# Structure-aware 3D Building Reconstruction

Huang, J.

**DOI**
10.4233/uuid:7406712d-7bc3-4565-9291-53f62f1e8758

**Publication date**
2025

**Document Version**
Final published version

**Citation (APA)**
Huang, J. (2025). *Structure-aware 3D Building Reconstruction*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:7406712d-7bc3-4565-9291-53f62f1e8758

# STRUCTURE-AWARE 3D BUILDING RECONSTRUCTION

# STRUCTURE-AWARE 3D BUILDING RECONSTRUCTION

## Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus Prof. dr. ir. T.H.J.J. van der Hagen
Chair of the Board for Doctorates
to be defended publicly on
Tuesday 18 March 2025 at 10:00 o'clock

by

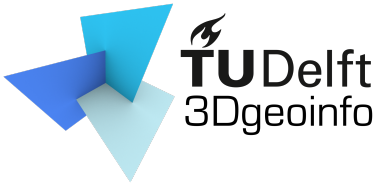
**Jin HUANG**

Master of Engineering in Aerospace Manufacturing Engineering,
Nanjing University of Aeronautics and Astronautics, China,
born in Jiangxi, China.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

| | |
|---|---|
| Rector Magnificus | chairperson |
| Prof. dr. J. E. Stoter | Delft University of Technology, promotor |
| Dr. L. Nan | Delft University of Technology, copromotor |
| | |
| *Independent members:* | |
| Prof. dr. -Ing. H. Fan | Norwegian University of Science and Technology, Norway |
| Prof. dr. -Ing. N. Pfeifer | Vienna University of Technology, Austria |
| Dr. ir. S. J. Oude Elberink | University of Twente, Netherlands |
| Prof. dr. ing. S. Nijhuis | Delft University of Technology |
| Dr. R. C. Lindenbergh | Delft University of Technology |
| Prof. dr. ing. U. Pottgiesser | Delft University of Technology, reserve member |





This research was financially supported by China Scholarship Council (CSC) under grant 201906830100.

*Keywords:* Building reconstruction, point clouds, MVS meshes, integer programming, regularization, symmetrization, geometry optimization



ISBN: 978-94-6518-031-1

A digital version of this dissertation is available at http://repository.tudelft.nl/.

*Dedicated to all the people who love me and whom I love*

Jin Huang

# CONTENTS

# SUMMARY

Lightweight and accurate building models have been widely used in diverse applications such as urban planning, virtual reality, and navigation. In recent years, structure-aware building reconstruction has emerged as a crucial research area. Despite significant advancements in measurement techniques such as Light Detection and Ranging (LiDAR) and photogrammetry, the raw data often contains different types of imperfections, such as noise, outliers, and missing regions. These imperfections pose challenges for the accurate and efficient reconstruction of complex building structures. Therefore, this thesis aims to address these challenges by proposing methods for automatic Level of Detail 2 (LoD2) building reconstruction from airborne LiDAR point clouds, and semi-automatic Level of Detail 3 (LoD3) building reconstruction from Multi-View Stereo (MVS) meshes. Throughout the reconstruction process, structural details are easily distorted in the final output due to the inaccuracies and imperfections of input data. Given that regularities such as symmetry are prevalent in building models, they can be leveraged to recover lost or distorted building structures. To facilitate the recovery of symmetry, building elements are projected into facade planes to be two-dimensional (2D) polygonal shapes. Therefore, to obtain accurate and aesthetically pleasing models, this thesis also focuses on recovering the symmetry of these 2D polygonal shapes generated from buildings.

My first contribution is city-scale LoD2 building reconstruction from airborne LiDAR point clouds. While LiDAR data provides rich geometric information, reconstructing detailed building models at such a large scale remains an open problem. This thesis proposes a novel method to tackle this problem, achieving accurate city-scale LoD2 building reconstruction. Firstly, I use footprint data to segment out the point clouds of individual building instances. Then, I detect planar primitives using a region-growing algorithm and infer wall primitives by applying a vertical assumption on the missing regions. Then an abundant set of candidate faces is generated by intersecting the planes derived from roof and wall primitives. Finally, I can obtain a compact building model by selecting the optimal subset of candidate faces through solving an integer programming problem. Geometry constraints are enforced to ensure that the final model is manifold and watertight.

My second contribution is a semi-automatic method for reconstructing LoD3 building models from MVS meshes. While MVS techniques can generate dense and detailed triangular surfaces, creating compact and accurate LoD3 models from them remains challenging due to the limited data resolution. The proposed method is designed to strike a balance between human interactions and automation, aiming to maximize efficiency while minimizing user efforts. The process begins with a coarse segmentation using variational shape approximation [45]. Then, simple and intuitive operations are introduced to refine the segmentation results by solving a multi-label optimization problem. At this stage, the user's involvement is minimal and limited to providing high-level guidance, ensuring that the system remains user-friendly. Importantly, these interac-

tions are kept to a minimum, allowing users to make adjustments without requiring precise input, making the process more efficient than manual reconstruction. Finally, the face normals and vertices of the mesh are updated based on the refined segmentation, and the layout of the model is regularized to produce an accurate LoD3 building model. This semi-automatic approach combines the strengths of both user input and automated computation, offering a practical solution for detailed building reconstruction that is both effective and user-friendly.

My third contribution is a novel algorithm to automatically symmetrize 2D polygonal shapes, which is essential to regularize the shapes and enhance the visual aesthetics of building models. The method follows a hypothesis-and-selection pipeline. Taking a 2D polygonal shape generated from a building model as input, I first generate a set of potential symmetric edge pairs. Then the initial set is pruned by two simple geometric tests. Finally, a perfectly symmetric shape is obtained by solving a mixed integer quadratic programming problem. Two hard constraints are imposed to ensure that the final shape to be symmetric. The method is also designed to handle partial symmetry in cases where perfect symmetry is not achievable.

In summary, I first automatically reconstruct LoD2 building models from airborne LiDAR point clouds. Then, I reconstruct LoD3 building models from MVS meshes by incorporating user guidance, which depicts a more detailed representation of building models. To obtain more accurate and visually pleasing building models, I propose to symmetrize the 2D polygonal shapes generated from facade elements of reconstructed models.

# SAMENVATTING

Lichtgewicht en nauwkeurige gebouwmodellen worden veelvuldig gebruikt in diverse toepassingen zoals stedelijke planning, virtuele realiteit en navigatie. In de afgelopen jaren is structuur-gebaseerde gebouwreconstructie naar voren gekomen als een cruciaal onderzoeksgebied. Ondanks aanzienlijke vooruitgang in meetmethoden zoals LiDAR (Light Detection and Ranging) en fotogrammetrie, bevat de ruwe data vaak verschillende soorten onvolkomenheden, zoals ruis, uitschieters en ontbrekende delen. Deze onvolkomenheden vormen uitdagingen voor de nauwkeurige en efficiënte reconstructie van complexe gebouwstructuren. Daarom richt deze scriptie zich op het aanpakken van deze uitdagingen door methoden voor te stellen voor automatische reconstructie van gebouwen op detailniveau 2 (LoD2) uit lucht-LiDAR-puntwolken, en semi-automatische reconstructie op detailniveau 3 (LoD3) uit Multi-View Stereo (MVS) meshes.

Tijdens het reconstructieproces worden structurele details vaak vervormd in het uiteindelijke resultaat vanwege de onnauwkeurigheden en onvolkomenheden van de invoergegevens. Aangezien regelmatigheden zoals symmetrie vaak voorkomen in gebouwmodellen, kunnen deze worden benut om verloren of vervormde gebouwstructuren te herstellen. Om de symmetrie te herstellen, worden bouwelementen geprojecteerd op gevelvlakken, zodat ze tweedimensionale (2D) polygonale vormen worden. Om nauwkeurige en esthetisch aantrekkelijke modellen te verkrijgen, richt deze scriptie zich ook op het herstellen van de symmetrie van deze 2D-polygonale vormen die uit gebouwen zijn gegenereerd.

Mijn eerste bijdrage is de stadsbrede LoD2-gebouwreconstructie uit lucht-LiDAR-puntwolken. Hoewel LiDAR-gegevens rijke geometrische informatie bieden, blijft het reconstrueren van gedetailleerde gebouwmodellen op zo'n grote schaal een open probleem. In deze scriptie wordt een nieuwe methode voorgesteld om dit probleem aan te pakken en nauwkeurige stadsbrede LoD2-gebouwreconstructie te bereiken. Eerst gebruik ik footprintgegevens om de puntwolken van individuele gebouwinstellingen te segmenteren. Vervolgens detecteer ik vlakke primitieven met behulp van een regiogroei-algoritme en leid muurprimitieven af door een verticale aanname toe te passen op de ontbrekende gebieden. Daarna wordt een uitgebreide set kandidaatvlakken gegenereerd door de vlakken van dak- en muurprimitieven te kruisen. Uiteindelijk kan ik een compact gebouwmodel verkrijgen door de optimale subset van kandidaatvlakken te selecteren via het oplossen van een integer-programmeringsprobleem. Geometrische beperkingen worden afgedwongen om ervoor te zorgen dat het uiteindelijke model veelzijdig en waterdicht is.

Mijn tweede bijdrage is een semi-automatische methode voor het reconstrueren van LoD3-gebouwmodellen uit MVS-meshes. Hoewel MVS-technieken dichte en gedetailleerde driehoekige oppervlakken kunnen genereren, blijft het creëren van compacte en nauwkeurige LoD3-modellen hieruit een uitdaging vanwege de beperkte resolutie van de gegevens. De voorgestelde methode is ontworpen om een balans te vinden tussen

menselijke interactie en automatisering, met als doel de efficiëntie te maximaliseren en tegelijkertijd de inspanningen van de gebruiker te minimaliseren. Het proces begint met een grove segmentatie met behulp van variational shape approximation. Vervolgens worden eenvoudige en intuïtieve bewerkingen geïntroduceerd om de segmentatieresultaten te verfijnen door een multi-label optimalisatieprobleem op te lossen. Op dit punt is de betrokkenheid van de gebruiker minimaal en beperkt tot het geven van algemene richtlijnen, wat ervoor zorgt dat het systeem gebruiksvriendelijk blijft. Deze interacties worden tot een minimum beperkt, zodat gebruikers aanpassingen kunnen maken zonder nauwkeurige invoer te vereisen, waardoor het proces efficiënter wordt dan handmatige reconstructie. Ten slotte worden de normale vlakken en hoekpunten van de mesh bijgewerkt op basis van de verfijnde segmentatie, en wordt de indeling van het model geregulariseerd om een nauwkeurig LoD3-gebouwmodel te produceren. Deze semi-automatische aanpak combineert de sterke punten van zowel gebruikersinput als geautomatiseerde berekeningen, en biedt een praktische oplossing voor gedetailleerde gebouwreconstructie die zowel effectief als gebruiksvriendelijk is.

Mijn derde bijdrage is een nieuw algoritme om 2D-polygonale vormen automatisch te symmetrizeren, wat essentieel is om de vormen te regulariseren en de visuele esthetiek van gebouwmodellen te verbeteren. De methode volgt een hypothese-en-selectie-pijplijn. Door een 2D-polygonale vorm gegenereerd uit een gebouwmodel als invoer te nemen, genereer ik eerst een reeks potentiële symmetrische randenparen. Vervolgens wordt de initiële set gesnoeid door twee eenvoudige geometrische tests. Uiteindelijk wordt een perfect symmetrische vorm verkregen door het oplossen van een mixed integer kwadratisch programmeringsprobleem. Twee harde beperkingen worden opgelegd om ervoor te zorgen dat de uiteindelijke vorm symmetrisch is. De methode is ook ontworpen om gedeeltelijke symmetrie aan te kunnen in gevallen waarin perfecte symmetrie niet haalbaar is.

Samenvattend, reconstrueer ik eerst automatisch LoD2-gebouwmodellen uit lucht-LiDAR-puntwolken. Vervolgens reconstrueer ik LoD3-gebouwmodellen uit MVS-meshes door gebruikersbegeleiding te integreren, wat een gedetailleerdere weergave van gebouwmodellen oplevert. Om nauwkeurigere en visueel aantrekkelijkere gebouwmodellen te verkrijgen, stel ik voor om de symmetrie te herstellen van de 2D-polygonale vormen die zijn gegenereerd uit de gevel elementen van gereconstrueerde modellen.

# ACRONYMS

**LiDAR** Light Detection and Ranging.

**LoD2** Level of Detail 2.

**LoD3** Level of Detail 3.

**LoDs** Levels of Detail.

**MVS** Multi-View Stereo.

**3D** Three Dimensional.

**OGC** Open Geospatial Consortium.

**ALS** Airborne LiDAR Scanning.

**SfM** Structure from Motion.

**RTG** Roof Topology Graph.

**DSM** Digital Surface Model.

**DEM** Digital Elevation Model.

**RANSAC** Random Sample Consensus.

**TIN** Triangulated Irregular Network.

**MDL** Minimum Description Length.

**NeRF** Neural Radiance Fields.

**GS** Gaussian Splatting.

# 1

# INTRODUCTION

*This chapter provides a brief overview of the applications of building models and introduces the motivation for 3D reconstruction of urban buildings. It then outlines the main challenges for reconstructing high level of detail building models. The chapter concludes with a discussion of the research questions that will be addressed, as well as the organization of the work.*

1

## 1.1. MOTIVATION

Representing real-world objects in three-dimension has fascinated humans for centuries. In the field of architecture, the desire to capture and preserve the essence of buildings can be traced back to early civilizations. During these periods, detailed sculptures and physical models were used to record architectural information. However, these manual methods are labor-intensive, limiting their further practicality and scalability. The late twentieth and early twenty-first centuries experienced a significant progress in measurement technology. The widespread use of digital cameras, and advanced software allowed researchers to develop better methods for creating building models from images. The availability of Light Detection and Ranging (LiDAR) technology extends the potential for capturing detailed building models even further.

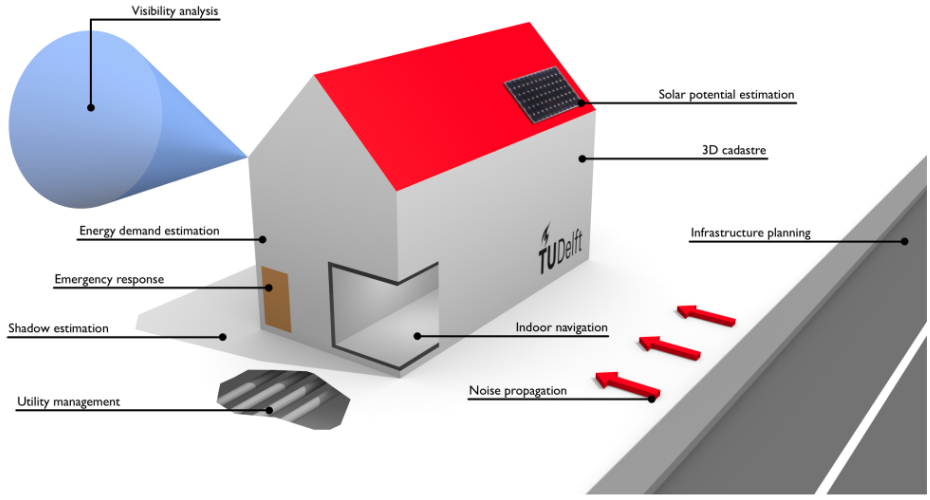



Figure 1.1: A variety of applications for building models. Figure taken from Biljecki et al. [19].

Nowadays, 3D building models are increasingly used in a variety of applications as summarized in Figure 1.1, such as urban planning[40, 203], simulation[119, 12, 60, 161, 180], robotic mapping, navigation[34, 47], visualization [195] and entertainment [88]. For example, planners use 3D building models to make more effective urban planning and development strategies, because they provide a detailed representation of existing structures and urban landscapes. Specifically, during natural disasters such as earthquakes, floods, or fires, accurate 3D building models are essential for assessing damage and planning rescue operations. Building models, often generated from LiDAR or photogrammetry data, provide essential references for robots to comprehend and navigate complex urban environments. These models enable robots to plan optimal paths, avoid obstacles, and accurately localize themselves within the 3D urban space. By comparing real-time sensor data with offline 3D maps, robots can make accurate decisions, thereby enhancing safety and efficiency. As a result, the automated reconstruction of urban buildings has attracted increasing attention in the community of computer graphics

and geomatics.

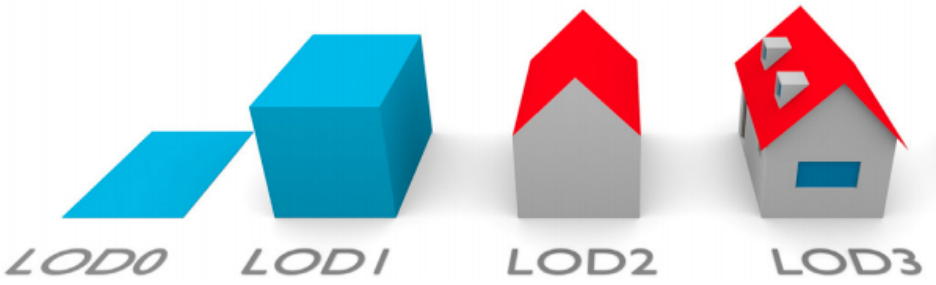



Figure 1.2: Four LoDs of CityGML 3.0[95]. Figure adapted from Biljecki et al. [18].

According to the newest standard[95] of the Open Geospatial Consortium (OGC), four different Levels of Detail (LoDs) are defined for a building model, as shown in Fig.1.2. LoD0 is the 2D representation of a building, which may consist of the building's footprint or outline, as seen in datasets such as 3DBAG[138]. LoD1 is a prismatic model that can be obtained by extruding the footprint of a building in the 3D space. Extending a projected roofplane or building outline to the ground is another way of generating LoD1 building models. LoD2 is a solid model, enriched with roof shapes and can also represent multi-surface geometries. LoD3 is an architecturally detailed model, enhanced with facade elements and roof superstructures, e.g., windows and chimneys. Higher LoD models contain more geometric details. The necessity for such detail varies depending on specific applications. For example, Biljecki et al.[20] demonstrate that in some spatial analyses, a LoD1 model can yield results that are as accurate as a LoD2 model, while a lower LoD model can bring benefits to memory storage and computational efficiency. The choice of the most appropriate LoD depends on the requirements of specific applications. Considering the diverse requirements across different applications regarding the level of detail in building models, and acknowledging the availability of LoD0 and LoD1 building datasets, my PhD research focuses on advancing the reconstruction of urban buildings at both LoD2 and LoD3 levels.

For LoD2 building reconstruction, airborne LiDAR point clouds have been the first choice due to their ability to capture non-vertical elements with high precision. They provide extensive coverage of large urban areas, ensuring consistent and uniform data, which makes them suitable for large-scale LoD2 building reconstruction. Despite their effectiveness in capturing roof structures, airborne LiDAR scanners typically struggle with acquiring sufficient data points for building facades. Furthermore, the obtained point clouds often contain noises and outliers due to imperfect outdoor scanning conditions, such as different observation distances and reflectance. These data defects hinder reconstruction results from being fully automatic and accurate. Moreover, there are no widely applicable patterns for buildings due to the variant architectural style across the world. Therefore, most existing reconstruction approaches are only applied to buildings with specific structural properties and are hard to adapt to general cases (see Chapter 2). For example, Li et al. [109] propose a fully automatic approach for reconstruct-

ing buildings from point clouds by exploiting the Manhattan-world assumption on the output models. Peters et al. [138] propose an automated method for the reconstruction of LoD2 building models and have created the 3DBAG dataset, a comprehensive 3D building dataset. However, as noted by Kaiser [87], this dataset often lacks structural regularity. These irregularities pose challenges for applications requiring precise and consistent models. For example, regularized building models are crucial for urban planners to accurately simulate future developments, assess compliance with building regulations—such as height restrictions—and optimize land use. Additionally, these models are essential for conducting shadow analysis and evaluating building density. To address these issues, one goal of my research is to develop a novel framework for the automatic generation of LoD2 building models from airborne LiDAR point clouds, with a focus on producing regular and accurate building models to meet the demands of these applications.

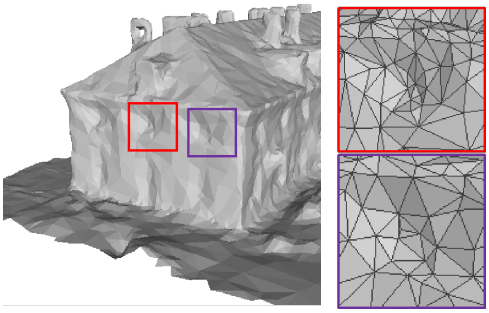


Figure 1.3: The mesh generated by Multi-View Stereo (MVS). There are only a few triangle faces for these windows.

While LoD2 models contain fundamental roof structures, LoD3 models introduce facade elements and roof superstructures, thereby increasing complexity. Airborne LiDAR scans exhibit high potential to reconstruct non-vertical elements such as roofs, yet they often fall short of capturing full 3D building elements. By contrast, meshes derived from oblique images using Structure from Motion (SfM) and Multi-View Stereo (MVS) workflows contain facades with rich details. This is because the MVS system is effective in capturing all sides of urban buildings and robust against occlusions. Bouzas et al. [25] introduce an optimization-based approach to generate LoD2 building models, concluding the geometric information of MVS meshes is sufficient for LoD2 building reconstruction. The MVS meshes which contain rich geometry and topology description offer the opportunity to reconstruct higher level of detail building models. However, achieving fully automatic LoD3 building reconstruction from MVS meshes remains infeasible due to inherent limitations in mesh resolution. As shown in Figure 1.3, only a few triangle faces represent a window on the facade, posing big challenges for automatic detection of detailed facade planes. Instead of reconstructing roof superstructures and facade components from data, Verdie et al. [172] model them as icons, which are not detailed and accurate. To address these challenges, I propose to combine the strengths of both user guidance and automated computation to generate LoD3 building models

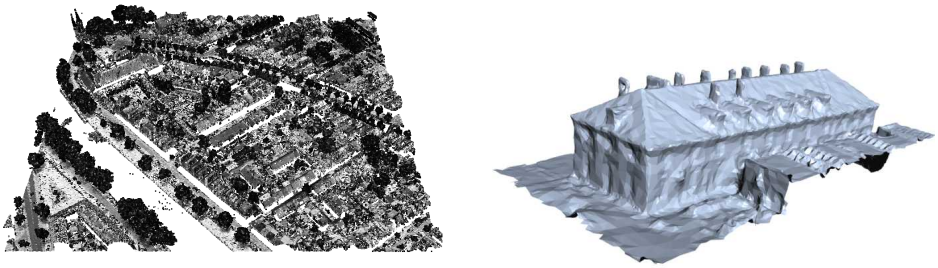
from MVS meshes. Computers are good at expensive and repeated computations such as solving complex optimization problems, and for building reconstruction. It is already possible to automatically detect the main roof structures. However, it is challenging to reliably detect meaningful building components in an automated way, such as windows, doors, and dormers due to data imperfections. Therefore, I propose to incorporate user guidance into the reconstruction process, which allows for high-level human interactions to generate accurate and high-quality building models. Specifically, the interactive approach strikes a balance between manual and automated reconstruction, ensuring reconstructed models meet users' demands with minimal human interventions.

Figure 1.4: Symmetry in real-life architecture.

During the process of data collection and LoD2/LoD3 building reconstruction, structural regularities such as orthogonality, parallelism, and symmetry (as shown in Figure 1.4) are often distorted or lost. Since these regularities are fundamental characteristics of building models, they can be leveraged as geometric priors to restore missing or distorted details in the reconstructed models [127]. Building structures typically contain two core elements [126]: (a) the components comprising the whole shape and (b) the intricate relationships among these components. Since real-world building models can be effectively approximated by piece-wise planar segments, I specify them as

1

the identification of basic planar primitives within the input data alongside their adjacency relationships. To achieve structure-aware 3D building reconstruction, this thesis goes beyond optimizing local geometry and preserving detailed features. Here, "structure awareness" refers to the ability to reconstruct models that (1) retain fundamental architectural characteristics, such as planar surfaces; (2) restore disrupted architectural regularities, including orthogonality, parallelism, and symmetry. To accomplish this, I propose a method specifically designed to correct distorted symmetries in reconstructed building models. Since there exist repetitive and distorted facade elements in building models, I project these elements onto the corresponding facade plane, generating basic 2D polygonal shapes for symmetrization. Subsequently, the symmetrized shapes are reprojected back into 3D space to enhance the overall regularity of building models.



(**a**) AHN3 point clouds (**b**) MVS meshes

Figure 1.5: Two typical data sources used as input for urban building reconstruction.

In summary, given a point cloud or MVS mesh (as shown in Figure 1.5), my goal is to reconstruct the highest possible level of detail building model. The main challenge lies in ensuring the robustness of the method across various input data types, each potentially containing defects that significantly influence the design of the reconstruction pipeline. In order to address these challenges, this thesis incorporates logical geometric assumptions tailored to the unique characteristics of various types of input data, as elaborated in Chapter 3 and 4.

## 1.2. RESEARCH QUESTIONS

The main research question of this thesis is defined as follows:

*How to achieve structure-aware 3D building reconstruction with the highest possible level of detail utilizing airborne LiDAR point clouds or Multi-View Stereo (MVS) meshes as input?*

Derived from this main question, three sub-questions (SQs) are proposed to delve deeper into the study of the research.

(1) How to reconstruct LoD2 building models using airborne LiDAR point clouds respecting the regularities of buildings?

Some important structures (e.g., vertical walls) of buildings are typically missing in airborne LiDAR point clouds due to the restricted positioning and moving trajectories of airborne scanners. This data imperfection impedes the accuracy of LoD2 building re-

construction. Additionally, real-world buildings exhibit complex structures with diverse architectural styles, which further complicates the reconstruction process. The sparse and noisy LiDAR point clouds provide limited structural cues, introducing ambiguities and making it difficult to generate topologically correct surface models. Structural regularities, such as parallelism, collinearity, and orthogonality, are common features in buildings and can serve as important priors to resolve ambiguities in the reconstruction process, thereby enhancing the accuracy and robustness of LoD2 models. The reconstruction of LoD2 models constitutes a critical foundation for structure-aware 3D building reconstruction, as it provides the essential framework for airborne LiDAR data. Addressing the limitations of LiDAR point clouds—such as sparsity and missing vertical structures—ensures geometric completeness, directly supporting the main research goal of generating accurate 3D building models.

(2) How to reconstruct LoD3 building models using MVS meshes, while balancing the amount of user guideline and automated computation?

While LoD2 models provide a foundational understanding of urban buildings, the transition to LoD3 signifies a refinement towards higher fidelity representations and higher details (e.g., windows, doors, chimneys). This shift necessitates a semi-automatic approach that not only leverages advanced computational techniques but also considers user interventions to ensure accuracy and details in the final building models. I aim to bridge the gap between automated reconstruction methodologies and user interactions, thereby advancing the highest LoD in urban building reconstruction. Advancing this fulfills the main question's objective of maximizing detail while maintaining structure awareness.

(3) How to recover or enhance the regularity of reconstructed building models?

Real-life buildings commonly exhibit regular geometric properties such as orthogonality, parallelism, and symmetry. However, the data acquisition process and subsequent LoD2/LoD3 reconstruction may lead to distortion or loss of these inherent characteristics. To further improve the quality of reconstructed models, post-processing steps to recover and enhance the regularity of the reconstructed models become imperative. As mentioned in Section 1.1, to achieve structure-aware 3D building reconstruction, the generated LoD models must not only capture intricate details but also maintain structure regularity. Solving this sub-question reinforces the structure awareness of LoD2/LoD3 building models.

### INTERDEPENDENCY OF SUB-QUESTIONS

The sub-questions are not independent but rather constitute a closed-loop framework. This interconnected structure ensures a comprehensive approach to achieve structure-aware 3D building reconstruction. The relationships among the SQs are as follows:

- **SQ1 → SQ2**: LoD3 models, derived from MVS meshes, serve as a higher extension of LoD2 models, which are reconstructed from airborne LiDAR point clouds. This transition enables the incorporation of finer architectural details while maintaining consistency with the underlying geometric framework.
- **SQ2 → SQ3**: Reconstructing building models inevitably introduces geometric distortions, particularly in intricate architectural features. Regularity recovery tech-

niques are required to ensure the structural realism of these high-LoD models, forming the core objective of SQ3. Addressing these distortions is crucial for improving the consistency and accuracy of reconstructed models.

- **SQ3 → SQ1/SQ2**: The restored regularity features, such as symmetry derived from SQ3, can be reintegrated into the workflows of both SQ1 and SQ2 as a post-processing step. This feedback loop enhances the geometric and structural quality of reconstructed models, ensuring that both LoD2 and LoD3 representations benefit from improved regularity and aesthetic coherence.

## 1.3. SCOPE OF THE RESEARCH

Here, I give out the scope of our research as follows,

- This research only focuses on building reconstruction. Therefore, other city objects like vegetation and ground are excluded.
- The output models are restricted to be composed of planar shapes. Despite of the simplicity, these shapes allow us to represent a large number of urban buildings.
- The LoD2 building models should necessitate adaptability to diverse roof shapes, along with characteristics such as low weight, compactness, and watertight. In addition, these models must exhibit topological validity, ensuring the absence of self-intersections and the adherence to manifold properties.
- It is essential to include detailed elements in LoD3 building models, going beyond simple roof structures to cover important features like roof superstructures and facade components. These can comprise architectural components like dormers, windows, and chimneys.
- The symmetrization of 2D polygonal shapes should be fully automatic and robust to various inputs, enhancing the regularity of reconstructed facade elements.

## 1.4. STRUCTURE OF THIS THESIS

The organization of this thesis is briefly illustrated as follows:

- Chapter 1 introduces the motivation behind the research, the research objectives, and the scope of this thesis.
- Chapter 2 presents the essential background and the basic knowledge of this thesis.
- Chapter 3 outlines an automatic method for LoD2 building reconstruction from airborne LiDAR point clouds.
- Chapter 4 introduces a semi-automatic method for LoD3 building reconstruction from MVS meshes.
- Chapter 5 presents an automatic method for symmetrization of 2D polygonal shapes, which is intended to enhance the regularity of reconstructed facade elements.

- Chapter 6 offers a summary of conclusions and discusses potential avenues for future research.

# 2

# BACKGROUND AND PRELIMINARIES

*This chapter aims to present the background knowledge related to structure-aware 3D building reconstruction. I begin by providing a concise overview of the characteristics of 3D geographic data obtained through airborne LiDAR scanning and aerial photogrammetry in section 2.1. Furthermore, I present the relevant literature review in sections 2.2-2.3, focusing on two key aspects: coarse building reconstruction, and detailed building reconstruction. By covering these topics, I aim to establish a solid foundation and contextual understanding for the subsequent chapters of this thesis.*

## 2.1. DATA ACQUISITION

There are mainly two types of source data used in this thesis, i.e., airborne LiDAR point clouds and MVS meshes.



### 2.1.1. AIRBORNE LIDAR POINT CLOUDS

A LiDAR system uses lasers to measure distances between the sensor and the urban environment. The equipment is often mounted on an aircraft, which allows it to collect highly accurate and detailed elevation data over vast areas. The system emits laser pulses to the ground plane. When the laser beam strikes objects, it reflects back to the sensor (see Figure 2.1). The system automatically measures the time-of-flight $\Delta T$, and computes the distance $R$ between the laser sensor and the object by using,

$$R = \frac{1}{2}\Delta T c \tag{2.1}$$

where $c$ is the speed of light (approximately 300,000 km/s). The LiDAR system combines these distance measurements with the aircraft's precise position and orientation with respect to a global reference system to create a point cloud, representing the 3D spatial distribution of all objects on the Earth's surface. It has been used in various applications, including forestry analysis [6, 152, 162, 210] and building reconstruction [164, 76, 138].

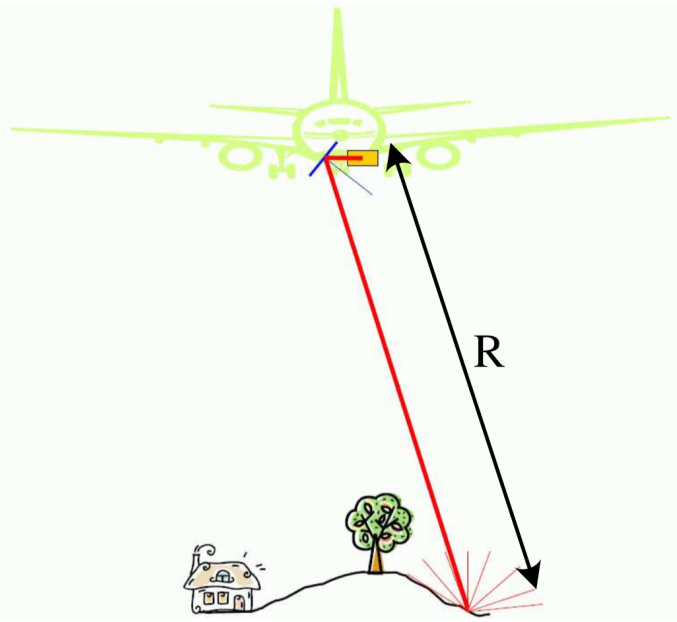



Figure 2.1: LiDAR range measurement. Figure taken from Ledoux et al. [100].

### 2.1.2. MULTI-VIEW STEREO MESHES

Aerial photogrammetry is also an important technique used to create 3D models of the Earth's surface by leveraging multiple overlapping images taken from cameras mounted

on airplanes, drones, or satellites. Before using the captured aerial images, the camera must be calibrated, which involves determining the intrinsic and extrinsic parameters [68]. Subsequently, features such as edges or corners [64] that exhibit strong identifiability across a range of images are detected. The keypoints are matched [136] between overlapping images to establish correspondences (see Figure 2.2). Once the keypoints are matched, triangulation [10] is performed to calculate the positions of the keypoints using the known camera parameters. To generate a more detailed 3D model, additional points are incorporated into the initially sparse point cloud using MVS techniques [117, 111], which rely on images captured from multiple viewpoints. MVS processes these images to estimate depth information and reconstruct the missing points, resulting in a denser point cloud. This densification step is crucial for improving the geometric detail of the 3D model, as it significantly enhances surface coverage and accuracy by filling in gaps and providing finer details that were not captured in the sparse representation. Various interpolation [79, 145] and filtering [148] techniques are utilized to estimate the 3D coordinates of additional points. Finally, surface reconstruction algorithms like Delaunay triangulation [44] or Possion reconstruction [89] are applied to create a dense mesh representation of the terrain. Applications of aerial photogrammetry include land surveying [53, 132, 21], urban planning [9, 197, 81], and environmental monitoring [33, 72, 63], which require detailed 3D models of large areas.

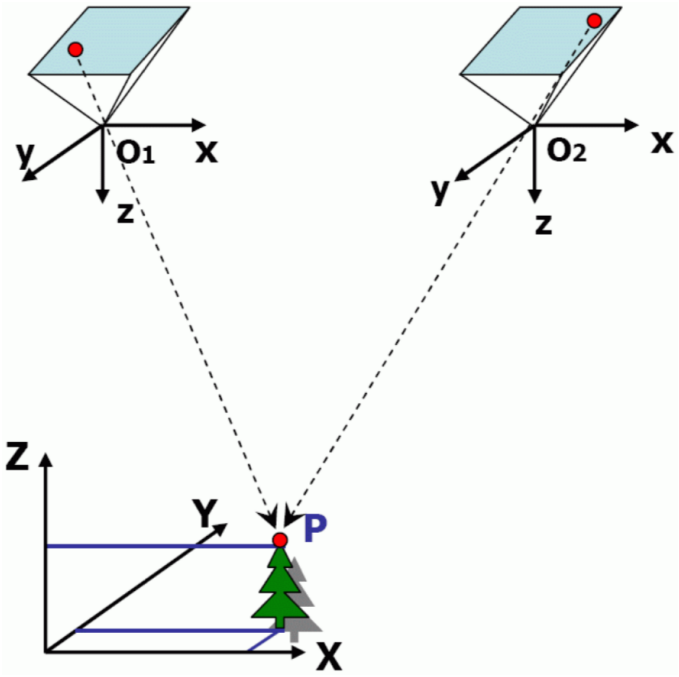



Figure 2.2: Photogrammetry measurement. Figure taken from Ledoux et al. [100].

