural network's prediction of the signed distance. After associating the representatives to the polyhedra, we express the geometric fidelity by voting on each polyhedron under the form

$$D(X) = \frac{1}{|C|} \sum_{i \in C} d_i(C_i, x_i) \tag{3.12}$$

where $d_i(C_i, x_i) = |probability(C_i) - x_i|$, and the probability of each polyhedron is further expressed as

$$probability(C_i) = sigmoid(SD_i \cdot volume_i) \tag{3.13}$$

where SD_i is the signed distance prediction of x_i from the neural network, $volume_i$ is the volume of x_i, and $sigmoid(x) = \frac{1}{1+e^{-x}}$. The sigmoid function normalises the signed distance to $(0, 1)$. Intuitively, a polyhedron with larger volume should weight higher than that of smaller volume though given identical signed distance.

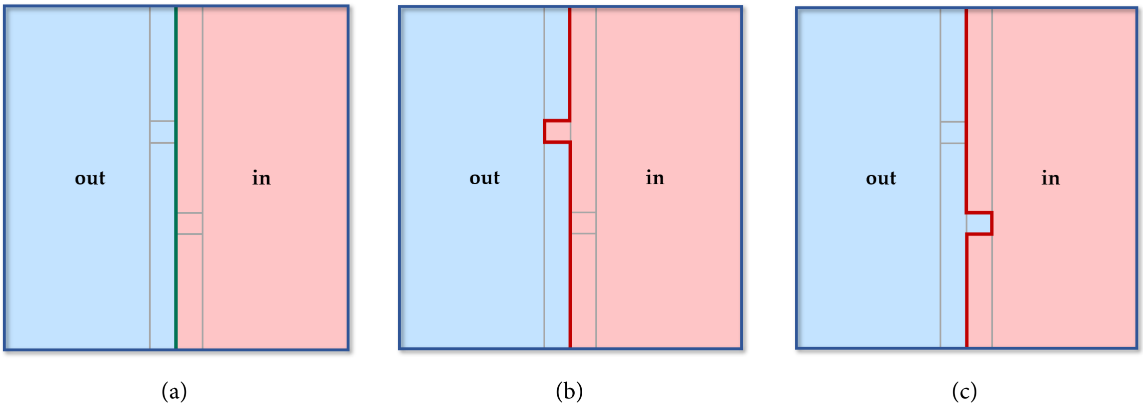


Figure 3.17: Complexity term penalises zigzagging artefact. Our combinatorial optimisation tends to reconstruct a compact surface (a) rather than a zigzagging one (b and c).

The other term $V(x)$ functions by penalising the complexity of the output building surface by its area, where a lower one is favoured. Intuitively, the zigzagging artefact on the surface would yield a higher $V(x)$ value, while a compact surface would minimise this term, as shown in Figure 3.17. The complexity term is expressed by

$$V(X) = \frac{1}{A} \sum_{\{i,j\} \in C} a_{ij} \cdot 1_{x_i \neq x_i} \tag{3.14}$$

where $\{i, j\} \in C$ represents pairs of adjacent polyhedra in the complex, a_{ij} denotes the shared area between polyhedron i and j, and A is a normalisation factor specified as the maximum area of all facets in the cell complex.

The energy formulation in Equation 3.11 implies an MRF, where $D(x)$ and $V(x)$ define the unary potential and the pairwise potential, respectively. If λ is assigned 0, i.e., no regularisation is imposed, the energy formulation is equivalent to a naive extraction of cells.

3.4.2 Surface extraction

The candidate polyhedra in the cell complex form a graph embedding, where each polyhedron represents a node in the graph. From adaptive binary space partitioning, the adjacency information among the polyhedra is dynamically obtained, representing links in the graph. Figure 3.18 illustrates how a cell complex can be interpreted as a graph. Since the energy expressed in Equation 3.11 satisfies the MRF formulation, we employ the graph-cut algorithm to solve this optimisation efficiently.

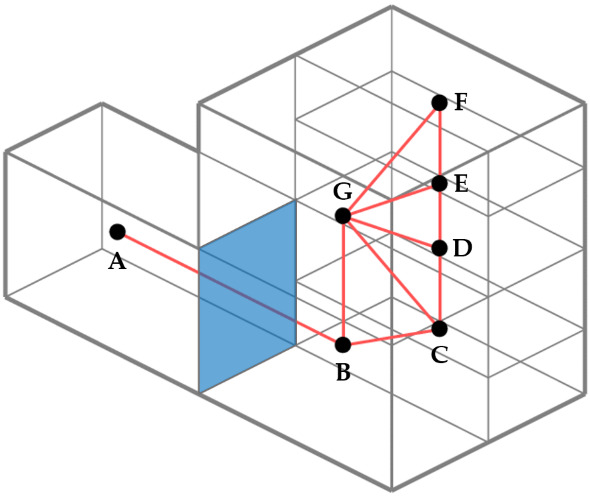
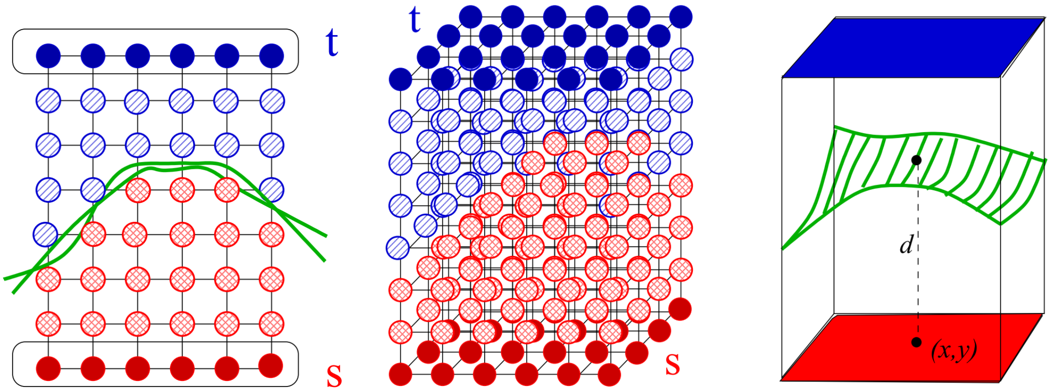


Figure 3.18: Cell complex interpreted as a graph. Links exist in between adjacent polyhedra with shared facets.

Let $G =< N, L >$ be a graph formed by a set of nodes N and a set of undirected links L that connect them. In N there are two distinctive *terminal* nodes which are defined as the *source* and the *sink*, and the other nodes are *non-terminal* nodes P. Such type of graph is called *s-t* graph and Figure 3.19 shows an example. Non-negative and undirected weight $w(p, q)$ is assigned to the link connecting node p and q in the graph. A link is called a *t-link* if it connects a *non-terminal* node with a *terminal*, and is called an *n-link* if it connects two *non-terminal* nodes.



(a) Cut on 2D grid

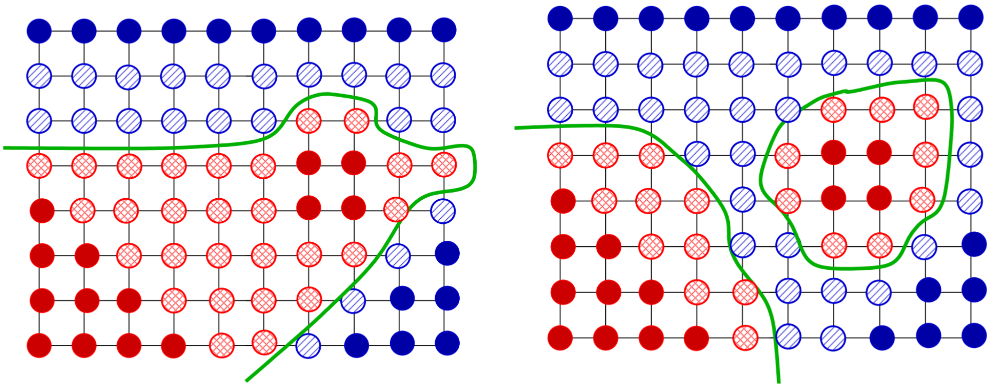
(b) Cut on 3D grid with hypersurface

Figure 3.19: Graph cuts as hypersurfaces [Boykov and Funka-Lea, 2006]

We denote a cut as CT. The cost of the cut $|CT|$ aggregates the weights on CT, which can be expressed as

$$|CT| = \sum_{e \in C} w_e. \tag{3.15}$$

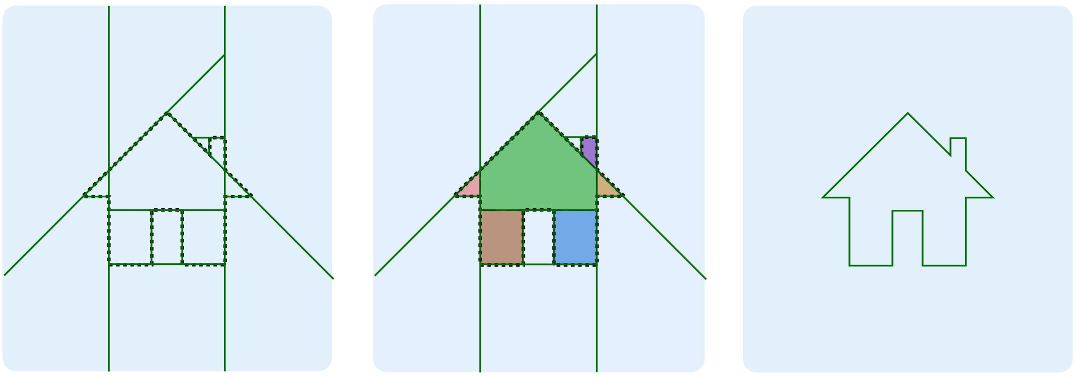
Intuitively, a cut is a segmentation of the nodes in G into two disjoint subsets, denoted S and T, such that the source s belongs to S and the sink t belongs to T. As illustrated in Figure 3.19, a cut is a hypersurface over the links, separating the graph into two subsets, namely polyhedra inside the building's shape and those outside. A minimal cut is the partition corresponding with a minimal value of $|CT|$, which, according to Boykov and Funka-Lea [2006], can be solved by finding the maximum flow from s to t, which is equal to the cost of the minimal cut. Figure 3.20 illustrates different topological properties for separating S and T with hypersurfaces where multiple disjoint components can present. For building reconstruction, both connected segment and disjoint ones are rational since one building can consist of one or multiple components.


(a) Connected source segment

(b) Disjoint source segments

Figure 3.20: Different topological properties for separating source S and terminal T with hypersurfaces [Boykov and Funka-Lea, 2006]

The surface to be extracted lies at the intersection between every pair of adjacent polyhedra where one is classified as inside and the other one as outside the building; this is exactly where the hypersurfaces cut the graph. Therefore, as illustrated in Figure 3.21, the surface can be directly retrieved where the cut is performed whose interface can be cached during the computation of facet area. Compared with the integer programming solver used by Nan and Wonka [2017], our graph-cut solver is substantially more efficient.


(a) Candidates

(b) Labelled as building

(c) Surface

Figure 3.21: Surface extraction from labelled cell complex. Building Surface (c) can be extracted from the labelled cell complex (b) where the hypersurfaces cut the graph.

A consistent orientation contributes to a topologically valid building surface representation. Though orientation information is not stored in our data structure during adaptive binary space partitioning, it can be redressed once the surface is obtained alongside a valid adjacency graph for all polyhedra. Specifically, we start from an arbitrary polygon, trace its adjacent polygons and reverse their orientation if the shared edge is expressed in the same direction (see Figure 3.22). The predicate recursively evaluates every pair of adjacent facets until they all are of opposite directions. This post-processing results in a consistent orientation of the building's surface.

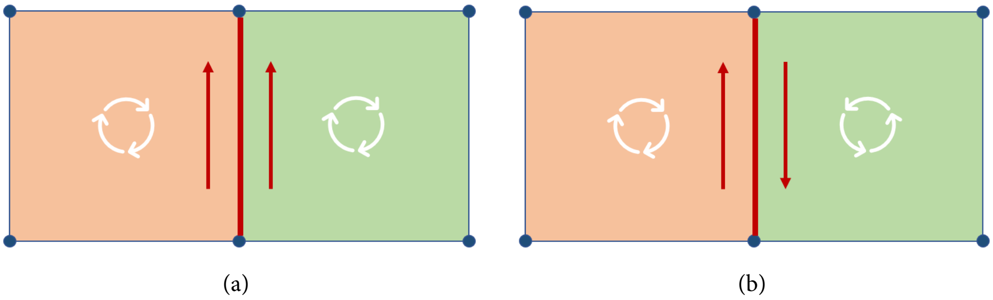


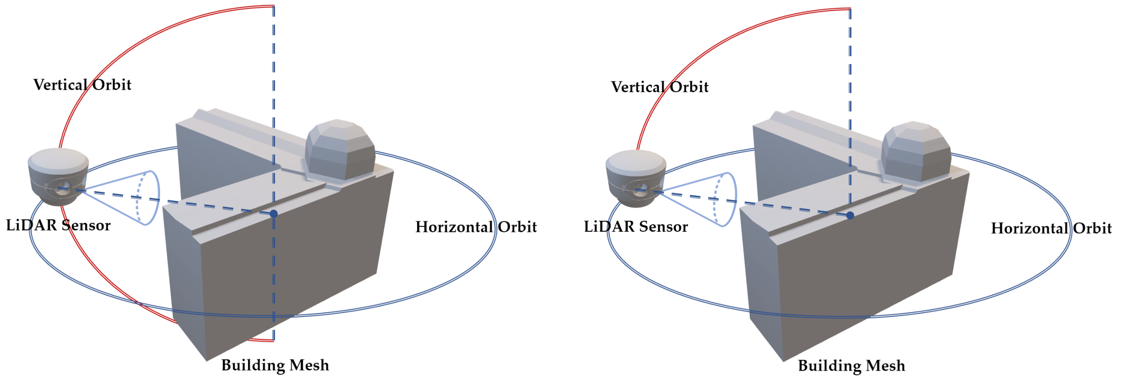
Figure 3.22: Consistent orientation of building's surface. The shared edge between every pair of adjacent facets is recursively evaluated where (a) is reversed to (b).

4 Implementation details

This chapter reveals the implementation details. Section 4.1 explains how the datasets are prepared including synthetic data and real-world scans. Section 4.2 describes the metric under which the proposed method is evaluated. The exact numerical representation of geometry is key to the robustness of our implementation, which is explained in Section 4.3. Finally, Section 4.4 lists the main libraries and software used to implement this thesis.

4.1 Datasets

A variety of datasets are used to evaluate the proposed method. Since this thesis targets building reconstruction with a learning-based approach, high-quality point cloud-surface mesh pairs are required as training data. Due to the lack of data in this domain, however, we simulate our own point clouds based on the Helsinki LoD2 CityGML models[1]. Specifically, we pick 678 watertight building meshes for training, 45 for validation and another 45 for testing. Notice that, due to the patch-based architecture described in Section 3.3.2, a large number of diverse patches are produced from each mesh as training samples.



(a) (b)

Figure 4.1: Simulation with Blensor. A LiDAR scanner rotates on a sphere around the building mesh to generate a full-view point cloud (a). The scanner's position is constrained on the upper half of the sphere to simulate a no-bottom point cloud (b).

The point clouds are generated by simulated scanning on the building meshes. For pre-processing, the watertight building meshes are translated to the origin and scaled uniformly to unit length. To simulate point clouds P on the buildings S, a LiDAR sensor is configured from random perspectives. We use Blensor [Gschwandtner et al., 2011] to simulate the scanning, intentionally with various levels of Gaussian noise and artefacts such as occlusions and light reflections. Each Helsinki building model is scanned randomly from 10 to 30 times to simulate the heterogeneous point distribution in real-world measurement. For each scan, we follow the configuration described in Erler et al. [2020] and position the scanner at a random location on a circumsphere centred at the origin, with a scanning radius set in $U[3R, 5R]$, where R is the largest side of the building's 3D bounding box. The scanner is oriented to target the building but with a slight random shift, between $U[-0.1R, 0.1R]$ along each axis, and rotated randomly around the

[1] https://kartta.hel.fi/3d/

scanning direction. Each scan yields approximately 25,000 points, minus missing measurements given intentionally simulated scanning artefacts. The final point cloud comprises the point clouds of several scans on the same building. Figure 4.1a illustrates the simulation process. Notice that due to a limitation of Blensor's configuration, an equivalent workaround is implemented[2].

For evaluating our method against various levels of noise, we simulate multiple versions of the Helsinki dataset, each with different amounts of Gaussian noise configured. Various Gaussian noise simulates measurement inaccuracies onto the depth values. We prepare point clouds with Gaussian noise whose standard deviation is randomly set in $U[0, 0.005R]$, and use this version for training the neural network. We also create multiple versions where the Gaussian noise is fixed per dataset, from {0, 0.001R, 0.005R, 0.010R, 0.050R}, for evaluating the robustness of the proposed method and its competitors.

