การตัดแยกองค์ประกอบโครงสร้างภายในอาคาร 3 มิติจากข้อมูลพอยท์คลาวด์โดยใช้ ดีบีสแกน และ แรนแซ็คปรับปรุงด้วยเงื่อนไขการเบี่ยงเบนปกติ

นายธนพล ดวงภุมเมศ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาคณิตศาสตร์ประยุกต์และวิทยาการคณนา
ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2565

3D BUILDING INTERNAL STRUCTURAL COMPONENT SEGMENTATION FROM POINT CLOUD DATA USING DBSCAN AND MODIFIED RANSAC WITH NORMAL DEVIATION CONDITIONS

Mr. Thanapon Doougphummet

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Applied Mathematics and
Computational Science
Department of Mathematics and Computer Science
Faculty of Science
Chulalongkorn University
Academic Year 2022

Thesis Title	3D BUILDING INTERNAL STRUCTURAL COMPONENT SEGMENTATION FROM POINT CLOUD DATA USING DBSCAN AND MODIFIED RANSAC WITH NORMAL DEVIATION CONDITIONS
By	Mr. Thanapon Doougphummet
Field of Study	Applied Mathematics and Computational Science
Thesis Advisor	Associate Professor Rajalida Lipikorn, Ph.D.
Thesis Co-advisor	Associate Professor Petarpa Boonserm, Ph.D.

Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

.... Dean of the Faculty of Science

(Professor Polkit Sangvanich, Ph.D.)

THESIS COMMITTEE

.... Chairman

(Associate Professor Krung Sinapiromsaran, Ph.D.)

.... Thesis Advisor

(Associate Professor Rajalida Lipikorn, Ph.D.)

.... Thesis Co-advisor

(Associate Professor Petarpa Boonserm, Ph.D.)

.... Examiner

(Associate Professor Nagul Cooharojananone, Ph.D.)

.... External Examiner

(Suriya Natsupakpong, Ph.D.)

ธนพล ดวงภุมเมศ : การตัดแยกองค์ประกอบโครงสร้างภายในอาคาร 3 มิติจากข้อมูลพอยท์คลาวด์โดยใช้ ดีบีสแกน และ แรนแซ็คปรับปรุงด้วยเงื่อนไขการเบี่ยงเบนปกติ. (3D BUILDING INTERNAL STRUCTURAL COMPONENT SEGMENTATION FROM POINT CLOUD DATA USING DBSCAN AND MODIFIED RANSAC WITH NORMAL DEVIATION CONDITIONS) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : รศ.ดร. รัชลิดา ลิปิกรณ์, อ.ที่ปรึกษาวิทยานิพนธ์ร่วม : รศ.ดร. เพชรอาภา บุญเสริม 50 หน้า.

ในปัจจุบัน กล้องเลเซอร์สแกนเป็นเครื่องมือที่เข้ามามีบทบาทสำคัญในการเก็บโครงสร้างอาคารในรูปแบบของพอยท์คลาวด์ ซึ่งสามารถนำมาใช้ในการสร้างพิมพ์เขียวหรือแบบแปลนชั้นสำหรับการบูรณะหรือการปรับปรุงอาคารเก่า ข้อมูลพอยท์คลาวด์แทนรูปร่างหรือวัตถุในสามมิติ การใช้พอยท์คลาวด์ร่วมกับการใช้ประโยชน์จากแบบจำลองสารสนเทศอาคารซึ่งเป็นกระแสงานที่ให้สารสนเทศเกี่ยวกับฐานและการวัดโครงสร้างของอาคาร วางแผน ออกแบบไปจนถึงการก่อสร้างที่ให้ความสะดวกสบายและมีประสิทธิภาพในหลาย ๆ ด้านมากกว่าการทำงานแบบดั้งเดิม แต่การใช้แบบจำลองสารสนเทศอาคารในการสร้างโครงสร้างภายในของอาคารเก่าจากพอยท์คลาวด์ด้วยมือจะใช้เวลามากและต้องใช้วิศวกรที่มีความเชี่ยวชาญสูง การพัฒนากระบวนวิธีที่สามารถสร้างโครงสร้างภายในของอาคารเก่าแบบอัตโนมัติจากพอยท์คลาวด์จะช่วยอำนวยความสะดวกการดำเนินการสร้างแบบจำลอง ในกระบวนการสร้างทั้งหมด การตัดแยกเป็นกระบวนการสำคัญที่ใช้ในการระบุส่วนประกอบหลักของอาคารที่ยังเป็นปัญหาที่นักวิจัยจะพัฒนาอัลกอริทึมที่สามารถตัดแยกส่วนประกอบโครงสร้างภายในของอาคาร