The use of photography in generating photogrammetric point clouds allows for the inclusion of color information of the target surface. This confers an advantage over Li-

DAR, which excels at capturing elevation data but is unable to acquire color information. However, it is important to note that the elevation data derived from aerial images in photogrammetry is generally less accurate than that obtained from LiDAR. As a consequence, when used for building reconstruction, photogrammetric data may produce surface models with distortions or misalignments, particularly in roofs and facades. These inaccuracies can compromise the quality and reliability of the building models, making them less suitable for applications that require accurate geometry. To address these issues, I make some reasonable assumptions about the structures and incorporate limited high-level user interactions to generate accurate building models, as detailed in Section 4.2.

In this thesis, I use airborne LiDAR point clouds to generate city-scale LoD2 building models and MVS meshes from aerial images to reconstruct LoD3 building models.

## 2.2. COARSE BUILDING RECONSTRUCTION

Coarse building reconstruction primarily concentrates on the shape of building roofs, excluding roof superstructures and facade elements such as dormers, chimneys, and windows. It is typically associated with LoD1 and LoD2 building reconstruction. Generally, these approaches can be classified into three categories: model-driven, data-driven, and learning-based methods.

### 2.2.1. MODEL-DRIVEN METHODS

Model-driven methods of 3D building reconstruction involve pre-building a library of known roof structures and adjusting the corresponding parameters to fit the roof shapes of input data. This type of approach is based on the observation that repetitive and simple roof shapes commonly exist in urban buildings.

Verma et al. [173] introduce a method that involves extracting building points from raw LiDAR data and constructing a Roof Topology Graph (RTG) to represent the relationships between various planar segments of a complex building. The RTG helps decompose a complex roof shape into multiple simpler polygonal patches. Geometry for these patches is described by a few parameters. Finally, the segmented ground LiDAR points are smoothed and the terrain surface is created by using Delaunay triangulation. Another approach [183] introduces a graph edit dictionary to correct errors in an RTG that arise from an erroneous set of planar primitives. Lafarge et al. [96] propose a different approach, where they extract 2D building outlines from a Digital Surface Model (DSM) and decompose them into several quadrilaterals. Each quadrilateral is assigned a building shape selected from a predefined shape library. The Gibbs model is utilized to control the assembling of these blocks, and a Bayesian framework is proposed to determine the optimal configuration of 3D blocks. Their method has been validated on multiple datasets such as satellites and DSMs in a wide resolution.

In summary, a significant advantage of model-driven methods is their ability to produce topologically valid models, provided the building templates are well-defined. Such generated building models are well-suited for certain tasks, like visualization, which do not require high LoD models. An obvious limitation of model-driven methods is their reliance on predefined shape templates. The accuracy of reconstructed models is con-

strained by these templates, while real-world roof shapes often exhibit greater complexity and variability beyond what the templates can fully cover.

### 2.2.2. DATA-DRIVEN METHODS



Compared to model-driven building reconstruction methods, the results of data-driven methods depend greatly on the quality and completeness of input data. These methods are not limited to pre-defined shape templates and are ideal for reconstructing urban scenes with a variety of distinct building shapes. A typical data-driven building reconstruction method involves two main steps: planar shape detection and surface reconstruction. Since buildings are human-made structures that generally feature large planar surfaces and sharp edges, researchers are motivated to approximate building geometry using polygonal surfaces. As a result, extracting planes from input data becomes a critical step in the process. Subsequently, a compact surface representation of the building can be constructed by leveraging the extracted planar primitives.

- **Planar shape detection**

  The most commonly used methods for planar shape detection are random sample consensus (RANSAC) [55] and its variants [149, 104, 211, 4, 185, 193, 114], which are robust against noise and outliers, and methods based on region growing [143, 36, 39, 50, 133, 26], which iteratively propagate planar regions by boundary advancement. The main differences between existing region-growing methods arise from variations in their seed point initialization and region expansion criteria. Yu et al. [198] propose a refinement approach incorporating five types of operations: merging, splitting, transfer, insertion, and exclusion, to enhance the planar segmentation results. Zhang et al. [201] introduce a method for automatically segmenting and fitting surfaces with accurate geometric parameters from 3D models. The approach involves employing a novel clustering algorithm to create superfaces based on local 3D geometric information, followed by an efficient MRF framework to partition the model into meaningful surface patches. Additionally, an iterative optimization algorithm is presented for fitting rolling-ball blending patches by recovering rolling center trajectory and ball radius parameters. The 3D Hough transform algorithms [23, 142, 78] employ a voting mechanism, wherein all possible plane instances accumulate votes from input points. Detected shapes are those that receive the highest number of votes [165]. This type of method is particularly effective in handling dense point clouds.

  As deep learning techniques have gained more and more popularity, various deep learning-based segmentation methods have also been proposed for planar shapes. Li et al. [105] introduce SPFN, which utilizes PointNet++ [140] to extract point-wise features for individual points, enabling the estimation of the planar primitive types and parameters. CPFN [98] employs an adaptive patch sampling network to combine detection results from global and local primitives at different scales. Yan et al. [191] integrate three learned features with adaptive weights and utilize a mean-shift clustering module to extract primitives. PrimitiveNet [77] jointly trains an embedding network and a discriminator to learn local surface properties. Li et al. [113] present a novel surface and edge detection network (SED-Net) for accu-

rate geometric primitive fitting of point clouds. The key idea is to learn parametric surfaces and edges simultaneously that can be assembled into a regularized and seamless Computer-Aided Design (CAD) model in one single framework. The core is a two-stage feature fusion mechanism that can fully utilize the type, edge, and geometry features. Overall, learning-based methods have shown limitations in handling real-world data, mainly because the training datasets differ from real scenarios as they lack defects and often consist of synthetic CAD models. In this thesis, I employ the variational shape approximation method [45] to extract coarse planar primitives given its simplicity and robustness.

- **Surface reconstruction**

  Zhou et al. [206] achieve this by simplifying the 2.5D triangulated irregular network (TIN) of buildings, which may result in artifacts in building contours due to its limited capability in capturing complex topology. To address this issue, the authors propose an extended 2.5D contouring method with improved topology control [205]. To cope with missing walls, Chauve et al. [35] incorporate additional primitives inferred from input point clouds. Given that buildings typically exhibit piece wise planar regions, many methods [109, 129, 15, 54, 85, 106, 194] have been proposed to obtain an arrangement of extracted planar primitives to represent the building geometry. These methods first detect a set of planar primitives from input point clouds and then hypothesize a set of polyhedral cells or polygonal faces using the supporting planes of extracted planar primitives, as shown in Figure 2.3. Finally, a compact polygonal mesh is extracted from hypothesized cells or faces by minimizing an energy function incorporating the terms of data fidelity and complexity. The details about solving this formulated integer programming problem can be found in the Appendix. These methods focus on the assembly of detected planar primitives, for which obtaining a complete set of planar primitives from airborne LiDAR point clouds is still a challenge.

  In this thesis, I extend an existing hypothesis-and-selection-based polygonal surface reconstruction method [129] to reconstruct buildings from airborne LiDAR point clouds. Based on the observation that urban buildings typically consist of planar roofs connected to vertical walls to the ground, I propose an approach to infer the vertical walls directly from input data. With the planar segments of both roofs and walls, I hypothesize the faces of the building surface. The final model is obtained by minimizing an extended energy function for LoD2 building reconstruction, as detailed in Section 3.2.3.

### 2.2.3. LEARNING-BASED METHODS

As the popularity of deep learning techniques continues to grow, there is a rise in the development of learning-based approaches for building reconstruction. Li et al. [103] present an end-to-end framework for constructing 3D building roof models from airborne LiDAR point clouds. They formulate the task of building roof modeling as a problem of vertex detection and edge prediction, which can be effectively addressed using deep neural networks. However, this method is constrained to processing simple building structures, due to the limited complexity of the synthetic dataset used for its training.

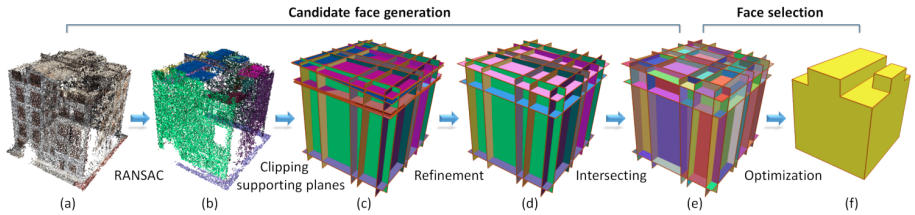



Figure 2.3: Illustrations of polygonal surface reconstruction from point clouds. Figure taken from Nan et al. [129].

Chen et al. [42] introduce a learning-based framework employing a deep implicit field for building reconstruction. A Markov Random Field (MRF) formulation is also proposed to extract surface from the occupancy learned by a neural network. Additionally, Chen et al. [41] propose PolyGNN, a polyhedron-based graph neural network designed for reconstructing compact polygonal building models from point clouds. It learns to assemble the polyhedra to achieve watertight and compact reconstruction, which is formulated as an end-to-end graph node classification problem. To enhance the representation of arbitrary-shaped polyhedra within the neural network, they introduce three different sampling strategies for selecting representative points as polyhedron-wise queries, enabling efficient occupancy inference. Liu et al. [118] employ an autoregressive model to sequentially predict mesh elements conditioned on input point clouds. Their network comprises two main modules: a vertex module that includes a point cloud feature extractor and a block for vertex generation, and a face module that sequentially generates mesh faces connecting the vertices to form the object surface. Akwensi et al. [7] propose a dynamic multi-scale attention-based network that integrates both local and global semantic information for generating complete building shapes from partial airborne LiDAR point clouds. They further introduce a neural building reconstruction framework to handle various building styles and complexities. The aforementioned LoD2 building reconstruction methods cannot consistently ensure the manifoldness or watertightness of reconstructed building models. In conclusion, existing learning-based methods are limited by the scarcity of large-scale datasets, which constrains their effectiveness and generalization. In addition, it is hard to add hard constraints to the results to ensure geometric correctness. To address these challenges and advance the development of learning-based methods, I propose an automatic building reconstruction approach and introduce a new LoD2 building dataset, as detailed in Section 3.2.

## 2.3. DETAILED BUILDING RECONSTRUCTION

To obtain more detailed building models, it is necessary to reconstruct both facade elements and roof superstructures, including windows, dormers, and chimneys. Furthermore, upgrading models from LoD2 to LoD3 requires improving the regularity of building details, which are often compromised or distorted during the reconstruction process. This section primarily reviews related works on building detail reconstruction and regularity enhancement.

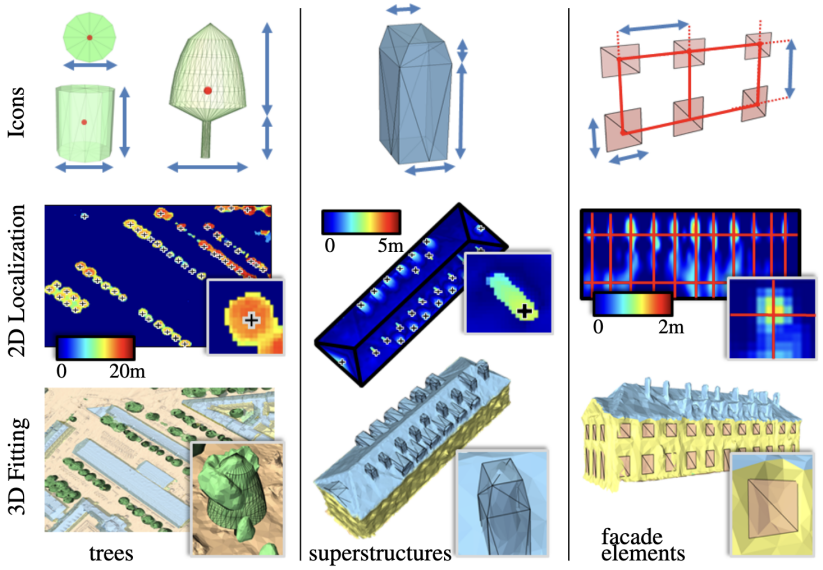



Figure 2.4: Iconization results of roof superstructures and facade elements. Figure taken from Verdie et al. [172].

### 2.3.1. BUILDING DETAIL RECONSTRUCTION

Bredif et al. [28] present an automatic method for the detection and detailed reconstruction of 3D building models that include roof superstructures from a very high-resolution Digital Elevation Model (DEM). The roof plane geometry and the set of superstructures are optimized alternatively by minimizing a Minimum Description Length (MDL) energy function. Verdie et al. [172] introduce a method that reconstructs entire urban scenes for different levels of detail using raw MVS meshes as input. This approach consists of three main steps: classification, abstraction, and reconstruction. For LoD3 building models, architectural details such as windows and doors are added through ionization (as shown in Figure 2.4), whereby the geometry best represents the detected objects. Nevertheless, the simple abstraction of facade elements and roof superstructures using the predefined templates limits the applicability of this method to only specific types of buildings.

Recognizing the limitations of fully automatic reconstruction techniques, interactive methods have been proposed to achieve more detailed urban building reconstruction. Nan et al. [130] propose an interactive system called SmartBox, designed for rapidly modeling architectural buildings from LiDAR point clouds. The system assumes Manhattan world scenes and excels in reconstructing facades with repetitive axis-aligned structures. Arikan et al. [11] propose an interactive method for reconstructing polygonal models from unorganized point clouds. This method enables users to iteratively refine the surface and automatically snap the components together through optimization. However, it requires a significant amount of human intervention, particularly when

dealing with complex building models.

Airborne point clouds and MVS meshes typically exhibit the 3D structures of building models, but are often incomplete or noisy. Meanwhile, imagery, as a 2D data source, provides high-resolution and dense information. By integrating these diverse inputs, the reconstruction process can be enhanced, generating more detailed and accurate building models. For example, Nan et al. [131] propose a novel framework to reconstruct building details by assembling 3D templates on coarse building models. Firstly, an initial coarse model approximating a point cloud and a set of 3D templates of facade elements are created. Next, the initial coarse model is optimized to enforce consistency between geometry and appearance. Then, building details are reconstructed by assembling templates on the facades of the coarse model. The 3D templates are automatically located and added through an optimization-based template assembly algorithm. Wen et al. [178] present a LoD3 building reconstruction method using point clouds and oblique remote sensing images as input. Building plane primitives are extracted using the RANSAC algorithm and used as constraints during feature line extraction from oblique images. These feature lines improve the reconstruction accuracy of building outlines. Finally, interactive topology editing and texture models are used to refine building models.

Compared to these methods, my method does not rely on specific assumptions regarding the shape of roof surfaces or multi-source input data, and it only requires limited high-level guidance from the user, as elaborated in Section 4.2.1.

### 2.3.2. REGULARITY ENHANCEMENT

Structural regularities, including orthogonality, parallelism, and symmetry, are essential characteristics of building models. Numerous methods have been proposed to enhance or restore these lost regularities. These methods can be categorized into four classes: building contour regularization, facade layout regularization, symmetry detection, and symmetry transform.

- **Building contour regularization**

  Building contours, recognized as crucial architectural elements, can be derived from detected roof primitives. However, these coarse building contours are often compromised by noise. Collinearity and rectangularity are the most commonly used constraints to refine the building contours. Meng et al. [121] propose a method that projects points onto a 2D grid map and extracts building contours by using a morphological operator. Chen et al. [36] introduce a novel algorithm that calculates roof primitive boundaries via Voronoi subgraph. To obtain a simplified outline shape, line simplification techniques such as the Douglas-Peucker algorithm [51] have been widely used to reduce the number of vertices [200, 110, 184]. This algorithm decomposes outlines into line segments by iteratively identifying points with the maximum distance to the simplified segments. If the distance of a point to the approximating segments falls below a threshold, that point is discarded. This recursion continues until the distances of all points to the corresponding simplified segments meet the threshold criterion. Zhou and Neumann [207] calculate the principal directions of a building and then enforce the roof boundary segments along with these directions. Jarzabek-Rychard et al. [82] employ

RANSAC to detect line segments in a height map generated from the input points. The boundary lines are then regularized by merging closely parallel segments and adjusting angles based on the main orientation, calculated from the longest line segments. Rectangularity or parallelism constraint is enforced, and different orientations or angles are ignored. He et al. [70] delineate the initial building boundary using the $\alpha$-shape algorithm and utilize a vertex-driven Douglas-Peucker method to generate polygonal hypotheses. An energy function is employed to evaluate these hypotheses, and the optimal polygon is chosen by minimizing the whole energy. Albers et al. [8] first extract raw building contours from input point clouds based on the $\alpha$-shape algorithm and then employ the Hough transform to determine the main directions of a building. The boundary line segments are regularized by minimizing a direction-constrained energy function. Sohn et al. [159] present a global optimization method to refine building boundaries. It minimizes a function that incorporates terms for model approximation and boundary complexity. Yi et al. [196] propose a contour extraction method from 2D sectional points based on a spectral residual clustering algorithm. Their results demonstrate its robustness to occlusions and noisy corruption. To enhance the regularity of building contours, Zhu et al. [209] introduce a two-stage scheme. They reorient segments that are nearly parallel to the facade directions to achieve alignment and then merge coplanar segments into one. Li et al. [108] simplify the contours using the Douglas-Peucker algorithm [51], followed by a normal-guided refinement for contour vertices. Soft collinearity and rectangularity constraints are enforced during the refinement process.

- **Facade layout regularization**

  Facade layout regularization refers to improving regularities in layouts consisting of a certain number of architectural elements. This problem arises commonly in user-created contents, such as room designs and facades, in which specifying the precise relationships among the elements is tedious and time-consuming. Motivated by the fact that human can unambiguously identify desired layouts of graphic elements by viewing all elements as a whole, Xu et al. [189, 188] present a new user interface for visualizing and editing the inferred relationships in a global way. It significantly improves the efficiency compared to the traditional interactive snap-dragging and command-based alignment tools for 2D and 3D layout regularization tasks. To automate layout regularization, Jiang et al. [83] formulate constraint detection as an integer program. They improve the imperfect layout by detecting and subsequently enforcing the desired constraints, such as alignment, same size, and equal distance between elements. Following that, Jiang et al. [84] present an automatic method specially for symmetrization of the layouts of building facades. This method also follows a two-stage pipeline: symmetry detection and optimization. They optimize the structural abstractions extracted from images and focus on symmetrizing the original layouts while minimizing the modifications. The regularity of the layout is enhanced by redistributing and aligning elements. These methods are designed mainly for regularizing the overall layout consisting of a set of man-made elements, and thus cannot be directly applied for

the symmetrization of a single shape.

Several interactive beautification methods have also been proposed, advancing the enhancement of facade layouts. Igarashi et al. [80] introduce a new interactive system called Pegasus to facilitate rapid geometric design, contributing to the beautification of facade elements. It can receive the freestrokes provided by users and beautify them by considering the geometric constraints among segments. Orbay et al. [134] present a new technique for turning digital design sketches into polished line drawings. This method involves a trainable stroke clustering technique that groups strokes into curves and orders them for smoother drawing, providing designers with more freedom and less structure. Fišer et al. [56] propose ShipShape, a tool that enhances freehand sketches by automatically correcting geometric relations without requiring advanced drawing skills or knowledge of software. Parakkat et al. [137] propose an algorithm that groups rough strokes drawn by users and represents them with simple curves. The algorithm uses Delaunay triangulation to group the strokes, identifies open curves, and reconstructs broken strokes. These methods demonstrate the potential of interactive beautification techniques in enhancing the quality of facade designs and improving the user experience. In contrast to these interactive methods, my research proposes a fully automatic method to achieve symmetrization of 2D polygonal shapes, enhancing the regularity of reconstructed facade elements.

- **Symmetry detection**

  Sophisticated strategies are commonly exploited to search for the optimal symmetry in a given shape. They typically require either brute-force validation [93] or random sample consensus [16] of the potential symmetry candidates to determine the optimal symmetry configuration, which is computationally expensive due to the large search space for exploration and is also sensitive to noise and outliers. To improve efficiency, local shape descriptors (mostly curvature-based) [124, 30, 155, 22, 101] are proposed to significantly reduce the search space. These methods take advantage of the rich geometric properties of shapes that are invariant under the considered symmetry transformations. For example, Mitra et al. [124] propose to match simple local shape signatures in pairs and use these matches to accumulate evidence for symmetry detection in the transformation space, as shown in Figure 2.5. After that, potential symmetric matches are clustered to form significant symmetries of the shape, from which the strongest symmetric match gives the final symmetry. This method relies on curvature estimation for point matching and candidate filtering, which is sensitive to noise and some user-specified parameters (e.g., patch radii). Shi et al. [155] adopt the same framework and introduce a more robust metric in the transformation space. Cailliere et al. [30] improve this method for global symmetry plane detection by replacing the clustering step with Hough transform to obtain better results with higher computation efficiency. It is worth noting that even with feature-based pruning of an initial set of symmetry candidates, the procedure to determine the optimal symmetry can still be time-consuming [58]. Other methods are also proposed for detecting symmetry axes or planes. An ICP-based algorithms for symmetry plane detection in point clouds

is used in [46, 52]. Sipiran et al. [158] propose a method for symmetry plane detection on incomplete objects. It uses feature extraction based on the Heat Kernel Signature descriptor [163] and a voting scheme. For shapes that have few high-curvature points, this method tends to fail because the descriptor has difficulty recognizing the symmetric features, which limits its applicability. Hruda et al. [73] introduce a novel differentiable measure for symmetry plane detection, where a fast gradient-based optimization is utilized to find symmetry in a given shape. This method does not perform well for non-uniformly sampled point clouds and may fail in the presence of scattered outliers. Ruchay et al. [147] determine the optimal symmetry plane of 3D point clouds with the help of a modified Hausdorff metric.

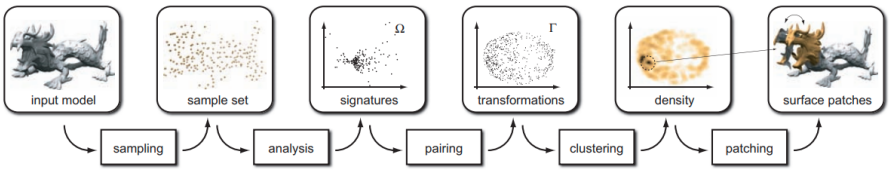



Figure 2.5: The pipeline of symmetry detection. Figure taken from Mitra et al. [124].

The above works concentrate on detecting extrinsic symmetries, which are defined as invariance under rigid transformations and scaling. At the same time, a wide class of deformations, such as articulated motion in humans, preserve the object's internal structure. These deformations leave intact intrinsic symmetries of an object. Therefore, there have also been advances in the field of intrinsic symmetry detection [202, 135, 187, 186, 175]. The work of Mitra et al. [125] mainly deals with shapes that are close to being intrinsically symmetric. Zheng et al. [202] develop a symmetrization method for intrinsically asymmetric shapes, extract and enhance the approximate intrinsic symmetries therein. It measures intrinsic distances over a curve skeleton, symmetrizes the skeleton, and then propagates the symmetrization to the shape.

Specific symmetry detection methods have also been proposed for specific applications. Xue et al. [190] introduce a derivative-free optimization-based approach for detecting architectural symmetries from point clouds. Haunert et al. [69] introduce a symmetry detector specially for urban-space analysis. These approaches rely on local geometric features or shape descriptors (mostly curvature-based) for symmetry detection, and their results highly depend on the quality of handcrafted features or descriptors, making it sensitive to the varying quality of the input data.

As deep learning techniques have gained popularity in recent years, numerous learning-based methods for symmetry detection or application of symmetry have been proposed. Shi et al. [154] introduce an end-to-end deep neural network to detect both reflectional and rotational symmetry of 3D shapes from single-view RGB-D images. They use a multi-task learning approach to avoid overfitting, and the network is trained to predict symmetry correspondences as well. Gao et al. [59] propose a new learning framework that uses a 3D convolutional neural network

to automatically discover planar reflective symmetry of a 3D shape. They introduce a dedicated symmetry distance loss and regularization loss to prevent the generation of duplicate symmetry planes. Qiao et al. [141] propose a learning-based method for detecting intrinsic reflectional symmetry using a functional map matrix that is computed based on the signs of Laplacian eigenfunctions. Seo et al. [151] introduce EquiSym, a group-equivariant convolutional network for symmetry detection that leverages equivariant feature maps with respect to a dihedral group of reflection and rotation. Moreover, Shi et al. [153] propose a 3D symmetry detection method that uses weakly supervised learning to detect symmetry from single-view RGB-D images and generate plausible shapes using a symmetry-aware shape prior. These learning-based methods typically require expensive data preparation and learning processes. In contrast, my symmetrization method can be integrated into computer-aided design software as a plug-in, enabling its immediate use without significant additional preparation.

- **Symmetry transform**

  In computer vision, several general symmetry transforms defined on all pixels of an in image have been proposed for robust detection of rotational symmetry in natural images. Reisfeld et al. [144] define a generalized symmetry transform for local symmetries, and several variants are proposed in [17, 43]. These methods have been proven to be effective in finding the local symmetries in noisy images. Podolak et al. [139] introduce a planar reflective symmetry transform (PRST) mainly for 3D meshes that captures a continuous measure of the symmetry with respect to all possible planes. An efficient Monte Carlo sampling algorithm [139] is also proposed to compute the transform for surfaces. This method requires rasterizing the input objects, which is usually computationally expensive. Furthermore, the initial results are not precise enough, and a post-processing step called Iterative Symmetric Points (ISP) is required for refinement. Inspired by this method, Xu et al. [187] introduce a voting scheme to compute an intrinsic reflectional symmetry axis (IRSA) of a closed manifold mesh. It is robust thanks to its statistical nature and the aid of a modified region growing method and an iterative refinement step. However, the voting scheme may fail when symmetry is present on relatively small parts of a complex model.

  Compared to existing methods relying on a dedicated step to explicitly detect symmetry, my symmetrization framework tackles symmetry detection and symmetry optimization jointly in a single optimization step, which can avoid the impact of ambiguities or errors in symmetry detection on the subsequent optimization. The details are elaborated in Section 5.2.1.

# 3

# LoD2 Building Reconstruction from Airborne LiDAR Point Clouds

*This chapter presents a fully automatic approach that I developed for reconstructing compact 3D building models from large-scale airborne point clouds. A major challenge of urban reconstruction from airborne LiDAR point clouds is that points on the vertical walls are often missing. Based on the observation that urban buildings typically consist of planar roofs connected to vertical walls to the ground, I propose an approach to infer the vertical walls directly from the data. With the planar segments of both roofs and walls, I hypothesize the faces of the building surface. The final model is obtained by using an extended hypothesis-and-selection-based polygonal surface reconstruction framework. Specifically, I introduce a new energy term to encourage roof preferences and two additional hard constraints into the optimization step to ensure correct topology and enhance detail recovery. Experiments on various large-scale airborne LiDAR point clouds have demonstrated that the method is superior to the state-of-the-art methods in terms of reconstruction accuracy and robustness. In addition, I have generated a new dataset with my method consisting of the point clouds and 3D models of 20k real-world buildings. This dataset can stimulate research in urban reconstruction from airborne LiDAR point clouds and the use of 3D city models in urban applications.*

This chapter has been published in Remote Sensing [76].

## 3.1. Introduction

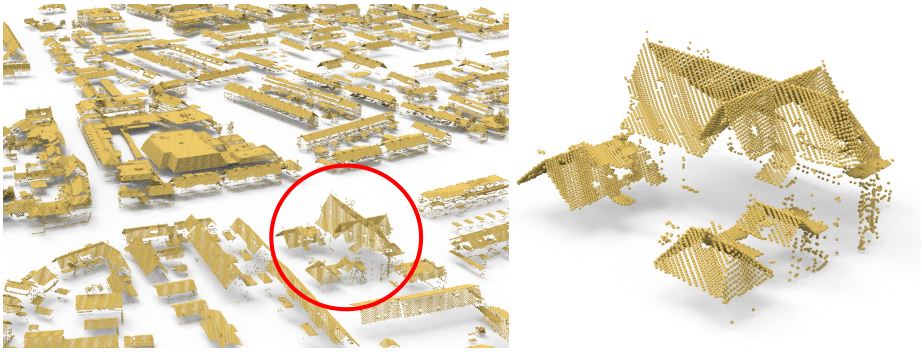
Digitizing urban scenes is an important research problem in computer vision, computer graphics, and photogrammetry communities. High-accuracy and compact LoD2 building models have become the infrastructure for a variety of applications. As seen in Chapter 2, existing building reconstruction methods strive to bring in a great level of detail and automate the process for large-scale urban environments. Interactive reconstruction techniques are successful in reconstructing accurate 3D building models with great detail [130, 131], but they require either high-quality laser scans as input or considerable amounts of user interaction. These methods can thus hardly be applied to large urban scenes. To facilitate practical applications that require large-scale 3D building models, researchers have attempted to address the reconstruction challenge using various data sources [204, 66, 172, 110, 29, 14, 108, 99]. Existing methods based on aerial images [66, 110, 29] and dense triangle meshes [172] require good coverage of the buildings, which imposes challenges in data acquisition [208]. Approaches based on airborne LiDAR point clouds alleviate data acquisition issues. However, the accuracy and geometric details are usually compromised [204, 14, 108, 99]. Following previous works using widely available airborne LiDAR point clouds, I strive to recover desired geometric details of real-world buildings while ensuring topological correctness, reconstruction accuracy, and good efficiency.

The challenges for large-scale urban reconstruction from airborne LiDAR point clouds include:

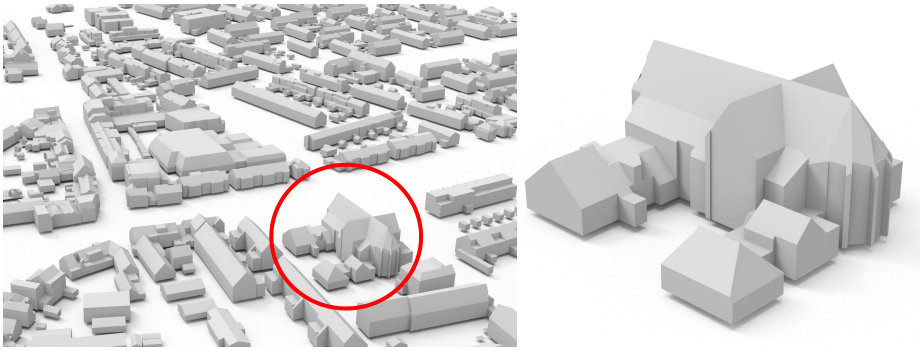
- Building instance segmentation. Urban scenes are populated with diverse objects, such as buildings, trees, city furniture, and dynamic objects (e.g., vehicles and pedestrians). The cluttered nature of urban scenes poses a severe challenge to the identification and separation of individual buildings from the massive point clouds. This has drawn considerable attention in recent years [140, 168].
- Incomplete data. Some important structures (e.g., vertical walls) of buildings are typically not captured in airborne LiDAR point clouds due to the restricted positioning and moving trajectories of airborne scanners.
- Complex structures. Real-world buildings demonstrate complex structures with varying styles. However, limited cues about structure can be extracted from the sparse and noisy point clouds, which further introduces ambiguities in obtaining topologically correct surface models.

In this work, I address the above challenges with the following strategies. Firstly, I address the building instance segmentation challenge by separating individual buildings using increasingly available vectorized building footprint data. Secondly, I exploit prior knowledge about the structures of buildings to infer their vertical planes. Based on the fact that vertical planes in airborne LiDAR point clouds are typically walls connecting the piecewise planar roofs to the ground, I propose an algorithm to infer the vertical planes from incomplete point clouds. My method has the option to extrude outer walls directly from the given building footprint. Finally, I approach surface reconstruction by introducing the inferred vertical planes as constraints into an existing hypothesis-and-selection-based polygonal surface reconstruction framework [129], which favors good

fitting to the input point cloud, encourages compactness, and enforces manifoldness of the final model (see Figure 3.1 for an example of the reconstruction results). The main contributions of this work include:



(**a**) Input airborne LiDAR point cloud



(**b**) My reconstruction result

Figure 3.1: The automatic reconstruction result of all the buildings in a large scene from the AHN3 dataset [5].

- A robust framework for fully automatic reconstruction of large-scale urban buildings from airborne LiDAR point clouds.
- An extension of an existing hypothesis-and-selection-based surface reconstruction method for buildings, which is achieved by introducing a new energy term to encourage roof preferences and two additional hard constraints to ensure correct topology and enhance detail recovery.
- A novel approach for inferring vertical planes of buildings from airborne LiDAR point clouds, for which I introduce an optimal-transport method to extract polylines from 2D bounding contours.
- A new dataset consisting of the point clouds and reconstructed surface models of 20k real-world buildings, available for researchers and urban practitioners.

## 3.2. Methodology

### 3.2.1. Overview

The proposed approach takes as input a raw airborne LiDAR point cloud of a large urban scene and the corresponding building footprints, and it outputs 2-manifold and watertight 3D polygonal models of the buildings in the scene, regardless of the density of the point cloud. Figure 3.2 shows the pipeline of the proposed method. It first extracts the point clouds of individual buildings by projecting all points onto the ground plane and collecting the points lying inside the footprint polygon of each building. Then, I reconstruct a compact polygonal model from the point cloud of each building.

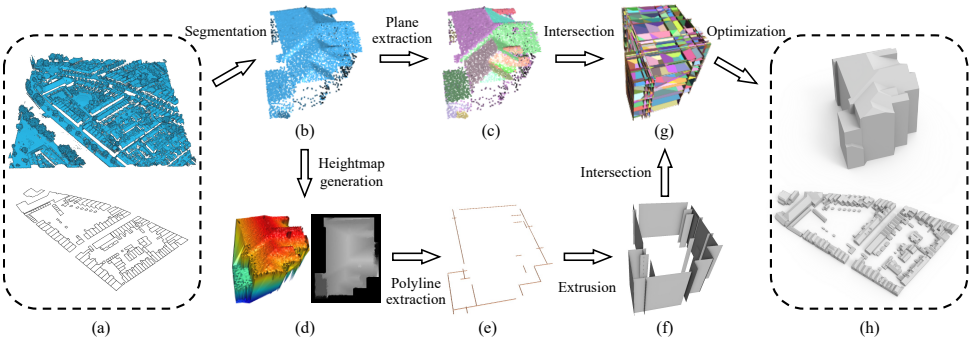



Figure 3.2: The pipeline of the proposed method (only one building is selected to illustrate the workflow). (**a**) Input point cloud and corresponding footprint data. (**b**) A building extracted from the input point cloud using its footprint polygon. (**c**) Planar segments extracted from the point cloud. (**d**) The heightmap (right) generated from the TIN (left, colored as a height field). (**e**) The polylines extracted from the heightmap. (**f**) The vertical planes obtained by extruding the inferred polylines. (**g**) The hypothesized building faces generated using both the extracted planes and inferred vertical planes. (**h**) The final model obtained through optimization.

My reconstruction of a single building is based on the hypothesis-and-selection-based framework of PolyFit [129], which is for reconstructing general piecewise-planar objects from a set of planar segments extracted from the point cloud. My method exploits not only the planar segments directly extracted from the point cloud but also the vertical planes inferred from the point cloud. From these two types of planar primitives, I hypothesize the faces of the building. The final model is then obtained by choosing the optimal subset of the faces through optimization.