In addition to the simulated point clouds, the training set also contains a set of query points X_S for each building. Since query points with smaller absolute distances are of higher importance for detailed surface reconstruction, we randomly sample 1000 points on the surface then apply perturbation in their normal direction by a random displacement in $U[-0.02R, 0.02R]$. In addition, we sample 1000 query points randomly distributed in the bounding box of the building, mounting up to 2000 query points in total per building. To enhance robustness of the learnt SDF, we randomly drop out 1000 query points and use the other 1000 samples for training. We denote this full-view dataset as *Helsinki full-view*. Figure 4.2 shows the dataset examples.

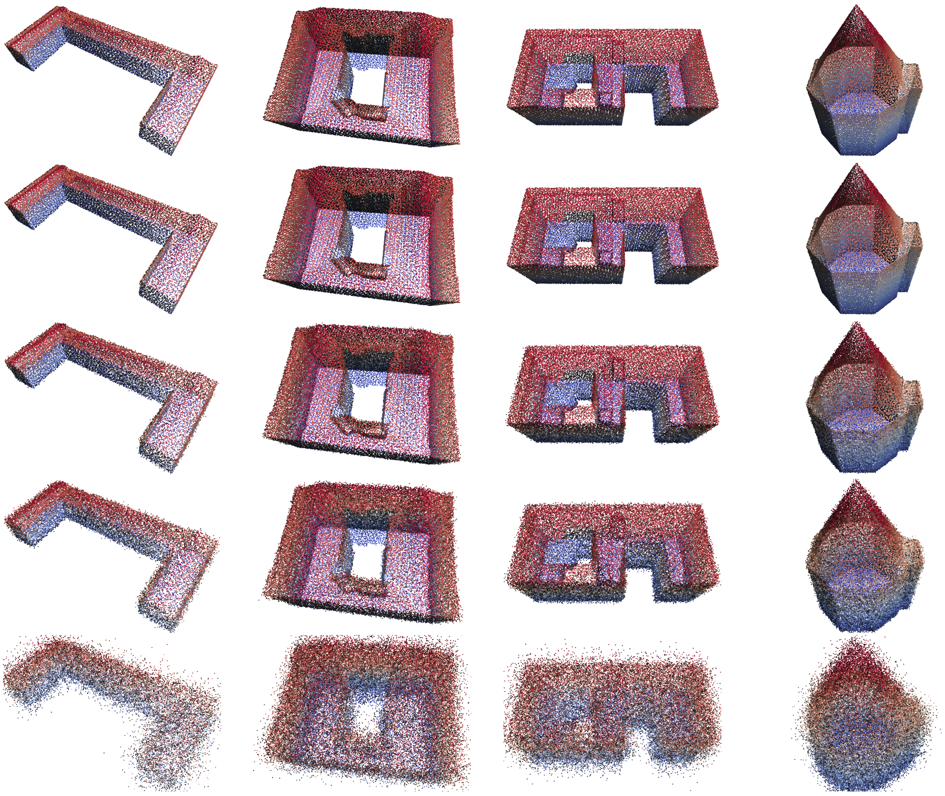


Figure 4.2: Helsinki dataset examples. From top to bottom: Gaussian noise ranging from 0.001 to 0.050.

We further constrain the scanner's position such that no bottom view is captured and create the *Helsinki no-bottom* dataset. The simulated point clouds resemble the real-world data of buildings acquired via MVS or LiDAR, where rare points from the bottom of a building can be captured. Like with *Helsinki full-view*, we generate a version of point clouds with various Gaussian noise, along with the query points, as training data, and a series of point clouds

[2] https://github.com/ErlerPhilipp/points2surf/issues/6

with fixed noise for evaluation purpose only. This simulation without the bottom view is illustrated in Figure 4.1b. Figure 4.3 shows the difference between *Helsinki full-view* and *Helsinki no-bottom* with examples.

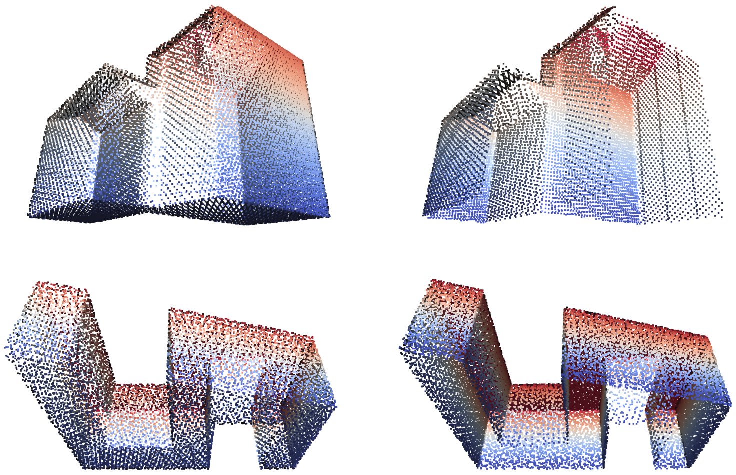


Figure 4.3: *Helsinki full-view* (left) versus *Helsinki no-bottom* (right).

In case the simulated point clouds still deviate from real-world scans, we further evaluate our method on the real-world photogrammetric point clouds of six buildings in Shenzhen, China, produced from aerial images with MVS. The original images were captured by UAVs from top and lateral perspectives. The main bodies of buildings are visible with well-captured roof structures. However, the building facades are only partially visible with missing areas due to occlusions and poor lighting conditions at lower part of the buildings. Examples of the *Shenzhen* point clouds are shown in Figure 4.4.

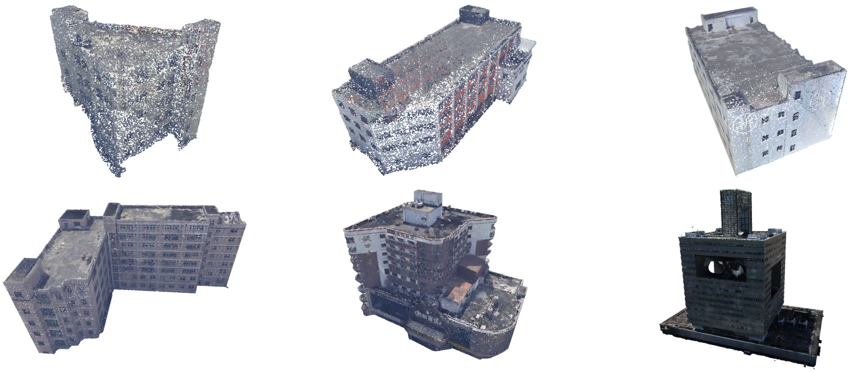


Figure 4.4: Shenzhen dataset examples

Table 4.1 summarises the characteristics of the datasets used for evaluation in this thesis. Since global shape priors are dataset-dependent, conceivably the neural network trained on one dataset may not capture the characteristics of another. Nonetheless, we train our neural network for SDF estimation on the *Helsinki full-view* dataset and *Helsinki no-bottom* dataset with various noise only, individually. The trained model on the former is used for evaluation on full-view point clouds, while the latter on no-bottom point clouds of typical building scans where the bottom view is not visible to the scanner.

Name	Type	Perspective			Quantity	Usage
		Top	Bottom	Lateral		
Helsinki full-view	Simulated LiDAR	✓	✓	✓	768	Training + evaluation
Helsinki no-bottom	Simulated LiDAR	✓	✗	✓	768	Training + evaluation
Shenzhen	Real-world MVS	✓	✗	✓	6	Evaluation

Table 4.1: Datasets overview

4.2 Evaluation metrics

We evaluate the proposed method in terms of geometric fidelity and complexity, as well as its computational efficiency. Since each building in both *Helsinki full-view* dataset and *Helsinki no-bottom* has a ground truth surface, we sample both the reconstructed surface and the reference each with 10,000 points. The symmetric mean Hausdorff (SMH) distance is used as the metric to assess the geometric discrepancy between the reconstructed surface and its reference. The one-sided distance $Dist_{A,B}$ from point set A to set B is denoted as

$$Dist_{a,B} = min_{b \in B}(\|a - b\|) \qquad Dist_{A,B} = max_{a \in A}(Dist_{a,B}). \tag{4.1}$$

As this distance is non-symmetric, the SMH is computed by taking the maximum of $Dist_{A,B}$ and $Dist_{B,A}$:

$$Dist_{SMH} = max(Dist_{A,B}, Dist_{B,A}). \tag{4.2}$$

In the context of surface reconstruction from point clouds,

$$Dist_{SMH} = max(Dist_{SR}, Dist_{RS}) \tag{4.3}$$

where $Dist_{SR}$ and $Dist_{RS}$ represent the one-side Hausdorff distances from surface to reference, and from reference to surface, respectively. Because there are no ground truth surfaces present in *Shenzhen* dataset, we instead calculate the one-side Hausdorff distance $Dist_{SR}$ from the surface to the input point cloud as reference. Notice that for visualising the error distribution over the reconstructed building surfaces, we generate the error map showing the discrepancy between the vertices and their closest reference points.

The geometric compactness of the output building models is assessed by the number of facets it comprises, where the more facets are used to represent a building, the more geometrically complex the reconstructed model is. To clarify, this measure mainly addresses complexity in terms of data structure, instead of visual compactness, since multiple adjacent coplanar facets are of the same visual complexity as one combined.

In addition to the distance measures, we also evaluate the efficiency of our method on a single computer with an Intel i5 CPU clocked at 2.90 GHz and an NVIDIA GTX 2080Ti GPU. Specifically, the running time of cell complex construction, occupancy estimation and surface extraction are respectively timed in sequence for scenes of various complexity. One of the measures we adopt that particularly reflects scalability is the maximum number of planar primitives our method can process without exceeding 10^3 seconds.

Evaluation	Metric	Description
Fidelity	Hausdorff distance	From reconstructed surface to reference
Compactness	Number of facets	Number of facets that composite the surface
Efficiency	Execution time	Total running time for reconstruction

Table 4.2: Evaluation metrics

4.3 Numerical robustness

To tackle the notorious consistency error in geometry, we employ exact arithmetic in the construction of explicit geometry. Since a computer cannot precisely represent a point in $\mathbb{R}^3$, we cast all geometric primitives into the exact rational coordinate space $\mathbb{Q}^3$. This exactness underpins the robustness of the geometry processing including adaptive binary space partitioning and surface extraction because the set of vertices, polygons and polyhedra in $\mathbb{Q}^3$ are self-contained under Boolean spatial operations.

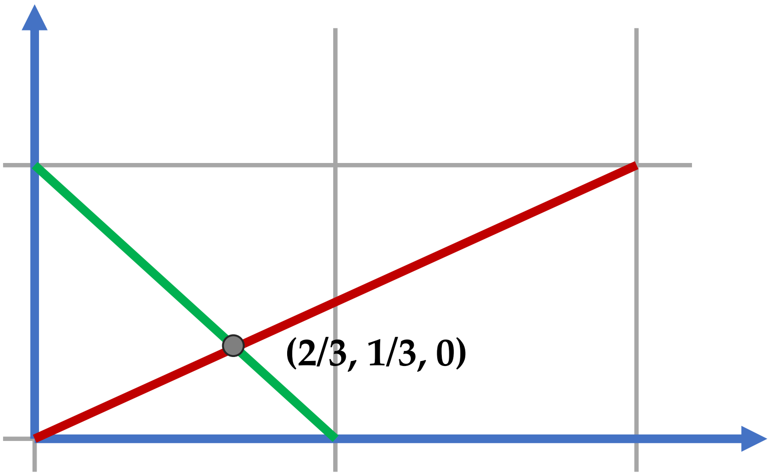
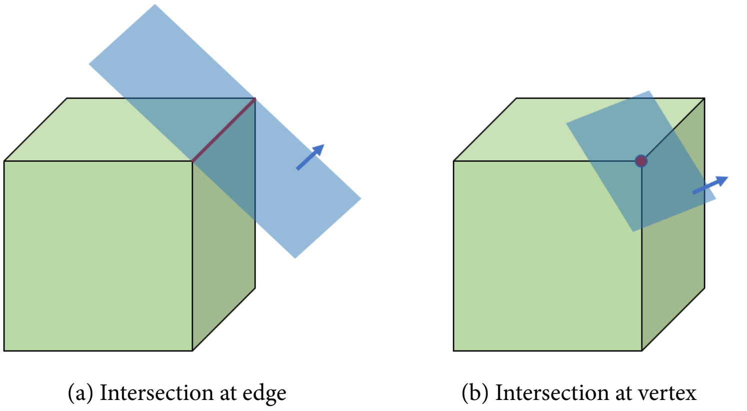


Figure 4.5: Exact representation by rational numbers. The intersection's position $(2/3, 1/3, 0)$ cannot be precisely described with floating-point arithmetic.

The floating-point coordinate space $\mathbb{F}^3$ does not retain the exactness. For example, the intersection between line segments shown in Figure 4.5 lies at position $(2/3, 1/3, 0)$ which cannot be precisely described with floating-point arithmetic. The exact rational space $\mathbb{Q}^3$ contains the floating-point space as a subset: $\mathbb{F}^3 \in \mathbb{Q}^3$. Therefore, given the input planar primitives with floating-point vertex positions, we can losslessly cast them to the exact space. Given the output building's surfaces, we can still cast them into floating-point coordinate space, but with a compromised precision.


(a) Intersection at edge

(b) Intersection at vertex

Figure 4.6: Example of degenerate geometry due to floating-point arithmetic. A half-space may intersect with an existing polyhedron at its edge (a) or vertex (b). Both have degenerate volumetric configurations.

The rational-based exact representation effectively avoids degenerate cases that may otherwise cause inconsistencies in the geometry. For example, as illustrated in Figure 4.6, empty polyhedron, possibly formed by supporting planes intersecting at one vertex or one plane in adaptive binary space partitioning, can be easily identified and further eliminated when the exact representation is used.

4.4 Libraries and software used within this thesis

The main libraries used in this thesis are listed as follows:

- **SageMath** [The Sage Developers, 2021] is a mathematics software system with features covering algebra, combinatorics, graph theory, numerical analysis, number theory, calculus and statistics. It is used for implementing the adaptive space partitioning with its class *Polyhedra*. It also provides the *ring* of rational numbers which facilitates numerical robustness.
- **PyTorch** [Paszke et al., 2019] is an optimised tensor library for deep learning. The neural network architecture for occupancy learning in function space is built with PyTorch.
- **NetworkX** [Hagberg et al., 2008] is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks. It supports the graph structure dynamically constructed during binary space partitioning, and provides an efficient implementation for computing the node partition of a minimum $s-t$ cut based on the max-flow min-cut theorem [Boykov and Funka-Lea, 2006].
- **Blensor** [Gschwandtner et al., 2011] is a simulation package binding with Blender for LiDAR and Kinect sensors. It is used for simulating the point clouds from the Helsinki city models.
- **Easy3D** [Nan, 2018] is a lightweight library for 3D modelling, geometry processing, and rendering. It is used for planar primitive detection from point clouds, and visualisation of the 3D geometry including points and surface meshes.

More libraries are used under the hood, such as *scipy* and *Qhull* for convex hull calculation, *sklearn* for PCA, *rtree* for spatial indexing, etc.

All methods in comparison within this thesis are either authors' implementations or from *CGAL*[3] as openly available software, including Poisson reconstruction, Points2Surf, PolyFit, Manhattan-world reconstruction, 2.5D Dual Contouring, QEM, SAMD and VSA. The non-commercial SCIP[4] solver is used for PolyFit.

[3] https://www.cgal.org/

[4] https://www.scipopt.org/

5 Results and discussion

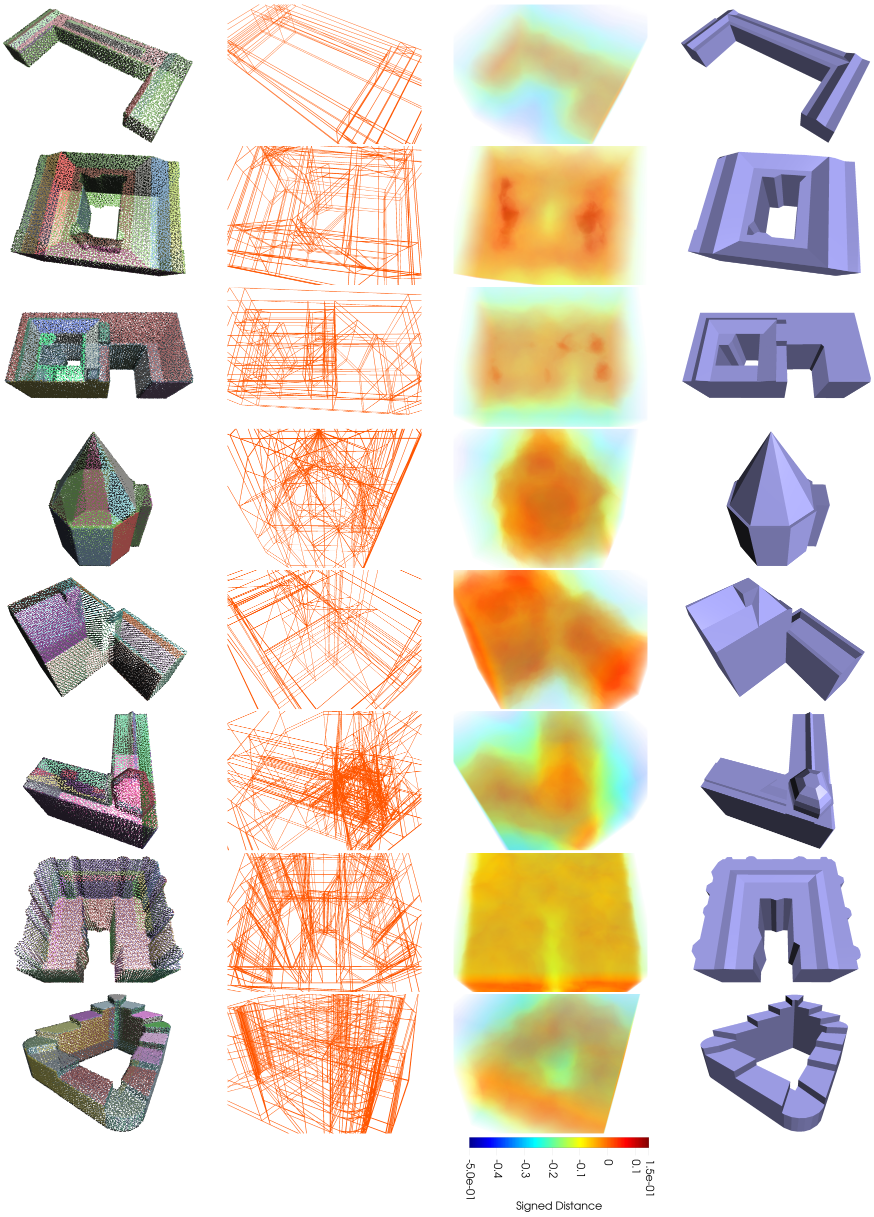


Figure 5.1: Reconstructions from *Helsinki full-view* point clouds. From left to right: input point cloud (coloured randomly per primitive), wireframe of candidate polyhedra, volume rendering of SDF, and reconstructed building model. Best viewed digitally.

This chapter presents the experimental results. Reconstruction results on several datasets are shown in Section 5.1. Section 5.2 extensively evaluates the proposed method in comparison with state-of-the-art methods. In Section 5.3 and Section 5.4 we delineate the limitations and the application scope respectively.

5.1 Reconstruction results

With the neural network trained on *Helsinki full-view* point clouds, the proposed method can reconstruct buildings of various architectural styles. Figure 5.1 presents a few reconstruction results from *Helsinki full-view* data. The SDF can accurately describe occupancy of the buildings even though their styles may be distinct from the data that the neural network is trained on.

Table 5.1 reveals the reconstruction error both visually and quantitatively. Most errors occur where subtle structures exist, due to the abstraction from planar primitive detection and the regularisation imposed to surface extraction. Nonetheless, all building surfaces can be effectively retrieved from the point clouds with plausible visual quality and SMH distance less than 0.3%.

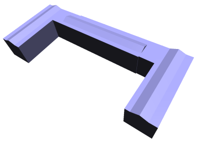
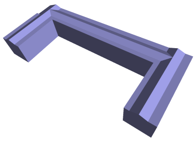
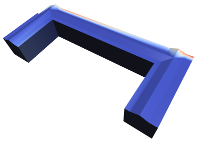
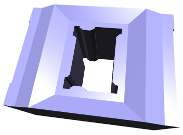
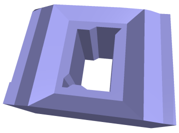
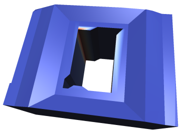
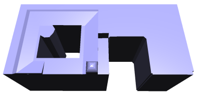
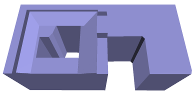
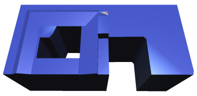
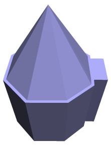
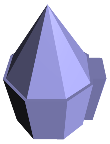
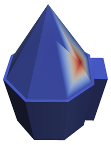
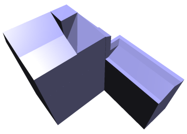
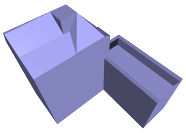
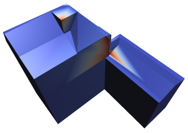
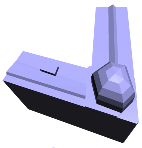
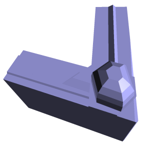
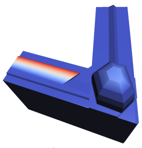
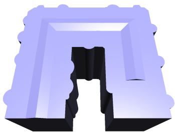
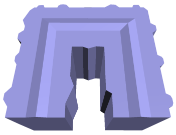
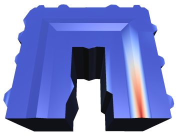
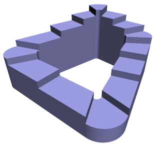
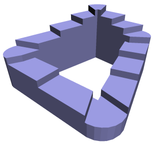
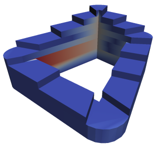
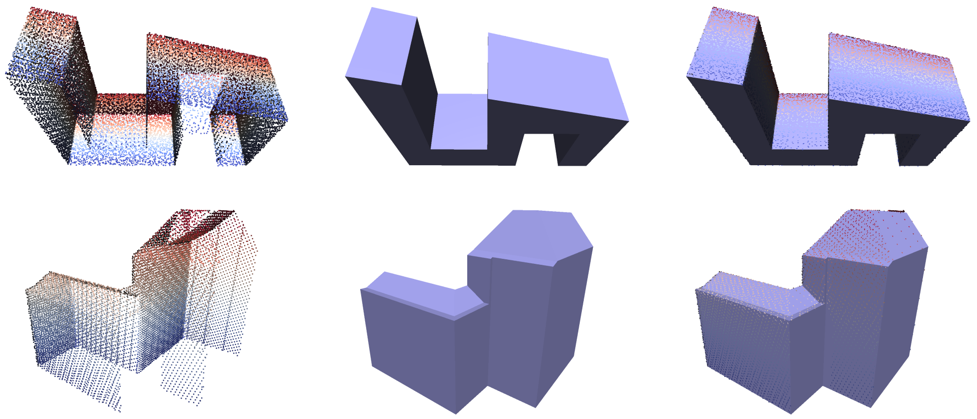
Index	Reference	Reconstructed	Error Map	$Dist_{SMH}$ (%)
#1				0.04541
#2				0.04440
#3				0.11673
#4				0.22028
#5				0.21887
#6				0.08891
#7				0.14784
#8				0.02611

Table 5.1: Error analysis on *Helsinki full-view* data. $Dist_{SMH}$ represents SMH distance.

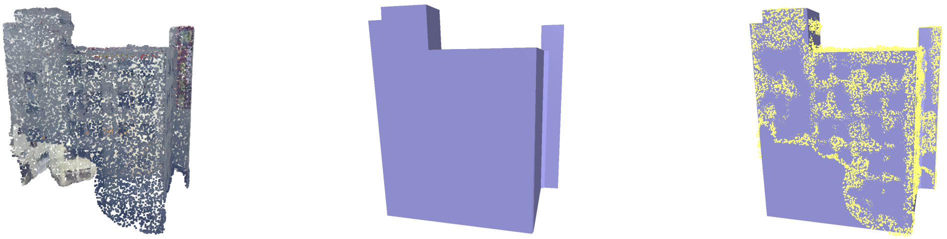
The global information estimating the sign of the SDF ensures the shape prior is preserved even when the local point samples are contaminated, e.g., missing the bottom view from the measurements. To demonstrate the capacity of our approach on these incomplete measurements, we additionally train the neural network then evaluate on *Helsinki no-bottom* point clouds. As shown in Figure 5.2, our method can still correctly infer the occupancy of the entire buildings, resulting in complete reconstruction. Unlike PolyFit [Nan and Wonka, 2017], we make no assumption that the model is closed by the detected primitives. Instead, with appended facets of the AABB, the candidate polyhedra embedding is always closed. Therefore, complete facades can be reconstructed though barely associated with points. The learnable strong prior for SDF estimation enables our method to adapt to possibly deficient scans of other kinds, e.g., from terrestrial laser scanner (TLS) where the roof structure may be invisible.



(a) Point cloud (b) Surface (c) Overlay

Figure 5.2: Reconstructions from *Helsinki no-bottom* point clouds. Our method can reconstruct a closed building model even given insufficient scans on the bottom or facades. Point clouds coloured by height.

Since the deep implicit field trained on *Helsinki no-bottom* data takes advantage of both local geometry and the general shape prior characterising point clouds of urban buildings, our reconstruction can reasonably generalise from synthetic data to real-world scans. Figure 5.4 shows the reconstruction results from *Shenzhen* point clouds directly using the neural network trained on *Helsinki no-bottom* data. The inferred distance fields still conform to the real-world point clouds of styles unseen by the network during training. Though lower parts of the buildings are insufficiently measured due to occlusion and unfavourable lighting conditions, our method can still successfully reconstruct the entire building (see Figure 5.3).



(a) Point cloud (b) Surface (c) Overlay

Figure 5.3: Complete reconstruction from insufficient scans of *Shenzhen* data.

Table 5.2 presents the error analysis on *Shenzhen* data. Since no ground truth surface is available, the error is measured asymmetrically from the reconstructed surface to the input point cloud as a reference. Therefore, the most prominent error lies where the measurement is missing, because no proximate reference exists for the correctly reconstructed surface at lower part of the building. In addition, the intricate roof structures occasionally introduce tangible errors when approximated with piecewise-planar abstraction. For instance, the protuberance on top of building #4 of *Shenzhen* data is approximated with a superfluous cuboid. Building #6 of *Shenzhen* data contains a nested structure which causes ambiguity in interior/exterior classification.

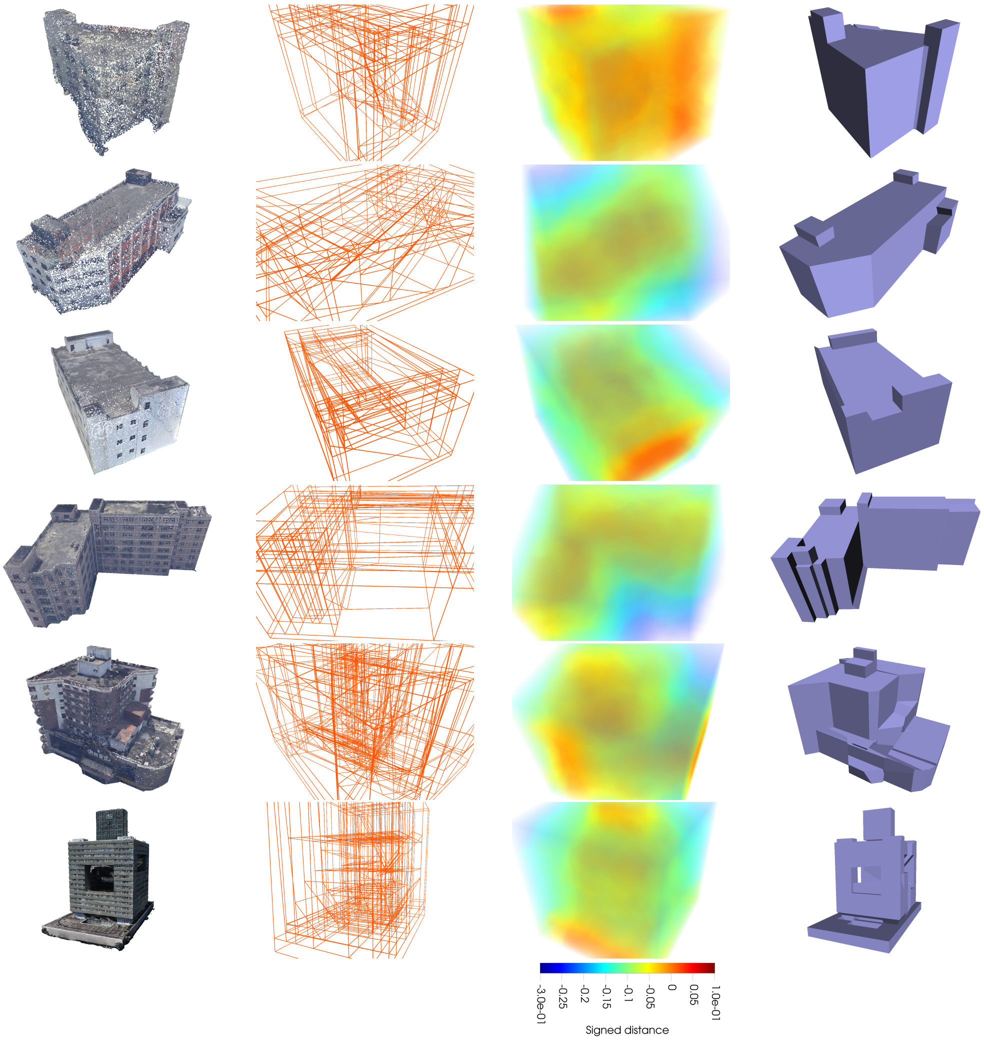


Figure 5.4: Reconstructions from *Shenzhen* point clouds. From left to right: input point cloud (with RGB), wireframe of candidate polyhedra, volume rendering of SDF, and reconstructed building model. Best viewed digitally.

5.2 Evaluation

In this section, we analyse the proposed method on various metrics over fidelity, complexity, robustness and computational efficiency, in comparison with state-of-the-art methods in smooth surface reconstruction, piecewise-planar shape reconstruction, surface approximation and geometry simplification.

5.2.1 Fidelity and complexity

Figure 5.5 compares the surfaces reconstructed by our method and two methods for smooth surface reconstruction, namely Poisson reconstruction [Kazhdan et al., 2006] and Points2Surf [Erler et al., 2020]. The former is considered the golden standard for non-data-driven surface reconstruction, while the latter is a recent learning-based approach that partly constitutes the occupancy learning in our framework. Unlike our approach describing a surface with

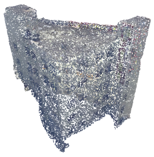
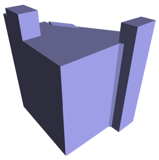
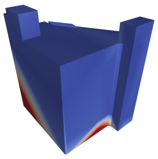
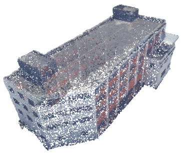
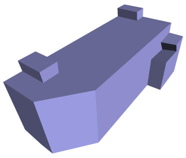
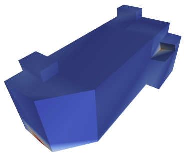
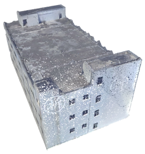
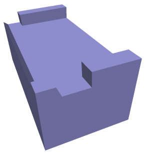
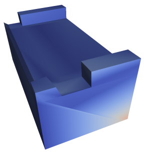
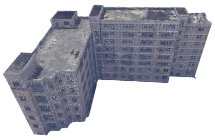
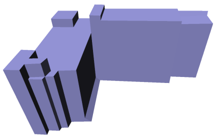
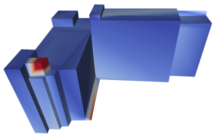
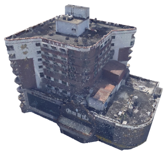
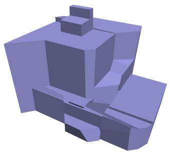
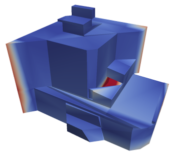
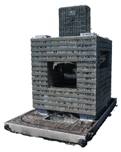
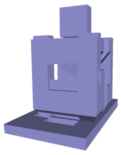
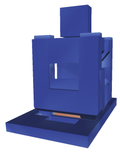
Index	Reference	Reconstructed	Error Map	$Dist_{SR}$ (%)
#1				3.8815
#2				2.0045
#3				1.1059
#4				1.5759
#5				2.5830
#6				3.3763

Table 5.2: Error analysis on *Shenzhen* data. $Dist_{SR}$ represents Hausdorff distance from surface to reference.

piecewise planarity, both these two methods produce a massive number of triangle facets. This not only leads to extra memory consumption, but also potentially results in non-watertight surfaces.

Figure 5.6 compares our results on *Helsinki full-view* and *Shenzhen* with PolyFit [Nan and Wonka, 2017], Manhattan-world reconstruction [Li et al., 2016b] and 2.5D Dual Contouring [Zhou and Neumann, 2010]. Neither Manhattan-world reconstruction nor 2.5D Dual Contouring can deliver high-quality building models. The former constrains facets to be axis-aligned thus cannot faithfully represent those of arbitrary orientation, while the latter produces a prohibitive number of facets that do not properly address the piecewise planarity of urban buildings but with undesirable discontinuity. However, both Manhattan-world reconstruction and 2.5D Dual Contouring allows users to specify the ground plane through their graphical user interface (GUI) such that, for incomplete point clouds from *Shenzhen* data, the lower part of the buildings can be recovered.