The differences between my method and PolyFit are: (1) my method is dedicated to reconstructing urban buildings, and it makes use of vertical planes as hard constraints, for which I propose a novel algorithm for inferring the vertical planes of buildings that are commonly missing in airborne LiDAR point clouds. (2) I introduce a new *roof preference* energy term and two additional hard constraints into the optimization to ensure correct topology and enhance detail recovery.

In the following sections, I detail the key steps of my method with an emphasis on the processes that differ from PolyFit [129].

### 3.2.2. INFERENCE OF VERTICAL PLANES

With airborne LiDAR point clouds, important structures like vertical walls of a building are commonly missed due to the restricted positioning and moving trajectories of the scanner. In contrast, the roof surfaces are usually well captured. This inspired me to infer the missing walls from the available points containing the roof surfaces. I infer the vertical planes representing not only the outer walls but also the vertical walls within the footprint of a building. I achieve this by generating a 2D rasterized height map from its 3D points and looking for the contours that demonstrate considerable variations in the height values. To this end, an optimal-transport method is proposed to extract closed polylines from the contours. The polylines are then extruded to obtain the vertical walls. The process for inferring the vertical planes is outlined in Figure 3.2 (d)–(f).



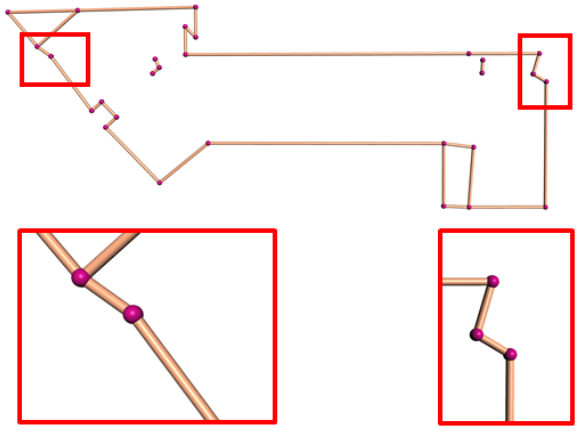
After obtaining the point cloud of a building, I project the points onto the ground plane, from which I create a height map. To cope with the non-uniform distribution of the points (e.g., some regions have holes while others may have repeating points), I construct a TIN model using 2D Delaunay triangulation. The TIN model is a continuous surface and naturally completes the missing regions. Then, a height map is generated by rasterizing the TIN model with a specified resolution $r$. The issue of small holes in the height maps (due to uneven distribution of roof points) is further alleviated by image morphological operators while preserving the shape and size of the building [74]. After that, a set of contours is extracted from the height map using the Canny detector [32], which serves as the initial estimation of the vertical planes. I propose an optimal-transport method to extract polylines from the initial set of contours.

**Optimal-transport method for polyline extraction**. The initial set of contours are discrete pixels, denoted as $S$, from which I would like to extract simplified polylines that best describe the 2D geometry of $S$. My optimal-transport method for extracting polylines from $S$ works as follows. First, a 2D Delaunay triangulation $T_0$ is constructed from the discrete points in $S$. Then, the initial triangulation $T_0$ is simplified through iterative edge collapse and vertex removal operations. In each iteration, the most suitable vertex to be removed is determined in a way such that the following conditions are met:

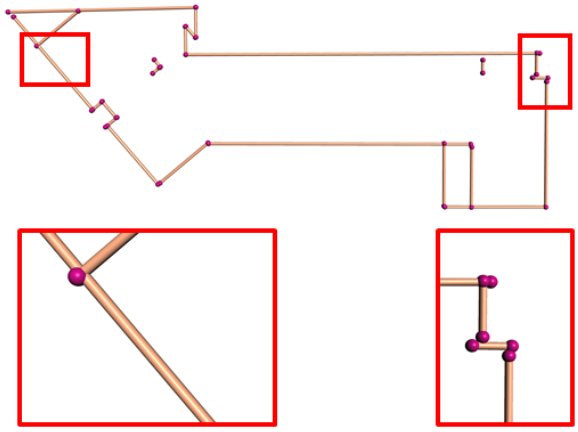
- The maximum Hausdorff distance from the simplified mesh $T_0$ to $S$ is less than a distance threshold $\epsilon_d$.
- The increase of the total transport cost [48] between $S$ and $T_0$ is kept at a minimum.

In each iteration, a vertex satisfying the above conditions is removed from $T_0$ by edge collapse, and the overall transportation cost is updated.

As the iterative simplification process continues, the overall transportation cost will increase. The edge collapse operation stops until no vertex can be further removed, or the overall transportation cost has increased above a user-specified tolerance $\epsilon_c$. After that, I apply an edge filtering step [48] to eliminate small groups of undesirable edges caused by noise and outliers. Finally, the polylines are derived from the remaining vertices and edges of the simplified triangulation using the procedure described in [48]. Compared to [48], my method not only minimizes the total transport cost but also provides control over local geometry, ensuring that the distance between every vertex in the final polylines and the initial contours is smaller than the specified distance threshold $\epsilon_d$.



(**a**) Before (28 segments)



(**b**) After (22 segments)

Figure 3.3: The effect of the clustering-based regularity enhancement on the polylines inferring the vertical walls. (**a**) Before regularity enhancement. (**b**) After regularity enhancement.

**Regularity enhancement**. Due to noise and uneven point density in the point cloud, the polylines generated by the optimal-transport algorithm are unavoidably inaccurate and irregular (see Figure 3.3 (a)), which often leads to artefacts in the final reconstruction. I alleviate these artefacts by enforcing structure regularities that commonly dominate urban buildings. I consider the structure regularities, namely parallelism, collinearity, and orthogonality, defined by [112]. Please note that since all the lines will be extruded vertically to obtain the vertical planes, the verticality regularity will inherently be satisfied. I propose a clustering-based method to identify the groups of line segments that potentially satisfy these regularities. My method achieves structure regularization in two steps: clustering and adjustment.

*Clustering.* In this work, I cluster the line segments of the polylines generated by the optimal-transport algorithm based on their orientation and pairwise Euclidean distance [150]. The pairwise Euclidean distance is measured by the minimum distance between a line segment and the supporting line of the other line segment.

*Adjustment.* For each cluster that contains multiple line segments, I compute its average direction. Then each line segment in the cluster is adjusted to align with the average direction. In case the building footprint is provided, the structure regularity can be further improved by aligning the segments with the edges in the footprint. After average adjustment, the near-collinear and near-orthogonal line segments are adjusted to be perfectly collinear and orthogonal, respectively (I use an angle threshold of 20°).

After regularity enhancement, the vertical planes of the building can be obtained by vertical extrusion of the regularized polylines. The effect of the regularity enhancement is demonstrated in Figure 3.3, from which I can see that it significantly improves structure regularity and reduces the complexity of the building outlines. Here, we provide the option to incorporate vertical planes from the footprint data or to omit them. The impact of utilizing these planes is analyzed in Section 3.3.5.

### 3.2.3. RECONSTRUCTION

My surface reconstruction involves two types of planar primitives, i.e., vertical planes inferred in the previous step (see Section 3.2.2) and roof planes directly extracted from the point cloud. Unlike PolyFit [129] that hypothesizes faces by computing pairwise intersections using all planar primitives, I compute pairwise intersections using only the roof planes, and then the resulted faces are cropped with the outer vertical planes (see Figure 3.2 (g)). This process ensures that the roof boundaries of the reconstructed building can be precisely connected with the inferred vertical walls. Additionally, since the object to be reconstructed is a real-world building, I introduce a *roof preference* energy term and a set of new hard constraints specially designed for buildings into the original formulation. My reconstruction is obtained by finding the optimal subset of the hypothesized faces. I formulate this as an optimization problem, with an objective function consisting of three energy terms: *data fitting*, *model complexity*, and *roof preference*. Specifically, my objective for obtaining the model faces $F^*$ can be written as

$$F^* = \arg\min_X \lambda_d E_d + \lambda_c E_c + \lambda_r E_r, \tag{3.1}$$

where $X = \{x_i | x_i \in \{0,1\}\}$ denotes the binary variables for the faces (1 for *selected* and 0 otherwise). $E_d, E_c, E_r$ denotes data fitting term, model complexity term, and roof refer-

ence term. The first two terms are the same as in [129]. In the following, I introduce all these terms and provide the final complete formulation.

- **Data fitting**. It is defined to measure how well the final model (i.e., the assembly of the chosen faces) fits to the input point cloud. It is used to encourage selecting faces supported by more points,

$$E_d = 1 - \frac{1}{|P|} \sum_{i=1}^{|F|} x_i \cdot support(f_i), \tag{3.2}$$

  where $|P|$ is the number of points in the point cloud. $support(f_i)$ measures the number of points that are $\epsilon$-close to a face $f_i \in F$ (i.e., points $p$ satisfying $dist(p, f_i) < \epsilon$), and $x_i \in \{0, 1\}$ denotes the binary status of the face $f_i$ (1 for *selected* and 0 otherwise). $|F|$ denotes the total number of hypothesized faces.

- **Model complexity**. To avoid defects introduced by noise and outliers, this term is introduced to encourage large planar structures, and favors simple planar structures.

$$E_c = \frac{1}{|E|} \sum_{i=1}^{|E|} corner(e_i), \tag{3.3}$$

  where $|E|$ denotes the total number of pairwise intersections in the hypothesized face set. $corner(e_i)$ is an indicator function denoting if choosing two faces connected by an edge $e_i$ results in a sharp edge in the final model (1 for *sharp* and 0 otherwise).

- **Roof preference**. I have observed in rare cases that a building in aerial point clouds may demonstrate more than one layer of roofs, e.g., semi-transparent or overhung roofs. In such a case, I assume a higher roof face is preferable to the ones underneath. I formulate this preference as an additional *roof preference* energy term,

$$E_r = \frac{1}{|F|} \sum_{i=1}^{|F|} x_i \cdot \frac{z_{max} - z_i}{z_{max} - z_{min}} \tag{3.4}$$

  where $z_i$ denotes the $Z$ coordinate of the centroid of a face $f_i$. $z_{max}$ and $z_{min}$ are, respectively, the highest and lowest $Z$ coordinates of the building points.

**Hard constraints**. I impose two hard constraints to enhance the topological correctness of the final reconstruction.

- *Single-layer roof.* This constraint ensures that the reconstructed 3D model of a real-world building has a single layer of roofs, which can be written as,

$$\sum_{k \in V(f_i)} x_k = 1, (1 \le i \le |F|)$$

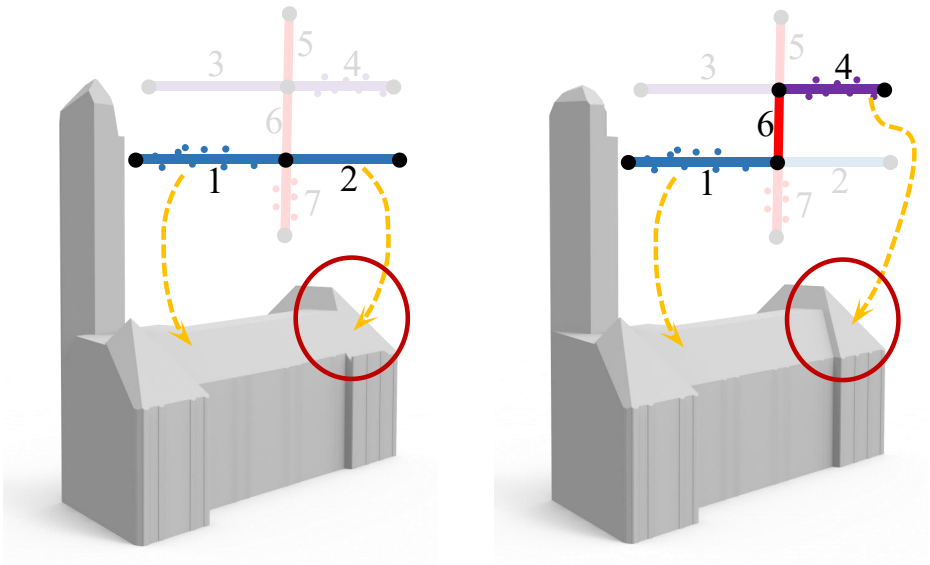
  where $V(f_i)$ denotes the set of hypothesized faces that overlap with face $f_i \in F$ in the vertical direction.

- *Face prior.* This constraint enforces that for all the derived faces from the same planar segment, the one with the highest confidence value is always selected as a prior. Here, the confidence of a face is measured by the number of its supporting points. This constraint can be simply written as

$$x_l = 1,$$

where $x_l$ is the variable whose value denotes the status of the most confident face $f_l$ of a planar segment. This constraint resolves ambiguities if two hypothesized faces are near coplanar and close to each other, which preserves finer geometric details. The effect of this constraint is demonstrated in Figure 3.4.






(**a**) (**b**)

Figure 3.4: The effect of the *face prior* constraint. The insets illustrate the assembly of the hypothesized faces in the corresponding marked regions (each line segment denotes a hypothesized face, and line segments of the same color represent faces derived from the same planar primitive). (**a**) Reconstruction without the *face prior* constraint. (**b**) Reconstruction with the *face prior* constraint, for which faces 1 and 4 both satisfy the *face prior* constraint. The numbers 1-7 denote the 7 candidate faces.

With all the constraints, the complete optimization problem is written as

$$\begin{aligned} &\min_X \lambda_d E_d + \lambda_c E_c + \lambda_r E_r \\ &\text{s.t.} \quad \begin{cases} \sum\limits_{k \in V(f_i)} x_k = 1, (1 \le i \le |F|) \\ \sum\limits_{j \in N(e_i)} x_j = 0 \quad or \quad 2, \quad (1 \le j \le |E|) \\ x_l = 1, \\ x_i \in \{0,1\}, \quad \forall i \in N \end{cases} \end{aligned} \tag{3.5}$$

where the first constraint is called *single roof*, which ensures that the reconstructed building model has a single layer of roofs. The second constraint enforces that in the final model an edge is associated with two adjacent faces, ensuring the final model to be watertight and manifold. The third constraint is called *face prior*, which ensures that, for the faces derived from the same planar segment, the one with the highest confidence value is selected as a prior.

By solving the above optimization problem, the final surface model of the building can be obtained by solving the optimization problem given in Equation (3.5).



Figure 3.5: Reconstruction of a large scene from the AHN3 dataset [5].

## 3.3. Results and Evaluation

My method is implemented in C++ using CGAL [167]. All experiments were conducted on a desktop PC with a 3.5 GHz AMD Ryzen Threadripper 1920X and 64 GB RAM.

### 3.3.1. Test Datasets

I have tested my method on three datasets of large-scale urban point clouds including more than 20 k buildings.

- AHN3 [5]. An openly available country-wide airborne LiDAR point cloud dataset covering the entire Netherlands, with an average point density of 8 points/$m^2$. The corresponding footprints of the buildings are obtained from the Register of Buildings and Addresses (BAG) [13]. The geometry of footprint is acquired from aerial photos and terrestrial measurements with an accuracy of 0.3 m. The polygons in

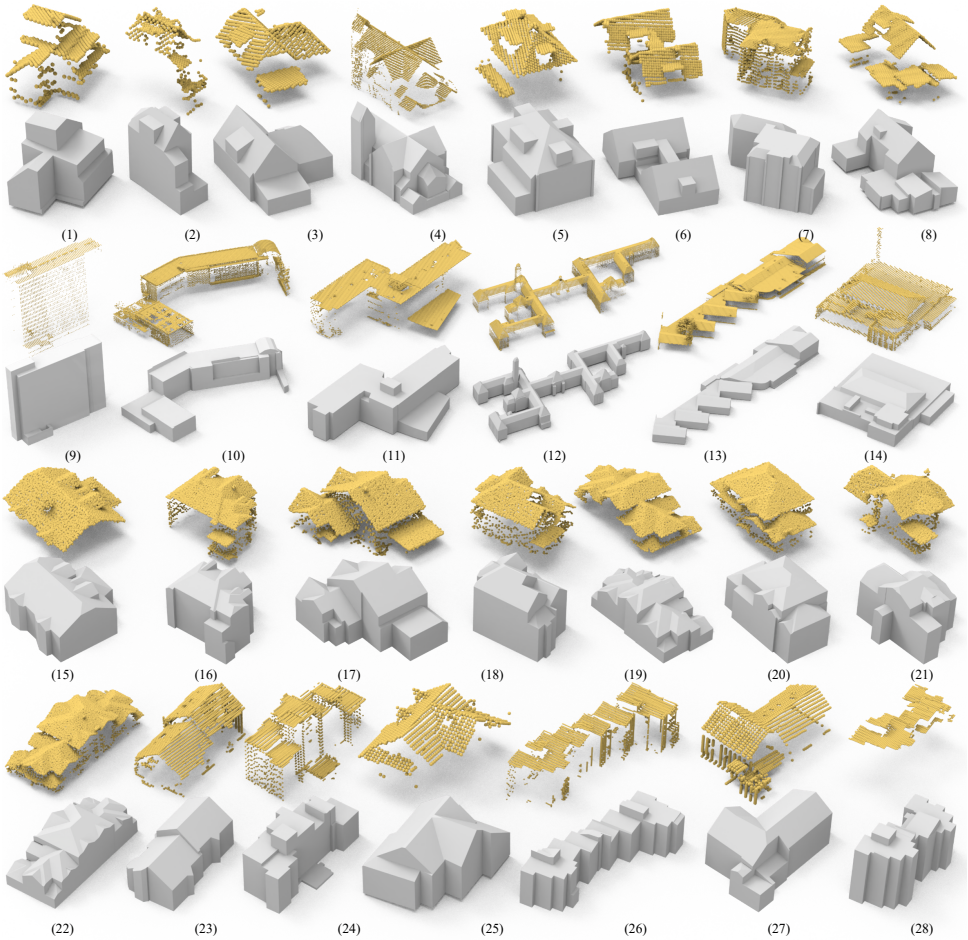



Figure 3.6: The reconstruction results of a set of buildings from various datasets. (**1**–**14**) are from the AHN3 dataset [5], (**15**–**22**) are from the DALES dataset [171], (**23**–**28**) are from the Vaihingen dataset [146].

the BAG represent the outlines of buildings as their outer walls seen from above, which are slightly different from footprints. I still use 'footprint' in this thesis.

- DALES [171]. A large-scale aerial point cloud dataset consisting of forty scenes spanning an area of 10 $km^2$, with instance labels of 6 k buildings. The data was collected using a Riegl Q1560 dual-channel system with a flight altitude of 1300 m above ground and a speed of 72 m/s. Each area was collected by a minimum of 5 laser pulses per meter in four directions. The LiDAR swaths were calibrated using the BayesStripAlign 2.0 software and registered, taking both relative and absolute errors into account and correcting for altitude and positional errors. The average point density is 50 points/$m^2$. No footprint data is available in this dataset.

- Vaihingen [146]. An airborne LiDAR point cloud dataset published by ISPRS, which has been widely used in semantic segmentation and reconstruction of urban scenes. The data were obtained using a Leica ALS50 system with 45° field of view and a mean flying height above ground of 500 m. The average strip overlap is 30% and multiple pulses were recorded. The point cloud was pre-processed to compensate for systematic offsets between the strips. I use in my experiments a training set that contains footprint information and covers an area of 399 m × 421 m with 753 k points. The average point density is 4 points/$m^2$.

### 3.3.2. Reconstruction results

**Visual results**. I have used my method to reconstruct more than 20 k buildings from the aforementioned three datasets. For the AHN3 [5] and Vaihingen [146] datasets, the provided footprints were used for both building instance segmentation and extrusion of the outer walls. My inferred vertical planes were used to complete the missed facade faces. For the DALES [171] dataset, I used the provided instance labels to extract building instances, and I used my inferred vertical walls for the reconstruction.

Figures 3.1 and 3.5 show the 3D reconstruction of all buildings in two large scenes from the AHN3 dataset [5]. For the buildings reconstructed in Figure 3.1, their models are simplified polygonal meshes with an average face count of 34. To better reveal the quality of my reconstructed building models, I demonstrate in Figure 3.6 a set of individual buildings reconstructed from the three test datasets. From these visual results, I can see that although the buildings have diverse structures of different styles, and the input point clouds have varying densities and different levels of noise, outliers, and missing data, my method succeeded in obtaining visually plausible reconstruction results. These experiments also indicate that my approach is successful in inferring the vertical planes of buildings from airborne LiDAR point clouds and it is effective to include these planes in the 3D reconstruction of urban buildings.

**Quantitative results**. I have also evaluated the reconstruction results quantitatively. Since ground-truth reconstruction is not available for all buildings in the three datasets, I chose to use the commonly used accuracy measure, Root Mean Square Error (RMSE), to quantify the quality of each reconstructed model. In the context of surface reconstruction, RMSE is defined as the square root of the average of squared Euclidean distances from the points to the reconstructed model. In Table 3.1, I report the statistics of my quantitative results on the buildings shown in Figure 3.6. I can see that my method has

| Dataset | Model | #Points | #Faces | RMSE ($m$) | Time ($sec$) |
|---|---|---|---|---|---|
| AHN3 | (1) | 732 | 23 | 0.07 | 3 |
| | (2) | 532 | 42 | 0.12 | 4 |
| | (3) | 1165 | 31 | 0.04 | 3 |
| | (4) | 20,365 | 127 | 0.15 | 62 |
| | (5) | 1371 | 48 | 0.04 | 5 |
| | (6) | 1611 | 45 | 0.06 | 4 |
| | (7) | 3636 | 68 | 0.21 | 18 |
| | (8) | 2545 | 52 | 0.04 | 8 |
| | (9) | 15,022 | 63 | 0.11 | 28 |
| | (10) | 23,654 | 262 | 0.26 | 115 |
| | (11) | 13,269 | 102 | 0.11 | 34 |
| | (12) | 155,360 | 1520 | 0.09 | 2520 |
| | (13) | 24,027 | 176 | 0.24 | 141 |
| | (14) | 28,522 | 227 | 0.15 | 78 |
| DALES | (15) | 8662 | 39 | 0.04 | 11 |
| | (16) | 11,830 | 73 | 0.1 | 8 |
| | (17) | 10,673 | 47 | 0.07 | 7 |
| | (18) | 7594 | 33 | 0.07 | 14 |
| | (19) | 13,060 | 278 | 0.05 | 145 |
| | (20) | 11,114 | 55 | 0.06 | 24 |
| | (21) | 8589 | 51 | 0.06 | 15 |
| | (22) | 18,909 | 282 | 0.08 | 86 |
| Vaihingen | (23) | 7701 | 51 | 0.24 | 25 |
| | (24) | 6845 | 99 | 0.12 | 8 |
| | (25) | 1007 | 24 | 0.11 | 2 |
| | (26) | 11,591 | 206 | 0.17 | 10 |
| | (27) | 4026 | 42 | 0.26 | 6 |
| | (28) | 5059 | 61 | 0.22 | 9 |

Table 3.1: Statistics on the reconstructed buildings shown in Fig. 3.6. For each building, the number of points in the input, number of faces in the reconstructed model, fitting error (i.e., RMSE in meters), and running time (in seconds) are reported.

| Region | #Points | #Building | RMSE($m$) BAG3D | RMSE($m$) Ours |
|---|---|---|---|---|
| (a) | 1,694,247 | 198 | 0.088 | **0.079** |
| (b) | 329,593 | 387 | 0.139 | **0.138** |
| (c) | 224,970 | 368 | 0.140 | **0.132** |
| (d) | 80,447 | 160 | 0.146 | **0.128** |

Table 3.2: Quantitative comparison with the BAG3D [1] on four urban scenes (a)–(d). Both BAG3D and my method used the point clouds from the AHN3 dataset [5] as input. The bold font indicates smaller RMSE values.

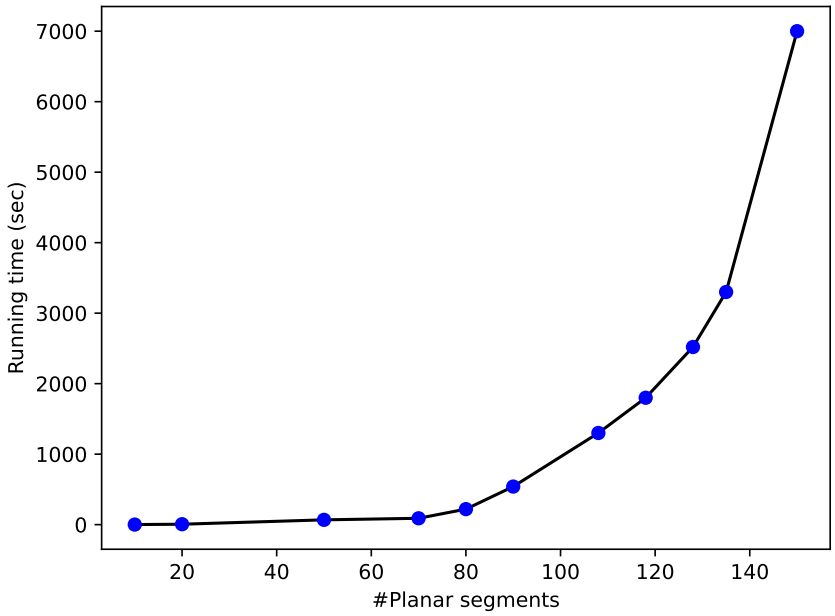



Figure 3.7: The running time of my method with respect to the number of the detected planar segments. These statistics are obtained by testing on the AHN3 dataset.

obtained good reconstruction accuracy, i.e., the RMSE for all buildings is between 0.04 m to 0.26 m, which is quite promising for 3D reconstruction of real-world buildings from noisy and sparse airborne LiDAR point clouds. As observed from the number of faces column of Table 3.1, my results are simplified polygonal models and are more compact than those obtained from commonly used approaches such as the Poisson surface reconstruction method [89] (that produces dense triangles). Table 3.1 also shows that the running times for most buildings are less than 30 s. The reconstruction of the large complex building shown in Figure 3.6 (12) took 42 min. This long reconstruction time is due to that my method computes the pairwise intersection of the detected planar primitives and inferred vertical planes, and it generates a large number of candidate faces and results in a large optimization problem [129] (see Section 3.3.7). The running time with respect to the number of detected planar segments for the reconstruction of more buildings is reported in Figure 3.7.

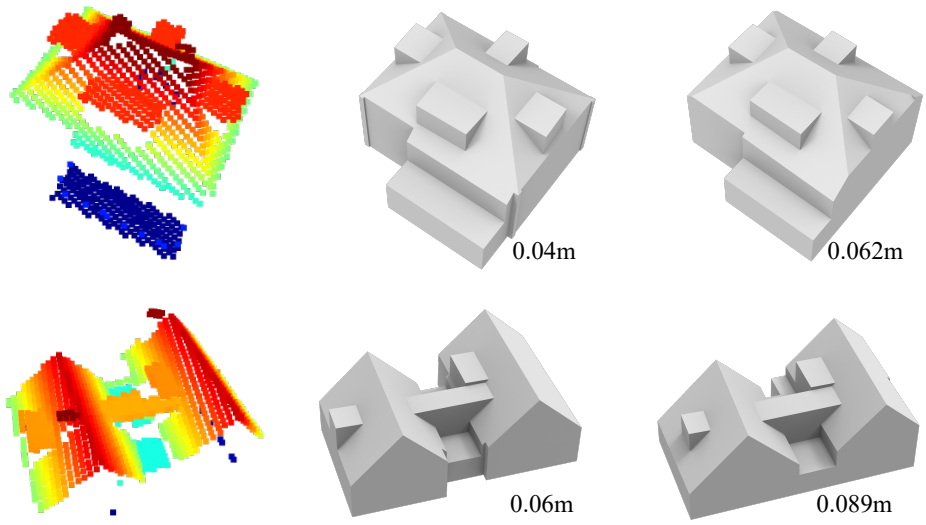
**New 3D Building models dataset**. My method has been applied to city-scale building reconstruction. The results are released as a new dataset consisting of 20 k buildings (including the reconstructed 3D models and the corresponding airborne LiDAR point clouds). I believe this dataset can stimulate research in urban reconstruction from airborne LiDAR point clouds and the use of 3D city models in urban applications.

| Dataset | Method | #Faces | RMSE ($m$) | Time ($sec$) |
|---|---|---|---|---|
| AHN3 | 2.5D DC [206] | 12,781 | 0.213 | 13 |
| | PolyFit [129] | 1848 | 0.242 | 160 |
| | Ours | 2453 | 0.128 | 380 |
| DALES | 2.5D DC [206] | 2297 | 0.204 | 10 |
| | PolyFit [129] | 444 | 0.287 | 230 |
| | Ours | 583 | 0.184 | 670 |
| Vaihingen | 2.5D DC [206] | 2695 | 0.168 | 6 |
| | PolyFit [129] | 647 | 0.275 | 102 |
| | Ours | 798 | 0.157 | 212 |

Table 3.3: Statistics on the comparison of 2.5D Dual Contouring [206], PolyFit [129], and my method on the reconstruction from the AHN3 [5], DALES [171], and Vaihingen [146] datasets. Total face numbers, running times, and average errors are reported.



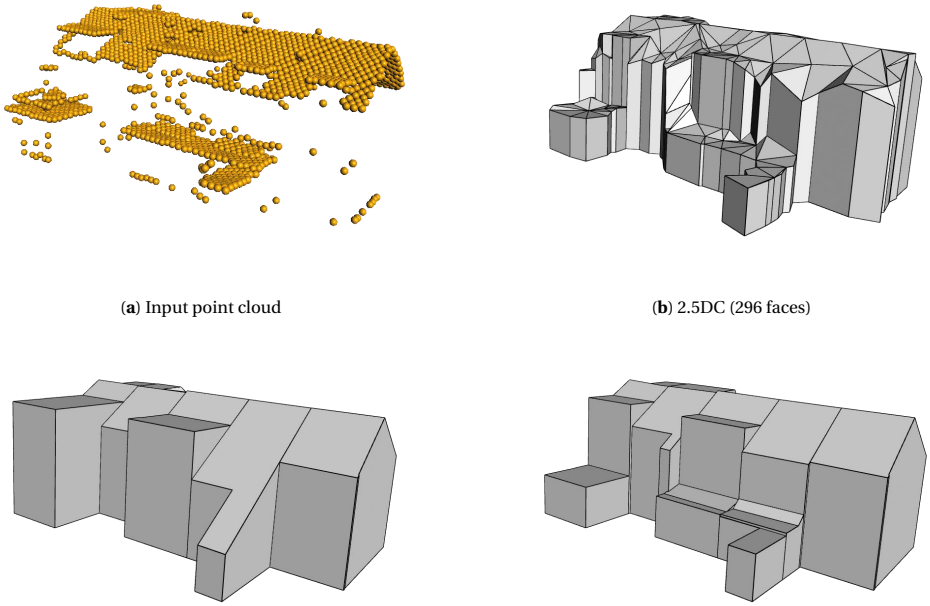
### 3.3.3. COMPARISONS




(a) Input (b) With footprint (c) Without footprint

Figure 3.10: Comparison between the reconstruction *with* (**b**) and *without* (**c**) footprint data on two buildings (**a**) from the AHN3 dataset [5]. The number below each model denotes the root mean square error (RMSE).

I have compared my method with two successful open-source methods, i.e., 2.5D Dual Contouring (dedicated for urban buildings) [206] and PolyFit (for general piecewise-planar objects) [129], on the AHN3 [5], DALES [171], and Vaihingen [146] datasets. The city block from the AHN3 dataset [5] is sparse and contains only 80,447 points for 160 build-



(**a**) Input point cloud

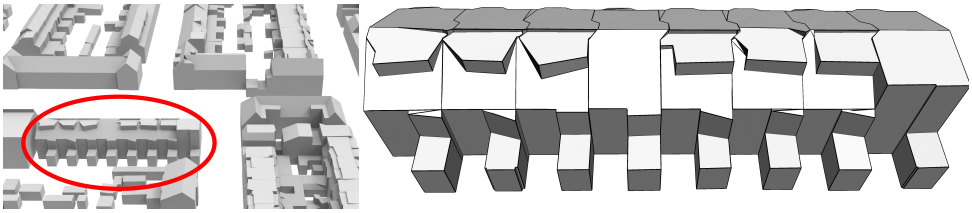
(**b**) 2.5DC (296 faces)

(**c**) PolyFit (58 faces)

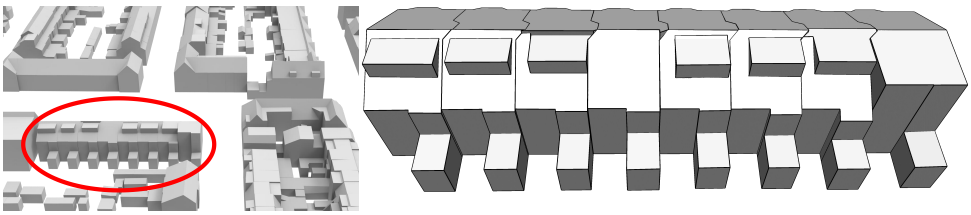
(**d**) Ours (86 faces)

Figure 3.8: Comparison with 2.5D Dual Contouring (2.5DC) [206] and PolyFit [129] on a single building from the AHN3 dataset [5].

ings (i.e., on average 503 points per building). The city region from DALES is denser and contains 214,601 points for 41 buildings (i.e., on average 5,234 points per building). The city area from the Vaihingen dataset contains 69,254 points for 57 buildings (i.e., on average 1,215 points per building). The walls of all the point clouds are severely occluded. Figure 3.8 shows the visual comparison of one of the buildings. PolyFit assumes a complete set of input planar primitives, which is not the case for airborne LiDAR point clouds because the vertical walls are often missing. For PolyFit to be effective, I added my inferred vertical planes to its initial set of planar primitives. From the result, I can observe that both PolyFit and my method can generate compact building models, and the number of faces in the result is an order of magnitude less than that of the 2.5D Dual Contouring method. It is worth noting that even with the additional planes, PolyFit still failed to reconstruct some walls and performed poorly in recovering geometric details. In contrast, my method produces the most plausible 3D models. By inferring missing vertical planes, my method can recover internal building features such as facade faces, which enhance the overall geometry by adding more detailed structural elements to the final LoD2 building model. Table 3.3 reports the statistics of the comparison, from which I can see that the reconstructed building models from my method have the highest accuracy. In terms of running time, my method is slower than the other two, but it is still



(**a**) Result from BAG3D



(**b**) Ours result

Figure 3.9: A visual comparison with BAG3D [1]. A building from Table 3.2 (b) is shown.

acceptable in practical applications (on average 4.9 s per building).

I also performed an extensive quantitative comparison with the 3D building models from the BAG3D [1], which is a public 3D city platform that provides 3D models of urban buildings at the LoD2 level. For this comparison, I picked four different regions consisting of 1113 buildings in total from the BAG3D. In Figure 3.9, I demonstrate a visual comparison, from which I can see that my models demonstrate more regularity. The quantitive result is reported in Table 3.2, from which it can be seen that that my results have higher accuracy.

### 3.3.4. PARAMETERS

My method involves a few parameters that are empirically set to fixed values for all experiments, i.e., the distance threshold $\epsilon_d = 0.25$ and the tolerance for overall transportation cost $\epsilon_c = 2.0$. The resolution $r$ for the rasterization of the TIN model to generate heightmaps is dataset dependent due to the difference in point density. It is set to 0.20 m from AHN3, 0.15 m for DALES, and 0.25 m for Vaihingen. The weight of the *roof preference* energy term $\lambda_r = 0.04$ (while the weights for the data fitting and model complexity terms are set to $\lambda_d = 0.34$ and $\lambda_c = 0.62$, respectively).