Among these methods, PolyFit is most comparable with ours for generating compact piecewise-planar building surfaces. In fact, both PolyFit and our method adopt the *hypothesizing-and-selection* strategy: we both tessellate the ambient space into a candidate set with detected planar primitives, and then seek a proper arrangement of the candidates with combinatorial analysis. However, the optimisation goal in PolyFit is hardcoded with proper weights distributed for model-fitting, coverage and complexity. Instead, with our approach, the optimisation is essentially guided by the implicit field estimated from the neural network with regularization imposed by the MRF.

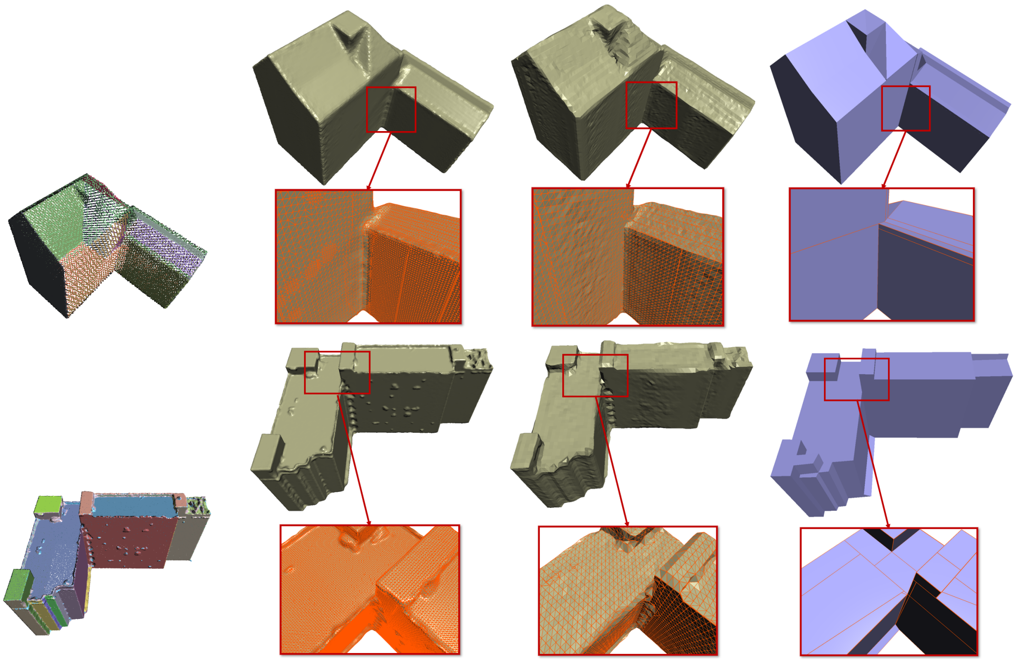


Figure 5.5: Comparison with smooth surface reconstruction. From left to right: point cloud (coloured randomly per primitive), Poisson reconstruction, Points2Surf, and ours.

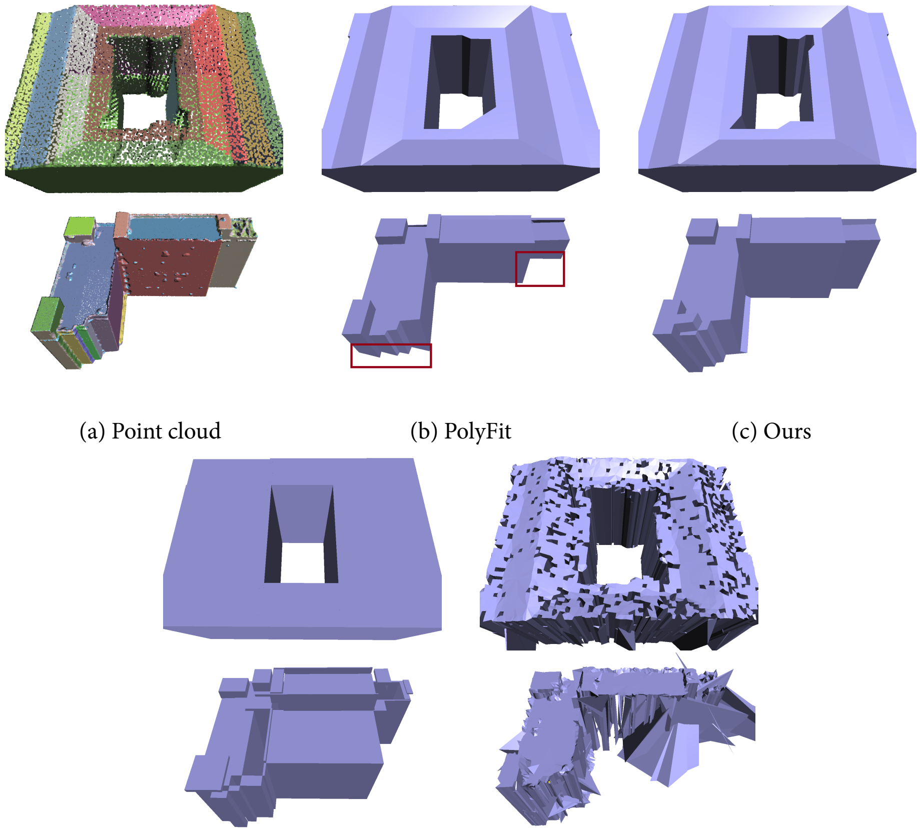
Since PolyFit assumes all necessary planes are provided to form a closed surface, the substance of a building is not guaranteed to be recovered when a pertinent planar primitive fails to be extracted, which, unfortunately, happens to most real-world buildings scans where the ground plane is missed out. For instance, PolyFit fails to recover the lower part of the building from the *Shenzhen* point clouds, as shown in Figure 5.4. Instead, we relax this assumption by initialising with a bounding box of the point cloud for cell complex construction, which effectively guarantees complete reconstructions from incomplete measurements.

	Ours				**PolyFit**			
Index	**Fidelity**			**Complexity**	**Fidelity**			**Complexity**
	$Dist_{SR}$	$Dist_{RS}$	$Dist_{SMH}$	#Facets	$Dist_{SR}$	$Dist_{RS}$	$Dist_{SMH}$	#Facets
#1	0.01858	0.04541	0.04541	102	0.01758	0.04543	0.04543	158
#2	0.03438	0.04441	0.04441	117	0.15216	0.08322	0.15216	643
#3	0.11674	0.06079	0.11673	155	0.02774	0.06174	0.06174	816
#4	0.02369	0.22029	0.22029	112	0.02125	0.22144	0.22144	180
#5	0.08391	0.21888	0.21888	108	0.02937	0.22151	0.22151	271
#6	0.02372	0.08892	0.08892	162	0.02506	0.09165	0.09165	1278
#7	0.02940	0.14785	0.14785	312	0.03461	0.14145	0.14145	1919
#8	0.02612	0.02368	0.02612	203	0.02038	0.02182	0.02182	849

Table 5.3: Evaluation of fidelity and complexity. $Dist_{SR}$, $Dist_{RS}$ and $Dist_{SMH}$ represent Hausdorff distance from surface to reference, from reference to surface and SMH distance, respectively.

Table 5.3 further quantitatively compares the fidelity and complexity of the *Helsinki full-view* building models generated by our method and by PolyFit. The comparable SMH distances indicate no significant fidelity advantage from either method. However, by overlaying the point cloud on the reconstructed surface in Figure 5.7, we notice, with default complexity configuration[1], PolyFit tends to output a more regularised surface while ours conforms more

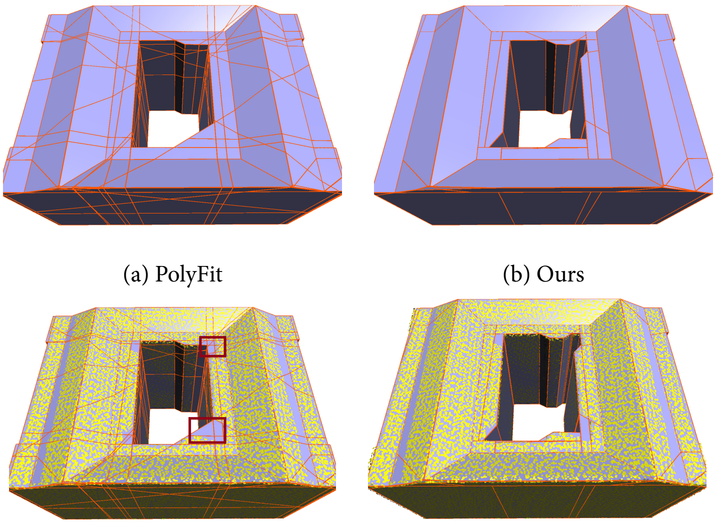
[1] 0.46, 0.27, and 0.27 as weights for model-fitting, coverage, and complexity in PolyFit, respectively; $\lambda = 0.001$ for complexity in ours.



(a) Point cloud (b) PolyFit (c) Ours

(d) Manhattan-world (e) 2.5D DC

Figure 5.6: Comparison with piecewise-planar reconstruction methods from both *Helsinki full-view* (top) data and *Shenzhen* (bottom) data.



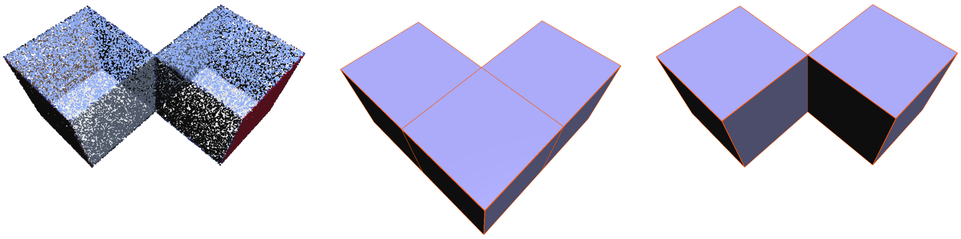
(a) PolyFit (b) Ours

(c) PolyFit overlay (d) Ours overlay

Figure 5.7: Comparison with PolyFit. PolyFit outputs a more regularised surface with more facets, while ours conforms more to the geometric detail exhibited in the building with fewer facets.

to the geometric detail exhibited in the data. As in the scope of urban building reconstruction, there is always a trade-off between the fidelity and complexity of reconstructed building models.

Compared with exhaustive partitioning adopted by PolyFit, our adaptive strategy can produce building surfaces with lower complexity, measured by the number of facets on the reconstructed surface, as shown qualitatively in Figure 5.7 and quantitatively in Table 5.3. Note that this measure of complexity may not fully comply with the visual compactness of a building model but only for data structure, because multiple adjacent coplanar facets are of the same visual complexity as one combined. Through polygonisation as post-processing, the number of facets generated by PolyFit can be reduced. Nevertheless, our reconstruction method has the advantage of directly generating compact building models. Moreover, we argue the adaptive partitioning strategy cannot be plugged into PolyFit since it would otherwise break the manifold precondition asserted in PolyFit.



(a) Point cloud (b) PolyFit (c) Ours

Figure 5.8: A non-manifold example. For the input point cloud (a), our method can faithfully reconstruct the surface (b). PolyFit is under its manifold constraint and includes an impertinent space in the reconstruction (c).

Through evaluation on synthetic point clouds, we observe the non-manifold structure (see Figure 3.15b) can significantly affect the surface reconstruction, as exemplified by Figure 5.8. Since our method constrains only watertightness but not manifoldness, the reconstructed surface respects faithfully the geometry of the point cloud that exhibits the non-manifold structure. For PolyFit, however, an impertinent space is included in its reconstruction, as a consequence of its manifoldness constraint. The non-manifold structure is expected in urban building models, for which our method applies while PolyFit does not.

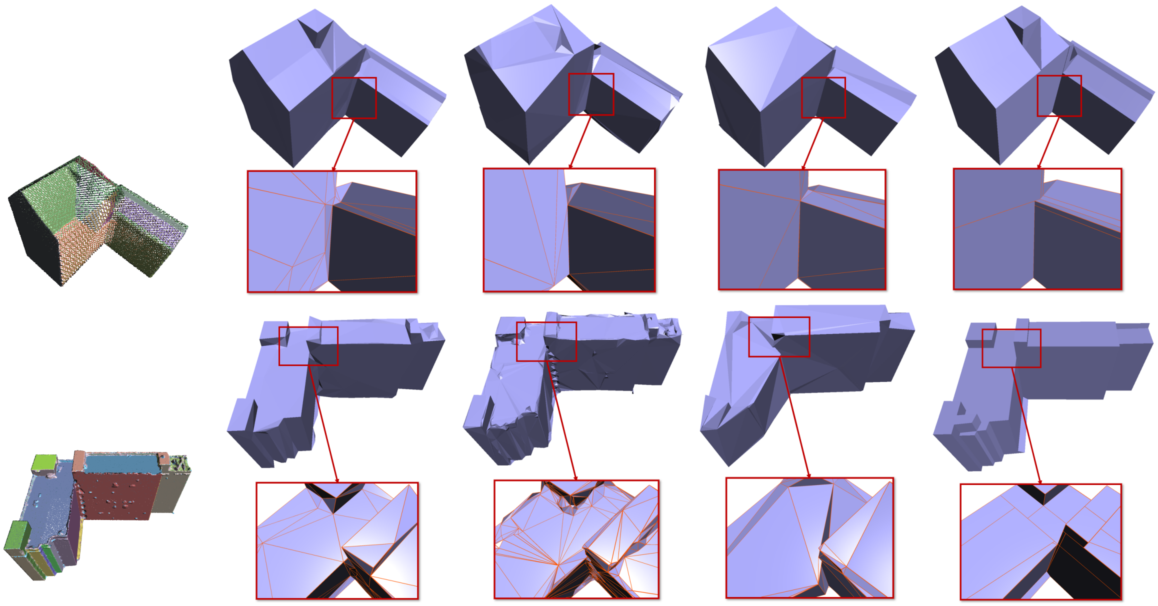


Figure 5.9: Comparison with surface approximation methods. From left to right: point cloud (coloured randomly per primitive), QEM, SAMD, VSA, and ours.

In addition to the aforementioned reconstruction methods, we further compare our method with state-of-the-art surface approximation methods, namely QEM [Garland and Heckbert, 1997], SAMD [Salinas et al., 2015] and VSA [Cohen-Steiner et al., 2004], which all can approach a compact surface by simplifying a smooth surface comprised of dense triangles, i.e., the dense triangular surfaces from smooth surface reconstruction serves as input for QEM, SAMD and VSA.

We set up the expected number of vertices after approximation equal to ours and compare the quality of the generated building surfaces with comparable complexity. As shown in Figure 5.9, our reconstruction method can produce surfaces of significantly more piecewise planarity, with sharp edges kept, compared with QEM, SAMD and VSA. The quadric error metrics in QEM contract edges without attention onto planar structures thus generate fractured surfaces. However, its output is the closest to our reconstruction among the three approximation methods in comparison. SAMD, though claiming structure-aware, struggles to deliver building models with compact surfaces, partially because of its strong compliance to the detected primitives, which, when given in low quality, limits its compactness on the contrary. VSA produces surfaces with the lowest quality given the same complexity. Notice QEM, SAMD and VSA all output surfaces with triangles instead of arbitrary-sided polygons as with our method.

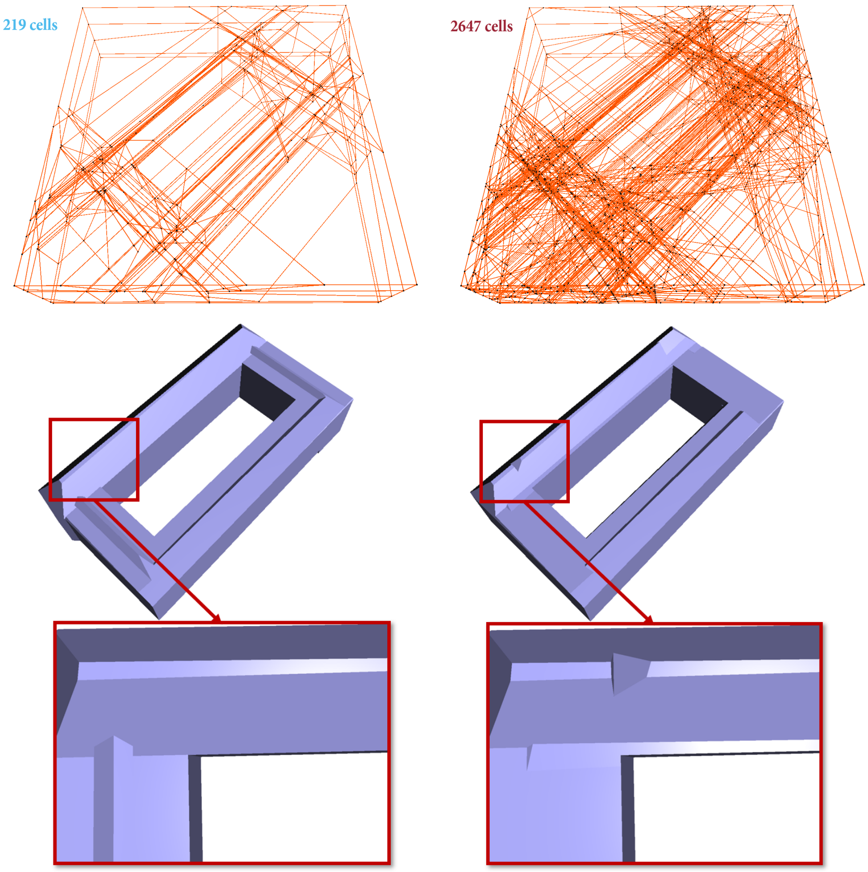


Figure 5.10: Adaptive partitioning (left) versus exhaustive partitioning (right). Adaptive space partitioning avoids over-tessellation and the subsequent 'caved' artefact.