### 3.3.5. IMPACT OF USING FOOTPRINT

My method can infer the vertical planes of a building from its roof points, and then the outer walls are completed using the vertical planes. It also has the option to directly use given footprint data for reconstruction. With a given footprint, vertical planes are firstly obtained by extruding the footprint polygons. Then these planes and those extracted from the point clouds are intersected to hypothesize the model faces, followed by the optimization step to obtain the final reconstruction. Figure 3.10 shows such a compar-

ison on two buildings. The quantative analysis shows that using the inferred vertical planes slightly increases reconstruction errors.

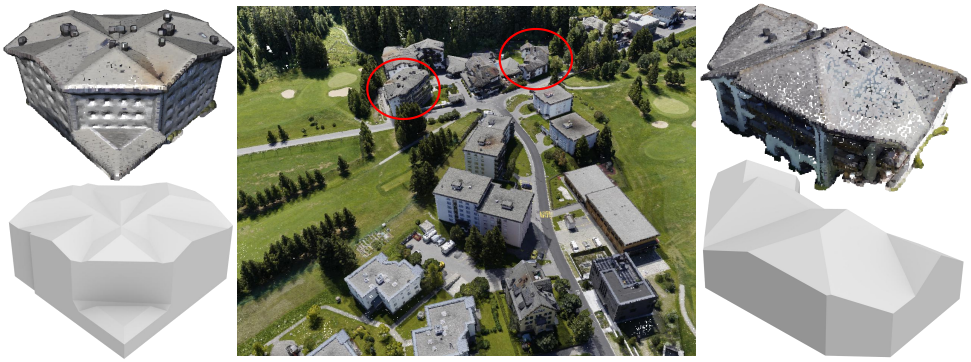


Figure 3.11: Reconstruction from aerial point clouds. In these point clouds, the vertical walls can be extracted from the point clouds and directly used in reconstruction, and thus the vertical plane inference step was skipped. The dataset is obtained from [31].

### 3.3.6. Impact of redundant vertical planes

The methodology presented in my thesis only focuses on airborne LiDAR point clouds, in which vertical walls of buildings are typically missing. In practice, my method can be easily adapted to work with other types of point clouds that contain points of vertical walls, e.g., point clouds reconstructed from drone images. For such point clouds, my method can still be effective by replacing the inferred vertical planes with those directly detected from the point clouds. Figure 3.11 shows two such examples.

### 3.3.7. Limitations

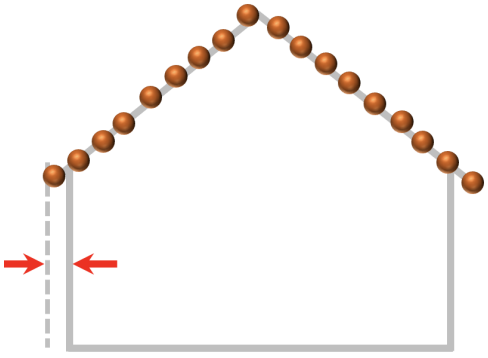


Figure 3.12: Mis-alignment with ground-truth footprint.

While the proposed framework advances vertical plane inference for wall reconstruction, two constraints still exist. First, the reliance on rooftop-derived vertical plane ge-

ometry introduces potential misalignment between reconstructed walls and ground-truth footprints (as shown in 3.12), particularly when input LiDAR data lacks sufficient vertical resolution. This underscores the necessity of integrating high-quality georeferenced footprint data to mitigate geometric discrepancies in practice. Second, the computational scalability of the PolyFit-derived framework is inherently constrained by its hypothesis-and-selection architecture. Buildings exhibiting an excessive number of planar regions may incur prohibitive processing times, necessitating algorithmic optimizations or hierarchical decomposition for large-scale urban applications.



## 3.4. CONCLUSIONS AND FUTURE WORK

I have presented a fully automatic approach for large-scale 3D reconstruction of urban buildings from airborne LiDAR point clouds. I propose to infer the vertical planes of buildings that are commonly missing from airborne LiDAR point clouds. The inferred vertical planes play two distinct roles during the reconstruction. The outer vertical planes directly become part of the exterior walls of the building, and the inner vertical planes enrich building details by splitting the roof planes at proper locations and forming the necessary facade faces in final models. My method can also incorporate provided building footprints for reconstruction. When footprints are used, they are extruded to form the exterior walls of the models, and the inferred inner planes enrich building details. Extensive experiments on different datasets have demonstrated that inferring vertical planes is an effective strategy for building reconstruction from airborne LiDAR point clouds, and the proposed *roof preference* energy term and the novel hard constraints ensure topologically correct and accurate reconstruction.

My current framework uses only planar primitives and it is sufficient for reconstructing most urban buildings. In the real world, there still exist buildings with curved surfaces, which my current implementation could not handle. However, my hypothesize-and-selection strategy is general and can be extended to process different types of primitives. As a future work direction, my method can be extended to incorporate other geometric primitives, such as spheres, cylinders, or even parametric surfaces. With such an extension, buildings with curved surfaces can also be reconstructed.

Having discussed an LoD2 building reconstruction method in this chapter, the focus now shifts to a more detailed explanation of LoD3 building reconstruction. Building upon the foundational concepts established in the introduction chapter, the next chapter aims to achieve a higher level of detail in the results. Through a combination of simple user interactions and intelligent algorithms, the LoD3 building reconstruction method seeks to further refine and enhance the representation of buildings in the digital environment, ultimately contributing to the overall advancement of urban modeling.

# 4

# SEMI-AUTOMATIC LOD3 BUILDING RECONSTRUCTION FROM MVS MESHES

*This chapter introduces a semi-automated method to reconstruct compact building models at Level of Detail 3 (LoD3). In this thesis, I introduce a semi-automated method to reconstruct robust building models at Level of Detail 3 (LoD3). Unlike most existing reconstruction methods that rely on the automatic extraction of planar primitives, I combine the advantages of automated optimization and user guidance to obtain a refined set of planar primitives. First, I utilize a variational shape approximation technique to generate an initial mesh segmentation. A refined segmentation is then obtained by incorporating a few user-delineated strokes, which serve as instructive cues to indicate the spatial arrangement and contour of the underlying building structures. Subsequently, the geometry of the input mesh is updated based on the segmentation, resulting in a compact polygonal mesh. Finally, I enhance the overall regularity of the simple polygonal mesh to generate visually pleasing building models. My method has been validated using Multi-View Stereo (MVS) meshes of buildings with various styles. Experimental results demonstrate the effectiveness of the proposed approach in generating LoD3 building models.*

## 4.1. INTRODUCTION

In recent years, the use of 3D city models has become more and more popular, driven by their wide range of applications, including computational fluid dynamics simulations, solar potential analysis, and heating energy estimation. As the need for greater levels of detail in these models increases, there has been a shift toward more advanced building representations, specifically LoD3 models. These models not only represent basic roof structures but also capture intricate architectural details, including facade elements and roof superstructures. Such enhanced representations are essential for applications requiring high geometric accuracy and visual realism. The ability to reconstruct precise LoD3 models has become increasingly critical for urban modeling, as well as in fields where fine geometric detail and realism are paramount, positioning this as a key research area in 3D reconstruction and urban modeling technologies.



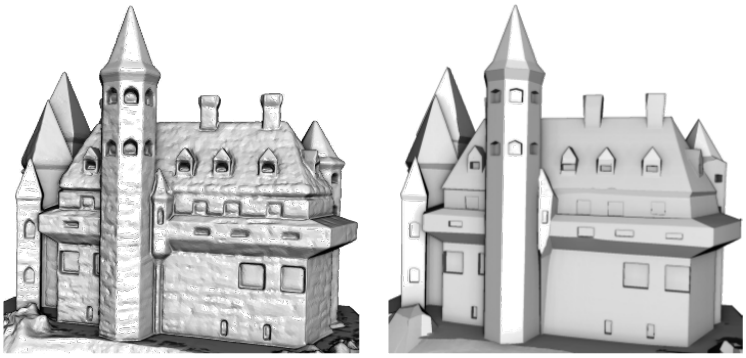
Considering the existing 3D building datasets mainly cover LoD1 and LoD2 [138, 103, 176], there remains a scarcity of models with higher detail, specifically LoD3 models (see Figure 4.1 (b)). Some applications require more intricate building representations. For example, energy demand can be better computed if the roofs and windows of the building are considered [19]. For these cases, LoD3 building models that include slanted roofs and necessary facade details are desired. Unfortunately, due to imperfections in real-life data (e.g., laser scans of buildings), such as noise, outliers, and missing regions, there is currently no fully automatic method capable of generating accurate LoD3 models. As a result, few researchers have turned to semi-automatic approaches for more detailed building reconstruction. However, these methods are constrained by the Manhattan-world assumption, which restricts their broader applicability, or they demand a significant amount of user interaction. To address these issues, I introduce a semi-automatic method for creating LoD3 buildings that effectively combines user guidance and automatic detection, as shown in Figure 4.1.

As demonstrated in Chapter 2, compared with airborne LiDAR point clouds, MVS meshes show greater potential for generating highly detailed building models. Therefore, I have selected MVS meshes as the input for the detailed building reconstruction process. However, there are still three challenges that need to be addressed to effectively achieve the highest level of detail (i.e., LoD3) reconstruction from MVS meshes:

1. MVS meshes demonstrate over-smoothed roof superstructures and facade elements (see Figure 4.1 (a)). This is due to that the reconstruction process of MVS meshes often involves excessive smoothing or alteration of the original details and intricacies. To achieve high-fidelity reconstruction, it is required to recover these lost sharp features.

2. Urban buildings typically exhibit repetitive patterns (e.g., same or similar windows on the same building), but the instances of these structures in an MVS mesh have varying qualities. To enhance the efficacy and efficiency of the overall reconstruction pipeline, it is essential to intelligently identify and leverage repetitive patterns commonly presented in facades.

3. Structural regularities (e.g., same size and equal spacing between windows on the same floor) have been distorted in MVS meshes. These irregularities are due to in-



(a) Input (b) Our result



Figure 4.1: My method takes as input a typical dense MVS mesh (a) and reconstructs a compact polygonal mesh (b) from the input.

accuracies and imperfections in the image-based reconstruction pipeline. Therefore, it is also crucial to enhance the regularities and rectify these distortions.

In this thesis, I leverage high-level user guidance, automatic building segmentation, and efficient geometry optimization in the reconstruction workflow and introduce a novel method for the semi-automatic reconstruction of LoD3 building models that tackle the aforementioned challenges. My method is based on the observation that buildings can be effectively approximated by planar surfaces, consisting of repeated structures that are commonly represented by singular or composite arrangements of fundamental primitives, such as boxes and prisms. This work is guided by two primary insights: Firstly, for the accurate reconstruction of facade elements or roof superstructures, the system must devise a solution capable of identifying the supporting planes associated with these entities. Secondly, the modeling process requires the optimal utilization of interactive input (i.e., high-level user guidance) and automatic optimization techniques to improve the regularity of the geometry. My approach begins with the utilization of the variational shape approximation method [45] to obtain a coarse segmentation of the input mesh, followed by a multi-label optimization-based step that incorporates simple user-drawn strokes to achieve a finer segmentation. Subsequently, a compact polygonal model is generated through the optimization of face normals and vertex positions of the mesh. Finally, I refine the 3D building models to improve the overall regularity, resulting in accurate and visually pleasing results.

In summary, my contributions are as follows:

- I introduce a novel semi-automatic framework for LoD3 building reconstruction from MVS meshes, which requires limited high-level user guidance.

- I propose a multi-label optimization method to refine mesh segmentation by incorporating high-level user guidance.

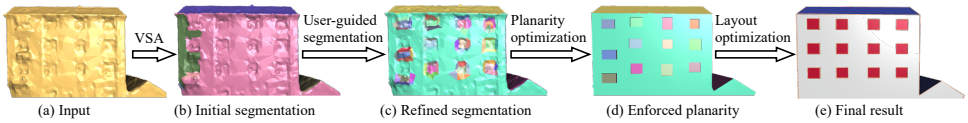



Figure 4.2: The pipeline of the proposed method. Given an MVS mesh of a building (a), I first generate an initial planar segmentation (b) by using the variational shape approximation (VSA) method [45]. Then, the initial segmentation is iteratively refined (c) by incorporating user guidance, followed by a planarity optimization step that enhances the symmetry, verticality, and horizontality of the refined planar regions (d). Finally, the layouts of facade elements are optimized, resulting in regular facade structures in the final model (e).

4

- I offer a set of algorithms that refine the 3D building model by enhancing the overall structural regularity.

## 4.2. METHODOLOGY

My method focuses on reconstructing a compact polygonal mesh from a dense MVS mesh representing a single building. The goal is to approximate the input as closely as possible while maintaining computational efficiency and accuracy in capturing architectural details. Figure 4.2 illustrates the pipeline of the proposed method.

The first step in my process is to decompose the input dense triangle mesh into a set of planar regions using the Variational Shape Approximation (VSA) method [45]. I select this method because of its effectiveness in decomposing complex surfaces into a set of planar segments, which is crucial for representing architectural features with small geometric errors. This approach allows for a more manageable and computationally feasible initial segmentation of the MVS mesh.

Following the initial segmentation, the method incorporates human guidance in the form of roughly drawn strokes to iteratively refine the segmentation result. This step leverages human intuition to correct errors, thereby enhancing the accuracy of the initial segmentation. By incorporating user input, I address potential deficiencies, such as inaccuracies or missing planar primitives in the previous automatic segmentation, which may result from occlusions or noise in the input data. A multi-label optimization algorithm is developed to improve mesh segmentation by updating the assignments of labels to mesh faces. This approach allows for greater flexibility and precision in delineating facade components and roof superstructures, which are often characterized by complex geometries. During the interactive phase, this optimization is repeated to enhance mesh segmentation progressively. The iterative nature of this process ensures that incremental refinements are made based on the updated segmentation result and user input. Subsequently, the planarity of each segment is enforced through an optimization process that iteratively adjusts face normals and vertex positions, which is critical to achieving geometrically accurate and visually coherent surfaces.

Finally, the layouts of facade elements are improved by solving a quadratic programming problem. This optimization step aligns facade elements more consistently and aesthetically with the overall building geometry, ensuring that the result is visually pleasing. The choice of quadratic programming is driven by its ability to efficiently manage the lin-

ear constraints and quadratic objectives that are commonly encountered in alignment problems.

In the following subsections, I provide a detailed explanation of each step in my methodology, explaining the specific algorithms implemented at each stage.

### 4.2.1. PLANAR SEGMENTATION

Given a dense MVS mesh of a building, I employ the VSA method [45] to obtain an initial planar segmentation of the input mesh. The initial segmentation often exhibits artifacts due to inherent imperfections in the data, such as holes, noise, outliers, and over-smoothed geometric features. These imperfections in segmentation lead to inaccuracies in the reconstruction process. Figure 4.3 (a) shows such an example, where the four supporting planes of the window are not fully detected, resulting in an incomplete shape recovery, as shown in the right figure. As automatic methods often struggle with accurately capturing complex architectural details, I propose integrating simple yet intuitive user interactions to refine the initial segmentation results. This approach leverages human hints to guide the segmentation process, ensuring more accurate and reliable results.



The user interactions in my approach are categorized into two distinct classes: contour operations and refinement operations. Contour operations are primarily designed to assist in constructing components characterized by box- or prism-like geometries, which commonly exist in architectural structures. These operations allow users to define and adjust the boundaries of planar segments, ensuring that the segmentation aligns with the expected geometric patterns. On the other hand, refinement operations serve as auxiliary functionalities to enhance the segmentation further, particularly in cases where object shapes deviate from standard box- or prism-like forms. These operations enable users to make flexible and detailed adjustments, correcting any remaining inaccuracies and ensuring that the final segmentation accurately represents the complex geometries of architectural features. By incorporating these user-driven operations, my method achieves a balance between fully automatic and manual reconstruction, allowing for a more accurate and reliable reconstruction process.

#### CONTOUR OPERATIONS

This category of user interactions is in the form of roughly drawn 2D sketches of a shape, followed by an automated process that identifies potential planar planes within the underlying region in the mesh. My method implicitly disregards the depth dimension and directly works in the locally projected 2D space (i.e., the supporting plane of the local facade region), considerably reducing the complexity of user interaction and decreasing the overall effort required for achieving a satisfied planar segmentation.

In this work, the roof superstructures and facade components of buildings are abstracted by singular or compound groups of basic primitives, such as boxes and prisms. To effectively capture such common structures in buildings, I consider two types of contours.

- **Rectangle**. Given that many architectural components, such as windows, can be approximated by basic boxes, a rubber-band rectangle is employed to loosely fit the contours of these elements, as illustrated in Figure 4.4 (a).

(a) Reconstruction from initial segmentation

(b) Reconstruction from refined segmentation

Figure 4.3: A building window reconstructed from different segmentation results. Top row: an under-segmentation leading to an incomplete reconstruction. Bottom row: a refined segmentation resulting in the expected reconstruction. The red arrow points to an over-smoothed region.

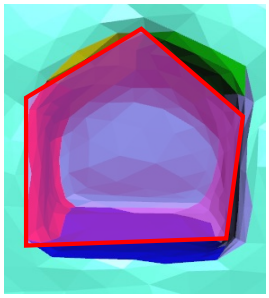
- **Polygon**. Users can roughly sketch the contours of complex elements by consecutively clicking in the screen space. The algorithm will then automatically determine within a local neighborhood the optimal planar segmentation representing a general prim, as illustrated in Figure 4.4 (b).

With the user input, I refine the planar segmentation result by using Markov Random Field (MRF) optimization. Given a set of $n$ triangular faces within a local region $P$, $k$ labels, and $n \cdot k$ costs, the objective is to assign the appropriate label $l_p$ for each face $f_p$. This determination is guided by utilizing the user-drawn sketches as a reference, thereby refining the initial segmentation result. The key observation is that each polygon edge is accompanied by an associated planar primitive, as shown in Figure 4.4. Let us take a simple rectangle as an example, as shown in Figure 4.5 (a) to explain my algorithm. Note that this can be easily generalized to more complex polygonal shapes. In this example, $l_p \in \{0,1,2,3,4,5\}$. Specifically, if the face $f_p$ belongs to either the ground or top planes of the box, I assign the label as 4 or 5, respectively. Conversely, labels ranging from 0 to 3 correspond to the four sides of the box. The specific label belonging to each region is illustrated in Figure 4.5 (b). I formulate the refinement of the segmentation problem as a multi-label graph cut minimization [49]. My objective function consists of three terms: data, smoothness, and label costs.

**Data cost.** $D(f_p, l_i)$ measures how well a face $f_p$ fits to label $l_i$. The likelihood of a face $f_p$ for assignment to the planar group associated with edge $e_i$ of the user-drawn polygon $M$ is positively correlated with its proximity to that edge. Additionally, faces exhibiting a similar normal orientation to either the ground or top plane are more likely to be assigned to their respective planar groups. Consequently, the data term is defined as follows,

$$E_d = \sum_{p \in P} D(f_p, l_i), \tag{4.1}$$

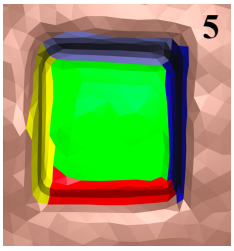
(a) A rectangular outline



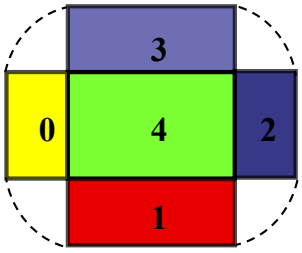
(b) A polygon outline

Figure 4.4: Two basic operations for indicating the outlines of building elements. (a) A sketch of a simple rectangle, resulting in a box-like shape. (b) A sketch of a general polygon, resulting in a prism.






(a) Expected segmentation




(b) Flattened regions

Figure 4.5: A rectangular example showing the objective of the MRF-based segmentation. (a) The expected planar segmentation of the 3D facade structure (a window in this case). (b) The flattened planar regions and their corresponding labels. The number denoting each region indicates the expected label $l_p$ for all the faces in the region.

$$D(f_p, l_i) = \begin{cases} \dfrac{d(f, l_i)}{1.2 * \text{avg}(L_{ei})}, & \text{if } l_i \in \{0,1,2,3\} \\ 1 - |\cos\langle n_i, n_p\rangle|, & \text{if } l_i \in \{4,5\} \end{cases}, \tag{4.2}$$

where $d(f, l_i)$ denotes the distance between face $f_p$ and edge $e_i$ in the user-drawn polygon, while $L_{ei}$ denotes the length of edge $e_i$. $n_p$ refers to the normal of face $f_p$, and $n_i$ represents the normal of the ground or top plane of the box.

**Smoothness cost.** The smoothing term, denoted as $E_s$, is employed to prevent abrupt label transitions within localized regions resulting in irregularities of the segmentation boundaries. When two neighboring faces exhibit minimal differences in their surface normals, the probability of them belonging to the same planar segment significantly increases. Consequently, I introduce a penalization function, denoted as $w(p, q)$, which discourages assigning different labels to two similar faces, $f_p$ and $f_q$. The magnitude of $w(p, q)$ increases as the degree of similarity between the adjacent faces intensifies. This

smoothness energy term is defined as follows,

$$E_s = \sum_{f_p \bigcap f_q = e} w(p,q) \cdot \delta(l_p, l_q), \tag{4.3}$$

where $w(p,q)$ and $l(p,q)$ can be computed as,

$$w(p,q) = exp(-\frac{\theta^2_{n_p,n_q}}{2\sigma^2}), \tag{4.4}$$

$$\delta(l_p, l_q) = \begin{cases} 0, & \text{if } \ l_p = l_q \\ 1, & \text{otherwise} \end{cases}, \tag{4.5}$$

respectively. Here $\theta$ represents the angle between the face normals $n_p$ and $n_q$, and $l_p$ and $l_q$ denote the label assigned to face $f_p$ and $f_q$, respectively. $\sigma$ corresponds to the kernel size of the Gaussian function and is empirically set to 0.5.

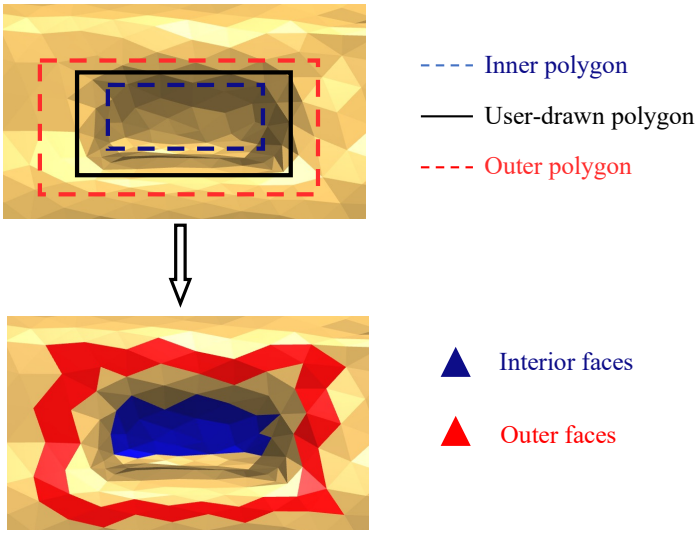



Figure 4.6: The user-drawn polygon and inner/outer offset polygons, and the corresponding interior/outer faces.

**Label cost.** This term is designed to ensure that every planar primitive in the local region, even in extreme cases where the regions are prone to be completely planar, has at least one labeled face. To achieve a refined segmentation result based on this assumption, I create two offset polygons, an inner polygon and an outer polygon (as illustrated in Figure 4.6), with an offset set to 0.25 times the average length of the edges in the user-drawn polygon. I observe that the interior faces of the inner polygon are affiliated with the ground planar region of the box, while the faces intersecting the outer polygon are more likely to belong to the top planar region. Additionally, faces intersecting the edges of the user-drawn polygon have a high potential to belong to the corresponding planar

region. As a result, I allocate the pre-assigned labels to these faces. To penalize the overall energy when the final labels differ from the pre-assigned labels, I define the label cost term $E_l$ as follows,

$$E_l = \sum_{i \in N} L(f_p, l_i), \tag{4.6}$$

where $L(f_p, l_i)$ is computed as,

$$L(f_p, l_i) = \begin{cases} 0, & \text{if } \quad l_i = l_e \\ 1, & \text{otherwise} \end{cases}, \tag{4.7}$$

where $l_e$ represents the expected label of face $f_p$.

**Overall objective function.** With the aforementioned energy terms, the complete objective function can be expressed as

$$E = \lambda_d E_d + \lambda_s E_s + \lambda_l E_l, \tag{4.8}$$

where $\lambda_d$, $\lambda_s$, and $\lambda_l$ are used to balance these three terms. To efficiently minimize the labeling problem described above, I employ the multi-label graph-cut algorithm [49]. After solving this equation, the label for each face can be determined.

I have used the simple box as an example to illustrate my semi-automated segmentation refinement method. The overall framework can be easily extended to general prism cases as illustrated in Figure 4.4 (b). The only difference is that the face labels should be replaced by $\{0, 1, 2, 3, 4, ..., N_e + 1\}$, where $N_e$ indicates the number of edges in the user-drawn polygon.



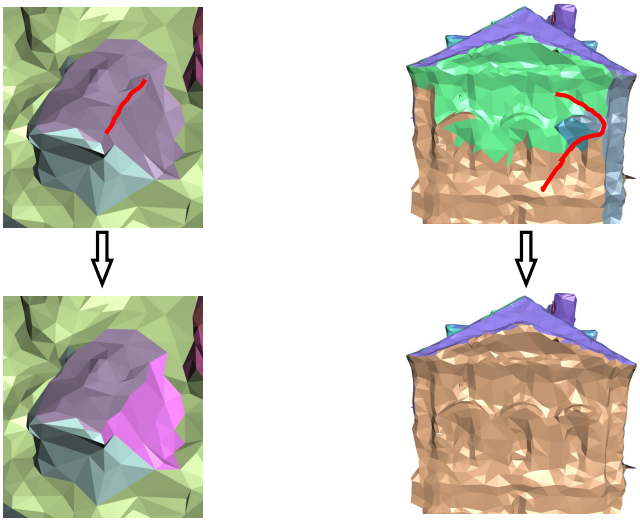
#### REFINEMENT OPERATIONS

Real-world buildings may present complex structures other than boxes and prisms. To bring more flexibility to the interactive process, I also introduce two basic refinement operations, split and merge. These two interactions are intended to improve the accuracy of segmentation results further.

- **Split**. The split operation is dedicated to breaking under-segmented regions. The original region is divided into two distinct parts along the line by merely sketching a line near the expected boundary. This segmentation follows a similar optimization framework (see Equation 4.8). The difference is that the number of labels is decreased to 2. See Figure 4.7 (a) for an illustration.
- **Merge**. This operation is intended to merge over-segmented regions. It merges the regions under the user stroke, as shown in Figure 4.7 (b).

#### LAYOUT DETECTION

To improve the efficiency of user interactions, the contour operations (see Sec. 4.2.1) also trigger the detection of repetitive elements on the same facade. The detected layout will be further optimized in the final step to maximize its regularity (see Sec. 4.2.3). Upon marking out a single representative instance of the facade elements (i.e., the template), my method automatically identifies other elements that are identical or highly similar to the template. Following the template assembly approach of Nan et al. [131], my proposed method consists of three steps, as illustrated in Figure 4.8.



(a) Split (b) Merge

Figure 4.7: Two types of refinement operations. (a) Split: the user roughly sketches a line to break a region into two. (b) Merge: the user roughly sketches over multiple regions to merge them.

- First, I project the vertices delineating the region encompassing a single element onto the facade plane, yielding the template height map, denoted as $T$.
- Second, I project the vertices of the entire facade region including all facade elements onto the supporting plane of the facade, resulting in a rasterized height map of the facade, denoted as $S$. In practice, since no semantic information is available, I treat all vertical planar regions that enclose multiple interior elements as facades.
- Finally, I perform template matching using the Normalized Cross-Correlation (NCC)-based algorithm [27], which accurately locates the other instances of facade elements similar to the template.

After determining the positions of similar components, the refinement operations are carried out to obtain a refined segmentation result, as shown in Figure 4.8 (d).

### 4.2.2. PLANARITY OPTIMIZATION

After refining the segmentation, I optimize the geometry of the input mesh to enforce the planarity for all the regions of the segmented model. This is achieved by iteratively updating the face normals and vertex positions through respective optimization procedures.

#### NORMAL UPDATE

To maximize the planarity of all planar segments, employing a unique normal direction for all faces belonging to the same planar segment is straighforward. Besides, orthogonality and parallelism are common in building models, which also pose constraints

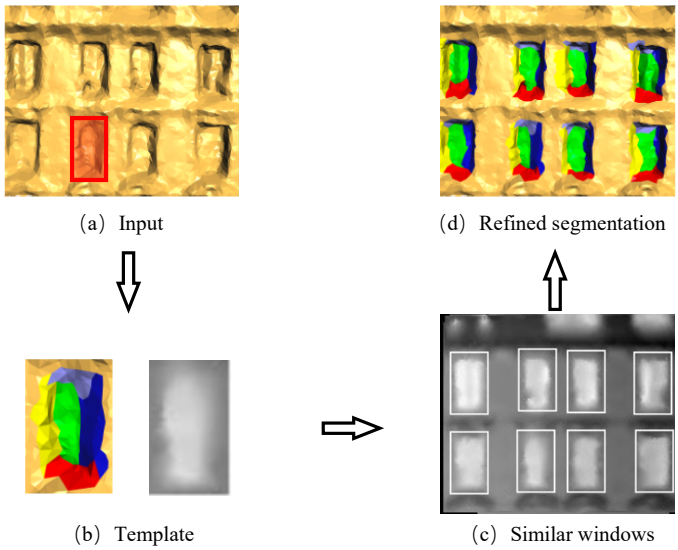



Figure 4.8: An example of layout detection. (a) A typical urban facade with repetitive windows and the user sketched a rubber-band rectangle over one window. (b) The template heightmap $T$ (right) is generated from the region of interest (left, with the colors indicating different detected planar regions). (c) Window instances on the rasterized image $S$ are identified by template matching. (d) The refined segmentation result.

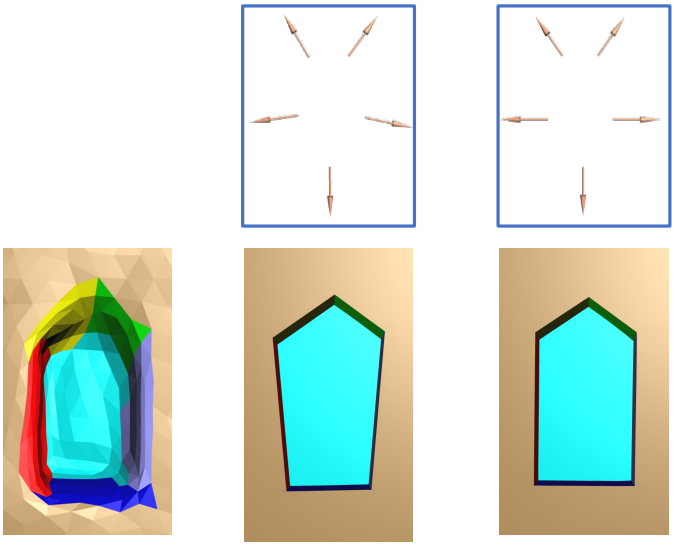


on the face normals to adhere to these geometric properties. In this work, potential geometric constraints related to orthogonality and parallelism are defined for pairs of planes and subsequently incorporated into the objective function. As illustrated in Figure 4.5, there is expected parallelism between the top region (denoted by '5') and bottom region (denoted by '4') of the box, indicating a pair of planes belonging to the parallel group $G_{\parallel}$. Similarly, any pair of adjacent side regions of the box are orthogonal to each other, resulting in four pairs of planes belonging to the orthogonal group $G_{\perp}$. I enforce these orthogonality and parallelism constraints with minimum change to the original normals of the regions, which is achieved through the following optimization:

$$\begin{aligned} \min \lambda_1 \sum_{i \in M} \| W_i (n_i - n_i^*) \|_2^2 + \lambda_2 \sum_{(k,l) \in G_{\perp}} \| n_k \cdot n_l \|_2^2 \\ + \lambda_3 \sum_{(p,q) \in G_{\parallel}} \| n_p - n_q \|_2^2, \end{aligned} \tag{4.9}$$

where $n_i$ and $n_i^*$ represent the normal before and after the update, respectively. The initial normal is computed using Principal Component Analysis (PCA) among the vertices belonging to the same planar region. $W_i$ denotes the number of vertices in the corresponding planar region. $n_k$ and $n_l$ denote the normals of two regions belonging to the orthogonal group $G_{\perp}$. Similarly, $n_p$ and $n_q$ are the normals of two regions belonging to the parallel group $G_{\parallel}$. The weights $\lambda_1$, $\lambda_2$, and $\lambda_3$ balance the three terms. I solve this optimization problem using the L-BFGS algorithm[115].

After updating the normals of all planar regions, I further enhance the overall regularity of these regions by incorporating symmetry, verticality, and horizontality to fur-



(a) Input (b) Initial result (c) Refined result

Figure 4.9: Normal refinement by enforcing symmetry, verticality, and horizontality. (a) Part of the input mesh (a window). (b) The initial result from planarity optimization. (c) The final result with normal refinement by enforcing symmetry, verticality, and horizontality. This step significantly improves the fidelity of the reconstructed model. The top row illustrates the normal vectors associated with different planar regions of the window.

ther refine the updated normals. Specifically, I achieve this by extending the method proposed by Huang et al. [75], which can simultaneously identify symmetric edge correspondences and optimize the symmetry of 2D shapes by solving a mixed-integer program. Different from the original method dedicated to 2D polygonal shapes, I use the normals of the planar regions as input, and I also incorporate energy terms to encourage verticality and horizontality. The overall formulation is given by

$$\begin{aligned} \min \quad & \sum \| n_t - n_t^* \|_2^2 + \omega \sum (1 - z_{ij}) \\ \text{s.t.} \quad & \begin{cases} z_{ij} \cdot S(n_i, n_j) = 0, & \forall (i,j) \in P \\ x_i^l - x_j^r = 0, & \text{if } n_i \in R_v \\ y_i^t - y^b = 0, & \text{if } n_i \in R_h \\ \sum\limits_{j \neq i} z_{ij} \leq 1, & \forall 1 \leq i \leq N \\ z_{ij} \in \{0,1\}, & \forall (ij) \in P \end{cases} , \end{aligned} \tag{4.10}$$

where the binary variable $z_{ij}$ denotes whether the normal pair $(n_i, n_j)$ is symmetric or not. The weight $\omega$ controls the degree of non-symmetry, and it is fixed to be 1 in this work. $S(e_i, e_j)$ measures how much the edge pair$(n_i, n_j)$ deviates from being perfectly symmetric. $x$ and $y$ with superscripts $t$, $b$, $l$, and $r$ denote the coordinates of the top, bottom, left, and right endpoints of a normal, respectively. $R_v$ and $R_h$ denote the set of vertical normal and horizontal normal, respectively. The problem given by Eq. 4.10 is a binary

integer program, and I solve it using Newton Barrier solver provided by Gurobi [65]. Figure 4.9 illustrates the effect of this normal refinement step, which clearly shows it is a crucial step for visually plausible results.

#### VERTEX UPDATE

After obtaining the refined normals, I update the mesh vertices such that the resulting mesh complies with the new normal field. This is achieved through an optimization step similar to the normal update (see Sec. 4.2.2). The objective function consists of three terms. (1) The first term encodes the deviation of the vertices; (2) The second term encourages the vertices to be transformed in a way such that the resulting face edges are orthogonal to previously refined normals. (3) The third term penalizes big changes in edge lengths, which is crucial to prevent edge collapse. The vertex update procedure can expressed as follows:

$$\min \sum \mu_1 \| v_i - v_i^* \|_2^2 + \mu_2 \sum_j \sum_{k=0}^{2} \| n_j \cdot e_k \|_2^2 + \mu_3 \sum \| e_l - e_l^* \|_2^2 \tag{4.11}$$

where $v_i$ and $v_i^*$ denote the vertices before and after update, respectively. $n_j$ is the face normal obtained in the previous normal update step. $e_k$ denotes an edge of a face. $e_l$ and $e_l^*$ represent the edge before and after update, respectively. The formulation given by Equation 4.11 is a least-squares problem, and I solve it also using the L-BFGS algorithm [115].

The above normal update and vertex update procedures are interchangeably carried out iteratively until convergence. Subsequently, the building model is simplified by merging the coplanar triangular faces into a compact polygonal model. It is worth noting that this step significantly reduces the number of faces/vertices in the model. For example, in Figure 4.2 (d), the number of faces is 158 compared to that of 8045 in Figure 4.2 (c).

### 4.2.3. LAYOUT OPTIMIZATION

Building facades typically exhibit repeated elements (e.g., windows), and these elements often form regular patterns. However, the compact building model obtained in the previous step does not inherently convey such regularities. This is because each facade element was reconstructed separately based on the refined planar segmentation and the structural regularities were ignored. In this section, I introduce an approach to optimize facade layouts detected in the previous step (see Sec. 4.2.1). The effect of layout optimization is illustrated in Figure 4.2 (e).

Given the fact that architectural elements in real-world buildings typically possess consistent sizes and are aligned and distributed evenly in the horizontal and/or vertical direction, I impose these regularities upon the layout of these elements that closely resemble real-world structures. In my work, I consider a basic facade element, symbolized as $e_i = \{x_i, y_i, w_i, h_i, d_i\}$, where $(x_i, y_i)$, $w_i$, $h_i$, and $d_i$ denote the coordinates of bottom left corner, width, height, and depth, respectively. Similar to Jiang et al. [83], I employ three distinct constraints, namely alignment, size, and spacing, as illustrated in Figure 4.10.

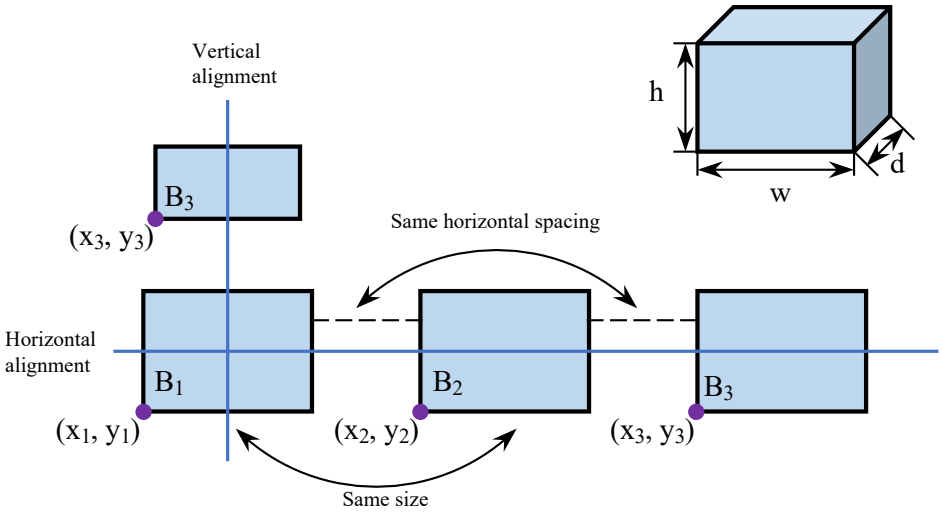



Figure 4.10: The three types of layout regularization constraints in my method. The vertical alignment constraint between $B_1$ and $B_3$ is formulated as $x_1 + w_1/2 - (x_3 + w_3/2) = 0$. The horizontal alignment constraint between $B_1$ and $B_2$ is expressed as $y_1 + h_1/2 - (y_2 + h_2/2) = 0$. The size constraints between $B_1$ and $B_2$ are $w_1 - w_2 = 0$, $h_1 - h_2 = 0$, and $d_1 - d_2 = 0$. The spacing constraint between $B_1$, $B_2$, and $B_3$ is formulated as $x_1 + w_1 - x_2 = x_2 + w_2 - x_3$.

- **Alignment constraint.** This constraint ensures two elements $e_i$ and $e_j$ align well in either the horizontal or vertical direction. For *horizontal* alignment, the constraint can be expressed as

$$x_i + \frac{w_i}{2} - (x_j + \frac{w_j}{2}) = 0. \tag{4.12}$$

  Similarly, alignment along the *vertical* direction can be enforced by

$$y_i + \frac{h_i}{2} - (y_j + \frac{h_j}{2}) = 0. \tag{4.13}$$

- **Size constraint.** This constraint ensures that similar elements share the same width, height, and depth, formulated as,

$$\begin{aligned} w_i - w_j &= 0, \\ h_i - h_j &= 0, \\ d_i - d_j &= 0. \end{aligned} \tag{4.14}$$

- **Spacing constraint.** This constraint ensures that elements are distributed evenly along a certain direction. Specifically, given two pairs of elements $(e_i, e_j)$ and $(e_m, e_n)$ exhibiting identical spacing along the horizontal direction, this constraint can be expressed as

$$x_i + w_i - x_j = x_m + w_m - x_n. \tag{4.15}$$

Similarly, the same spacing constraint in the vertical direction can be given by

$$y_i + h_i - y_j = y_m + h_m - y_n. \tag{4.16}$$

Along with the above constraints, I intend to alternate the model with minimum changes to its current shape. To this end, my objective function takes into consideration the deviations of both positions and sizes of facade elements, motivating two energy terms, namely the *position deviation* $E_{pos}$ and *size deviation* $E_{size}$, formulated as follows:

$$E_{pos} = \sum_{i=1}^{n} (x_i + \frac{w_i^*}{2} - x_i - \frac{w_i}{2})^2 + (y_i + \frac{h_i^*}{2} - y_i - \frac{h_i}{2})^2, \tag{4.17}$$

$$E_{sc} = \sum_{i=1}^{n} (w_i^* - w_i)^2 + (h_i^* - h_i)^2 + (d_i^* - d_i)^2, \tag{4.18}$$

where variables without and with '$*$' denote the value before and after layout optimization. The overall objective function is given by

$$E = E_{pos} + \tau E_{size}, \tag{4.19}$$

where $\tau$ is the weight balancing between *position deviation* $E_{pos}$ and *size deviation*. This is a quadratic programming problem, and I solve it using the off-the-shelf Simplex solver provided by Gurobi [65]. Finally, the resulting output yields a compact and regularized LoD3 building model.

## 4.3. RESULTS AND EVALUATION

My method has been implemented in C++ using Easy3D [128], OpenCV [27], and CGAL [167]. All experiments were conducted on a MacBook Pro with an M1 CPU and 32G RAM. Extensive experiments on a variety of inputs have demonstrated the effectiveness of the proposed method.

### 4.3.1. TEST DATASETS

I have validated my method on three different MVS mesh datasets, namely InstanceBuilding [38], UrbanBIS [192], and VexcelImaging meshes [1].

- InstanceBuilding. The buildings were reconstructed using *Bentley ContextCapture* [2] from UAV images captured by DJI.
- UrbanBIS. The acquisition devices include the *DJI PHANTOM 4 RTK* drone [3] with the built-in camera and the *DJI M300RTK* drone [4] loaded with five *HD PSDK* 102S aerial cameras. Dedicated aerial paths were adopted to improve the quality of the reconstructed models.

[1] https://www.vexcel-imaging.com/ultracam-osprey-4-1/
[2] https://www.bentley.com/software/contextcapture/
[3] https://enterprise.dji.com/phantom-4-rtk
[4] https://enterprise.dji.com/matrice-300

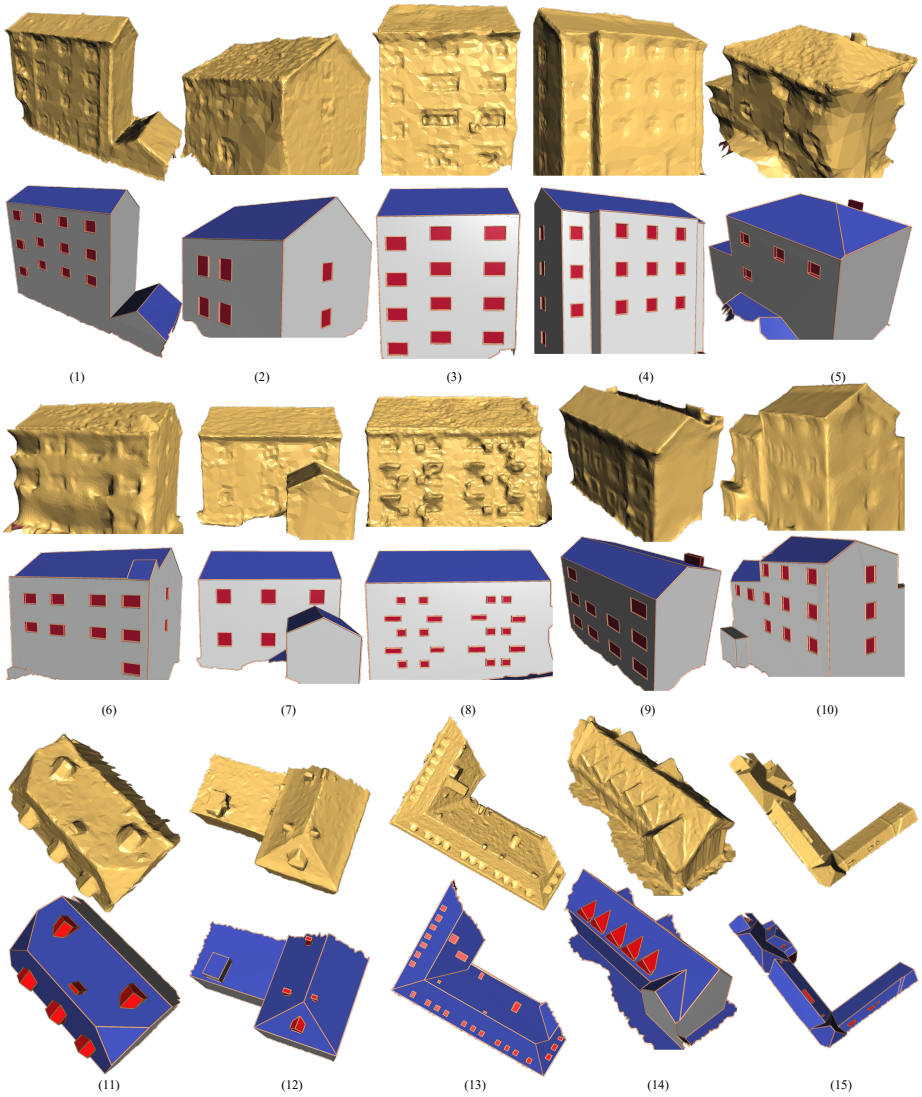



Figure 4.11: The reconstruction results of my method for buildings demonstrating diverse styles and sourced from different datasets. (1) - (5) are from InstanceBuilding [38], (6) - (10) are from UrbanBIS [192], (11) - (15) are from VexcelImaging mesh.

- VexcelImaging dataset. The buildings were reconstructed by utilizing UltraMap software, which processes UAV images obtained via the UltraCam Osprey 4.1 platform [5].

### 4.3.2. QUALITATIVE RESULTS

I have applied my method to reconstruct compact LoD3 building models across all three datasets. The buildings showcased in Figure 4.11 originate from various cities, each possessing unique structural properties. Buildings (1) to (5) are sourced from the InstanceBuilding dataset, (6) to (10) from the UrbanBIS dataset, and (11) to (15) from the VexcelImaging dataset.

I have tested my method on examples exhibiting varying degrees of imperfections, such as noise and holes. Notably, the input mesh in cases (1) to (10) exhibits significant blurring during data capture, leading to over-smoothing within the facades. Despite these challenges, my method adeptly recovered the structures, thanks to the enforcement of geometric constraints imposed on the planar segments. Take, for example, case (2), where the window resolution is exceptionally low and almost planar, creating complexities for automatic segmentation methods to decompose it into five planar primitives. In the face of these difficulties, distinct perceptible cues remained recognizable to the user. Leveraging this perceptual information, I utilized the introduced rubber-band tool to loosely outline window contours to facilitate finer segmentation. The windows were then successfully recovered after optimization.

Moving on to examples (11) to (15) from the VexcelImaging dataset, they differ from the preceding datasets by often featuring rich roof structure details but comparatively less facade information. My method, however, demonstrates effectiveness in recovering small-scale elements such as dormers or chimneys in these buildings.

These qualitative results show that my approach exhibits robustness across datasets, overcoming challenges posed by blurring, low resolutions, and diverse structural complexities, thereby showcasing its versatility and efficacy in reconstructing detailed LoD3 building models.

### 4.3.3. QUANTITATIVE RESULTS

To gain insights into the accuracy and computational efficiency of my LoD3 building model reconstruction method, I have conducted a comprehensive quantitative evaluation of the reconstructed LoD3 building models. Due to the absence of ground-truth reconstructions for all buildings within the datasets, I opted for the widely used metric, Root Mean Square Error (RMSE), to assess the fidelity of my results. The RMSE is defined as the square root of the average of squared Euclidean distances from the vertices of the input mesh to the closest faces on the reconstructed model. For a standardized evaluation, I utilized the diagonal length of each building's bounding box as a reference, allowing us to compute relative errors on a global scale.

Table 4.1 presents the statistical summary of quantitative results for the buildings showcased in Figure 4.11. Notably, the number of building faces has experienced a reduction by at least one order of magnitude, underscoring the efficiency and effectiveness

[5] https://www.vexcel-imaging.com/ultracam-osprey-4-1/

| Figure | Input faces | Output faces | RMSE |
|---|---|---|---|
| (1) | 8,045 | 158 | 0.008 |
| (2) | 9,258 | 96 | 0.023 |
| (3) | 15,584 | 119 | 0.006 |
| (4) | 6,811 | 186 | 0.006 |
| (5) | 12,743 | 61 | 0.013 |
| (6) | 40,548 | 179 | 0.010 |
| (7) | 8,145 | 71 | 0.009 |
| (8) | 24,104 | 180 | 0.010 |
| (9) | 27,552 | 167 | 0.011 |
| (10) | 25,807 | 234 | 0.013 |
| (11) | 6,468 | 186 | 0.008 |
| (12) | 9,385 | 276 | 0.005 |
| (13) | 29,939 | 164 | 0.005 |
| (14) | 6,535 | 42 | 0.004 |
| (15) | 27,072 | 98 | 0.005 |

Table 4.1: Statistical summary of quantitative results for the buildings showcased in Figure 4.11. The numbers of faces in the input and output and RMSE are reported.



of my reconstruction method. It is also worth noting that the entire process, from input mesh to reconstructed model, took between 10 to 20 minutes, demonstrating the feasibility of my approach for real-world applications.

### 4.3.4. IMPACT OF USER INPUT

The process of creating LoD3 building models from MVS meshes involves several steps where user input play a critical role in ensuring the quality and accuracy of the final output. My system is designed to incorporate progressive user interactions to refine these models, allowing for incremental improvements in detail and accuracy. Importantly, the required user input is minimal and involves only high-level guidance rather than precise specifications, ensuring the process remains efficient and user-friendly.

As illustrated in Figure 4.12, an increase in user input leads to the creation of more detailed and accurate building models. Note that, user interactions collected in my study refer to the actions such as sketching the outlines of facade elements using the *rectangle/polygon* tool and refining segmentation results with the *split/merge* tool. These actions guide the system in correcting inaccuracies in the initial segmentation of facade elements, thereby enhancing the quality of the final model. The recorded data in Figure 4.13 reveals a clear trend: increased user interactions contribute to the development of more accurate models, albeit at the cost of longer processing times.

Overall, increased user interaction enables the system to capture finer details and produce more comprehensive building models. This improvement in detail is largely attributed to the system's effective utilization of user input, which allows for the accurate segmentation of facade elements that are typically challenging for automated methods to capture. Consequently, user interaction is crucial in achieving high-fidelity building

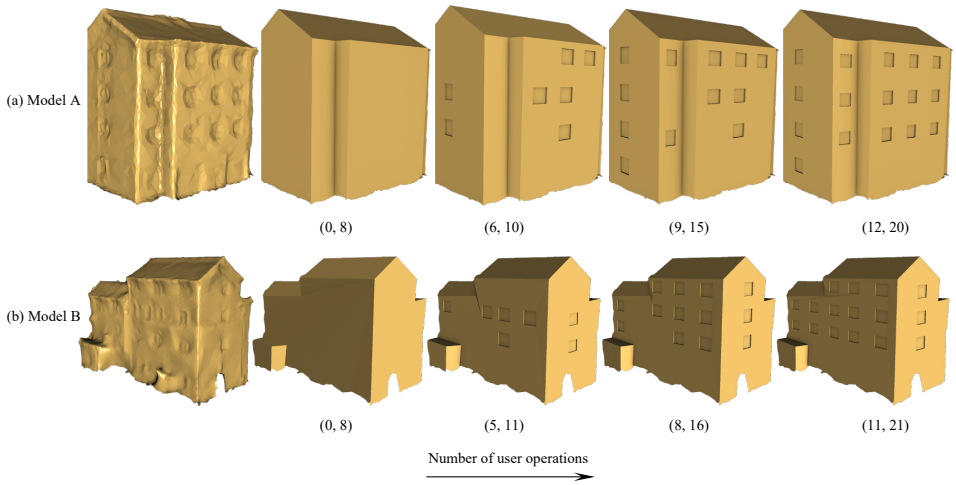



Figure 4.12: Results with incremental user input: the first column displays the input model, while the subsequent columns present the results as user interactions progressively increase. The numbers below each subfigure indicate the numbers of involved sketching operations and split/merge operations.

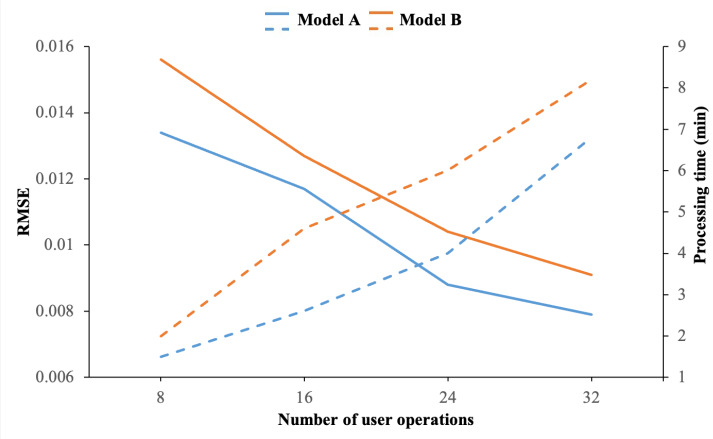



Figure 4.13: The plot of RMSE and processing time values for the reconstructed models in Figure 4.12 throughout the modeling process. The solid lines and dashed lines represent the RMSE values and processing times, respectively.

models that meet LoD3 standards. However, in our current system design, it is at the user's preference to trade-off between the level of detail and processing time, as more interactions, while improving model quality, also increase the time required to complete the reconstruction.

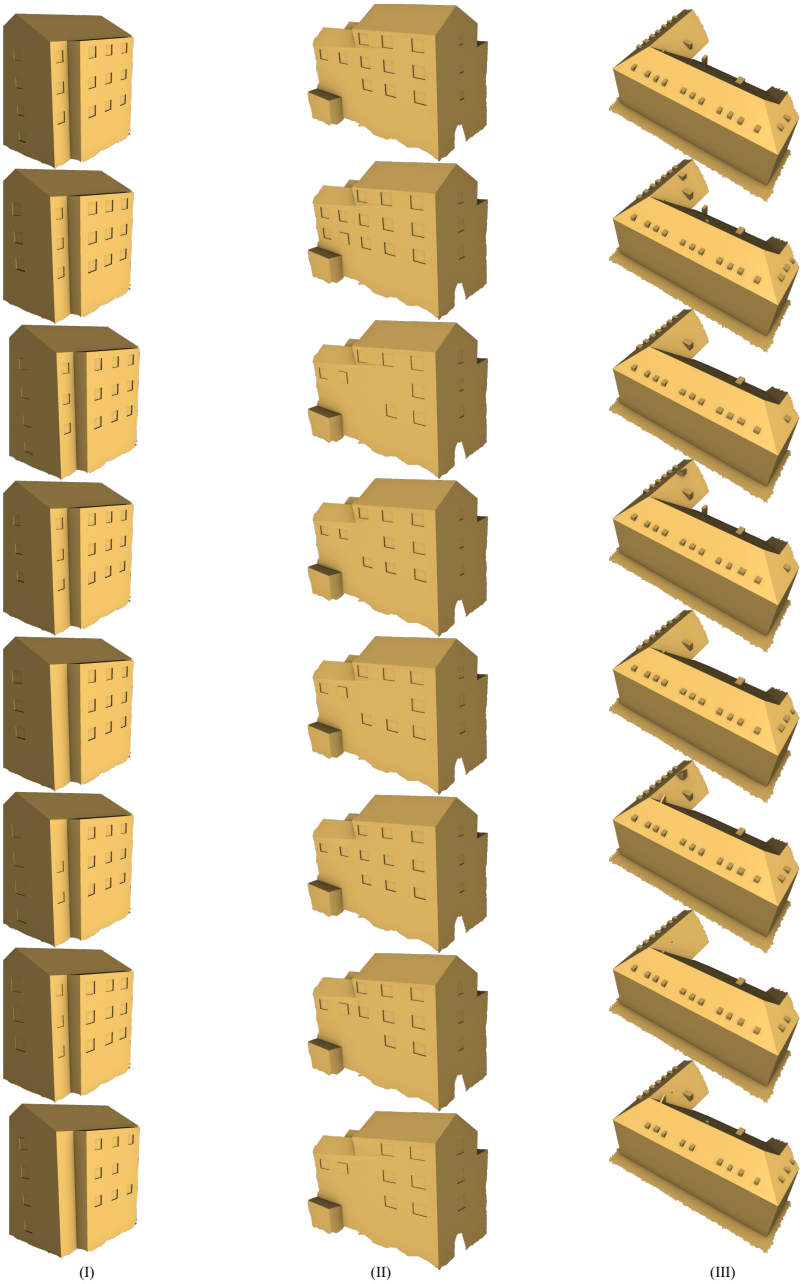



Figure 4.14: Reconstructed results from eight participants. The three examples correspond to building (4), (10), and (13) in Figure 4.11.

| Model | Group | Average Completion Time (minutes) | Average Accuracy (RMSE) | Average Number of User Interactions |
|---|---|---|---|---|
| 1 | I | 9.3 | 0.0095 | 28 |
| | II | 11.5 | 0.0123 | 36 |
| 2 | I | 8.5 | 0.0093 | 25 |
| | II | 12.8 | 0.0078 | 37 |
| 3 | I | 12.5 | 0.0053 | 34 |
| | II | 16.3 | 0.0068 | 47 |

Table 4.2: Statistical results for different models and participant groups. I and II represent the experienced group and the inexperienced group, respectively.



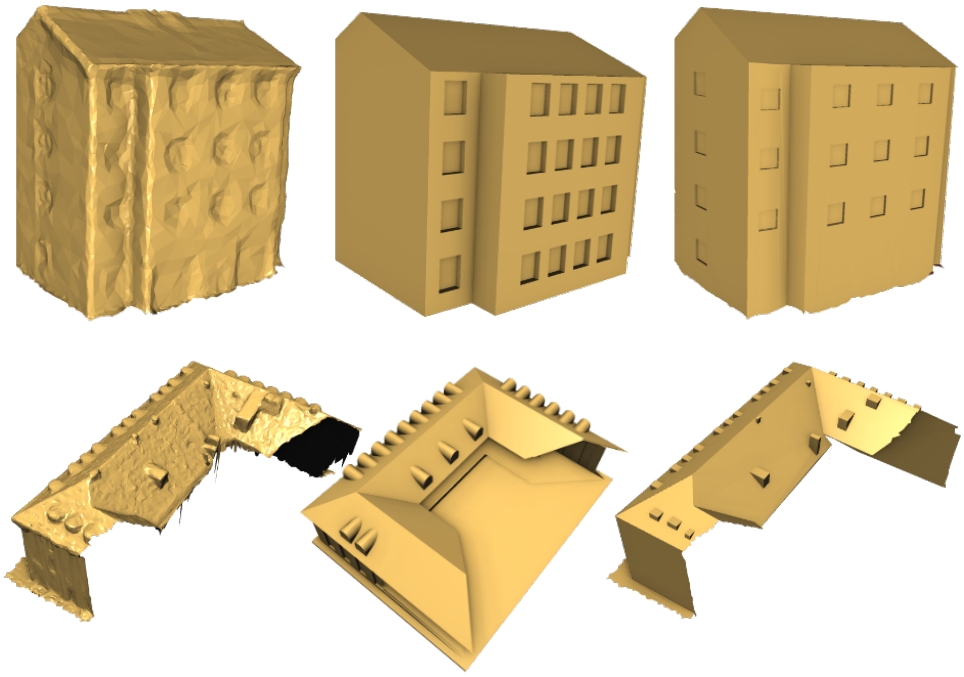
### 4.3.5. USER STUDY

To evaluate the general applicability of my modeling tool across users, I conducted a user study. 8 participants were selected from different backgrounds, including fields such as computer science and geomatics. To achieve a balanced representation, the participants were divided into two groups based on their prior experience with 3D modeling techniques: one group consisted of individuals with expertise in 3D modeling, while the other group included participants with limited experience in such techniques.

Before commencing the tasks, all participants, regardless of their previous experience, underwent an introduction to familiarize themselves with the interactive system. Following the training session, each participant was instructed to model three building models, with the goal of achieving the highest level of detail (LoD3) reconstruction within a standardized time limit of 30 minutes per model. Specific instructions were provided to highlight the importance of capturing intricate details, such as dormers and windows, which can be identified by the users. These tasks were designed to test their ability to apply the knowledge gained during the introduction and to assess the system's usability across different levels of modeling complexity. The time constraint was set to evaluate the system's efficiency.

The statistical results, including average completion times, average accuracy, and average number of user interactions across the two participant groups, are presented in Table 4.2. To visually present the results, Figure 4.14 displays all 24 reconstructed models (3 buildings per participant across 8 users). This figure highlights the level of detail each participant achieved in reconstructing the building models. As shown in Figure 4.14, participants generally succeeded in recovering basic roof structures and primary features such as windows and dormers within the allotted time. However, the specific details varied, particularly with regard to smaller features. While some participants chose to model even subtle details, others tended to ignore them, especially when these features were not prominent in the original data. Similar to a manual modeling tool, user interactions also play a crucial role in the modeling process.

Overall, my interactive system demonstrates its capability to facilitate the creation of a LoD3 building model with high-level user guidance. The intuitive interactions allow participants to iteratively refine their models, highlighting my system's potential to be an effective tool for both novice and experienced users in the field of 3D modeling.



(a) Input (b) AutoCAD (c) Ours

Figure 4.15: Comparison between our results and the models manually reconstructed using commercial software AutoCAD.

### 4.3.6. COMPARISONS WITH MANUAL RECONSTRUCTION USING COMMERCIAL SOFTWARE

I evaluate the performance of my system in comparison to manual reconstruction using the commercial software AutoCAD, in order to assess its effectiveness in 3D building modeling. To do this, I asked an expert with three years of experience in using AutoCAD to reconstruct the two building models. As demonstrated in Figure 4.15, the results produced by my system closely resemble those of manual modeling. Both my approach and AutoCAD are capable of reconstructing basic roof structures as well as detailed elements such as dormers and windows. My method constrains the output to polygonal shapes, whereas AutoCAD is capable of representing curved geometries, as shown in the subfigures on the bottom row. Additionally, my system is limited to reconstructing surfaces that are present in the input data, whereas software like AutoCAD, using tools such as "Edit" and "Add Face," allows for the recovery of missing or incomplete surfaces through more extensive user interactions. However, human-created models of varying complexity take 1-3 hours, while my tool only requires a maximum of 20 minutes. The instruction I gave to the expert was to model a polygonal shape that is as close as possible to the given mesh and meanwhile is as simple as possible. Observing the overall manual modeling process, I found that the expert follows this logic. First, the expert de-

lineated the geometric structures by outlining the building's silhouette. Following this, the expert generated the faces based on the constructed topology. This process required the expert to elaborately place vertices and edges to form the structures of the building. An important part of this workflow was the constant adjustment and verification of vertex positions from various perspectives, requiring frequently switched views to ensure the accuracy and consistency of reconstructed models.

Commercial software such as AutoCAD provides a comprehensive set of interactive tools, enabling users to model a wider range of curved structures. However, these manual modeling tools also come with certain limitations, especially concerning LoD3 building modeling. A significant limitation is that professional-grade tools often require extensive software skills to be used effectively. This complexity can be a barrier for users without substantial experience or training. For instance, the expert had to conduct a variety of complex operations including adjusting vertex positions in a 2D plane, cutting and extruding faces to shape the model, and rotating and translating grouped vertices to refine the geometry. In contrast, my system is designed to be more accessible, requiring minimal user input, just intuitively delineating the contour of facade elements without frequently adjusting the views. These improvements make it more user-friendly and efficient.



Overall, while commercial software presents strong and versatile modeling capabilities, my system aims to offer a more intuitive alternative designed to simplify the LoD3 reconstruction process for users.

### 4.3.7. COMPARISONS WITH OTHER METHODS

I compared my method against other mesh simplification/ polygonization-based methods, specifically GH's method [61] and BLN's method [26]. As illustrated in Figure 4.16, my method excels in reconstructing intricate structures such as chimneys and dormers, attributed to the flexible and robust user guidance incorporated. Contrastingly, fully automatic methods often overlook facade details or struggle to recover them accurately. Notably, my method achieves the most accurate and regularized reconstruction results, exhibiting the lowest RMSE values.

### 4.3.8. PARAMETERS

In my method, several parameters are empirically determined and set to fixed values for the experiments. Through iterative experimentation, I select parameter values that consistently yielded optimal performance in terms of accuracy and computational efficiency across a range of test cases. The weights in Equation 4.8 are set to $\lambda_d = 1$, $\lambda_s = 1.8$, and $\lambda_l = 1.3$, respectively. For Equation 4.9, the weights are $\lambda_1 = 1$, $\lambda_2 = 0.003$, and $\lambda_3 = 1$, respectively. Additionally, for Equation 4.11, $\mu_1 = 0.01$, $\mu_2 = 10$, and $\mu_3 = 5$ are the assigned weights. Finally, the weight for Equation 4.19 is set to $\tau = 5$.

### 4.3.9. LIMITATIONS

The methodology's reliance on planar approximations imposes two restrictions. First, the absence of volumetric gap-filling mechanisms brings challenges in generating watertight models. It limits utility in urban applications requiring strict topological consistency (e.g., urban simulations). Second, the current framework prioritizes planar ge-

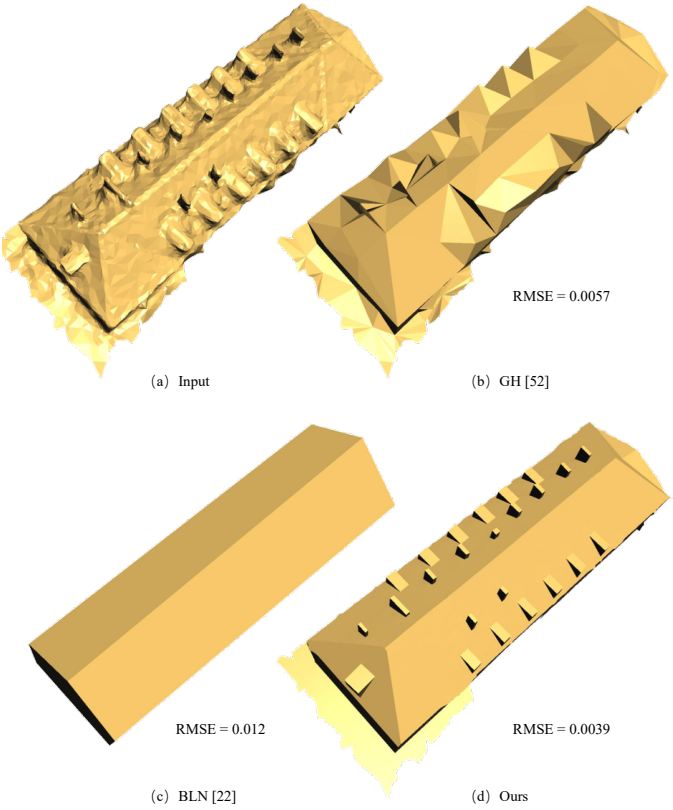



Figure 4.16: Comparison between my method and other methods. (a) Input mesh. (b) GH's method [61]. (c) BLN's method [26]. (d) My method.

ometries, thereby excluding non-planar architectural features (e.g., cylindrical towers, free-form façades). While this simplification enhances computational tractability, it reduces fidelity for structures dominated by complex curvature—a limitation that future work might address through hybrid parametric and data-driven geometric representations.

## 4.4. CONCLUSION AND FUTURE WORK

I present a novel method to semi-automatically reconstruct LoD3 building models from MVS meshes. My approach leverages the combined advantages of automatic detection and human guidance, enabling the effective reconstruction of the facade elements and roof superstructures of buildings. In particular, my method is robust against noise and diverse building styles. Given my underlying assumption that buildings are composed of planar segments, extending the framework to include cone, spherical, or cylindrical shapes could also be a promising direction.

Having proposed an LoD3 building reconstruction approach in this chapter, I now shift my focus to enhancing the geometric regularities of the reconstructed building models. While LoD2 or LoD3 models offer detailed depictions of buildings, the original reconstructed models often exhibit irregularities such as deviations from orthogonality, lack of parallelism, and absence of symmetry. To address this issue, the following chapter introduces a new method for symmetrizing 2D polygonal shapes. Repetitive building elements and superstructures are projected onto the facade plane to generate 2D abstract shapes. Subsequently, the algorithm simultaneously detects symmetric elements and symmetrizes the building abstract shapes, thereby improving their overall quality and enhancing their suitability for further urban design applications.

# 5

# AUTOMATIC SYMMETRIZATION OF 2D POLYGONAL SHAPES

*This chapter proposes a novel optimization-based framework that jointly detects and optimizes symmetry for 2D shapes represented as polygons. My method can detect and optimize symmetry using a single objective function. Specifically, I formulate symmetry detection and optimization as a mixed-integer program. The proposed method first generates a set of candidate symmetric edge pairs, which are encoded as binary variables in my optimization. The geometry of the shape is expressed as continuous variables, which are then optimized together with the binary variables. The symmetry of the shape is enforced through designed hard constraints. After optimization, both the optimal symmetric edge correspondences and the geometry are obtained. My method simultaneously detects all the symmetric primitive pairs and enhances the symmetry of the model while minimally altering its geometry. I have tested my method on a variety of shapes from designs, vectorizations, and reconstruction algorithms, and the results demonstrate its effectiveness.*

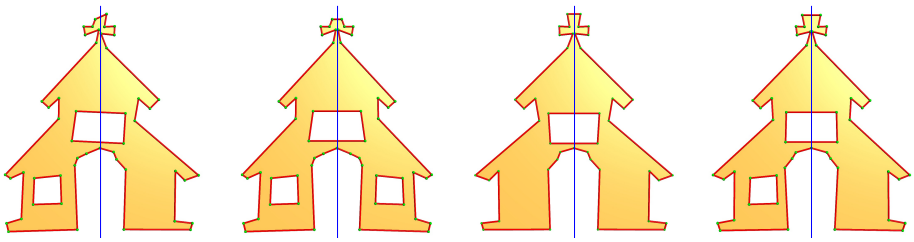
This chapter has been published in Computer-Aided Design [75].