The adaptive strategy in our approach effectively avoids over-tessellation and mitigates the subsequent 'caved' artefact, as shown in Figure 5.10. Exhaustive partitioning, on the contrary, produces numerous small-sized candidate polyhedra, especially near the building's surface. These unnecessary candidates are more likely to be misclassified than those with larger volumes generated by adaptive partitioning. Once an interior candidate is classified as outside the building instance, a 'caved' surface appears, and vice versa.

5.2.2 Computational efficiency

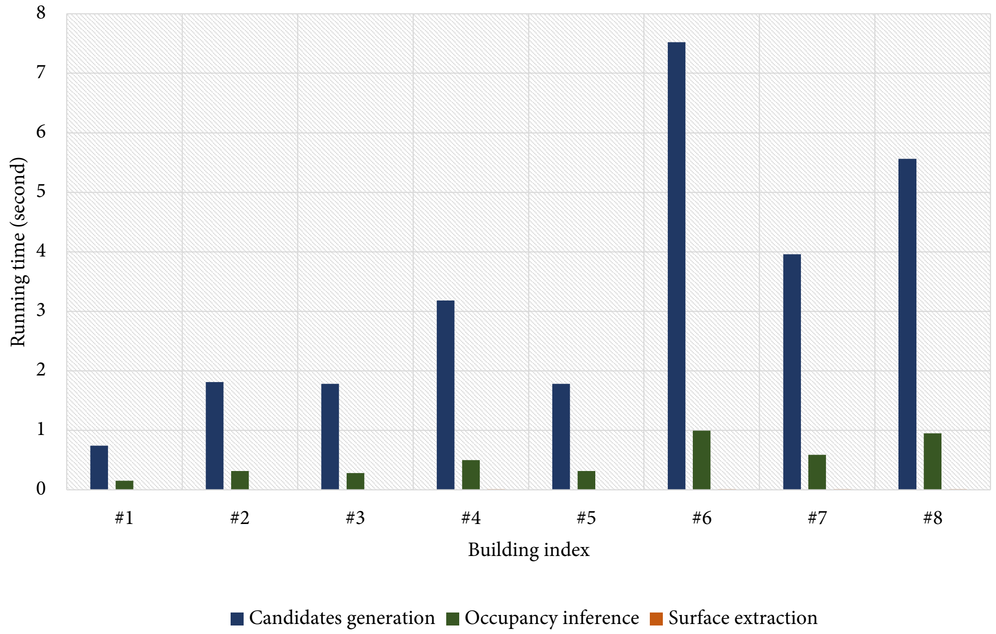


Figure 5.11: Distribution of execution time. Surface extraction consumes no more than 0.01 second for all of the eight buildings thus barely visible as bars.

Figure 5.11 presents the execution time distribution for each component of our method. Most running time is due to the creation of cell complex where the adjacency graph of polyhedra is dynamically obtained. The adoption of exact arithmetic slows computation, while being critical for the numerical robustness of our approach. For all the eight buildings shown in Figure 5.1, the surface extraction with our graph-cut solver consumes no more than 0.01 second, which is significantly faster than that of PolyFit with its integer programming solver. Inference of SDF is done by the neural network on a GPU in a batch manner, i.e., computations are parallelised across different query points and even across different objects. In our experiment, approximately one thousand query points can be processed per second, obtaining the aggregated signed distances of the same amount of polyhedra instantly.

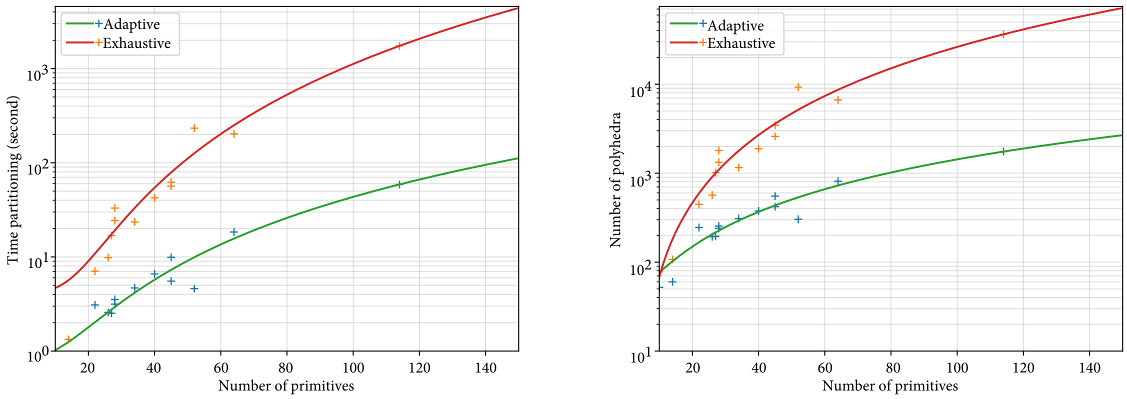


Figure 5.12: Efficiency of adaptive partitioning

We evaluate the efficiency of our adaptive binary space partitioning with exhaustive partitioning in comparison. Figure 5.12 shows the evolution of their construction time and complexity in function of the number of planar primitives. Adaptive space partitioning drastically reduces the computations for cell complex creation, yet generating much fewer polyhedra which in turn speeds up the follow-up occupancy inference and surface extraction procedures. With exhaustive partitioning by pairwise intersections, a massive number of candidate polyhedra are indiscriminately produced regardless of their spatial subordination; the partitioning time increases accordingly. The excessive amount of candidates not only hinders computation, but incline to defective surface on subtle structures where wrong labels are more likely to be assigned. Instead, the adaptive strategy avoids redundant partitioning thus is able to produce compact surfaces efficiently.

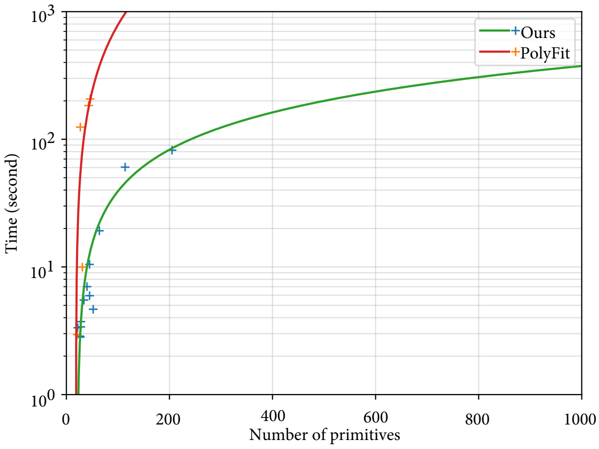
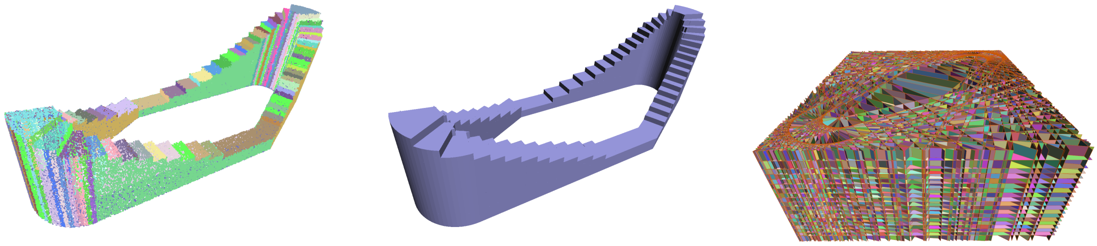


Figure 5.13: Scalability comparison with PolyFit

We further compare the scalability of our method with that of PolyFit in Figure 5.13. With pairwise intersection from a large number of planar primitives, PolyFit first generates a prohibitive number of candidate facets, from which an optimal combination is to be obtained by its integer programming solver. In our experiment, PolyFit struggles to reconstruct the surface when the number of primitives exceeds 100 and the number of polyhedra exceeds 20,000 as a consequence. For complex building models, PolyFit may even take days for solving its integer programming problem, only if solvable. Its solver may also fail with memory complaints when the scale of the candidate set exceeds capacity due to computationally intensive global optimisation. Instead, our method can process more than one thousand primitives at least one order of magnitude more efficiently by adopting the efficient graph-cut solver and minimising the candidate set with adaptive binary space partitioning. For instance, as shown in Figure 5.14, our method can produce a high-quality surface for a complex building with 260 detected primitives within 100 seconds, with only 0.13 second spent on solving the combinatorial optimisation problem, while PolyFit fails to solve its integer programming problem.



(a) Point cloud

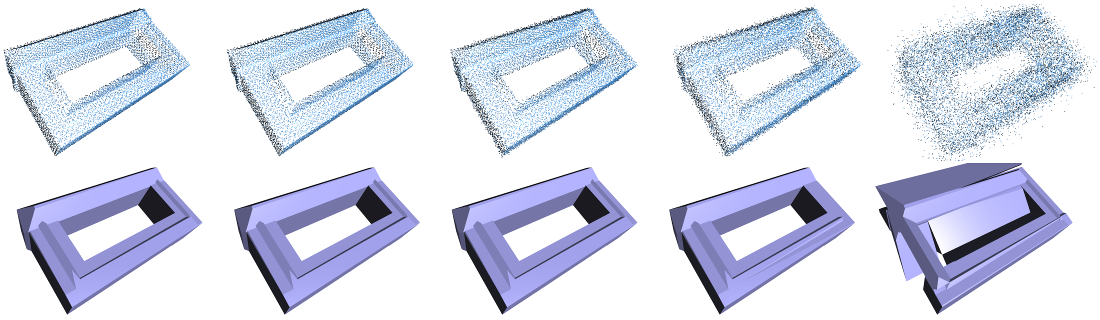
(b) Ours

(c) PolyFit

Figure 5.14: Complex building reconstruction. Our method can produce complex building model (b) from point cloud with 260 primitives (a), while PolyFit fails to solve its integer programming problem from massive candidate facets (c).

5.2.3 Robustness to noise and incomplete input

Since our deep implicit field is trained intentionally on the synthetic point clouds with various levels of Gaussian noise in between $U[0, 0.005R]$, the proposed method is robust to Gaussian noise within a reasonable range, as long as planar primitives can be accurately retrieved from the point clouds. Figure 5.15 shows the reconstruction results on input point clouds with various Gaussian noise ranging from $0R$ to $0.050R$. Our method can produce reasonable reconstruction until the noise reaches $0.010R$, which, for instance, may indicate a measurement error of as high as 1 meter against an 100 meter-sided building; this level of noise is already beyond a practical tolerance. A prohibitive level of noise larger than what the neural network is trained on, however, may hinder the occupancy estimation and eventually deteriorate the reconstruction. The reconstruction may still degrade due to poor explicit geometry from inaccurate primitive detection, even though the deep implicit field can accurately estimate the interior/exterior occupancy. Nonetheless, our refinement on detected primitives and the MRF formulation inclined to compact surfaces are designed to mitigate this inaccuracy.



(a) $0R$ (b) $0.001R$ (c) $0.005R$ (d) $0.010R$ (e) $0.050R$

Figure 5.15: Robustness to noise. Our neural network is trained with noise of range $U[0, 0.005R]$, while it reasonably extrapolates to until noise of $0.01R$.

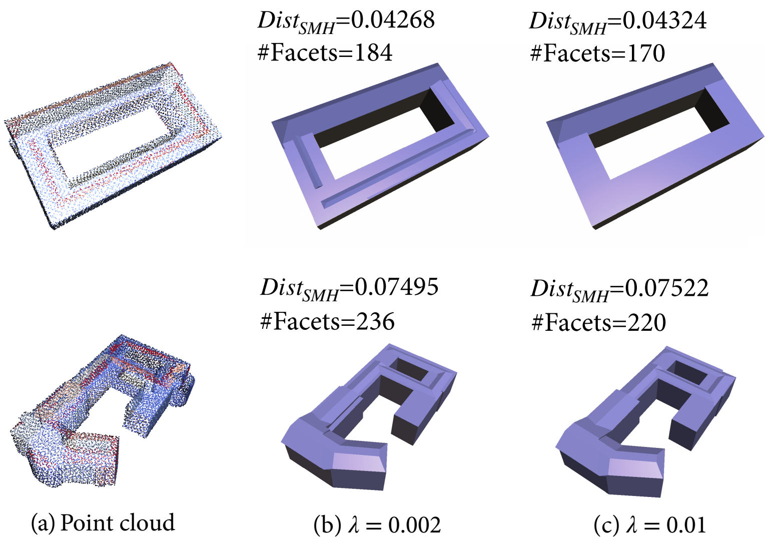
Moreover, since the neural network takes the global shape prior to complement the local geometry, the proposed method is robust to incomplete input as well, such as *Helsinki no-bottom* point clouds shown in Figure 5.2, and *Shenzhen* data shown in Figure 5.4. Initialising with bounding facets also contributes to a complete reconstruction from insufficient scans. However, the global prior is data-dependent: the learnt prior remains effective only when applied to point clouds with identical general characteristics as those of training data. For instance, *Helsinki no-bottom* and *Shenzhen* both lack measurements on their grounds, therefore the learnt SDF from the former applies to the latter.

5.2.4 Effect of parameter λ

λ in Equation 3.11 weights the complexity term formulated in the MRF for surface extraction from the cell complex. It controls the complexity of the output building surface: increasing λ leads to a more compact surface with fewer facets. However, a high λ results in shrinking of the surface where the geometric fidelity may deteriorate. Figure 5.16 illustrates the evolution of complexity and fidelity with respect to the choice of λ value. A high λ value may result in undesired shrinking of the surface model, as shown in Figure 5.17. In the experiment, λ is typically set to 0.001. We argue the limited choice of λ is a worthy compromise for its significant computational efficiency.

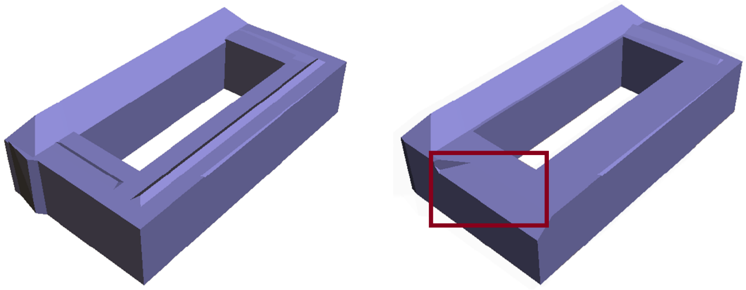
5.3 Limitations

Two main limitations exist within this research:



(a) Point cloud (b) $\lambda = 0.002$ (c) $\lambda = 0.01$

Figure 5.16: Impact of parameter λ. Increasing λ leads to a more compact surface with fewer facets, but meanwhile with larger geometric error.



(a) $\lambda = 0$ (b) $\lambda = 0.05$

Figure 5.17: A high λ value results in shrinking of the surface

Primitive detection. With *hypothesizing-and-selection*, our framework focuses on a proper assembly of planar primitives into piecewise-planar building surfaces, not on detecting them. An assumption is made that planar primitives can be accurately detected from inputs points, which is not always possible from contaminated or insufficient point clouds. Nevertheless, a few steps are designed to mitigate the inaccuracy from primitive detection, including primitive refinement and the complexity term in the MRF formulation. With accurately detected primitives, our method can produce high-quality building models. When the primitives are of low quality, however, the resulting building models may lose geometric detail, as shown in Figure 5.19. The piecewise-planar abstraction also leaves out other types of primitives that exist in real-world buildings, such as curved structures. Figure 5.18 shows how a curved surface is approximated with our assumption.