## 5.1. INTRODUCTION

Symmetry is an essential attribute of nature, and it also plays an important role in the design of various man-made objects such as buildings, furniture, and mechanical parts [127]. However, symmetry can easily be distorted in digitalisation processes. For example, in shape design, imperfect input data, like the raw shapes created by users in sketching-based modeling software, often do not exhibit the desired symmetry. Manually editing the models to enhance symmetry is a tedious and time-consuming task. It can be more challenging for large-scale architectural models, where symmetry is much more easily distorted due to the inevitable noise and outliers during the data acquisition process. In general, objects exhibiting symmetric structures are easier to perceive and understand, and it also serves as effective prior knowledge in a variety of applications [116], such as object alignment [139, 166], editing [177, 199], compression [157, 174], and reconstruction [158, 169].

Automatic symmetrization of 2D shapes is a challenging task. Simply copying and transforming object parts [202] usually does not lead to acceptable results. For example, the overall geometry of the shape shown in Figure 5.1 (a) is reflective symmetric, copying and transforming either the left or the right part cannot generate a satisfying result because the geometric information from the counterpart is simply ignored. This is confirmed by the symmetrization results shown in Figure 5.1 (b) and (c).





(**a**) Input (**b**) Copy-transform left part (**c**) Copy-transform right part (**d**) My result

Figure 5.1: Comparing two strategies for symmetrization of a 2D shape. (b) and (c) symmetrize the shape by copying and transforming half of the shape. My method automatically and jointly detects the optimal symmetry (defined by the edge correspondences) and optimizes the shape by minimal modification of the original shape, resulting in a more natural symmetrization result. Note that my method achieves strict symmetry for the majority parts of the shape, except for the left window that does not have a symmetric counterpart in the input shape.

As discussed in Chapter 2, existing approaches to shape symmetrization typically follow a two-stage process: explicit symmetry detection followed by symmetry enhancement. In the first stage, methods aim to identify the optimal symmetry using techniques such as brute-force validation or random sample consensus, which are both computationally intensive and sensitive to noise. Once the optimal symmetry is identified, the object is symmetrized through a transformation. Although these approaches are effective, their success heavily depends on the accuracy of the initial symmetry detection, making them prone to difficulties in the presence of noise or multiple local symmetries.

In this work, I introduce a novel optimization-based framework for the automatic symmetrization of 2D polygonal shapes that exhibit predominantly global extrinsic re-

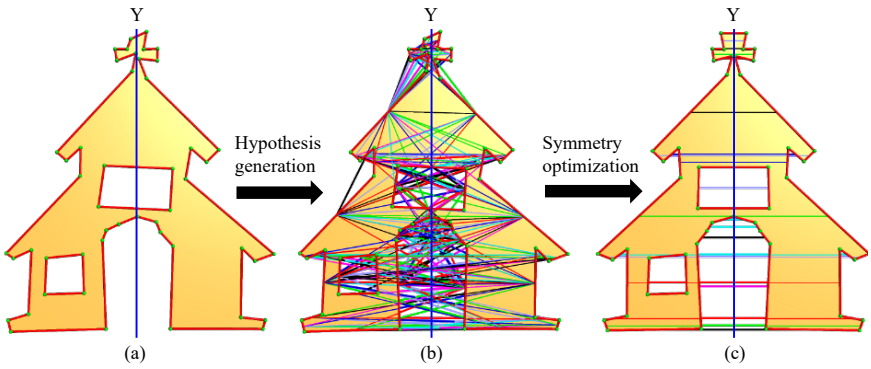



Figure 5.2: An illustration of the symmetrization method on a vectorized building shape with a symmetry axis aligned with the Y-axis. It takes as input a 2D shape (a) and first generates a set of candidate symmetric edge pairs (b). Every two edges connected by a line (in random color) in (b) have the potential to define a reflectional symmetry. After optimization, the symmetry of the shape is optimized by introducing minimal deviation to the shape's vertices (c), and meanwhile, the symmetric edge correspondences are identified. In this case, every two edges connected by the line in (c) are the symmetric edge pairs.



flectional symmetry. My motivation is to simultaneously detect and enhance symmetry within a single optimization stage. In other words, my method determines the optimal global symmetry axis of a shape, and when enhancing the symmetry using this symmetry axis, the change to the geometry of the shape is minimized. Different from existing works, I achieve symmetry detection and symmetrizations at the same time, within the same optimization process. To achieve this goal, I propose a *hypothesize-and-select* strategy to avoid wrong or sub-optimal symmetry detection that is commonly encountered in existing two-stage approaches. Given a 2D shape, I first generate a set of candidate symmetric edge pairs. Then a mixed-integer programming formulation is designed to select the most confident symmetric edge pairs and at the same time ensure that the optimal symmetrization is achieved by introducing minimum geometry change to the original shape. In my optimization formulation, I also introduce hard constraints that enforce the final shape to be strictly symmetric. My main contributions are two-fold:

- The *first* symmerization framework that can simultaneously determine edge correspondences and optimize the symmetry of 2D shapes.
- A novel mixed-integer programming formulation based on the *hypothesize-and-select* strategy, which guarantees the final shape to be symmetric.

## 5.2. METHODOLOGY

The input to my method is a 2D shape represented by one or multiple polygons consisting of $N$ edges. Without loss of generality, I assume that the symmetry axis approximately passes through the centroid of the bounding sphere of the input shape. In Sec. 5.2.1, I describe my method for shapes whose symmetry axis aligns with the Y-axis, and Sec. 5.2.2 elaborates on handling shapes with arbitrary symmetry axis.

### 5.2.1. SYMMETRIZATION FOR SHAPES WITH KNOWN SYMMETRIC AXES

For shapes whose symmetry axis aligns with their Y-axis, symmetrization is still challenging because the symmetric edge correspondence information is not known. I jointly find the edge correspondences and symmetrize the shape by exploring a set of symmetry hypotheses and optimizing the vertices of the shape in a way such that the symmetry of the shape is maximized with minimal change to its original shape. The idea of the proposed method is illustrated in Figure 5.2, and it consists of the following two parts:

- *Hypothesis generation.* I identify a set of candidate symmetric edge pairs. This initial set of edge pairs is further pruned based on two geometric tests, to reduce the size of the resulting optimization problem in the subsequent symmetry optimization step.
- *Symmetry optimization.* I optimize the symmetry of the shape in a way such that the overall change to the shape is minimal. The symmetrization is formulated as a mixed integer quadratic program, with an objective penalizing the overall deformation and with hard constraints enforcing strict symmetry of the final shape. The potential symmetric edge pairs are encoded as binary variables, and the geometry (i.e., vertices) of the shape is expressed as a set of continuous variables. After solving the optimization problem, both the optimal symmetric edge pairs and the geometry are obtained.



#### HYPOTHESIS GENERATION

I generate a sufficiently large number of candidate symmetric edge pairs, and I use these edge pairs to define my objective function for symmetry optimization. In theory, every edge pair has the potential to be symmetric, and the total number of such edge pairs is $\binom{N}{2}$. Since the potential edge pairs are encoded as binary variables in the subsequent symmetry optimization step, exhaustive enumeration of all the edge pairs will result in a large optimization problem that may not be feasible to solve within a reasonable time window. Based on the observation that two edges $u_i$ and $u_j$ are nearly symmetric if they are on the opposite sides of the symmetry axis and have sufficient proximity along the symmetry axis, illustrated in Figure 5.3 (a), I prune the edge pairs by two simple geometric tests:

- Side-of-axis test. The two edges must lie on the two sides of the symmetry axis, i.e.,

$$sign(e_i) * sign(e_j) < 0, \tag{5.1}$$

  where $sign(e)$ denotes the relative orientation of the edge $e$ concerning the Y-axis. Specifically, $sign(e)$ has a positive value if $e$ lies on the left side of the Y-axis and a negative value on the right side. The example shown in Figure 5.3 (b) will not pass the side-of-axis test.

- Proximity test. The two edges must be within sufficient proximity along the Y-axis. In my work, this condition is satisfied if two edges overlap or their distance is smaller than a threshold along the Y-axis, i.e.,

$$\|dist_Y(e_i, e_j)\| \leq d_t. \tag{5.2}$$

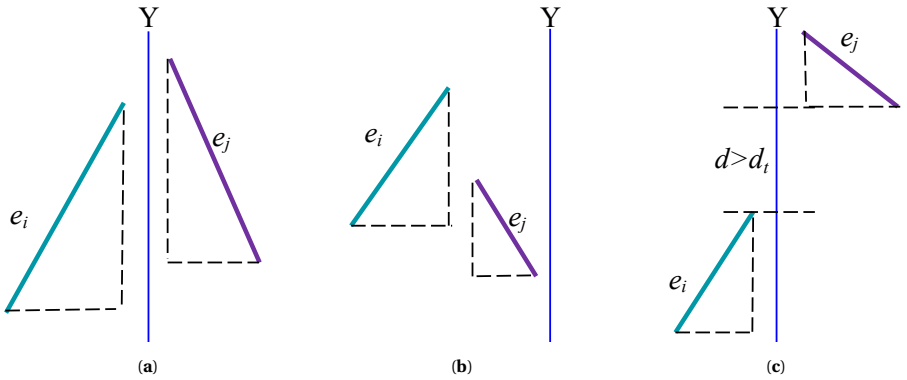



Figure 5.3: A few example cases processed by the geometric tests that prune candidate symmetric edges pairs with low confidence. This will reduce the size of the resulting optimization problem. The edge pair shown in (a) has a high probability to be symmetric, which will pass the geometric test. In contrast, the edges pairs in (b) and (c) do not have sufficient confidence (evaluated against some threshold detailed in Sec. 5.2.1) to represent meaningful symmetry and thus will be rejected.



In this work, to achieve a balance between the edge length involved in the test and the hypothesis numbers in the subsequent optimization, I empirically set $d_t = 0.5 * max\{length(e_i), length(e_j)\}$, where $length(e)$ denotes the length of an edge $e$. With this threshold, the example shown in Figure 5.3 (c) will not pass the proximity test.

By pruning the potential symmetry edge pairs using these simple tests, I can significantly reduce the number of candidate edge pairs. For example, in Figure 5.5 (13), the total number of candidate edge pairs is reduced from 2,346 to 450, speeding up the subsequent symmetry optimization step.

### SYMMETRY OPTIMIZATION

After obtaining the candidate symmetry edge pairs $P$, the next step is to jointly select an optimal subset of the candidate edge pairs and optimize the symmetry of the shape.

Let a binary variable $z_{ij}$ encode if an edge pair $(e_i, e_j)$ is valid ($z_{ij}$=1) or not ($z_{ij}$=0). By expressing the shape geometry (i.e., the coordinates of the vertices of the shape) in continuous variables, the symmetrization of the shape can be formulated as a mixed-integer program that balances two terms: *deformation* and *tolerance*.

- The *deformation* term measures how much the resulting shape deviates from its original geometry. This term is designed to minimize the change to the original shape while achieving symmetrization, which is defined as the sum of the weighted square deviation of all the vertices, i.e.,

$$E_d = \sum_i w_d \cdot \| v_i - v_i' \|^2, \tag{5.3}$$

where $v_i$ and $v_i'$ denote the coordinates of a vertex before and after symmetrization, respectively. $w_d$ is the coefficient of each vertex, which is computed as,

$$w_d = e^{\pi - \gamma}, \tag{5.4}$$

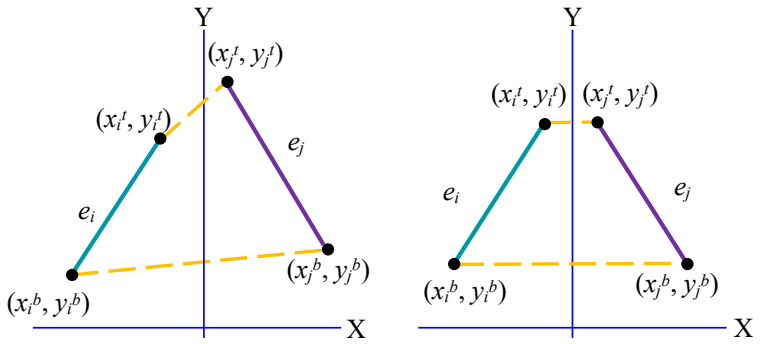
where $\gamma$ is the angle between the two incident edges of the vertex $v_i$. Intuitively, this weighting scheme encourages vertices with smaller $\gamma$ to have less freedom to move, which is intended to preserve sharp features.

- The *tolerance* term is designed to allow us to handle approximately symmetric shapes. It is simply defined as the number of the non-symmetric edge pairs, i.e.,

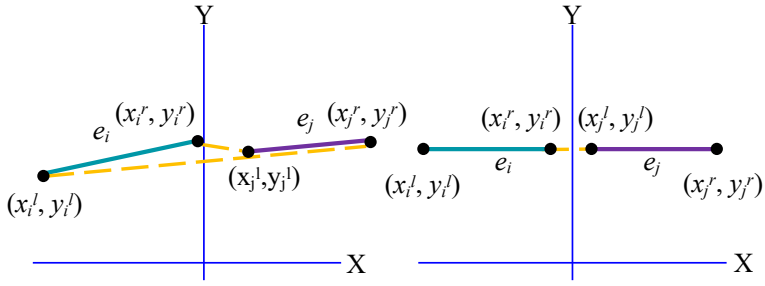
$$E_t = \sum_{(ij)\in P} (1 - z_{ij}), \tag{5.5}$$

where $P$ denotes the entire set of potential symmetric edge pairs. $z_{ij}$ has a value of 1 if two edges $e_i$ and $e_j$ are indeed symmetric after optimization.






(**a**) The *vertical* pattern




(**b**) The *horizontal* pattern

Figure 5.4: Two types of edge correspondence patterns. The vertices with superscript $t$, $b$, $l$, and $r$ represent the corresponding *top*, *bottom*, *left*, and *right* endpoint of an edge, respectively. (a) The *vertical* pattern. Two vertices of only the same type can be matched (i.e., only top-top or bottom-bottom is allowed). (b) The *horizontal* correspondence pattern. Two vertices of only different types can be matched (i.e., only left-right or right-left is allowed). For both (a) and (b), the left and right subfigures respectively illustrate the edge pair before and after symmetry optimization.

**Hard constraints**. To ensure strict symmetry in the final shape and encourage structure preservation, I also introduce two hard constraints in the symmetry optimization process.

- *Perfect symmetry.* This constraint enforces that the final shape is strictly symmetric. To achieve this, I define two edge correspondence patterns based on the relative orientation of a potentially symmetric edge pair, namely *vertical* and *horizontal*, as shown in Figure 5.4. Given an edge pair $(e_i, e_j)$, if the projected length of at least one of the two edges on the X-axis is longer than the corresponding projected length on the Y-axis, this edge pair is considered to have the *horizontal* pattern, and it is denoted as $(e_i, e_j) \in H$ for simplicity. Otherwise, the edge pair has the *vertical* pattern, denoted as $(e_i, e_j) \in V$. Considering these relative orientations of edge pairs, the *perfect symmetry* hard constraint can be formulated as

$$z_{ij} \cdot S(e_i, e_j) = 0, \quad \forall (e_i, e_j) \in P, \tag{5.6}$$

  where $P$ denotes the complete set of potentially matched edge pairs, and $S(e_i, e_j)$ measures how much the edge pair deviates from being perfectly symmetric, i.e.,

$$S(e_i, e_j) = \begin{cases} \|x_i^l + x_j^r\| + \|x_i^r + x_j^l\| + \\ \|y_i^r - y_j^l\| + \|y_i^l - y_j^r\|, \text{if } (e_i, e_j) \in H \\ \\ \|x_i^t + x_j^t\| + \|x_i^b + x_j^b\| + \\ \|y_i^t - y_j^t\| + \|y_i^b - y_j^b\|, \text{otherwise} \end{cases} \tag{5.7}$$

  The symbols and their superscripts are illustrated and explained in Figure 5.4.

- *Single matching.* In a symmetric shape, one edge is matched to only one other edge. This constraint is intended to preserve the structure of the shape by disallowing one edge to be symmetric with multiple other edges.

**The complete formulation**. With the aforementioned energy terms and hard constraints, the complete formulation for symmetry optimization can be written as

$$\begin{aligned} \min \quad & E_d + \lambda E_t \\ \text{s.t.} \quad & \begin{cases} z_{ij} \cdot S(e_i, e_j) = 0, & \forall (ij) \in P \\ \sum_{j \neq i} z_{ij} \leq 1, & \forall 1 \leq i \leq N \\ z_{ij} \in \{0, 1\}, & \forall (ij) \in P \end{cases} , \end{aligned} \tag{5.8}$$

where $\lambda$ is used to control the degree of non-symmetry, and $\sum_{j \neq i} z_{ij}$ counts the total number of edges that are symmetric with $e_i$ in the final shape. The formulation in Equation 5.8 is a standard mixed integer quadratic-constrained programming problem. In this formulation, the binary variables indicate whether the candidate edge pairs are symmetric or not, and the continuous variables correspond to the coordinates of the vertices in the shape. I solve this optimization problem using off-the-shelf Newton Barrier solver provided by Gurobi [65]. After optimization, the optimal values for both the binary and continuous variables are obtained, meaning the optimal symmetric edge pairs have been identified, and meanwhile the optimal positions of all the vertices have been optimized.

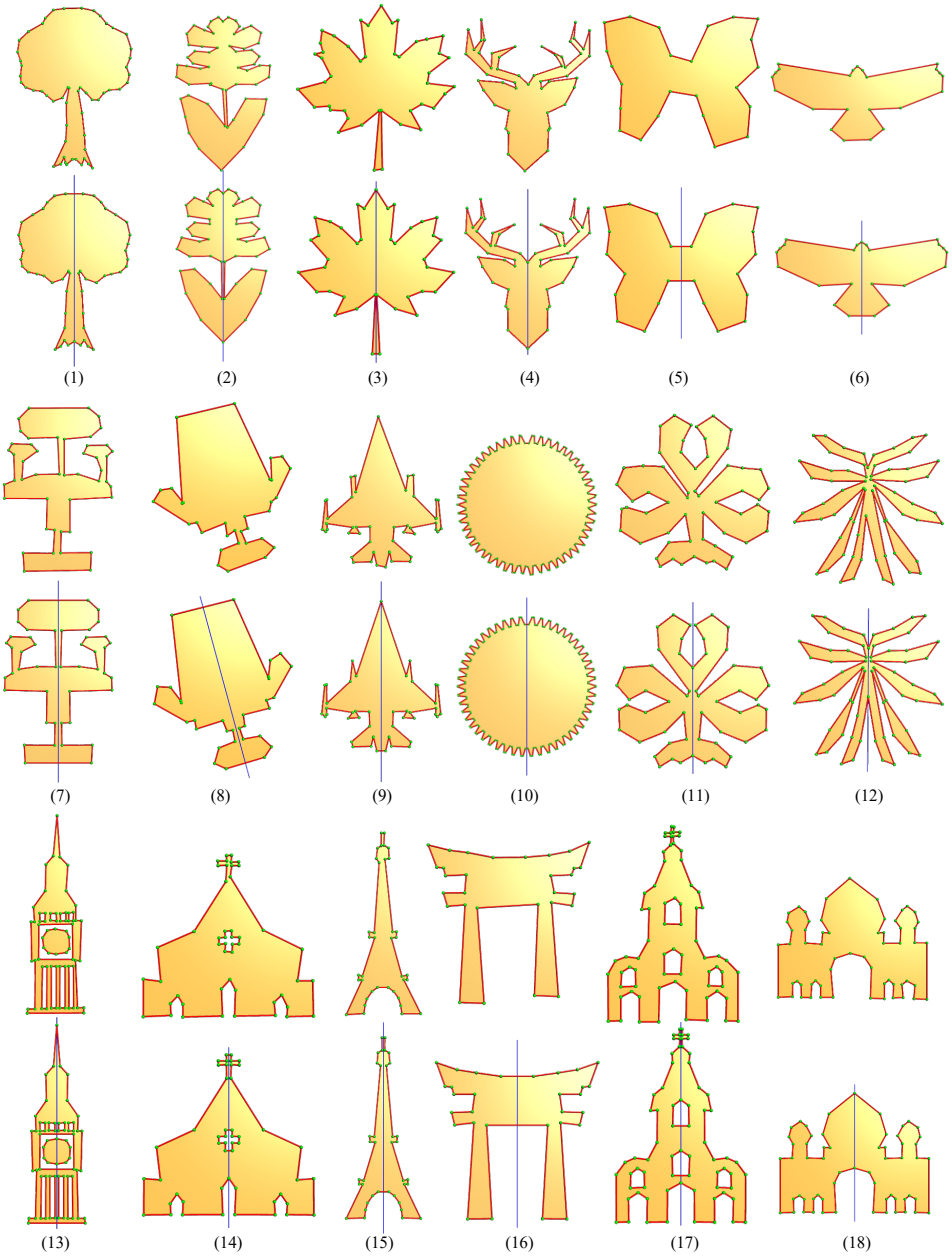



Figure 5.5: The symmetrization results of a set of 2D shapes from three categories: nature (1–6), design (7–12), and architecture (13–18).

### 5.2.2. SYMMETRIZATION OF SHAPES WITH AGNOSTIC AXES

In Sec. 5.2.1, I have described the symmetrization method for shapes whose symmetry axis is aligned with the Y-axis. In common architectural structures, such as windows, the symmetry axis is typically aligned vertically, allowing for straightforward recognition by the human eyes. However, for more complex or irregular shapes, the symmetry axis may be oriented arbitrarily, which significantly complicates the process of identifying symmetry. As a result, the formulation provided in Equation 5.8 is not directly applicable to these cases. In this section, I provide a simple strategy to achieve symmetrization of shapes with arbitrary axes without breaking the formulation given Equation 5.8. My idea is to find the symmetry axis such that, after symmetry optimization, the symmetry axis leads to a symmetrization result with the least deformation to the original shape.

Let $\theta$ denote the angle between the symmetry axis of a shape and its Y-axis, my goal is to detect the optimal symmetry axis (i.e., determine $\theta$) and meanwhile symmetrize the shape. These two subproblems constitute a *chicken-and-egg* problem since detecting the symmetry axis of a shape requires symmetric input and knowing the symmetry axis is also a precondition to optimize the symmetry of the shape. To tackle this problem, I first sample a discrete set of $\Theta = \{\theta_i\}$ uniformly with an interval of $\Delta\theta$. For each $\theta_i$, I rotate the object by the angle of $\theta$ (thus the symmetry axis will align with its Y-axis) and carry out the same symmetrization described in Sec. 5.2.1. In my experiments, I take into consideration the trade-off between accuracy and time complexity and empirically set $\Delta\theta$ to 10°. The residual of the symmetrization for $\theta_i$ is then measured by the overall change to the shape after symmetrization, which is the same as the objective function in Equation 5.8, i.e.,

$$r(\theta_i) = E_d(\theta_i) + \lambda E_t(\theta_i). \tag{5.9}$$

Finally, the optimal symmetrization is the one that yields the minimum residual, and the final symmetry axis is defined by the corresponding value of $\theta_i$, i.e.,

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, r(\theta_i). \tag{5.10}$$

## 5.3. RESULTS AND EVALUATION

I have implemented my method in C++ based on the Easy3D library [128]. The optimization problem given in Equation 5.8 is solved by using the Gurobi solver [65]. All experiments were conducted on a laptop MacBook Pro 2021 with an Apple M1 processor and 32GB RAM. Experiments on a variety of shapes have demonstrated the effectiveness of the proposed method.

### 5.3.1. SYMMETRIZATION RESULTS

I have tested my method on three categories of 2D shapes, namely *nature*, *design*, and *architecture*, as demonstrated in Figure. 5.5. (1)-(6) demonstrate the symmetrization of natural vegetation and animals, including trees, flowers, leaves, deers, butterflies, and eagles. (7)-(12) were originally captured from designs, among which (7)-(8) are commonly used chairs, (9)-(10) mechanical products, and (11)-(12) art logos. (13)-(18) are polygonal representations of line drawings of several well-known buildings, such as the Great Bell (13), the Eiffel Tower (15), and the Taj Mahal (18). From these visual results,

it can be seen that although the input shapes have diverse structures of different styles, my method succeeded in obtaining visually pleasing symmetrization results. It is worth noting that the global symmetry axis of (8) is not aligned with the Y-axis, for which my method determined its optimal symmetry axis and achieved the desired symmetrization of this shape.

My method is also robust in handling shapes with self-intersections. Figure 5.9 shows such an example, from which I can see that though the shape has three pairs of intersecting edges, my method still precisely determined the symmetric edge correspondences and achieved a promising symmetrization result. It is interesting to observe that my method also identified that the horizontal edge in the middle of the shape does not have a symmetric counterpart.

My experiments also show that when the local geometry of a shape is far from being perfectly symmetric, my symmetry optimization process modifies some edges in a way that is equivalent to the edge contraction operation (i.e., merging the two endpoints of the edge) to enforce strict symmetry in the final shape. As shown in Figure 5.6, the two adjacent edges of the black edge find their corresponding symmetric edges that share one common vertex. Therefore, the black edge is contracted, as the coordinates of its two endpoints become identical after symmetrization. This is attributed to the fact that my symmetrization formulation seeks to optimize the symmetry of the shape by introducing minimum deformation into its original geometry.



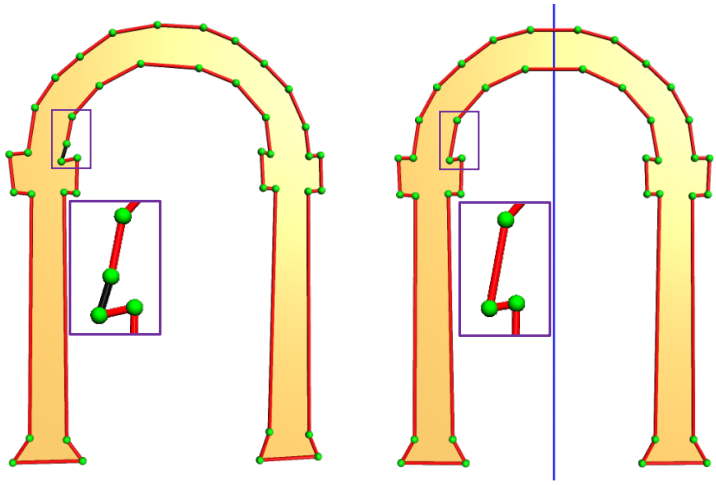
I also conducted tests on 2D shapes with limited openings, as shown in Figure 5.7. As can be observed from the figure, my method produces satisfactory results. While the symmetrization of the closed shape in (a) achieves perfect symmetry, the openings in (b) and (c) are not closed as there are no discernible clues in the shape to recover the geometry.

While there are existing methods proposed in the field of symmetrization, direct comparison with my method is challenging due to differing scenarios. Specifically, my method focuses on extrinsic symmetrization, whereas [202] focus on intrinsic symmetrization. As the authors of [202] have pointed out, the output of their method can serve as a good input for extrinsic symmetrization. Additionally, Mitra et al.'s work [125] is better suited for shapes that are nearly intrinsically symmetric. While these two papers are relevant to my work, the application scenarios are different. Therefore, I believe that my method offers a distinct contribution to the field of shape symmetrization.

### 5.3.2. ROBUSTNESS AND COMPLEXITY ANALYSIS

I have evaluated the robustness of my method by symmetrization of a shape representing a chair with an increasing amount of Gaussian noise. As shown in Figure 5.8, my method produces very promising results even when the noise level is as high as $\delta = 0.35$. When the noise becomes extremely large (e.g., $\delta \geq 0.45$) such that the input shape is completely contaminated, my method still obtains a visually convincing symmetrization result, and the overall structure of the chair has been recovered by my method. In practical applications such as design and digitalization of real-world shapes, the standard deviation of the noise introduced to the shapes is usually small, making my method quite applicable in optimizing or beautifying such shapes.

My method handles shapes represented as general polygons. The symmetry opti-



(**a**) Input (**b**) Symmetrization result

Figure 5.6: An example where my method achieves a strict symmetry by contracting an edge (i.e., merging its two endpoints). The black edge in the close-up view (left) becomes degenerated and is thus removed after optimization.



mization step involves solving a mixed integer quadratic program whose complexity depends on the number of edges in the input shape. To understand the scalability of my method, I have plotted a curve of the running times with respect to the complexity of the input shapes in Figure 5.10. Note that the running time I reported only considers that the symmetry axis is aligned with the Y-axis. The data were collected from the symmetrization of 20 shapes of various styles from different categories shown in Figures 5.5, 5.9, and 5.6. For most of the shapes in Figure 5.5, the input polygons have fewer than 180 edges, and the running times are less than 30 seconds. In Figure 5.5 (10), the gear shape has 192 edges, which led to 1152 candidate symmetric edge pairs (and thus the same number of integer variables). This resulted in a large mixed integer quadratic program, and my method took 56 minutes to solve it.

### 5.3.3. EXTENSION TO PARTIAL SYMMETRIC OBJECTS

My method is designed for 2D shapes with a single global symmetry axis, but it can be extended to shapes with partial symmetry. The basic idea is to reduce the influence of the *tolerance* term in my objective function given in Equation 5.8, which will allow leaving more edges non-symmetric during the optimization. I further studied the impact of the *tolerance* term by gradually changing the weight of this term. The results are reported in Figure 5.11, where the window of the building does not have a symmetric counterpart.

From Figure 5.11, it can be seen that when $\lambda$ has a small value (e.g., $\lambda \leq 20$ in (b) and (c)), most edges are in unpaired status, and only few edges are enforced to deform to be symmetric with their counterparts. By increasing $\lambda$, the optimization step imposes a stronger global symmetry constraint, and consequently, more and more edges are paired, and the overall shape including the middle window is strictly symmetrized. How-

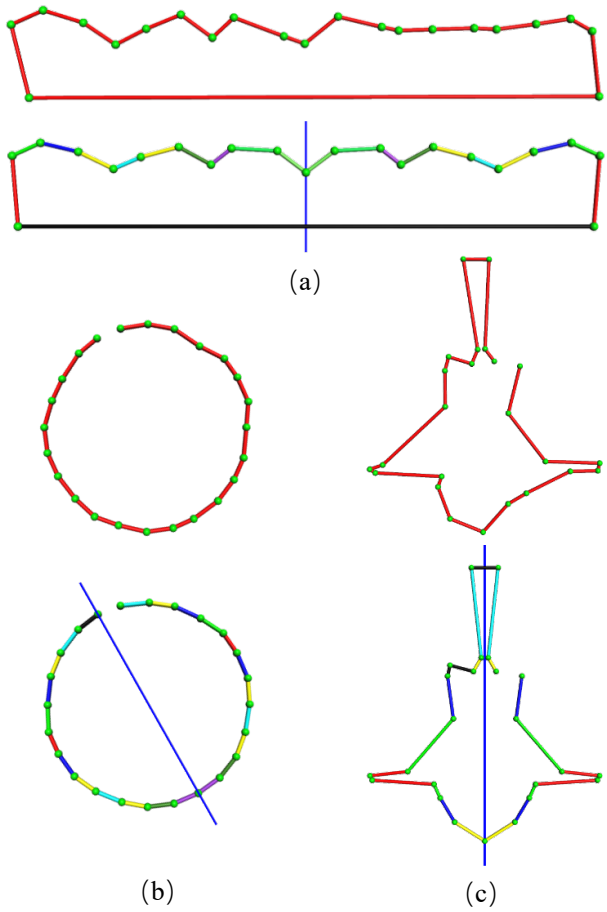
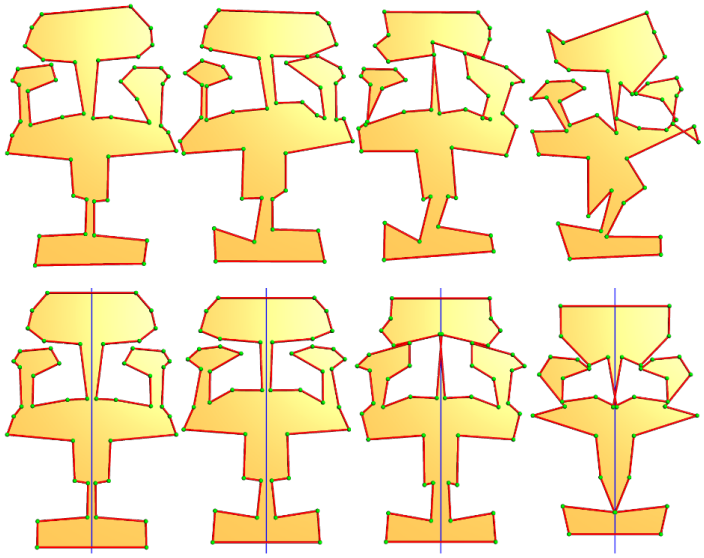



Figure 5.7: Symmetrization of shapes with and without openings. (a) is a closed polygonal shape where all symmetric edge pairs are identified by my method, resulting in perfect symmetry. In (b) and (c), my method identifies all potential symmetric edge pairs except for the openings where the geometry cannot be fully recovered due to the absence of clues. The edges with the same colors denote the identified symmetric edge pairs, while the black edge indicates that it is not symmetric with any other edges.



(a) $\sigma$=0.15 (b) $\sigma$=0.25 (c) $\sigma$=0.35 (d) $\sigma$=0.45

Figure 5.8: Symmetrization results of a shape with an increasing level of noise. The top row shows the input shapes, and the bottom row shows the corresponding symmetrization results. For each noise level, $\sigma$ indicates the standard deviation of Gaussian noise.

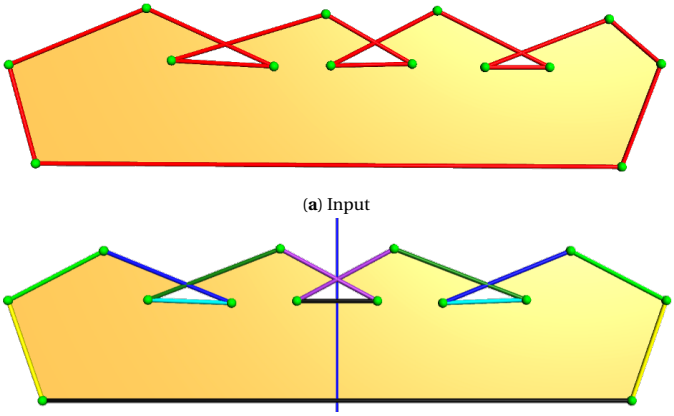


ever, some local regions such as the bottom-left window are left non-symmetric even when the influence of the *tolerance* term is increased to a very high value (i.e., $\lambda = 800$). These properties indicate that the weight of the *tolerance* term provides control over the desired level of non-symmetry, and my method is capable of symmetrizing global shapes while maintaining their non-symmetric local parts.

### 5.3.4. ADDITIONAL VERTICAL AND HORIZONTAL CONSTRAINTS

By incorporating the options of horizontal and vertical constraints into the original equation, I can improve the overall regularity of 2D shapes. This is because orthogonality and parallelism are common characteristics in these shapes, and the addition of these constraints allows for the consideration of these two types of regularities and symmetry in the equation.

Before the hypothesis generation step, I identify potential horizontal and vertical edges in the original shape by measuring the angle between each edge and the $X/Y$ axis. If the angle is less than a threshold of $\gamma$, the edge is added to either the set of vertical edges $R_v$ or the set of horizontal edges $R_h$ depending on its proximity to the two axis. In this work, $\gamma$ is set to 15°, which strikes a balance between capturing edges that are close to horizontal or vertical while allowing for small deviations due to noise or imperfections in the data. This threshold was chosen empirically to account for slight irregularities without misclassifying significantly non-aligned edges. After the symmetrization process, these edges are aligned well with the X/Y axis, achieved by adding a new hard



(**a**) Input

(**b**) Symmetrization result

Figure 5.9: Symmetrization of a shape with self-intersections. In (b), the edges with the same color reveal the symmetric edge pairs identified by my method, and the black edge indicates that it is not symmetric with any other edges.



constraint to equation 5.8, defined as follows,

$$s.t. \begin{cases} x_i^l - x_j^r = 0, & \text{if } e_i \in R_v \\ y_i^t - y^b = 0, & \text{if } e_i \in R_h \end{cases}, \tag{5.11}$$

where $x$ and $y$ with superscripts $t$, $b$, $l$, and $r$ denote the coordinates of the top, bottom, left, and right endpoint of an edge, respectively.

As illustrated in Figure 5.12, the introduction of the new constraints results in more edges in (c) to be either orthogonal or parallel to the symmetry axis, as seen in comparison with the symmetrization result (b), which further improves the overall regularity of the shape. Note that the proposed vertical or horizontal constraints are different from the vertical or horizontal patterns introduced in Section 5.2.1. The distinction lies in the fact that the patterns are defined on the candidate edge pairs, while the constraints are directly defined on the edges.

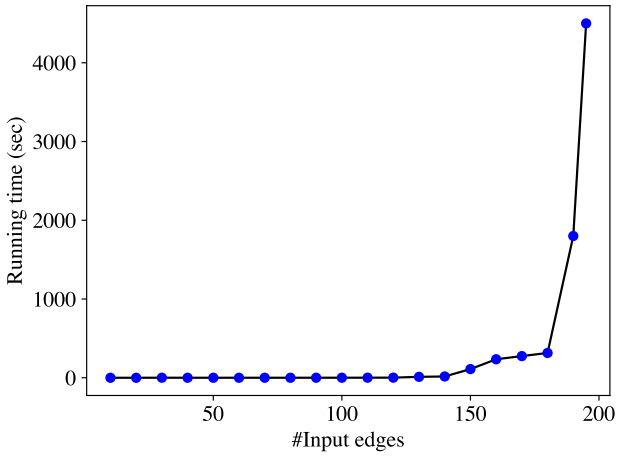



Figure 5.10: The running time (in seconds) of my method with respect to the number of input edges. The statistics are recorded from symmetrization of 20 shapes shown in Figures 5.5, 5.9, and 5.6.



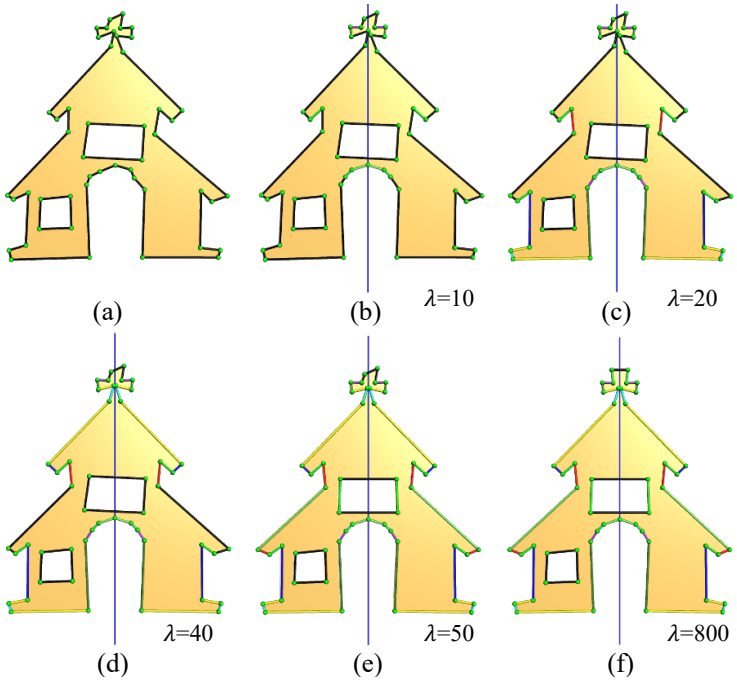
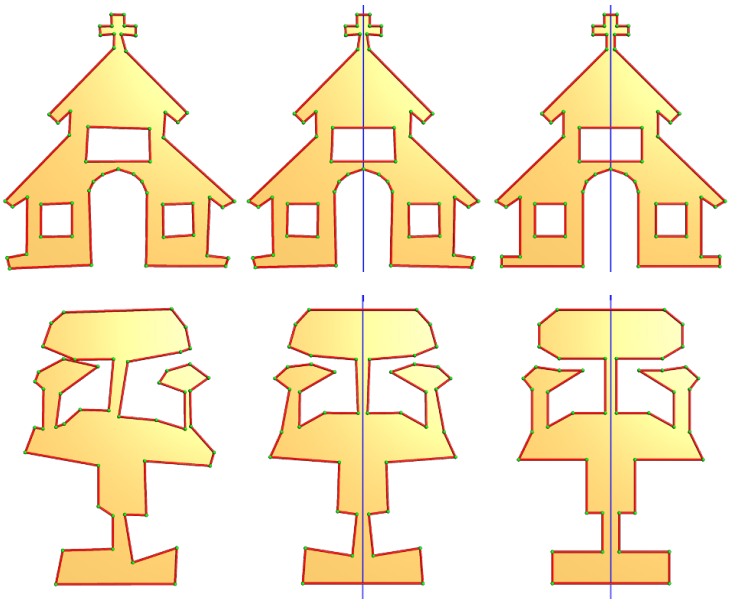
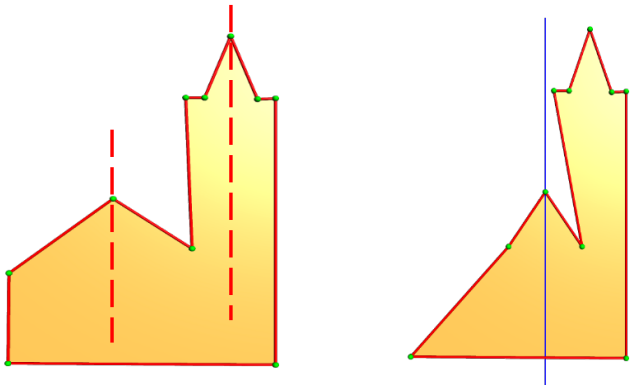



Figure 5.11: The effect of the *tolerance* term. Symmetrization results of a 2D shape (a) by gradually increasing the influence of the *tolerance* term. The value below each subfigure denotes the weight used in the optimization. The edges with the same color (except black indicating the non-symmetric edges) are paired and enforced to be symmetric in the final optimization.



(**a**) (**b**) (**c**)

Figure 5.12: The effect of additional vertical and horizontal constraints, demonstrated on two shapes. (a) Input. (b) Symmetrization result without these constraints. (c) Symmetrization result with these constraints.



(**a**) Input (**b**) Symmetrization result

Figure 5.13: A failure case of my method. This is due to that my method is designed for the symmetrization of shapes with a single global symmetry axis, while the input model has two dominant local ones.

### 5.3.5. LIMITATIONS

The symmetrization framework, though effective for shapes with a single dominant symmetry axis, encounters three key challenges: First, for shapes with multiple local symmetries, my method strives to enhance the most dominant symmetry (see Figure 5.11), and

it may fail to generate reasonable results when the multiple symmetries are comparably dominating the shape. Such a failure example is given in Figure 5.13, where my symmetry optimization identifies only the dominant symmetry. Second, the mixed-integer quadratic programming formulation exhibits exponential time complexity relative to the number of edges, necessitating pre-simplification for large polygons (e.g., polygons with more than 200 edges). Third, while self-intersections were not empirically observed, the theoretical absence of topological safeguards (e.g., intersection detection) introduces a risk of geometric invalidity under edge-case configurations.

## 5.4. CONCLUSION AND FUTURE WORK

I have presented an automatic method for the symmetrization of 2D shapes based on a mixed-integer programming formulation. My formulation aims to jointly identify the optimal symmetry axis (by determining the correct symmetric edge pairs) and optimize the symmetry of the shape simultaneously. Extensive experiments on various types of 2D shapes have demonstrated the effectiveness of the proposed method. In particular, my method is robust to noise, non-symmetric local geometry, and artifacts such as self-intersections.



My current symmetrization framework assumes that a shape has a single global symmetry axis. In the future, I would like to extend my method to handle shapes with multiple symmetry axes by incorporating user interactions. Introducing structural priors and regularities, such as repetition [130], into symmetry optimization is also an interesting direction. Additionally, I believe that extending my framework to handle intrinsic symmetry or 3D representations would be a promising direction for future research.

# 6

# Conclusion and Future Work

*In this section, I bring together the main contributions and key findings of the research discussed in the previous chapters. The research questions proposed in chapter 1 have been successfully answered, accomplishing the objectives of this thesis. Furthermore, I explore potential directions for future research, discussing areas that could benefit from deeper investigation.*

## 6.1. CONTRIBUTIONS AND KEY FINDINGS

Building models are used in various applications, including urban planning, simulation, and automatic navigation. However, research on high levels of detail building reconstruction is still scarce. To address this research gap, this thesis explores structure-aware 3D building reconstruction with the highest possible level of detail utilizing airborne LiDAR point clouds or MVS meshes as input. The main research question is divided into three sub-questions in Chapter 1, which are answered through three research cycles. In subsequent sections, I will elaborate on the core contributions to the research objective and the key findings regarding research questions, based on the relevant literature and the content of each chapter of this thesis.

### I. AUTOMATIC LOD2 BUILDING RECONSTRUCTION

- **Contribution:** Developed an automatic method of LoD2 building reconstruction from LiDAR point clouds.

- **Key findings:**

  **Q:** *How to reconstruct city-scale LoD2 building models using airborne LiDAR point clouds?*

  **A:** The reconstruction of city-scale LoD2 building models from airborne LiDAR point clouds can be effectively achieved through a "divide and conquer" strategy. This involves segmenting building instances from the raw LiDAR data using available footprint data, thereby enabling the reconstruction of LoD2 building models on an individual basis.

  Airborne LiDAR point clouds often exhibit data missing in facade regions due to the limited positioning and movement trajectories of airborne scanners. To address this, the reconstruction process infers the locations of these missing regions by assuming they typically consist of vertical planes. The methodology begins by projecting the entire point cloud of a single building onto the ground plane to create a height map. The edges within this map represent height discontinuities, which correspond to facade planes. Using a Canny edge detector and a refined polyline simplification method, these edges are extracted as polylines, which are subsequently extruded into vertical planes. This approach effectively identifies the missing vertical planes, thereby contributing to a comprehensive representation of the building structure. Following the identification of both wall and roof segments, the reconstruction framework accurately reconstructs complete building models. Given that building models typically consist of planar segments, I approximated them using detected and inferred planes from input data. The planar segments of both roofs and walls are used to hypothesize the faces of the building surface. The final model is obtained by solving an integer programming problem, with geometric constraints ensuring that the final results are manifold and watertight.

### II. SEMI-AUTOMATIC LOD3 BUILDING RECONSTRUCTION

- **Contribution:** Developed a semi-automatic method of LoD3 building reconstruction from MVS meshes.

- **Key findings:**

  **Q:** *How to reconstruct LoD3 building models using MVS meshes, while balancing the amount of user guideline and automated computation?*

  **A:** MVS meshes often exhibit overly smoothed roof superstructures and facade elements, a consequence of the reconstruction process that tends to excessively smooth or alter the original details and intricacies. To address this issue, I propose to incorporate limited user interactions, specifically through delineated strokes, which serve as critical instructional cues. These strokes guide the system in accurately interpreting the spatial disposition and contours of the building's architectural features, thereby aiding in the recovery of these lost details. This step is crucial in achieving a higher level of detail in the final model, particularly in complex architectural features that automated methods may struggle to capture fully.

  Given that human modeling processes often require frequent view switching to check and adjust vertices to match the geometry, my approach leverages advanced pattern recognition algorithms. These algorithms automatically detect and replicate similar architectural elements within the model, significantly reducing the need for extensive user intervention. This not only enhances reconstruction accuracy but also streamlines the modeling process, making it more efficient. Then, the geometry of the input mesh is updated, yielding a compact polygonal mesh that is both efficient and geometrically accurate. The final stage involves further enhancement of the overall layout of the polygonal mesh to produce a visually coherent and aesthetically pleasing LoD3 building model. It ensures that the model not only meets rigorous accuracy standards but also achieves a high level of visual fidelity, making it suitable for practical applications in urban planning, heritage conservation, and other fields that demand high-quality LoD3 representations.

### III. AUTOMATIC SYMMETRIZATION OF 2D POLYGONAL SHAPES

- **Contribution:** Developed an automatic symmetrization method for 2D polygonal shapes generated from facade elements.

- **Key findings:**

  **Q:** *How to recover or enhance the regularity of reconstructed building models?*

  **A:** Given the prevalence of repetitive roof superstructures and facade elements in building models, I project these elements onto facade planes to generate 2D polygonal shapes for symmetrization. By subsequently reprojecting the symmetrized 2D shapes back into the 3D space, the regularity of reconstructed building models can be enhanced.

Detecting symmetry elements and achieving symmetrization of 2D polygonal shapes is inherently a chicken-and-egg problem: accurate symmetry detection guides modifications to enhance symmetry, and as shapes become more symmetric, detecting symmetry elements becomes easier. To address this issue, I propose a novel hypothesis-and-selection-based framework. Unlike existing methods that follow a detect-then-symmetrize pipeline, I integrate these two sub-problems and formulate them as a mixed integer programming problem. First, I generate a set of hypotheses for candidate symmetric elements from input data. Two geometric tests are then conducted to prune the number of candidate symmetric elements, thereby reducing the complexity of the subsequent optimization problem. Next, I solve a mixed integer programming problem to achieve optimal symmetry detection and symmetrization. The objective function penalizes overall deformation while hard constraints enforce strict symmetry of the final shape. Finally, the 3D building models are regularized by reprojecting the symmetrized 2D shapes into the original space.

### IV. OVERALL DISCUSSION



The three research cycles in this thesis are designed as a progressive and interdependent framework, addressing the challenges of structure-aware 3D building reconstruction. The first research cycle establishes a foundational methodology for large-scale LoD2 reconstruction by leveraging airborne LiDAR data, demonstrating a reasonable geometric inference can overcome inherent data limitations. Building on this foundation, the second cycle refines the reconstruction process by integrating semi-automatic workflows with MVS meshes. It ensures that fine architecture details essential for LoD3 models are captured with a balance of automation and user input. The third cycle further enhances the accuracy of previous LoD2/LoD3 building reconstruction by introducing an optimization-based symmetrization framework, which systematically restores geometric regularity and reinforces architectural coherence. Together, these cycles form an integrated pipeline where each stage strengthens the next, contributing to a more robust and accurate building reconstruction process. By systematically addressing the data limitations, the interplay between automation and human input, and geometric regularity, this thesis advances the field of 3D reconstruction. The findings also underscore the importance of hybrid workflows (e.g., combining automatic algorithms with user guidance) in tackling complex real-world reconstruction tasks. This thesis not only answers the sub-questions but also provides a validated roadmap for future research in balancing efficiency, accuracy, and usability in 3D building modeling.

## 6.2. FUTURE WORK

This thesis represents an initial advancement in the area of reconstructing 3D urban building models, although on a small scale. However, it is important to recognize that my methods still have a few limitations, which collectively define the boundary conditions of the proposed methodologies, highlighting the inherent trade-offs between geometric fidelity, computational efficiency, and architectural generality. While the proposed ap-

proaches offer significant advancements in structure-aware 3D reconstruction, certain challenges persist. Future work will focus on three key areas: (1) integrating non-planar primitives and gap-filling algorithms to enhance topological robustness; (2) developing adaptive symmetry detection heuristics capable of handling multi-axis configurations; and (3) refining algorithms to improve scalability for large-scale or highly complex structures. By reflecting on these limitations, this thesis lays the groundwork for future advancements and identifies key areas for further research, which will be discussed in the following:

- **Extension to more complex primitives**

  At the beginning of this thesis, I limit the scope to approximate building models using planar primitives. However, it is essential to acknowledge that real-world buildings display greater complexity and variability. Consequently, a natural and necessary extension of this work involves incorporating a broader range of geometric primitives. By integrating spheres, cylinders, cones, tori, and even B-spline surfaces into the repertoire of primitive types, the modeling approach can be significantly enriched. This extension will not only enhance the overall richness of the models but also improve the accuracy of the reconstructed 3D building models. The primary challenge associated with this extension is the development of a unified framework capable of managing and intersecting these diverse primitive types to generate valid candidate faces. To address this, it is crucial to design robust algorithms that can seamlessly handle the intersections and unions of these different geometries. Additionally, my exhaustive intersection strategy to generate hypotheses significantly increases the number of candidate faces. Consequently, it is necessary to design an effective pruning strategy to reduce the number of candidate faces without sacrificing accuracy. This pruning mechanism mitigates computational complexity and reduces overall running time. Such enhancements are crucial to make the approach more efficient and feasible for practical applications, thereby optimizing performance and ensuring the scalability of the method to larger and more complex data.

- **Symmetry-based 3D building reconstruction**

  Symmetry is a fundamental characteristic of building models. This thesis is centered on the automatic symmetrization of 2D polygonal shapes, which are derived from the roof superstructures or facade elements of building models. The principal challenge in symmetrizing 3D shapes lies in the dual task of identifying optimal symmetric planes and achieving symmetrization, a process that can be likened to a chicken-and-egg problem. In 2D cases, I formulate a mixed-integer programming problem, which allows for the simultaneous resolution of the two interdependent tasks. The transition from 2D to 3D symmetrization significantly increases the complexity due to the additional dimension involved in 3D buildings. A promising direction involves directly utilizing symmetry as a prior to complete point clouds, which are then used in building reconstruction. Airborne laser scanning data often exhibit regions with missing information. Integrating a symmetry-based completion module, as demonstrated in previous research [160], enhances

the quality of airborne point clouds. The completed data can subsequently serve as input for the reconstruction task, thereby improving the overall accuracy and reliability of the reconstructed building models.

- **Integration of traditional geometric processing and deep learning methods**

  There are both strengths and weaknesses of traditional geometric optimization and deep learning methods. For example, traditional geometric optimization methods often employ regularization techniques or constraints to ensure that the resulting models adhere to geometric principles and minimize errors. However, these methods are sensitive to parameter settings. It is difficult to find a set of parameters that are suitable for all input data. Deep learning techniques, on the other hand, have the ability to automatically learn underlying features from raw input data. They excel at extracting complex patterns and hierarchical structures, enabling the capture of intricate building details. However, achieving good performance with deep learning typically requires a large amount of labeled training data. Acquiring and annotating such datasets for 3D building reconstruction can be time-consuming and costly. It is worth noting that designing a fully end-to-end deep learning network that is universally applicable to all geometric problems is quite challenging. Therefore, a more promising direction for future research lies in efficiently combining traditional geometric processing with deep learning. By selectively choosing the most suitable method at each step, I can obtain more favorable results.

- **A better 3D representation for urban modeling**

  The primary goal of this research is to introduce a method for reconstructing compact and accurate 3D models for real-life buildings. My approach utilizes 3D polygonal mesh models as the chosen representation. This choice offers several advantages over dense meshes or point clouds, as it significantly reduces storage requirements while preserving the sharp features inherent in building models. Nevertheless, recent advancements, particularly the emergence of Neural Radiance Fields (NeRF) [122], have shown considerable promise in overperforming point cloud, mesh, and volume models. NeRF models excel in generating realistic and high-quality renderings of complex scenes, successfully capturing intricate lighting effects and fine details. Recently, Google Maps started to utilize NeRF to render 3D open street maps, further highlighting its potential for capturing fine details, thus enhancing the overall immersive experience. Therefore, it is important to explore the broader utility of NeRF in diverse geometric modeling applications. This exploration may advance the development of building modeling. Recently, 3D Gaussian Splatting (GS) [92] has been proposed and holds significant potential in city modeling due to its ability to efficiently represent complex urban environments with high fidelity. By leveraging Gaussian functions to interpolate data points, such as building heights or density information, onto a 3D grid, it is able to create smooth and realistic representations of cities [182]. However, the disordered nature [37] of 3D GS makes geometric modeling very challenging. Despite the ability of 3D GS to produce highly realistic renderings, modeling internal structures of

objects within the GS framework remains unanswered and deserves further exploration. Investigating the application of 3D GS in geomatics tasks presents an intriguing avenue for research. This exploration holds the potential to enhance the precision and efficiency of spatial data analysis and management within the field.

- **Combination of multi-source data**

  In this thesis, I independently utilize airborne LiDAR point clouds and MVS meshes for LoD2 and LoD3 building reconstruction. A promising research direction involves designing methods that effectively integrate these two distinct types of data. These approaches must ensure that information from each source is appropriately combined to enhance the accuracy and completeness of the reconstructed building models. It is essential to consider that data from different sources, such as LiDAR and photogrammetry, may exhibit variations in noise levels, density, and scale [107]. Therefore, to ensure the seamless integration of multi-source data while preserving geometric and semantic details, developing robust data preprocessing techniques for registration is crucial. Additionally, incorporating street-view images and 2D footprint data contributes to reconstructing highly detailed building models. Designing a comprehensive building reconstruction framework that effectively combines these diverse data types is of significant value [90].

- **Large-scale LoD3 building dataset**

  Currently, learning-based approaches for building reconstruction are limited by the scarcity of comprehensive datasets specifically targeting LoD3 representations. Existing building datasets primarily emphasize lower Levels of Detail (LoD), ranging from LoD0 to LoD2. In Chapter 3, I developed a new LoD2 building dataset, consisting of point clouds and reconstructed surface models of 20k real-world buildings. To advance detailed building reconstruction and enable broader applications within the geomatics community, it is imperative to develop a feature-rich, large-scale LoD3 building dataset. Although my thesis has primarily focused on LoD3 building reconstruction methods for individual buildings, there is potential to scale these methods to larger urban areas by leveraging more advanced computational techniques. For example, after the user delineates a window in several sketches, the algorithm can efficiently identify all windows exhibiting the same pattern throughout the large-scale urban scene. A promising avenue for future research involves integrating multiple data sources [67], such as airborne laser scanning, mobile laser scanning, and street-view images, to generate detailed LoD3 building models. Recently, generative methods have gained significant attention in the field of computer graphics and remote sensing. For example, Siddiqui et al. [156] introduced MeshGPT, a novel method for generating compact triangle meshes by leveraging recent advancements in large language models. This method employs a sequence-based approach in which a transformer model is trained to predict subsequent triangles in a mesh, thereby facilitating the creation of efficient and sharp-edged triangulations. Given the availability of MeshGPT, which has been trained on an extensive dataset of general 3D shapes, it is feasible

to fine-tune this model using a smaller, detailed building dataset. This process can enable the generation of a substantial LoD3 building dataset.

# ACKNOWLEDGEMENTS

It is such a long and beautiful journey, and finally, it's time to say goodbye to everyone! The 5 years of doing a Ph.D. in the Netherlands are really amazing and unbelievable. I could never forget it. It is my first time being abroad and living alone for such a long time. Actually, I had many hard times. I suffered a lot during the COVID-19 pandemic, working from home for nearly two years. After contracting the virus, I had to quarantine at home. On top of that, negative and harsh comments from paper reviewer 2 upset me a lot, making things even more challenging. Despite these hard moments, there are also lots of joyful experiences. Celebrating my birthday with my colleagues from different countries in spring. Visiting Disneyland Park in France in summer. Exploring various Dutch museums in autumn. Having a Chinese hotpot with my friends in winter. It is too difficult to enumerate all these moments.

Firstly, I would like to express my sincere gratitude to my promotor Prof. dr. Jantien Stoter. Her exceptional qualities as a patient, rigorous, innovative, and enthusiastic researcher have greatly influenced my research. Her ability to provide valuable high-level suggestions, without getting stuck in technical details, is a testament to her invaluable expertise. This skill, undoubtedly, holds significant importance. Of course, I would express my sincere appreciation to my daily supervisor, Dr. Liangliang Nan. He has guided me in numerous areas, including research and personal development. He is not only a supervisor but also a true friend. As a dedicated computer scientist, he devotes his full energy to advancing his field. He upholds a rigorous commitment to research, from which I have greatly benefited. From my point of view, Dr. Nan is one of the best teachers I have encountered. I am immensely grateful for their countless insightful suggestions concerning my Ph.D. research as well as my next steps.

I would like to thank the colleagues of 3D Geoinformation group at TU Delft. Nail is a remarkably enthusiastic researcher whose confidence in research has always been an encouragement to me. Zexin provided invaluable assistance during my challenging times and offered numerous insightful suggestions for my presentations. Weixiao demonstrated exceptional creativity as a researcher, presenting novel ideas that inspired me a lot. Shenglan has brought me much joy, especially her astrological predictions for me. Ravi offered numerous valuable suggestions regarding LoD2 building reconstruction. Anna, our former office manager, organized a lot of interesting group activities. Nadine, our next-generation office manager, organized many 3D sports for us. Also thanks to other colleagues, Stelios, Xiaoxin, Zhaiyu, Camilo, Hugo, Clara, Ivan, Giorgio, Lucas, Akshay, Gina, Daan, Maarten, Miguel, Jasper, Daniele, and Amir. Additionally, I would like to express my appreciation to the secretaries of Urbanism, Margo, Danielle, Karin, and Astrid, who have shown great enthusiasm in assisting me whenever needed.

I want to express my appreciation to Dr. Wufan Zhao from Hong Kong University of Science and Technology (Guangzhou). He offered me a lot of information about working in industry and academia, which helped me a lot in deciding my next steps. I also want

to thank my friend Dr. Honghua Chen and Dr. Yan Wang, who gave me a lot of guidance about finding academic and industrial positions.

I want to thank China Scholar Council for funding my Ph.D. research. Without the scholarship from my home country, I cannot afford the cost of living in the Netherlands to complete my Ph.D. study. I would like to thank my master's supervisors, Prof. dr. Jun Wang and Prof. dr. Mingqiang Wei from Nanjing University of Aeronautics and Astronautics. They brought me to the world of research, and assisted me a lot in applying for this Ph.D. position and the scholarship. Many thanks to the Dutch government and TU Delft, for providing three-month funding to support the contract extension due to the coronavirus pandemic. I would like to thank all the reviewers of my papers, although sometimes I received negative comments from them, in return, I have learned a lot.

I would like to extend my heartfelt gratitude to my fiancée, Xiannan Liu, for her unwavering support and for being a courageous, gentle, empathetic, and passionate partner in life. Reflecting on the entirety of our relationship, I am struck by the challenges we faced together. During most of my Ph.D., we were separated by different time zones, often unable to be physically together like other couples. Nevertheless, she chose to stand by my side. I am grateful for her presence in my life.

Lastly, I would like to express sincere gratitude to my parents, Mr. Donglin Huang and Mrs. Junping Ouyang, and my sister Miss. Mengyu Huang. There is an old Chinese saying, "The other guys only care about how far you fly, but family members care about whether you are tired from flying". Every time I was upset and depressed, they came to me and encouraged me to pursue my dream. My father, a very traditional Chinese man, may never explicitly express his love to me, but I can feel the depth of his affection. My mother, on the other hand, constantly engaged in conversations with me, frequently checking up on me and worrying about my well-being, just like any mother would. Despite being much older than my younger sister, I can also sense her immense value and regard for me. I could not achieve my goals without my family by my side.

*Jin*
*September 2024*

# APPENDIX

## INTEGER PROGRAMMING

Suppose that I have a linear program, then it can be formulated as follows,

$$\begin{aligned} &\text{Maximize} && c^T x \\ &\text{subject to} && Ax \leq b, \end{aligned} \tag{6.1}$$

where $c \in R^n$, $b \in R^m$ are vectors and $A \in R^{m \times n}$ is a matrix. In contrast to linear programming, where variables can take continuous values, integer programming restricts the variables to integers [181], making it useful for modeling discrete decision-making processes.

### TYPE OF INTEGER LINEAR PROGRAMMING

The general form of an integer programming problem is to maximize or minimize an objective function meanwhile satisfying a set of linear constraints. The objective function and constraints involve linear relationships between the decision variables. The integer constraint imposes that the variables must take on integer values.

The basic one is called integer linear programming (ILP), where both the objective function and constraints are linear. A typical ILP problem can be formulated as follows,

$$\begin{aligned} &\text{Maximize} && c^T x \\ &\text{subject to} && Ax \leq b, \\ & && x \in \mathbb{Z}^n, \end{aligned} \tag{6.2}$$

where $c \in R^n$, $b \in R^m$ are vectors and $A \in R^{m \times n}$ is a matrix. A special case is that all variables are restricted to be 0 or 1, then it is binary integer programming (BIP). It is formulated as,

$$\begin{aligned} &\text{Maximize} && c^T x \\ &\text{subject to} && Ax \leq b, \\ & && x_i \in \{0, 1\}. \end{aligned} \tag{6.3}$$

In Chapter 3, I formulate LoD2 building reconstruction from airborne LiDAR point clouds as a binary integer programming problem.

Another type is called mixed integer programming (MIP). Some variables are restricted to be integers, while others are continuous values. This adds an additional level of complexity to the problem, as it involves both integer and continuous decision vari-

ables, comprising a combinatorial problem. It is formulated as,

$$\begin{aligned} \text{Maximize} \quad & c^T x + d^T y \\ \text{subject to} \quad & Ax + By \leq b, \\ & x \in \mathbb{Z}^n, \\ & y \in R^p, \end{aligned} \tag{6.4}$$

where $c \in R^n$, $d \in R^p$, $b \in R^m$ are vectors and $A \in R^{m \times n}$, $B \in R^{m \times p}$ is a matrix.

In Chapter 4, I formulate the symmetrization of 2D polygonal shapes as a mixed-integer programming problem.

### SOLUTION OF INTEGER PROGRAMMING

IP problems are generally more challenging to solve than pure linear programming problems due to the discrete nature of variables, which makes the feasible solution space more complex. There are several methods to solve IP problems:

- **Branch and Bound (B&B)**

  At each step, the problem is divided into two or more subproblems by fixing some integer variables [97]. The subproblems are solved recursively, and the process of division continues until an optimal solution is found or proven to be impossible.

- **Branch and Cut**

  As an extension of the branch and bound method, branch and cut incorporates cutting planes to tighten the relaxation of the problem at each node in the branch-and-bound tree [123]. It converges faster than the branch and bound method.

- **Linear Programming Relaxation**

  The first step in solving an IP problem is to solve its linear programming (LP) relaxation, where the integer constraints are relaxed to allow fractional values [86, 179]. The optimal LP solution provides an upper bound on the IP solution.

- **Cutting Plane Methods**

  Cutting planes are linear inequalities that help improve the relaxation [120] of the linear programming (LP) relaxation of the IP problem. By iteratively adding cutting planes that eliminate fractional solutions [102], the solution space is refined, eventually leading to an integer solution.

- **Heuristic Methods**

  These methods provide approximate solutions to IP problems quickly, without guaranteeing optimality. Examples include greedy algorithms [94, 181], rounding procedures [3], and constructive algorithms [57].

- **Metaheuristic Methods**

  Metaheuristic techniques like Genetic Algorithms [71], Simulated Annealing [170, 24], Tabu Search [62], and Particle Swarm Optimization [91] can be adapted to solve IP problems. These methods explore the solution space in a more flexible and exploratory manner, which can be beneficial for complex problems.

In this thesis, I mainly focus on formulating the building reconstruction or symmetrization as a linear/mixed integer programming problem. I do not propose any new mathematical methods to solve these problems. The off-the-shelf solvers such as SCIP [2] or Gurobi [65] are employed to resolve the formulated optimization problem.

# BIBLIOGRAPHY


[1] *3D BAG (v21.09.8)*. https://3dbag.nl/en/viewer. 2021.

[2] Tobias Achterberg. "SCIP: solving constraint integer programs". In: *Mathematical Programming Computation* 1 (2009), pp. 1–41.

[3] Tobias Achterberg, Timo Berthold, and Gregor Hendel. "Rounding and propagation heuristics for mixed integer programming". In: *Operations Research Proceedings 2011: Selected Papers of the International Conference on Operations Research (OR 2011), August 30-September 2, 2011, Zurich, Switzerland*. Springer. 2012, pp. 71–76.

[4] A Adam et al. "H-RANSAC: A hybrid point cloud segmentation combining 2D and 3D data". In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 4 (2018), pp. 1–8.

[5] AHN3. *Actueel Hoogtebestand Nederland (AHN)*. https://www.pdok.nl/ nl/ahn3-downloads. 2018.

[6] Abdullah Emin Akay et al. "Using LiDAR technology in forestry activities". In: *Environmental monitoring and assessment* 151 (2009), pp. 117–125.

[7] Perpetual Hope Akwensi, Akshay Bharadwaj, and Ruisheng Wang. "APC2Mesh: Bridging the gap from occluded building façades to full 3D models". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 211 (2024), pp. 438–451.

[8] Bastian Albers, Martin Kada, and Andreas Wichmann. "AUTOMATIC EXTRACTION AND REGULARIZATION OF BUILDING OUTLINES FROM AIRBORNE LIDAR POINT CLOUDS." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 41 (2016).

[9] M Alicandro and M Rotilio. "UAV photogrammetry for resilience management in reconstruction plan of urban historical centres after seismic events. A case study". In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 42 (2019), pp. 55–61.

[10] C Altuntas. "Triangulation and time-of-flight based 3D digitisation techniques of cultural heritage structures". In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 43 (2021), pp. 825–830.

[11] Murat Arikan et al. "O-snap: Optimization-based snapping for modeling architecture". In: *ACM Transactions on Graphics (TOG)* 32.1 (2013), pp. 1–15.

[12] Godfried Augenbroe. "Trends in building simulation". In: *Advanced building simulation*. Routledge, 2004, pp. 18–38.

[13] BAG. *Basisregistratie Adressen en Gebouwen (BAG)*. https://bag.basisregistraties.overheid.nl/datamodel. 2019.

[14] Jean-Philippe Bauchet and Florent Lafarge. “City reconstruction from airborne LiDAR: a computational geometry approach”. In: *3D GeoInfo 2019-14thConference 3D GeoInfo*. 2019.

[15] Jean-Philippe Bauchet and Florent Lafarge. “Kinetic shape reconstruction”. In: *ACM Transactions on Graphics (TOG)* 39.5 (2020), pp. 1–14.

[16] Alexander Berner et al. “A graph-based approach to symmetry detection.” In: *VG/PBG@ SIGGRAPH*. 2008, pp. 1–8.

[17] Josef Bigün. “Pattern recognition in images by symmetries and coordinate transformations”. In: *Computer Vision and Image Understanding* 68.3 (1997), pp. 290–307.

[18] Filip Biljecki, Hugo Ledoux, and Jantien Stoter. “An improved LOD specification for 3D building models”. In: *Computers, Environment and Urban Systems* 59 (2016), pp. 25–37.

[19] Filip Biljecki et al. “Applications of 3D city models: State of the art review”. In: *ISPRS International Journal of Geo-Information* 4.4 (2015), pp. 2842–2889.

[20] Filip Biljecki et al. “The variants of an LOD of a 3D building model and their influence on spatial analyses”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 116 (2016), pp. 42–54.

[21] Wolfgang Böhler and Guido Heinz. “Documentation, surveying, photogrammetry”. In: *XVII CIPA Symposium. Recife, Olinda. Brazil*. Vol. 1. 1999.

[22] M. Bokeloh et al. “Symmetry Detection Using Feature Lines”. In: *Computer Graphics Forum* 28.2 (2009), pp. 697–706. ISSN: 1467-8659.

[23] Dorit Borrmann et al. “The 3d hough transform for plane detection in point clouds: A review and a new accumulator design”. In: *3D Research* 2.2 (2011), pp. 1–13.