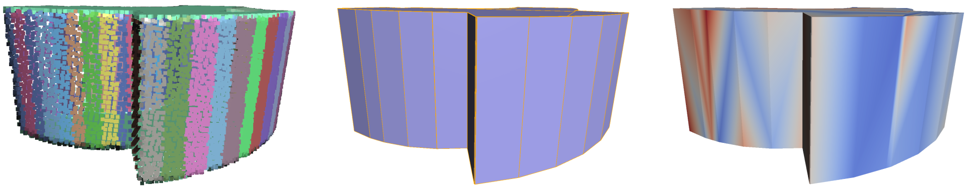


Figure 5.18: An example of non-planar buildings and how it is approximated

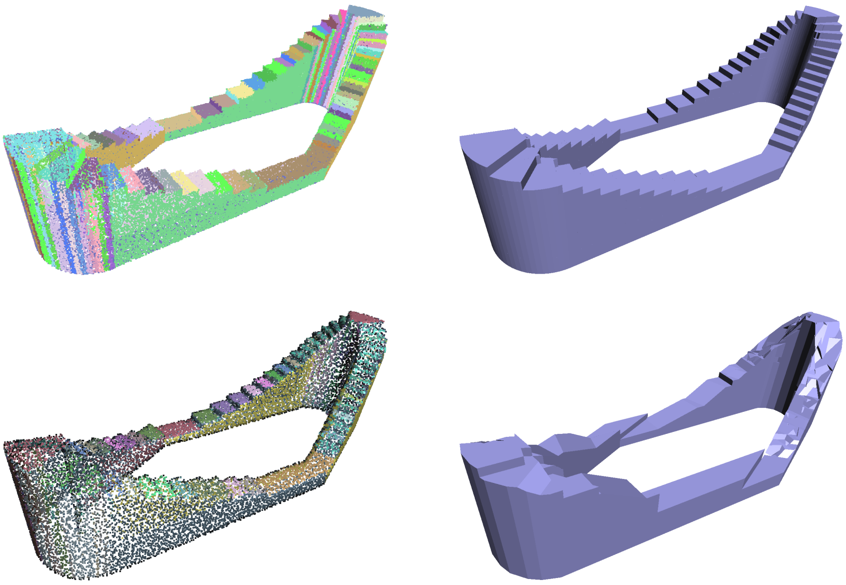


Figure 5.19: Quality of planar primitives impacts reconstruction. Our method can reconstruct high-quality building models from accurately detected primitives (top), but results in low-quality models from inaccurate detection (bottom).

The 'caved' artefact. Though the complexity term in our MRF formulation penalises irregularity on the surface, the 'caved' artefact can remain because a shrunk surface would have a smaller area which is favoured towards the goal of the optimisation (see Figure 5.20). Due to the formulation of MRF that only composes of the unary potential (i.e., energy defined on each polyhedron) and the pairwise potential (i.e., energy defined between polyhedra pairs), no global constraints can be imposed such as the total number of facets defined by Nan and Wonka [2017]. The 'caved' artefact can be mitigated with a carefully chosen λ value.

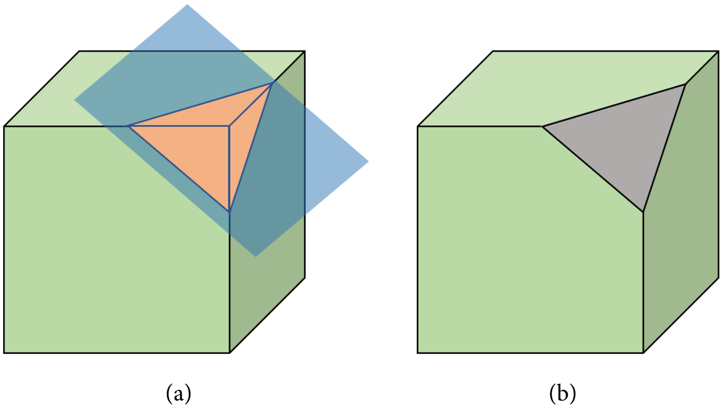


Figure 5.20: Formulation of the 'caved' artefact: area of a 'caved' surface (b) is smaller than a complete one (a).

5.4 Application scope

The proposed method is capable of reconstructing high-quality 3D building models from point clouds effectively and efficiently. The reconstructed building models can be employed in various downstream applications such as building information modelling (BIM), solar potential assessment, environmental simulation, etc (see Figure 1.1).

A by-product of our adaptive binary space partitioning is the spatial tessellation of a building, which may represent different functional zones inside the building. Therefore, by extracting and aggregating subordinate information for each polyhedron in the binary tree (Figure 3.8), building components can be analysed, as illustrated in Figure 5.21. This component information can enrich the reconstructed building models with additional semantics.

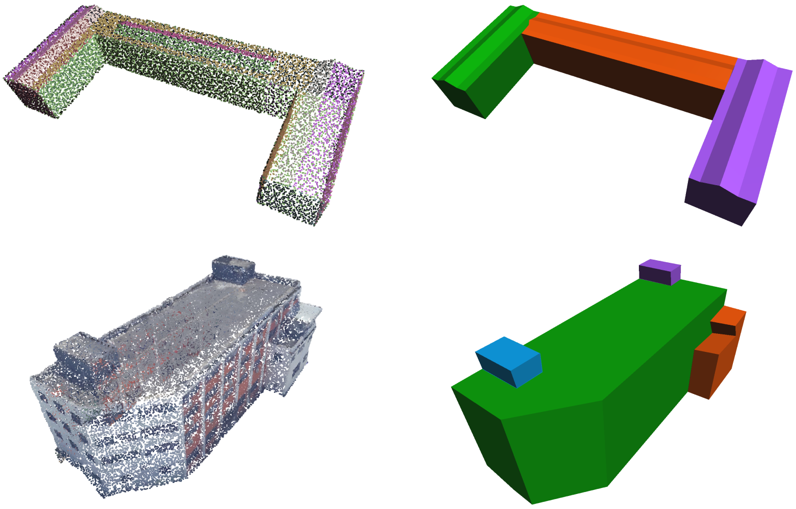


Figure 5.21: Application in building component analysis

Since we do not assert building-specific assumptions except for the priority assigned to planar primitives, our approach remains generic and can naturally be extended to free-form objects. Figure 5.22 shows the Stanford Bunny model reconstructed by our approach. Being an efficient representation for computing, storing and rendering, piecewise-planar surfaces produced by our method can represent the raw geometry with various scales. This finds applications in geometry compression, real-time rendering, physical simulations, etc.

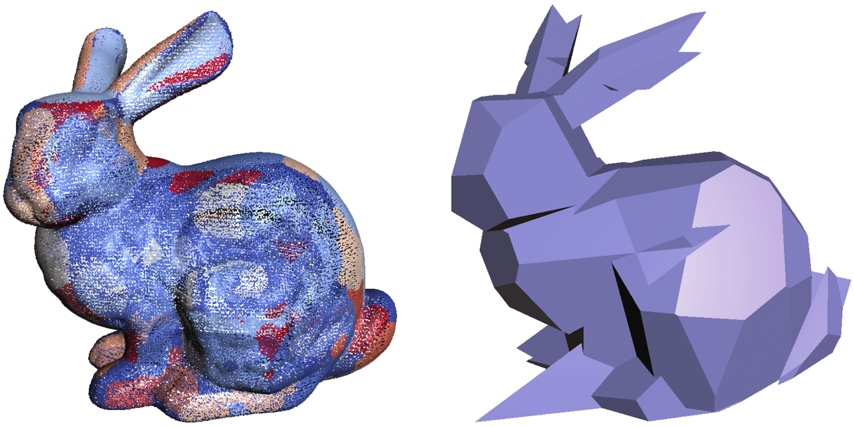


Figure 5.22: Application in free-form shape reconstruction

6 Conclusions and future work

In this chapter, the research questions of this thesis are revisited and answered with justification based on the experimental results in Section 6.1. We then highlight the main contributions of this thesis in Section 6.2, and conclude with a list of future work and an outlook in Section 6.3.

6.1 Research overview

To address the main research question on **how can deep implicit fields be used for compact building model reconstruction**, in this thesis, we present a novel framework utilising the learnt implicit representation as an occupancy indicator for explicitly constructed cell complex geometry from adaptive binary space partitioning. The indicator describes a scalar field in which the surface of a building is extracted from an MRF by an efficient graph-cut solver. With our neural-guided strategy, we demonstrate that high-quality building models can be obtained with significant advantages over fidelity, compactness and computational efficiency. To the best of our knowledge, this is the first work where a deep implicit field is explored for building reconstruction.

The sub-questions are addressed as follows:

- **How to incorporate a neural network architecture that leverages an implicit field?**

  A neural network is incorporated which learns the SDF such that, for a given query point in 3D space, it estimates the distance from the point to its closest surface whose sign signifies whether the point is inside or outside of the surface. To alleviate the notorious generalisation incapacity when representing the shape with one global feature vector, we follow the Points2Surf architecture [Erler et al., 2020] that takes the neighbouring geometry around the query point into account, and factorise the SDF into the absolute distance and the sign of the distance, separately. The local geometry facilitates accurate surface representation, while the global geometry enables the expression of general shape priors.

  We incorporate the deep implicit field to the explicit geometry of candidate polyhedra, and formulate surface reconstruction as a binary labelling problem framed in an MRF, where candidate selection is essentially neural-guided by the implicit field. This hybrid strategy takes advantage of high-quality primitives generated by RANSAC and the expression capacity of the deep neural network, allowing faithful reconstruction from point clouds of various architectural styles. Since real-world point clouds on buildings—whether captured by LiDAR or produced with MVS from images—lack the bottom view, we leverage the global shape prior learnt from the synthetic *Helsinki no-bottom* point clouds and further evaluate on real-world photogrammetric *Shenzhen* point clouds. Experimental results demonstrate that the learnt implicit fields can generalise reasonably well from synthetic scans to real-world measurements.

- **How to guarantee the reconstructed surface is compact and watertight?**

  We propose adaptive space partitioning for producing a cell complex with candidate polyhedra. The surface of the building is then obtained between every pair of inside polyhedron and outside polyhedron whose confidence is inferred by the neural network. Since the generated complex is a valid polyhedral embedding, the surface is guaranteed to be watertight. The output geometry can still be non-manifold, which we argue is desired for representing real-world buildings of non-manifoldness.

With our adaptive strategy that minimises the intersections while respecting the spatial layout, redundant candidate polyhedra resulting in unnecessary complexity can be avoided. Compared with exhaustive partitioning, our adaptive strategy drastically reduces the computations for cell complex creation, yet generating much fewer polyhedra which in turn speeds up the follow-up occupancy inference and surface extraction procedures. In addition, a complexity regularisation is imposed in formulation of the MRF, by the area of facets. Therefore a compact and watertight B-rep of the building's surface can be obtained.

- **To what extent does the proposed method generalise across different point clouds?**

  As observed from the experimental results, our method can be applied to different point clouds with various characteristics, across architectural styles, sensor's viewpoints, and even from synthetic scans to real-world measurements off-the-shelf. This strong generalisation comes from two aspects of our hybrid strategy. The one is that high-quality explicit geometry can be constructed regardless of the high-level shape information that is often data-dependent. The other one is the aggregation of local and global information in the implicit function, which allows both general shape priors and detailed geometry to be expressed, enforcing the neural network to learn robust local features in the point cloud.

  The trained deep implicit field inevitably fits more to the training data, and possibly deviates from other point clouds with completely unseen characteristics. For instance, For instance, *Helsinki no-bottom* and *Shenzhen* both lack measurements on their grounds; therefore the learnt SDF from the former applies to the latter. Directly using the implicit function trained on *Helsinki full-view* data for evaluation on *Helsinki no-bottom* data would fail completely, because their shape priors fundamentally differ from each other.

- **How sensitive is the method regarding contaminated (e.g., noisy) and incomplete point clouds?**

  As observed from the experimental results, our method is robust to noise and incomplete point clouds within a reasonable range. The robustness of occupancy estimation stems from the training data in which artefacts are intentionally coined to enhance the robustness of the neural network. The robustness of explicit geometry comes from several steps towards generating a high-quality candidate set of polyhedra, including incorporation of the point cloud's AABB, planar primitive refinement, voting representatives and the complexity term in the MRF formulation for surface extraction.

- **What are the (dis)advantages of the proposed method compared with state-of-the-art methods in terms of geometry accuracy and topology validity?**

  We extensively compare our method with Poisson reconstruction [Kazhdan et al., 2006] and Points2Surf [Erler et al., 2020] for smooth surface reconstruction, with PolyFit [Nan and Wonka, 2017], Manhattan reconstruction [Li et al., 2016b], 2.5D Dual Contouring [Zhou and Neumann, 2010] for piecewise-planar surface reconstruction, with QEM [Garland and Heckbert, 1997], SAMD [Salinas et al., 2015] and VSA [Cohen-Steiner et al., 2004] for surface approximation. Our method can produce 3D building models that are more compact than all other methods. Among these methods in comparison, only PolyFit can generate succinct building models as close to ours. However, our method has a significant advantage over PolyFit in terms of compactness and computational efficiency. Specifically, the graph-cut solver in our MRF formulation is at least one more magnitude more efficient than the integer programming solver employed by PolyFit. This underpins the scalability of our method and allows reconstructions of complex building models.

  As a learning-based approach, features that contribute to the SDF estimation are automatically extracted by the neural network from the point cloud, unlike PolyFit whose optimisation goal is hardcoded with crafted weights distributed for model-fitting, coverage and complexity. Therefore, the effectiveness of our approach heavily depends on the training data. In addition, though we notice the manifold constraint can possibly lead to unfaithful reconstructions, manifoldness remains critical for specific applications that our method falls short of.

6.2 Contributions

In this thesis, we propose a novel framework for 3D building model reconstruction from point clouds. The prominent contributions of this work are summarised as follows:

(i) We propose a learning-based framework for compact building model reconstruction. To the best of our knowledge, this is the first work where a deep implicit field is explored for building reconstruction. Our method shows significant performance and quality advantage over state-of-the-art methods for urban building reconstruction, especially for complex building models. In addition, our method remains generic for the reconstruction of arbitrary objects besides buildings.

(ii) We design an adaptive space partitioning solution for generating a cell complex of candidate polyhedra. Compared with the exhaustive baseline, our adaptive strategy can efficiently partition the space, minimising redundant polyhedra that hinder the follow-up SDF inference and surface extraction.

(iii) We formulate the surface extraction as an MRF optimization problem where an efficient graph-cut solver extracts the building's surface with complexity regularisation. Our solver is far more efficient than the integer programming solver used in state-of-the-art methods.

(iv) We provide along with the thesis an open synthetic building point cloud dataset for cultivating learning-based applications in the built environment.

6.3 Future work

We expect the following future work as an extension of the research done within this thesis:

- **End-to-end neural network architecture**. The current pipeline comprises explicitly constructed geometry and the learnt implicit function. The former is independent of the neural network that facilitates the latter. It is possible to further incorporate the cell complex generation into the network architecture, therefore making the network training end-to-end. The crux of this incorporation is to dynamically adapt the number of primitives for each shape, where a sequence-to-sequence architecture such as a recurrent neural network (RNN) can be employed for variable-length output. Appendix C presents alternative neural network formulations for SDF learning, as well as additional experimental results.

- **Self-adaptive λ value**. A high λ value weighting the complexity term results in shrinking of the surface, therefore the value has to be carefully chosen. Although in the experiments an empirical λ value of 0.001 balances the fidelity and complexity of most buildings, a self-adaptive λ value can possibly produce higher-quality reconstruction by automatically adapt the complexity to the input point cloud.

- **Interactive reconstruction**. The use of graph-cut solver enables fast editing of segments [Boykov and Funka-Lea, 2006] by efficiently recomputing the optimal solution that satisfies additional constraints. Therefore, user interactions can be integrated in the loop to adjust the reconstruction accordingly.

- **Generic shape reconstruction with 3D Delaunay triangulation**. By substituting our adaptive space partitioning with 3D Delaunay triangulation, our method is expected to handle generic shape reconstruction with non-uniformity provided by Delaunay tessellation. Appendix B describes this alternative cell complex formulation.

- **More primitive types**. This thesis targets the ubiquitous piecewise-planar structure that dominates the geometry of urban buildings. There are other types of primitives that constitute real-world buildings and can be parameterised as well, e.g., sphere, cylinder, torus. A proper assembly with more primitive types may result in more realistic and accurate 3D building models. This requires a more generic space partitioning mechanism other than BSP.

- **Complex scenes**. Restricted by the available datasets, only LoD2 building models are addressed in this thesis. However, the proposed method—as proved feasible on free-form objects—should apply to reconstructions of other LoD off the shelf. Moreover, we target watertight building models instead of urban scenes without closed boundaries. To adapt to non-watertight scenes the cell complex no longer applies as a candidate set. Nevertheless, this requires corresponding datasets for future performance evaluation.
- **Stronger generalisation to real-world data**. As a learning-based approach, the features that contribute to an accurate occupancy estimation are automatically learnt from the training data. Therefore, our method inevitably fits more to the training data, which may not fully reflect the characteristics of unseen buildings. In our experimental setup, the neural network is trained only on synthetic buildings, without augmentation from any real-world measurements. Though exhibiting reasonable generalisation capability, when dealing with real-world scans, our method is not as robust as non-learning methods such as PolyFit. We ascribe this to the lack of real-world variants in our training data. Therefore, the generalisation capability can possibly be enhanced by training on real-world point clouds.