[24] Kevin Boston and Pete Bettinger. “An analysis of Monte Carlo integer programming, simulated annealing, and tabu search heuristics for solving spatial harvest scheduling problems”. In: *Forest science* 45.2 (1999), pp. 292–301.

[25] Vasileios Bouzas, Hugo Ledoux, and Liangliang Nan. “Structure-aware Building Mesh Polygonization”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 167 (2020), pp. 432–442.

[26] Vasileios Bouzas, Hugo Ledoux, and Liangliang Nan. “Structure-aware Building Mesh Polygonization”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 167 (2020), pp. 432–442.

[27] Gary Bradski. “The openCV library.” In: *Dr. Dobb’s Journal: Software Tools for the Professional Programmer* 25.11 (2000), pp. 120–123.

[28] Mathieu Brédif et al. “3D building reconstruction with parametric roof superstructures”. In: *2007 IEEE International Conference on Image Processing*. Vol. 2. IEEE. 2007, pp. II–537.

[29] Mehmet Buyukdemircioglu, Sultan Kocaman, and Umit Isikdag. “Semi-automatic 3D city model generation from large-format aerial images”. In: *ISPRS International Journal of Geo-Information* 7.9 (2018), p. 339.

[30] David Cailliere et al. “3D mirror symmetry detection using Hough transform”. In: *2008 15th IEEE International Conference on Image Processing*. IEEE. 2008, pp. 1772–1775.

[31] Gülcan Can et al. “Semantic segmentation on Swiss3DCities: A benchmark study on aerial photogrammetric 3D pointcloud dataset”. In: *Pattern Recognition Letters* 150 (2021), pp. 108–114.

[32] John Canny. “A computational approach to edge detection”. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), pp. 679–698.

[33] Alessandra Capolupo et al. “Photogrammetry for environmental monitoring: The use of drones and hydrological models for detection of soil contaminated by copper”. In: *Science of the Total Environment* 514 (2015), pp. 298–306.

[34] Cindy Cappelle et al. “Virtual 3D city model for navigation in urban areas”. In: *Journal of Intelligent & Robotic Systems* 66.3 (2012), pp. 377–399.

[35] Anne-Laure Chauve, Patrick Labatut, and Jean-Philippe Pons. “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*. IEEE. 2010, pp. 1261–1268.

[36] Dong Chen, Ruisheng Wang, and Jiju Peethambaran. “Topologically aware building rooftop reconstruction from airborne laser scanning point clouds”. In: *IEEE Transactions on Geoscience and Remote Sensing* 55.12 (2017), pp. 7032–7052.

[37] Guikun Chen and Wenguan Wang. “A survey on 3d gaussian splatting”. In: *arXiv preprint arXiv:2401.03890* (2024).

[38] Jiazhou Chen et al. “3D Instance Segmentation of MVS Buildings”. In: *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022), pp. 1–14.

[39] Lei Chen et al. “Building detection in an urban area using lidar data and QuickBird imagery”. In: *International Journal of Remote Sensing* 33.16 (2012), pp. 5135–5148.

[40] Ran Chen. “The development of 3D city model and its applications in urban planning”. In: *2011 19th International Conference on Geoinformatics*. IEEE. 2011, pp. 1–5.

[41] Zhaiyu Chen et al. “PolyGNN: Polyhedron-based Graph Neural Network for 3D Building Reconstruction from Point Clouds”. In: *arXiv preprint arXiv:2307.08636* (2023).

[42] Zhaiyu Chen et al. “Reconstructing compact building models from point clouds using deep implicit fields”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 194 (2022), pp. 58–73.

[43] Il Choi and Sung-Il Chien. “A generalized symmetry transform with selective attention capability for specific corner angles”. In: *IEEE Signal Processing Letters* 11.2 (2004), pp. 255–257.

[44] Daniel Cohen-Or and Yishay Levanoni. *Temporal continuity of levels of detail in delaunay triangulated terrain*. IEEE, 1996.

[45] David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. “Variational shape approximation”. In: *ACM SIGGRAPH 2004 Papers*. 2004, pp. 905–914.

[46] Benoıt Combès et al. “Automatic symmetry plane estimation of bilateral objects in point clouds”. In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2008, pp. 1–8.

[47] Jonathan Crespo et al. “Semantic information for robot navigation: A survey”. In: *Applied Sciences* 10.2 (2020), p. 497.

[48] Fernando De Goes et al. “An optimal transport approach to robust reconstruction and simplification of 2d shapes”. In: *Computer Graphics Forum*. Vol. 30. 5. Wiley Online Library. 2011, pp. 1593–1602.

[49] Andrew Delong et al. “Fast approximate energy minimization with label costs”. In: *International journal of computer vision* 96 (2012), pp. 1–27.

[50] Zhen Dong et al. “An efficient global energy optimization approach for robust 3D plane segmentation of point clouds”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 137 (2018), pp. 112–133.

[51] David H Douglas and Thomas K Peucker. “Algorithms for the reduction of the number of points required to represent a digitized line or its caricature”. In: *Cartographica: the international journal for geographic information and geovisualization* 10.2 (1973), pp. 112–122.

[52] Aleksandrs Ecins, Cornelia Fermuller, and Yiannis Aloimonos. “Detecting reflectional symmetries in 3D data through symmetrical fitting”. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 1779–1783.

[53] R El Meouche et al. “UAV photogrammetry implementation to enhance land surveying, comparisons and possibilities”. In: *The international archives of the photogrammetry, remote sensing and spatial information sciences* 42 (2016), pp. 107–114.

[54] Hao Fang and Florent Lafarge. “Connect-and-Slice: an hybrid approach for reconstructing 3D objects”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 13490–13498.

[55] Martin A Fischler and Robert C Bolles. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”. In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[56] Jakub Fišer, Paul Asente, and Daniel Sỳkora. “ShipShape: a drawing beautification assistant”. In: *Proceedings of the workshop on Sketch-Based Interfaces and Modeling*. 2015, pp. 49–57.

[57] Marshall L Fisher and Jeremy F Shapiro. “Constructive duality in integer programming”. In: *SIAM Journal on Applied Mathematics* 27.1 (1974), pp. 31–52.

[58] Ran Gal and Daniel Cohen-Or. “Salient geometric features for partial shape matching and similarity”. In: *ACM Transactions on Graphics (TOG)* 25.1 (2006), pp. 130–150.

[59] Lin Gao et al. “PRS-Net: Planar reflective symmetry detection net for 3D models”. In: *IEEE Transactions on Visualization and Computer Graphics* 27.6 (2020), pp. 3007–3018.

[60] C García-Sánchez, DA Philips, and C Gorlé. “Quantifying inflow uncertainties for CFD simulations of the flow in downtown Oklahoma City”. In: *Building and environment* 78 (2014), pp. 118–129.

[61] Michael Garland and Paul S Heckbert. “Surface simplification using quadric error metrics”. In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 1997, pp. 209–216.

[62] Fred Glover and Manuel Laguna. *Tabu search*. Springer, 1998.

[63] J A Goncalves and Renato Henriques. “UAV photogrammetry for topographic monitoring of coastal areas”. In: *ISPRS journal of Photogrammetry and Remote Sensing* 104 (2015), pp. 101–111.

[64] Natasha Govender. “Evaluation of feature detection algorithms for structure from motion”. In: (2009).

[65] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2022. URL: https://www.gurobi.com.

[66] Norbert Haala, Mathias Rothermel, and Stefan Cavegn. “Extracting 3D urban models from oblique aerial images”. In: *2015 Joint Urban Remote Sensing Event (JURSE)*. IEEE. 2015, pp. 1–4.

[67] Xu Han et al. “WHU-Urban3D: An urban scene LiDAR point cloud dataset for semantic instance segmentation”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 209 (2024), pp. 500–513. ISSN: 0924-2716.

[68] Robert M Haralick. “Determining camera parameters from the perspective projection of a rectangle”. In: *Pattern Recognition* 22.3 (1989), pp. 225–230.

[69] Jan-Henrik Haunert. “A symmetry detector for map generalization and urban-space analysis”. In: *ISPRS journal of photogrammetry and remote sensing* 74 (2012), pp. 66–77.

[70] Yuxiang He, C Zhang, and Clive S Fraser. “An energy minimization approach to automated extraction of regular building footprints from airborne LiDAR data”. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2.3 (2014), p. 65.

[71] John H Holland. “Genetic algorithms”. In: *Scientific american* 267.1 (1992), pp. 66–73.

[72] E Honkavaara et al. “Autonomous hyperspectral UAS photogrammetry for environmental monitoring applications”. In: *ISPRS Archives*. International Society for Photogrammetry and Remote Sensing (ISPRS). 2014.

[73] Lukáš Hruda, Ivana Kolingerová, and Libor Váša. “Robust, fast and flexible symmetry plane detection based on differentiable symmetry measure”. In: *The Visual Computer* 38.2 (2022), pp. 555–571.

[74] Hai Huang, Claus Brenner, and Monika Sester. “A generative statistical approach to automatic 3D building roof reconstruction from laser scanning data”. In: *ISPRS Journal of photogrammetry and remote sensing* 79 (2013), pp. 29–43.

[75] Jin Huang, Jantien Stoter, and Liangliang Nan. “Symmetrization of 2D Polygonal Shapes Using Mixed-Integer Programming”. In: *Computer-Aided Design* (2023), p. 103572.

[76] Jin Huang et al. “City3D: Large-Scale Building Reconstruction from Airborne LiDAR Point Clouds”. In: *Remote Sensing* 14.9 (2022), p. 2254.

[77] Jingwei Huang, Yanfeng Zhang, and Mingwei Sun. “Primitivenet: Primitive instance segmentation with local primitive embedding under adversarial metric”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 15343–15353.

[78] Rostislav Hulik et al. “Continuous plane detection in point-cloud data based on 3D Hough Transform”. In: *Journal of visual communication and image representation* 25.1 (2014), pp. 86–97.

[79] Masud Husain, Stefan Treue, and Richard A Andersen. “Surface interpolation in three-dimensional structure-from-motion perception”. In: *Neural Computation* 1.3 (1989), pp. 324–333.

[80] Takeo Igarashi et al. “Interactive beautification: A technique for rapid geometric design”. In: *ACM SIGGRAPH 2007 courses*. 2007, 18–es.

[81] Chima Iheaturu et al. “A simplified structure-from-motion photogrammetry approach for urban development analysis”. In: *Remote sensing applications: Society and Environment* 28 (2022), p. 100850.

[82] M Jarzabek-Rychard. “Reconstruction of building outlines in dense urban areas based on LIDAR data and address points”. In: *ISPAr* 39 (2012), pp. 121–126.

[83] Haiyong Jiang et al. “Automatic constraint detection for 2d layout regularization”. In: *IEEE transactions on visualization and computer graphics* 22.8 (2015), pp. 1933–1944.

[84] Haiyong Jiang et al. “Symmetrization of facade layouts”. In: *Graphical Models* 85 (2016), pp. 11–21.

[85] Jingen Jiang et al. “Structure-Aware Surface Reconstruction via Primitive Assembly”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 14171–14180.

[86] Ellis L Johnson, George L Nemhauser, and Martin WP Savelsbergh. “Progress in linear programming-based algorithms for integer programming: An exposition”. In: *Informs journal on computing* 12.1 (2000), pp. 2–23.

[87] ISIS KAISER. *The possibilities and limitations of City3D as large-scale 3D building reconstruction model*. 2024. URL: https://research.geodan.nl/the-possibilities-and-limitations-of-city3d-as-large-scale-3d-building-reconstruction-model/.

[88] Antonios Kargas, Georgios Loumos, and Dimitrios Varoutas. “Using different ways of 3D reconstruction of historical cities for gaming purposes: The case study of Nafplio”. In: *Heritage* 2.3 (2019), pp. 1799–1811.

[89] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. “Poisson surface reconstruction”. In: *Proceedings of the fourth Eurographics symposium on Geometry processing*. Vol. 7. 2006, p. 0.

[90] Tom Kelly et al. “BigSUR: large-scale structured urban reconstruction”. In: *ACM Transactions on Graphics* 36.6 (2017).

[91] James Kennedy and Russell Eberhart. “Particle swarm optimization”. In: *Proceedings of ICNN'95-international conference on neural networks*. Vol. 4. IEEE. 1995, pp. 1942–1948.

[92] Bernhard Kerbl et al. “3D gaussian splatting for real-time radiance field rendering”. In: *ACM Transactions on Graphics* 42.4 (2023), pp. 1–14.

[93] Vladimir G Kim et al. “Möbius transformations for global intrinsic symmetry analysis”. In: *Computer Graphics Forum*. Vol. 29. 5. Wiley Online Library. 2010, pp. 1689–1700.

[94] Stavros G Kolliopoulos and Clifford Stein. “Approximating disjoint-path problems using greedy algorithms and packing integer programs”. In: *International Conference on Integer Programming and Combinatorial Optimization*. Springer. 1998, pp. 153–168.

[95] Tatjana Kutzner, Kanishk Chaturvedi, and Thomas H Kolbe. “CityGML 3.0: New functions open up new applications”. In: *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science* 88.1 (2020), pp. 43–61.

[96] Florent Lafarge et al. “Structural approach for building reconstruction from a single DSM”. In: *IEEE Transactions on pattern analysis and machine intelligence* 32.1 (2008), pp. 135–147.

[97] Eugene L Lawler and David E Wood. “Branch-and-bound methods: A survey”. In: *Operations research* 14.4 (1966), pp. 699–719.

[98] Eric-Tuan Lê et al. “Cpfn: Cascaded primitive fitting networks for high-resolution point clouds”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 7457–7466.

[99] Hugo Ledoux et al. “3dfier: automatic reconstruction of 3D city models”. In: *Journal of Open Source Software* 6.57 (2021), p. 2866.

[100] Hugo Ledoux et al. *Computational modelling of terrains*. Self-published, 2022. DOI: 10.5281/zenodo.3992107.

[101] Seungkyu Lee and Yanxi Liu. “Curved glide-reflection symmetry detection”. In: *IEEE transactions on pattern analysis and machine intelligence* 34.2 (2011), pp. 266–278.

[102] Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. “A faster cutting plane method and its implications for combinatorial and convex optimization”. In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. IEEE. 2015, pp. 1049–1065.

[103] Li Li et al. “Point2Roof: End-to-end 3D building roof modeling from airborne LiDAR point clouds”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 193 (2022), pp. 17–28.

[104] Lin Li et al. “An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells”. In: *Remote Sensing* 9.5 (2017), p. 433.

[105] Lingxiao Li et al. “Supervised fitting of geometric primitives to 3D point clouds”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2652–2660.

[106] Minglei Li and Liangliang Nan. “Feature-preserving 3D mesh simplification for urban buildings”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 173 (2021), pp. 135–150.

[107] Minglei Li, Shu Peng, and Liangliang Nan. “Hybrid geometry sets for global registration of cross-source geometric data”. In: *International Journal of Applied Earth Observation and Geoinformation* 128 (2024), p. 103733.

[108] Minglei Li, Franz Rottensteiner, and Christian Heipke. “Modelling of buildings from aerial LiDAR point clouds using TINs and label maps”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 154 (2019), pp. 127–138.

[109] Minglei Li, Peter Wonka, and Liangliang Nan. “Manhattan-world urban reconstruction from point clouds”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 54–69.

[110] Minglei Li et al. “Reconstructing building mass models from UAV images”. In: *Computers & Graphics* 54 (2016), pp. 84–93.

[111] Ying Li et al. “ADR-MVSNet: A cascade network for 3D point cloud reconstruction with pixel occlusion”. In: *Pattern recognition* 125 (2022), p. 108516.

[112] Yuan Li and Bo Wu. “Relation-Constrained 3D Reconstruction of Buildings in Metropolitan Areas from Photogrammetric Point Clouds”. In: *Remote Sensing* 13.1 (2021), p. 129.

[113] Yuanqi Li et al. “Surface and Edge Detection for Primitive Fitting of Point Clouds”. In: *ACM SIGGRAPH 2023 Conference Proceedings*. 2023, pp. 1–10.

[114] Zhixin Li and Jie Shan. “RANSAC-based multi primitive building reconstruction from 3D point clouds”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 185 (2022), pp. 247–260.

[115] Dong C Liu and Jorge Nocedal. “On the limited memory BFGS method for large scale optimization”. In: *Mathematical programming* 45.1-3 (1989), pp. 503–528.

[116] Yanxi Liu et al. “Computational symmetry in computer vision and computer graphics”. In: *Foundations and Trends® in Computer Graphics and Vision* 5.1–2 (2010), pp. 1–195.

[117] Yebin Liu, Qionghai Dai, and Wenli Xu. “A point-cloud-based multiview stereo algorithm for free-viewpoint video”. In: *IEEE transactions on visualization and computer graphics* 16.3 (2009), pp. 407–418.

[118] Yujia Liu et al. “Point2Building: Reconstructing Buildings from Airborne LiDAR Point Clouds”. In: *arXiv preprint arXiv:2403.02136* (2024).

[119] Alessandro Malerba et al. “Post carbon city: building valuation and energy performance simulation programs”. In: *New Metropolitan Perspectives: Local Knowledge and Innovation Dynamics Towards Territory Attractiveness Through the Implementation of Horizon/E2020/Agenda2030–Volume 2*. Springer. 2019, pp. 513–521.

[120] Hugues Marchand et al. “Cutting planes in integer and mixed integer programming”. In: *Discrete Applied Mathematics* 123.1-3 (2002), pp. 397–446.

[121] Xuelian Meng, Le Wang, and Nate Currit. “Morphology-based building detection from airborne LIDAR data”. In: *Photogrammetric Engineering & Remote Sensing* 75.4 (2009), pp. 437–442.

[122] Ben Mildenhall et al. “Nerf: Representing scenes as neural radiance fields for view synthesis”. In: *Communications of the ACM* 65.1 (2021), pp. 99–106.

[123] John E Mitchell. “Branch-and-cut algorithms for combinatorial optimization problems”. In: *Handbook of applied optimization* 1.1 (2002), pp. 65–77.

[124] Niloy J Mitra, Leonidas J Guibas, and Mark Pauly. “Partial and approximate symmetry detection for 3D geometry”. In: *ACM Transactions on Graphics (TOG)* 25.3 (2006), pp. 560–568.

[125] Niloy J Mitra, Leonidas J Guibas, and Mark Pauly. “Symmetrization”. In: *ACM Transactions on Graphics (TOG)* 26.3 (2007), 63–es.

[126] Niloy J Mitra et al. “Structure-aware shape processing”. In: *ACM SIGGRAPH 2014 Courses*. 2014, pp. 1–21.

[127] Niloy J Mitra et al. “Symmetry in 3D geometry: Extraction and applications”. In: *Computer Graphics Forum*. Vol. 32. 6. Wiley Online Library. 2013, pp. 1–23.

[128] Liangliang Nan. “Easy3D: a lightweight, easy-to-use, and efficient C++ library for processing and rendering 3D data”. In: *Journal of Open Source Software* 6.64 (2021), p. 3255.

[129] Liangliang Nan and Peter Wonka. “PolyFit: Polygonal Surface Reconstruction from Point Clouds”. In: (2017).

[130] Liangliang Nan et al. “Smartboxes for interactive urban reconstruction”. In: *ACM SIGGRAPH 2010 papers*. 2010, pp. 1–10.

[131] Liangliang Nan et al. “Template assembly for detailed urban reconstruction”. In: *Computer Graphics Forum*. Vol. 34. 2. Wiley Online Library. 2015, pp. 217–228.

[132] PRT Newby. “Quality management for surveying, photogrammetry and digital mapping at the ordnance survey”. In: *The Photogrammetric Record* 14.79 (1992), pp. 45–58.

[133] Abdul Nurunnabi, David Belton, and Geoff West. “Robust segmentation for multiple planar surface extraction in laser scanning 3D point cloud data”. In: *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. IEEE. 2012, pp. 1367–1370.

[134] Gunay Orbay and Levent Burak Kara. “Beautification of design sketches using trainable stroke clustering and curve fitting”. In: *IEEE Transactions on Visualization and Computer Graphics* 17.5 (2011), pp. 694–708.

[135] Maks Ovsjanikov, Jian Sun, and Leonidas Guibas. “Global Intrinsic Symmetries of Shapes”. In: *Computer Graphics Forum* 27.5 (2008), pp. 1341–1348. ISSN: 1467-8659.

[136] Alexandra I Papadaki and Ronny Hansch. “Match or no match: Keypoint filtering based on matching probability”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 1014–1015.

[137] Amal Dev Parakkat, Uday Bondi Pundarikaksha, and Ramanathan Muthuganapathy. “A Delaunay triangulation based approach for cleaning rough sketches”. In: *Computers & Graphics* 74 (2018), pp. 171–181.

[138] RY Peters et al. “Automated 3D Reconstruction of LoD2 and LoD1 Models for All 10 Million Buildings of the Netherlands”. In: *Photogrammetric Engineering and Remote Sensing* 88.3 (2022).

[139] Joshua Podolak et al. “A planar-reflective symmetry transform for 3D shapes”. In: *ACM SIGGRAPH 2006*. 2006, pp. 549–559.

[140] Charles R Qi et al. “Pointnet: Deep learning on point sets for 3D classification and segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[141] Yi-Ling Qiao et al. “Learning-based Intrinsic Reflectional Symmetry Detection”. In: *IEEE Transactions on Visualization and Computer Graphics* (2022), pp. 1–1.

[142] Tahir Rabbani and Frank Van Den Heuvel. “Efficient hough transform for automatic detection of cylinders in point clouds”. In: *Isprs Wg Iii/3, Iii/4* 3 (2005), pp. 60–65.

[143] Tahir Rabbani, Frank Van Den Heuvel, and George Vosselmann. “Segmentation of point clouds using smoothness constraint”. In: *International archives of photogrammetry, remote sensing and spatial information sciences* 36.5 (2006), pp. 248–253.

[144] Daniel Reisfeld, Haim Wolfson, and Yehezkel Yeshurun. “Context-free attentional operators: The generalized symmetry transform”. In: *International Journal of Computer Vision* 14.2 (1995), pp. 119–130.

[145] Dong Ren et al. “DEM extraction based on SFM using remote sensing images”. In: *2014 World Automation Congress (WAC)*. IEEE. 2014, pp. 779–783.

[146] Franz Rottensteiner et al. “The ISPRS benchmark on urban object classification and 3D building reconstruction”. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences I-3 (2012), Nr. 1* 1.1 (2012), pp. 293–298.

[147] Alexey Ruchay et al. “Fast 3D object symmetry detection for point cloud”. In: *Applications of Digital Image Processing XLIV*. Vol. 11842. SPIE. 2021, pp. 574–579.

[148] Enoc Sanz-Ablanedo et al. "Accuracy of unmanned aerial vehicle (UAV) and SfM photogrammetry survey as a function of the number and location of ground control points used". In: *Remote Sensing* 10.10 (2018), p. 1606.

[149] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. "Efficient RANSAC for point-cloud shape detection". In: *Computer graphics forum*. Vol. 26. 2. Wiley Online Library. 2007, pp. 214–226.

[150] Erich Schubert et al. "DBSCAN revisited, revisited: why and how you should (still) use DBSCAN". In: *ACM Transactions on Database Systems (TODS)* 42.3 (2017), pp. 1–21.

[151] Ahyun Seo et al. "Reflection and Rotation Symmetry Detection via Equivariant Learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 9539–9548.

[152] Ryan D Sheridan et al. "Modeling forest aboveground biomass and volume using airborne LiDAR metrics and forest inventory and analysis data in the Pacific Northwest". In: *Remote Sensing* 7.1 (2014), pp. 229–255.

[153] Yifei Shi et al. "Learning to Detect 3D Symmetry From Single-View RGB-D Images With Weak Supervision". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.4 (2023), pp. 4882–4896.

[154] Yifei Shi et al. "SymmetryNet: learning to predict reflectional and rotational symmetries of 3D shapes from single-view RGB-D images". In: *ACM Transactions on Graphics (TOG)* 39.6 (2020), pp. 1–14.

[155] Zeyun Shi et al. "Symmetry and orbit detection via lie-algebra voting". In: *Computer Graphics Forum*. Vol. 35. 5. Wiley Online Library. 2016, pp. 217–227.

[156] Yawar Siddiqui et al. "Meshgpt: Generating triangle meshes with decoder-only transformers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 19615–19625.

[157] Patricio Simari, Evangelos Kalogerakis, and Karan Singh. "Folding meshes: hierarchical mesh segmentation based on planar symmetry". In: *Symposium on geometry processing*. 2006, pp. 111–119.

[158] Ivan Sipiran, Robert Gregor, and Tobias Schreck. "Approximate symmetry detection in partial 3D meshes". In: *Computer Graphics Forum*. Vol. 33. 7. Wiley Online Library. 2014, pp. 131–140.

[159] Gunho Sohn et al. "An implicit regularization for 3D building rooftop modeling using airborne lidar data". In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 1.3 (2012), pp. 305–310.

[160] Hyeontae Son and Young Min Kim. "SAUM: Symmetry-aware upsampling module for consistent point cloud completion". In: *Proceedings of the Asian Conference on Computer Vision*. 2020.

[161] Jantien Stoter et al. "Automated reconstruction of 3D input data for noise simulation". In: *Computers, Environment and Urban Systems* 80 (2020), p. 101424.

[162] Juan C Suárez et al. “Use of airborne LiDAR and aerial photography in the estimation of individual tree heights in forestry”. In: *Computers & Geosciences* 31.2 (2005), pp. 253–262.

[163] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. “A concise and provably informative multi-scale signature based on heat diffusion”. In: *Computer graphics forum*. Vol. 28. 5. Wiley Online Library. 2009, pp. 1383–1392.

[164] Shaohui Sun and Carl Salvaggio. “Aerial 3D building detection and modeling from airborne LiDAR point clouds”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 6.3 (2013), pp. 1440–1449.

[165] Fayez Tarsha-Kurdi, Tania Landes, and Pierre Grussenmeyer. “Hough-transform and extended ransac algorithms for automatic detection of 3D building roof planes from lidar data”. In: 2007.

[166] Jefry Tedjokusumo and Wee Kheng Leow. “Normalization and alignment of 3D objects based on bilateral symmetry planes”. In: *International Conference on Multimedia Modeling*. Springer. 2007, pp. 74–85.

[167] The CGAL Project. *CGAL User and Reference Manual*. 5.5.1. CGAL Editorial Board, 2022. URL: https://doc.cgal.org/5.5.1/Manual/packages.html.

[168] Hugues Thomas et al. “Kpconv: Flexible and deformable convolution for point clouds”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 6411–6420.

[169] Sebastian Thrun and Ben Wegbreit. “Shape from symmetry”. In: *Tenth IEEE International Conference on Computer Vision (ICCV’05) Volume 1*. Vol. 2. IEEE. 2005, pp. 1824–1831.

[170] Peter JM Van Laarhoven et al. *Simulated annealing*. Springer, 1987.

[171] Nina Varney, Vijayan K Asari, and Quinn Graehling. “DALES: a large-scale aerial LiDAR data set for semantic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 186–187.

[172] Yannick Verdie, Florent Lafarge, and Pierre Alliez. “LOD generation for urban scenes”. In: *ACM Transactions on Graphics* 34 (2015), p. 30.

[173] Vivek Verma, Rakesh Kumar, and Stephen Hsu. “3D building detection and modeling from aerial lidar data”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*. Vol. 2. IEEE. 2006, pp. 2213–2220.

[174] Die Wang et al. “Progressive point set surface compression based on planar reflective symmetry analysis”. In: *Computer-Aided Design* 58 (2015), pp. 34–42.

[175] Hui Wang and Hui Huang. “Group representation of global intrinsic symmetries”. In: *Computer Graphics Forum*. Vol. 36. 7. Wiley Online Library. 2017, pp. 51–61.

[176] Ruisheng Wang, Shangfeng Huang, and Hongxin Yang. “Building3D: An Urban-Scale Dataset and Benchmarks for Learning Roof Structures from Point Clouds”. In: *arXiv preprint arXiv:2307.11914* (2023).

[177] Y. Wang et al. “Symmetry Hierarchy of Man-Made Objects”. In: *Computer Graphics Forum* 30.2 (2011), pp. 287–296. ISSN: 1467-8659.

[178] Xuedong Wen et al. “Accurate reconstruction of the LoD3 building model by integrating multi-source point clouds and oblique remote sensing imagery”. In: *ISPRS international journal of geo-information* 8.3 (2019), p. 135.

[179] Tomas Werner. “A linear programming approach to max-sum problem: A review”. In: *IEEE transactions on pattern analysis and machine intelligence* 29.7 (2007), pp. 1165–1179.

[180] Edmund Widl, Giorgio Agugiaro, and Jan Peters-Anders. “Linking Semantic 3D City Models with Domain-Specific Simulation Tools for the Planning and Validation of Energy Applications at District Level”. In: *Sustainability* 13.16 (2021), p. 8782.

[181] Laurence A Wolsey. *Integer programming*. John Wiley & Sons, 2020.

[182] Haozhe Xie et al. “Citydreamer: Compositional generative model of unbounded 3d cities”. In: *arXiv preprint arXiv:2309.00610* (2023).

[183] Biao Xiong, S Oude Elberink, and G Vosselman. “A graph edit dictionary for correcting errors in roof topology graphs reconstructed from point clouds”. In: *ISPRS Journal of photogrammetry and remote sensing* 93 (2014), pp. 227–242.

[184] Biao Xiong, S Oude Elberink, and G Vosselman. “FOOTPRINT MAP PARTITIONING USING AIRBORNE LASER SCANNING DATA.” In: *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 3.3 (2016).

[185] Bo Xu et al. “Investigation on the weighted ransac approaches for building roof plane segmentation from lidar point clouds”. In: *Remote Sensing* 8.1 (2015), p. 5.

[186] Kai Xu et al. “Multi-scale partial intrinsic symmetry detection”. In: *ACM Transactions On Graphics (TOG)* 31.6 (2012), pp. 1–11.

[187] Kai Xu et al. “Partial intrinsic reflectional symmetry of 3D shapes”. In: *ACM SIGGRAPH Asia 2009* (2009), pp. 1–10.

[188] Pengfei Xu et al. “Global Beautification of 2D and 3D Layouts with Interactive Ambiguity Resolution”. In: *IEEE transactions on visualization and computer graphics* (2019).

[189] Pengfei Xu et al. “Global Beautification of Layouts with Interactive Ambiguity Resolution”. In: *UIST ’14*. 2014.

[190] Fan Xue et al. “A derivative-free optimization-based approach for detecting architectural symmetries from 3D point clouds”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 148 (2019), pp. 32–40.

[191] Siming Yan et al. “Hpnet: Deep primitive segmentation using hybrid representations”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 2753–2762.

[192] Guoqing Yang et al. “UrbanBIS: a Large-scale Benchmark for Fine-grained Urban Building Instance Segmentation”. In: *ACM SIGGRAPH 2023 Conference Proceedings*. 2023, pp. 1–11.

[193] Lina Yang et al. “Efficient plane extraction using normal estimation and RANSAC from 3D point cloud”. In: *Computer Standards & Interfaces* 82 (2022), p. 103608.

[194] Shengming Yang et al. “Connectivity-aware Graph: A planar topology for 3D building surface reconstruction”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 191 (2022), pp. 302–314.

[195] Zhihang Yao et al. “3DCityDB-a 3D geodatabase solution for the management, analysis, and visualization of semantic 3D city models based on CityGML”. In: *Open Geospatial Data, Software and Standards* 3.1 (2018), pp. 1–26.

[196] Cheng Yi et al. “Urban building reconstruction from raw LiDAR point data”. In: *Computer-Aided Design* 93 (2017), pp. 1–14.

[197] Haci Murat Yilmaz et al. “Importance of digital close-range photogrammetry in documentation of cultural heritage”. In: *Journal of Cultural Heritage* 8.4 (2007), pp. 428–433.

[198] Mulin Yu and Florent Lafarge. “Finding Good Configurations of Planar Primitives in Unorganized Point Clouds”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 6367–6376.

[199] Hao Zhang et al. “Layered analysis of irregular facades via symmetry maximization”. In: *ACM Transactions on Graphics (TOG)* 32.4 (2013), pp. 1–13. ISSN: 0730-0301.

[200] Keqi Zhang, Jianhua Yan, and S-C Chen. “Automatic construction of building footprints from airborne LIDAR data”. In: *IEEE Transactions on Geoscience and Remote Sensing* 44.9 (2006), pp. 2523–2533.

[201] Long Zhang et al. “Blending surface segmentation and editing for 3D models”. In: *IEEE Transactions on Visualization and Computer Graphics* 28.8 (2020), pp. 2879–2894.

[202] Qian Zheng et al. “Skeleton-Intrinsic Symmetrization of Shapes”. In: *Computer Graphics Forum*. Vol. 34. 2. Wiley Online Library. 2015, pp. 275–286.

[203] Alexander Michael Zhivov et al. “Planning tools to simulate and optimize neighborhood energy systems”. In: *Green Defense Technology*. Springer, 2017, pp. 137–163.

[204] Qian-Yi Zhou. *3D urban modeling from city-scale aerial LiDAR data*. University of Southern California, 2012.

[205] Qian-Yi Zhou and Ulrich Neumann. “2.5 D building modeling with topology control”. In: *CVPR 2011*. IEEE. 2011, pp. 2489–2496.

[206] Qian-Yi Zhou and Ulrich Neumann. “2.5 d dual contouring: A robust approach to creating building models from aerial lidar point clouds”. In: *European conference on computer vision*. Springer. 2010, pp. 115–128.

[207] Qian-Yi Zhou and Ulrich Neumann. “Fast and extensible building modeling from airborne LiDAR data”. In: *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*. 2008, pp. 1–8.

[208] Xiaohui Zhou et al. “Survey on path and view planning for UAVs”. In: *Virtual Reality & Intelligent Hardware* 2.1 (2020), pp. 56–69.

[209] Lingjie Zhu et al. “Variational building modeling from urban MVS meshes”. In: *2017 International Conference on 3D Vision (3DV)*. IEEE. 2017, pp. 318–326.

[210] Daniel A Zimble et al. “Characterizing vertical forest structure using small-footprint airborne LiDAR”. In: *Remote sensing of Environment* 87.2-3 (2003), pp. 171–182.

[211] Marco Zuliani, Charles S Kenney, and BS Manjunath. “The multiransac algorithm and its application to detect planar homographies”. In: *IEEE International Conference on Image Processing 2005*. Vol. 3. IEEE. 2005, pp. III–153.

# CURRICULUM VITÆ

## Jin HUANG

Jin Huang was born in Jiujiang, P.R.China, in 1994. He received his bachelor's degree in Manufacturing Engineering of Aerospace from Nanjing University of Aeronautics and Astronautics, Nanjing, China in 2016. He got a master's degree in Manufacturing Engineering of Aerospace at Nanjing University of Aeronautics and Astronautics, Nanjing, China in 2019. In the same year, he continued a Ph.D. study in the 3D Geoinformation group, TU Delft under the supervision of Prof. dr. J.E.Stoter and Dr. L.Nan. His research mainly focused on structure-aware 3D building reconstruction from airborne LiDAR point clouds or MVS meshes, also the subject of this thesis.

# LIST OF PUBLICATIONS

1. **Jin Huang**, Jantien Stoter, Mingqiang Wei, Liangliang Nan. "MakeItSimple: User-Guided Compact Building Reconstruction from MVS Meshes." Manuscript in preparation for submission to IEEE Transactions on Visualization and Computer Graphics (TVCG).

2. **Jin Huang**, Jantien Stoter, Liangliang Nan. Symmetrization of 2D Polygonal Shapes Using Mixed-Integer Programming. Computer-Aided Design (2023): 103572.

3. **Jin Huang**, Jantien Stoter, Ravi Peters, Liangliang Nan. City3D: Large-scale building reconstruction from airborne LiDAR point clouds. Remote Sensing 14.9 (2022): 2254.