Moreover, we expect an enormous potential of deep implicit fields in the context of urban modelling, where this thesis serves only as a promising starting point. Aside from stand-alone buildings, multiple objects and their relations in the urban environment can be described by deep implicit fields. With various information from geographic information system (GIS), not only geometry but semantics can be incorporated to enrich the implicit field, contributing to higher-dimensional modelling of the urban environment.

Bibliography

Arikan, M., Schwärzler, M., Flöry, S., Wimmer, M., and Maierhofer, S. (2013). O-snap: Optimization-based snapping for modeling architecture. *ACM Transactions on Graphics (TOG)*, 32(1):1–15.

Atzmon, M. and Lipman, Y. (2020). SAL: Sign agnostic learning of shapes from raw data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2565–2574.

Barinova, O., Konushin, V., Yakubenko, A., Lee, K., Lim, H., and Konushin, A. (2008). Fast automatic single-view 3D reconstruction of urban scenes. In *European Conference on Computer Vision*, pages 100–113. Springer.

Berger, M., Tagliasacchi, A., Seversky, L. M., Alliez, P., Guennebaud, G., Levine, J. A., Sharf, A., and Silva, C. T. (2017). A survey of surface reconstruction from point clouds. In *Computer Graphics Forum*, volume 36, pages 301–329. Wiley Online Library.

Biljecki, F., Ledoux, H., and Stoter, J. (2016). An improved LOD specification for 3D building models. *Computers, Environment and Urban Systems*, 59:25–37.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., and Çöltekin, A. (2015). Applications of 3D city models: State of the art review. *ISPRS International Journal of Geo-Information*, 4(4):2842–2889.

Blut, C. and Blankenbach, J. (2021). Three-dimensional CityGML building models in mobile augmented reality: a smartphone-based pose tracking system. *International Journal of Digital Earth*, 14(1):32–51.

Boulch, A., de La Gorce, M., and Marlet, R. (2014). Piecewise-planar 3D reconstruction with edge and corner regularization. In *Computer Graphics Forum*, volume 33, pages 55–64. Wiley Online Library.

Bouzas, V., Ledoux, H., and Nan, L. (2020). Structure-aware building mesh polygonization. *ISPRS Journal of Photogrammetry and Remote Sensing*, 167:432–442.

Boykov, Y. and Funka-Lea, G. (2006). Graph cuts and efficient ND image segmentation. *International journal of computer vision*, 70(2):109–131.

Calakli, F., Ulusoy, A. O., Restrepo, M. I., Taubin, G., and Mundy, J. L. (2012). High resolution surface reconstruction from multi-view aerial imagery. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 25–32. IEEE.

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*, pages 1261–1268. IEEE.

Chen, J. and Chen, B. (2008). Architectural modeling from sparsely scanned range data. *International Journal of Computer Vision*, 78(2-3):223–236.

Chen, Z., Tagliasacchi, A., and Zhang, H. (2020). BSP-Net: Generating compact meshes via binary space partitioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 45–54.

Chen, Z. and Zhang, H. (2019). Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5939–5948.

Cohen-Steiner, D., Alliez, P., and Desbrun, M. (2004). Variational shape approximation. In *ACM SIGGRAPH 2004 Papers*, pages 905–914.

Coughlan, J. M. and Yuille, A. L. (2000). The Manhattan world assumption: Regularities in scene statistics which enable bayesian inference. In *NIPS*, volume 2, page 3.

Davies, T., Nowrouzezahrai, D., and Jacobson, A. (2020). On the effectiveness of weight-encoded neural implicit 3D shapes. *arXiv preprint arXiv:2009.09808*.

Deng, B., Genova, K., Yazdani, S., Bouaziz, S., Hinton, G., and Tagliasacchi, A. (2020). CvxNet: Learnable convex decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 31–44.

Erler, P., Guerrero, P., Ohrhallinger, S., Mitra, N. J., and Wimmer, M. (2020). Points2Surf: Learning implicit surfaces from point clouds. In *European Conference on Computer Vision*, pages 108–124. Springer.

Garland, M. and Heckbert, P. S. (1997). Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216.

Genova, K., Cole, F., Sud, A., Sarna, A., and Funkhouser, T. (2020). Local deep implicit functions for 3D shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4857–4866.

Gkioxari, G., Malik, J., and Johnson, J. (2019). Mesh R-CNN. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9785–9795.

Gschwandtner, M., Kwitt, R., Uhl, A., and Pree, W. (2011). Blensor: Blender sensor simulation toolbox. In *International Symposium on Visual Computing*, pages 199–208. Springer.

Hagberg, A., Swart, P., and S Chult, D. (2008). Exploring network structure, dynamics, and function using NetworkX. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

Häne, C., Tulsiani, S., and Malik, J. (2017). Hierarchical surface prediction for 3D object reconstruction. In *2017 International Conference on 3D Vision (3DV)*, pages 412–420. IEEE.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.

Herbert, G. and Chen, X. (2015). A comparison of usefulness of 2D and 3D representations of urban planning. *Cartography and Geographic Information Science*, 42(1):22–32.

Ikehata, S., Yang, H., and Furukawa, Y. (2015). Structured indoor modeling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1323–1331.

Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7.

Kazhdan, M. and Hoppe, H. (2013). Screened Poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13.

Koutsourakis, P., Simon, L., Teboul, O., Tziritas, G., and Paragios, N. (2009). Single view reconstruction using shape grammars for urban environments. In *2009 IEEE 12th international conference on computer vision*, pages 1795–1802. IEEE.

Labatut, P., Pons, J.-P., and Keriven, R. (2009). Hierarchical shape-based surface reconstruction for dense multi-view stereo. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 1598–1605. IEEE.

Lafarge, F. and Alliez, P. (2013). Surface reconstruction through point set structuring. In *Computer Graphics Forum*, volume 32, pages 225–234. Wiley Online Library.

Li, M., Nan, L., Smith, N., and Wonka, P. (2016a). Reconstructing building mass models from UAV images. *Computers & Graphics*, 54:84–93.

Li, M., Wonka, P., and Nan, L. (2016b). Manhattan-world urban reconstruction from point clouds. In *European Conference on Computer Vision*, pages 54–69. Springer.

Li, Y., Bu, R., Sun, M., Wu, W., Di, X., and Chen, B. (2018). PointCNN: Convolution on X-transformed points. *Advances in neural information processing systems*, 31:820–830.

Li, Y. and Wu, B. (2021). Relation-constrained 3D reconstruction of buildings in metropolitan areas from photogrammetric point clouds. *Remote Sensing*, 13(1):129.

Liu, C., Kim, K., Gu, J., Furukawa, Y., and Kautz, J. (2019). Planercnn: 3D plane detection and reconstruction from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4450–4459.

Liu, C., Yang, J., Ceylan, D., Yumer, E., and Furukawa, Y. (2018). PlaneNet: Piece-wise planar reconstruction from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2588.

Liu, S., Zhang, Y., Peng, S., Shi, B., Pollefeys, M., and Cui, Z. (2020). DIST: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2019–2028.

Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169.

Machete, R., Falcão, A. P., Gomes, M. G., and Rodrigues, A. M. (2018). The use of 3D GIS to analyse the influence of urban context on buildings' solar energy potential. *Energy and Buildings*, 177:290–302.

Meagher, D. (1982). Geometric modeling using octree encoding. *Computer graphics and image processing*, 19(2):129–147.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. (2019). Occupancy networks: Learning 3D reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470.

Moenning, C. and Dodgson, N. A. (2003). Fast marching farthest point sampling. Technical report, University of Cambridge, Computer Laboratory.

Monszpart, A., Mellado, N., Brostow, G. J., and Mitra, N. J. (2015). RAPter: rebuilding man-made scenes with regular arrangements of planes. *ACM Trans. Graph.*, 34(4):103–1.

Mura, C., Mattausch, O., and Pajarola, R. (2016). Piecewise-planar reconstruction of multi-room interiors with arbitrary wall arrangements. In *Computer Graphics Forum*, volume 35, pages 179–188. Wiley Online Library.

Murali, T. and Funkhouser, T. A. (1997). Consistent solid and boundary representations from arbitrary polygonal data. In *Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 155–ff.

Nan, L. (2018). Easy3D: a lightweight, easy-to-use, and efficient C++ library for processing and rendering 3D data. https://github.com/LiangliangNan/Easy3D.

Nan, L. and Wonka, P. (2017). PolyFit: Polygonal surface reconstruction from point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2353–2361.

Oesau, S., Lafarge, F., and Alliez, P. (2016). Planar shape detection and regularization in tandem. In *Computer Graphics Forum*, volume 35, pages 203–215. Wiley Online Library.

Ohori, K. A. (2016). *Higher-dimensional modelling of geographic information*. Lulu. com.

Pan, J., Han, X., Chen, W., Tang, J., and Jia, K. (2019). Deep mesh reconstruction from single RGB images via topology modification networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9964–9973.

Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. (2019). DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174.

Partovi, T., Fraundorfer, F., Bahmanyar, R., Huang, H., and Reinartz, P. (2019). Automatic 3D building model reconstruction from very high resolution stereo satellite imagery. *Remote Sensing*, 11(14):1660.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, 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 Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660.

Qian, Y. and Furukawa, Y. (2020). Learning pairwise inter-plane relations for piecewise planar reconstruction. In *European Conference on Computer Vision*, pages 330–345. Springer.

Remelli, E., Lukoianov, A., Richter, S. R., Guillard, B., Bagautdinov, T., Baque, P., and Fua, P. (2020). MeshSDF: Differentiable iso-surface extraction. *arXiv preprint arXiv:2006.03997*.

Rupnik, E., Pierrot-Deseilligny, M., and Delorme, A. (2018). 3D reconstruction from multi-view VHR-satellite images in MicMac. *ISPRS Journal of Photogrammetry and Remote Sensing*, 139:201–211.

Salinas, D., Lafarge, F., and Alliez, P. (2015). Structure-aware mesh decimation. In *Computer Graphics Forum*, volume 34, pages 211–227. Wiley Online Library.

Schindler, F., Wörstner, W., and Frahm, J.-M. (2011). Classification and reconstruction of surfaces from point clouds of man-made objects. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 257–263. IEEE.

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient RANSAC for point-cloud shape detection. In *Computer graphics forum*, volume 26, pages 214–226. Wiley Online Library.

Schöning, J. and Heidemann, G. (2015). Evaluation of multi-view 3D reconstruction software. In *International conference on computer analysis of images and patterns*, pages 450–461. Springer.

Shannon, C. (1949). Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21.

Stoter, J., Peters, R., Commandeur, T., Dukai, B., Kumar, K., and Ledoux, H. (2020). Automated reconstruction of 3D input data for noise simulation. *Computers, Environment and Urban Systems*, 80:101424.

The Sage Developers (2021). *SageMath, the Sage Mathematics Software System (Version 9.1)*.

Van Kreveld, M., Van Lankveld, T., and Veltkamp, R. C. (2011). On the shape of a set of points and lines in the plane. In *Computer Graphics Forum*, volume 30, pages 1553–1562. Wiley Online Library.

Verdie, Y., Lafarge, F., and Alliez, P. (2015). LOD generation for urban scenes. *ACM Transactions on Graphics*, 34(ARTICLE):30.

Vo, A.-V., Truong-Hong, L., Laefer, D. F., and Bertolotto, M. (2015). Octree-based region growing for point cloud segmentation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 104:88–100.

Wang, H., Schor, N., Hu, R., Huang, H., Cohen-Or, D., and Huang, H. (2018a). Global-to-local generative model for 3D shapes. *ACM Transactions on Graphics (TOG)*, 37(6):1–10.

Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., and Jiang, Y.-G. (2018b). Pixel2Mesh: Generating 3D mesh models from single RGB images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67.

Xie, L., Hu, H., Zhu, Q., Li, X., Tang, S., Li, Y., Guo, R., Zhang, Y., and Wang, W. (2021). Combined rule-based and hypothesis-based method for building model reconstruction from photogrammetric point clouds. *Remote Sensing*, 13(6):1107.

Xu, M., Li, M., Xu, W., Deng, Z., Yang, Y., and Zhou, K. (2016). Interactive mechanism modeling from multi-view images. *ACM Transactions on Graphics (TOG)*, 35(6):1–13.

Yang, F. and Zhou, Z. (2018). Recovering 3D planes from a single image via convolutional neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100.

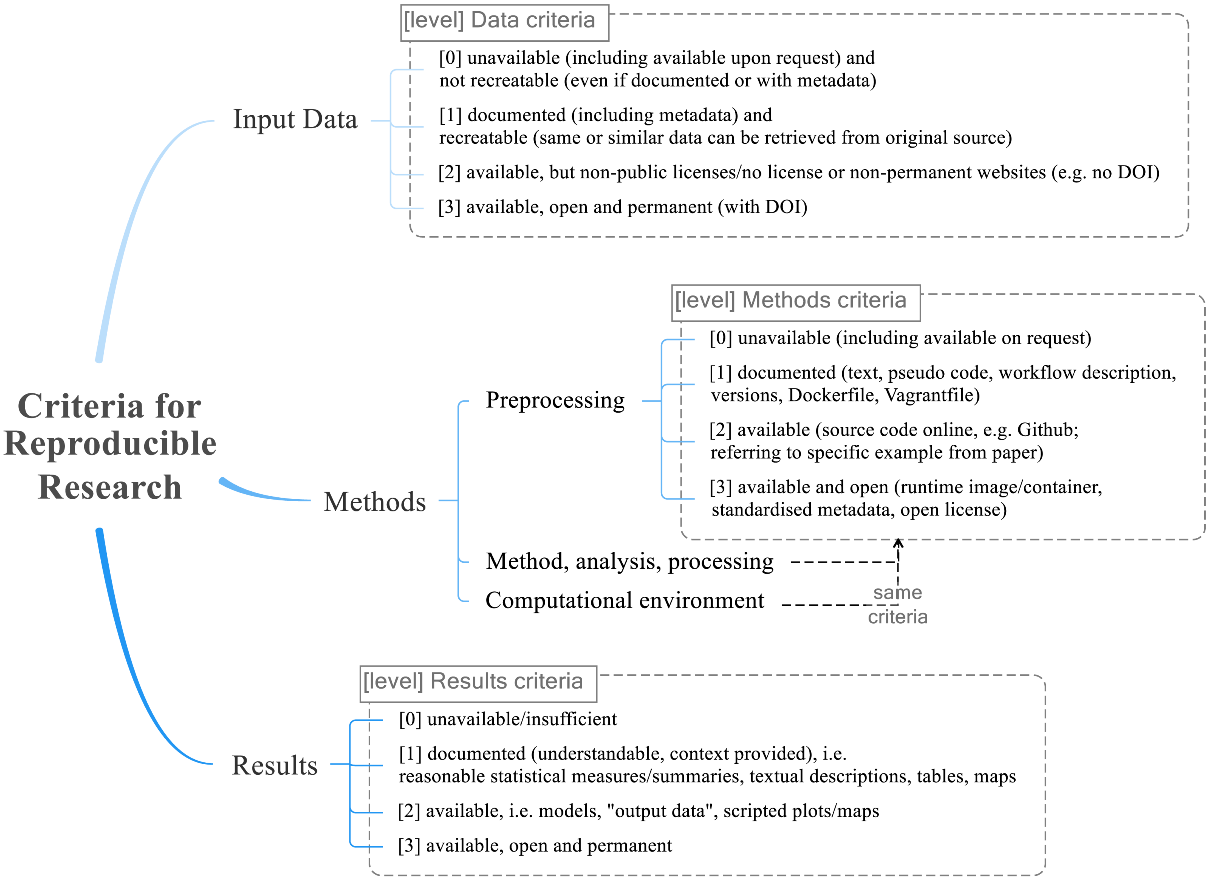
Yu, D., Ji, S., Liu, J., and Wei, S. (2021). Automatic 3D building reconstruction from multi-view aerial images with deep learning. *ISPRS Journal of Photogrammetry and Remote Sensing*, 171:155–170.

Yu, Z., Zheng, J., Lian, D., Zhou, Z., and Gao, S. (2019). Single-image piece-wise planar 3D reconstruction via associative embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1029–1037.

Zhou, Q.-Y. and Neumann, U. (2010). 2.5D Dual Contouring: A robust approach to creating building models from aerial LiDAR point clouds. In *European conference on computer vision*, pages 115–128. Springer.

A Reproducibility self-assessment

A.1 Marks for each of the criteria



[level] Data criteria
[0] unavailable (including available upon request) and not recreatable (even if documented or with metadata)
[1] documented (including metadata) and recreatable (same or similar data can be retrieved from original source)
[2] available, but non-public licenses/no license or non-permanent websites (e.g. no DOI)
[3] available, open and permanent (with DOI)
Input Data
Criteria for Reproducible Research
Methods
Preprocessing
[level] Methods criteria
[0] unavailable (including available on request)
[1] documented (text, pseudo code, workflow description, versions, Dockerfile, Vagrantfile)
[2] available (source code online, e.g. Github; referring to specific example from paper)
[3] available and open (runtime image/container, standardised metadata, open license)
Method, analysis, processing
Computational environment
same criteria
Results
[level] Results criteria
[0] unavailable/insufficient
[1] documented (understandable, context provided), i.e. reasonable statistical measures/summaries, textual descriptions, tables, maps
[2] available, i.e. models, "output data", scripted plots/maps
[3] available, open and permanent

Figure A.1: Reproducibility criteria to be assessed.

Table A.1 presents the marks under the criteria for reproducible research described in Figure A.1.

Category	Criteria			Mark
	Available	Open	Permanent	
Input data	✓	✓	✗	2
Preprocessing	✓	✓	✓	3
Method, analysis, processing	✓	✓	✓	3
Computational environment	✓	✓	✓	3
Results	✓	✓	✗	2

Table A.1: Reproducibility evaluation

A.2 Self-reflection

The research is considered reproducible in every criterion. The simulated data are openly available while the real-world point clouds in Shenzhen are available upon request to the authors of Li and Wu [2021]. All source codes concerning preprocessing, method, analysis and processing are openly available at `https://github.com/chenzhaiyu/points2poly`[1], with adaptive space partitioning maintained independently as a submodule at `https://github.com/chenzhaiyu/absp`. The computational environment employed in this research is open-sourced. The results are available while not open for clutter concerns.

[1] Available upon submission of this thesis.

B Alternative cell complex formulation

3D Delaunay triangulation also provides the possibility of partitioning a 3D space into a cell complex of tetrahedron (Figure B.1). Compared with hyperplane arrangement, it faithfully respects the distribution of the raw point cloud without distilling higher-level shape information. To mitigate this, farthest point sampling as proposed by Moenning and Dodgson [2003] can be used to select points of high importance to the global shape. Hyperplane arrangement and Delaunay triangulation complement each other, targeting shapes of piecewise planarity and generalising to general objects (such as the Stanford bunny in Figure 5.22), respectively.

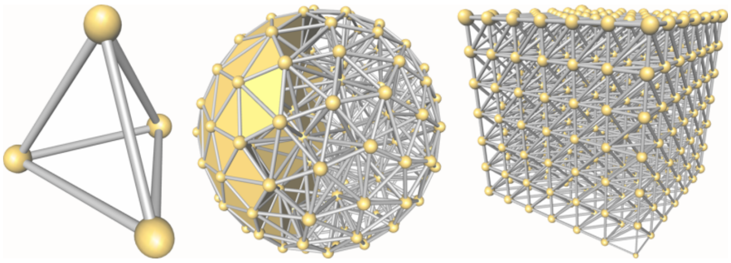
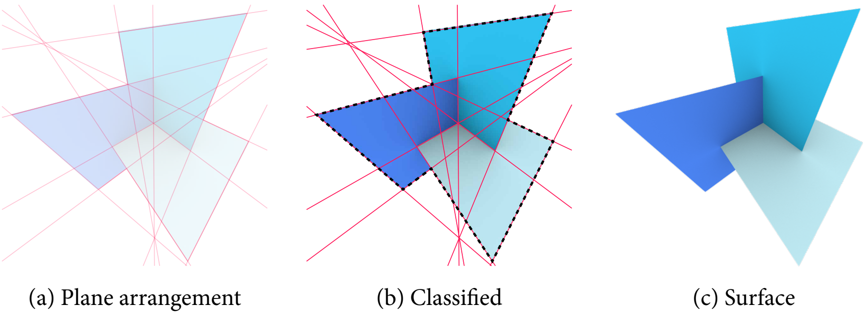


Figure B.1: Cell complex (tetrahedra) constructed with 3D Delaunay triangulation. Figure from `https://doc.cgal.org/latest/Triangulation_3/index.html`.

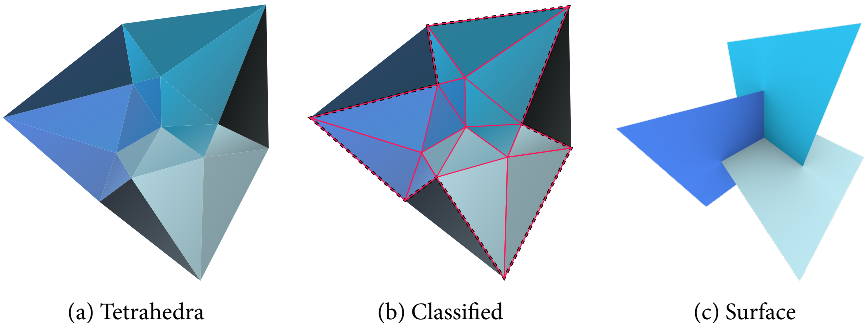


(a) Plane arrangement

(b) Classified

(c) Surface

Figure B.2: Surface extraction from a cell complex generated via hyperplane arrangement



(a) Tetrahedra

(b) Classified

(c) Surface

Figure B.3: Surface extraction from a cell complex generated via 3D Delaunay triangulation

C Alternative deep implicit field formulations

The X-Conv proposed by Li et al. [2018] may be employed for encoding point clouds, followed by an MLP producing the feature vector. Given any query point, its coordinates are concatenated with the encoded shape features, feeding to the implicit decoder [Chen and Zhang, 2019; Park et al., 2019], which learns whether the query point is inside or outside the shape, as illustrated in Figure C.1.

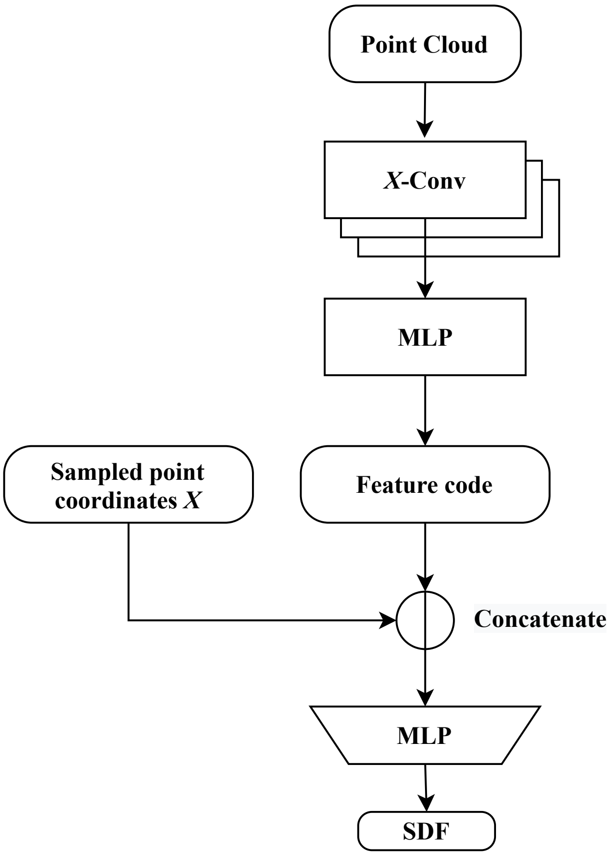


Figure C.1: Network architecture of implicit shape indicator

Based on BSP-Net [Chen et al., 2020], we propose an alternative neural network architecture for implicit field learning, as illustrated in Figure C.2. The original BSP-Net takes as input either voxels or images. By substituting the encoder with X-Conv operators [Li et al., 2018], the adapted neural network learns to reconstruct piecewise-planar objects from point clouds. Specifically, the X-Conv operators followed by an MLP produce the canonical parameters of p planes $P_{p\times 4}$. These plane parameters are multiplied with the homogeneous coordinates of n sampled points $X_{n\times 4}$ to yield the signed distance from each sampled point to each plane $D_{n\times p}$. Then a learnable binary matrix $T_{p\times c}$ selectively form convexes $C_{n\times c}$ from $D_{n\times p}$. At the last layer, the convexes are merged into one shape $S_{n\times 1}$.

Figure C.3 presents the LoD2 building reconstruction results on the AHN[1] dataset with candidate cell complexes generated by hyperplane arrangement. Arguably, LoD2 reconstruction limits the capability of the learnt implicit

[1] https://www.ahn.nl/

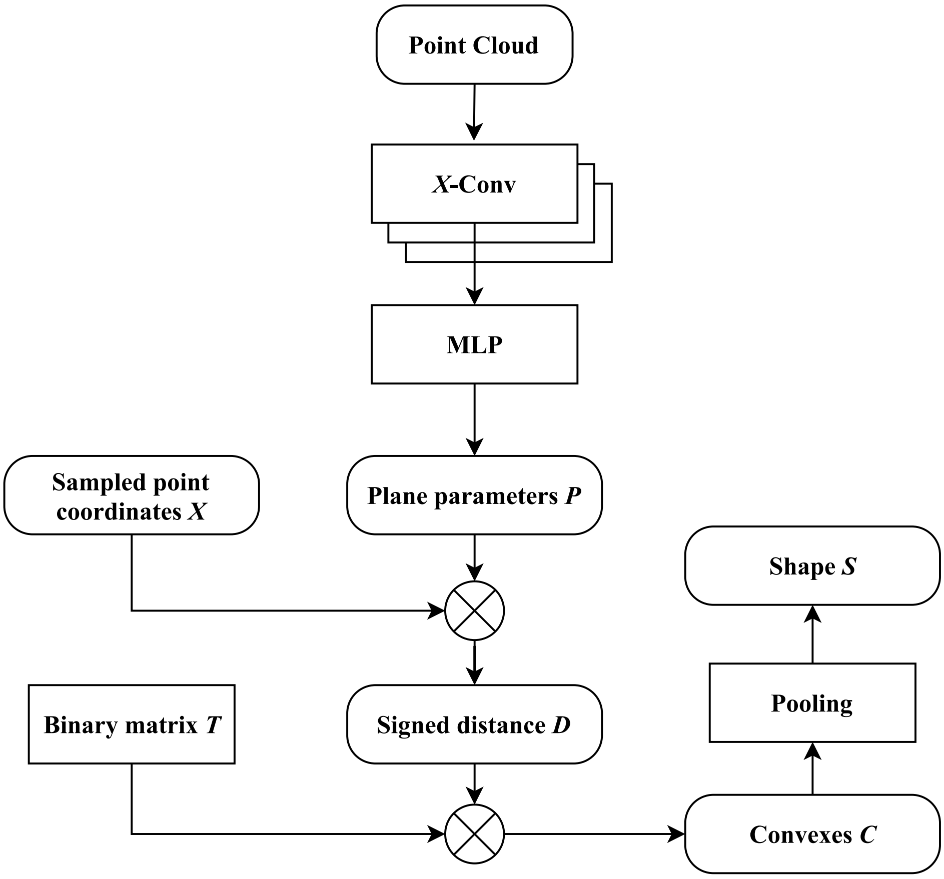


Figure C.2: Adapted BSP-Net for shape reconstruction from point clouds

indicator function due to a lack of correspondence between the sparse aerial point clouds and the inaccurate ground truth surfaces provided by PolyFit [Nan and Wonka, 2017]. Nevertheless, the reconstructed surfaces are comparable with the ground truth ones.

Figure C.4 presents the reconstruction results on a collection of ShapeNet[2] point clouds. Benefiting from the self-supervised learning scheme, piecewise-planar surfaces can be generated even though the input point cloud does not fulfil the assumption, e.g., the point clouds of the lamps and that of the car exhibit little planarity, but the piecewise-planar surfaces can be reconstructed with reasonable geometric fidelity.

[2] https://shapenet.org/

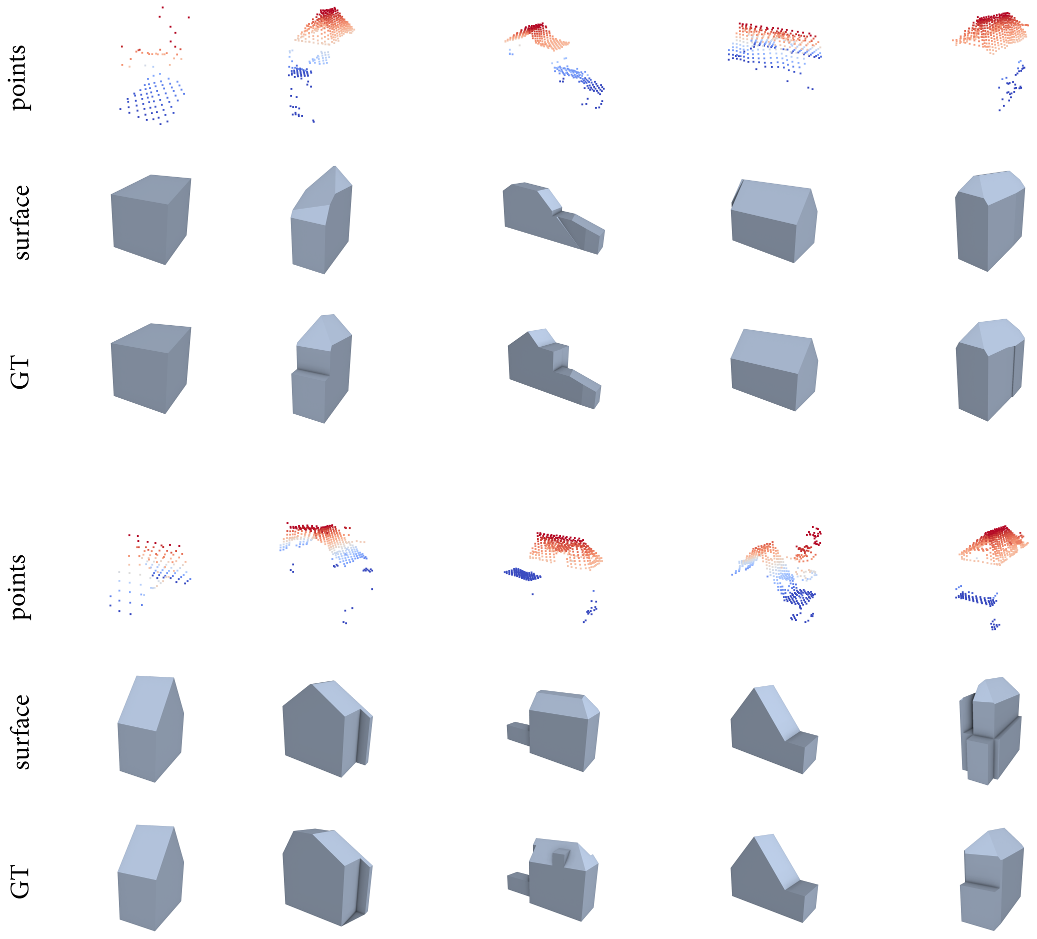


Figure C.3: Building reconstruction from AHN point clouds. Points rendered based on height.

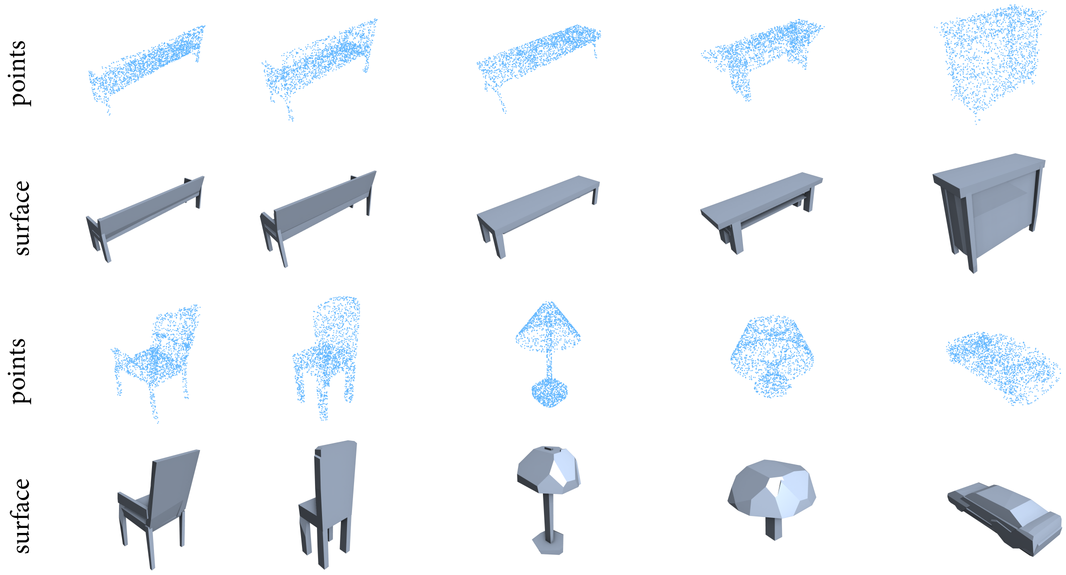


Figure C.4: Reconstruction from ShapeNet point clouds

Colophon

This document was typeset using LaTeX, using the KOMA-Script class `scrbook`. The main font is Minion Pro.

Cover

🅭🅯 The cover image was adapted from *Isometric Cityscape* by *Coen Pohl* licensed under a Creative Commons Attribution 4.0 International License. The signed distance field and wireframe were created with Adobe Photoshop.