วัตถุประสงค์ของงานวิจัยนี้คือเพื่อพัฒนาระเบียบวิธีตัดแยกที่สามารถสกัดโครงสร้างเชิงระนาบของอาคารจากพอยท์คลาวด์ด้วยอัลกอริทึมตัวอย่างจากการสุ่มที่มีความสอดคล้องสูง (RANSAC) ปรับแต่ง อัลกอริทึมตัวอย่างจากการสุ่มที่มีความสอดคล้องสูงต้นฉบับถูกดัดแปลงโดยการลดทอนความซับซ้อนในการคำนวณโดยใช้การสุ่มตัวอย่างเฉพาะที่ และคุณภาพของการตัดแยกถูกทำให้ดีขึ้นโดยการรวมเงื่อนไขค่าเบี่ยงเบนแนวฉากกับภาวะเชื่อมต่อในการคำนวนคะแนนของอัลกอริทึมต้นฉบับ อัลกอริทึมต้นฉบับและระเบียบวิธีที่นำเสนอถูกประเมินผลบนชุดข้อมูลเกณฑ์มาตรฐานของ สมาคมระหว่างประเทศสำหรับการรังวัดด้วยภาพถ่ายและ

การรับรู้จากระยะไกล (ISPRS) ผลที่ได้จากระเบียบวิธีที่นำเสนอถูกนำมาเปรียบเทียบกับผลที่ได้จากอัลกอริทึมต้นฉบับ และจากการวัดด้วยสายตาจะเห็นได้ว่าส่วนประกอบโครงสร้างที่ถูกสกัดด้วยระเบียบวิธีที่นำเสนอมีความใกล้เคียงกับส่วนประกอบโครงสร้างจริงมากกว่าและยังสามารถรักษาลักษณะเฉพาะโดยรวมของอาคารไว้

ภาควิชา คณิตศาสตร์และวิทยาการคอมพิวเตอร์ ลายมือชื่อนิสิต ....

ลายมือชื่อ อ.ที่ปรึกษาหลัก ....

สาขาวิชา คณิตศาสตร์ประยุกต์และวิทยาการคณนา ลายมือชื่อ อ.ที่ปรึกษาร่วม................

ปีการศึกษา 2565

6270045723 : MAJOR APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE

KEYWORDS : POINT CLOUD / BUILDING INFORMATION MODELING (BIM) / THREE-DIMENSIONAL (3D) SEGMENTATION / RANSAC / DBSCAN / NORMAL DEVIATION

THANAPON DOOUGPHUMMET : 3D BUILDING INTERNAL STRUCTURAL COMPONENT SEGMENTATION FROM POINT CLOUD DATA USING DBSCAN AND MODIFIED RANSAC WITH NORMAL DEVIATION CONDITIONS. ADVISOR : ASSOC. PROF. RAJALIDA LIPIKORN, Ph.D., THESIS COADVISOR : ASSOC. PROF. PETARPA BOONSERM, Ph.D., 50 pp.

Nowadays, the laser scanner plays an important role as a tool to capture building structure in a form of point cloud which can be used to draw a blueprint or a floor plan for reconstruction or renovation of an existing building. The point cloud data represent a shape or an object in three dimensions. This point cloud together with the utilization of Building Information Modeling (BIM) which is a workflow that provides information about the foundation and structure measurement of a building, make a planning and design to construction more convenient and efficient in several aspects than the traditional way. However, using BIM to design or construct the internal structure of an existing building from point cloud manually can be very time consuming and requires a highly skilled engineer. The development of a method that can automatically construct the internal structure of an existing building from point cloud can facilitate the process of model creation. Among the whole process of construction, segmentation is an important process that identifies the main components of a building, which is still a challenging problem for researchers to develop an algorithm that can automatically segment the internal structural components of the building.

The objective of this research is to develop a segmentation method that can extract the planar structures of a building from point cloud using the modified Random Sample Consensus (RANSAC) algorithm. The original RANSAC is modified by reducing the computational complexity using localized sampling, and the quality of segmentation is improved by adding normal deviation with connectivity constraints to RANSAC's score calculation. The original RANSAC and the proposed method were evaluated on the International Society for Photogrammetry and Remote Sensing (ISPRS) benchmark dataset. The segmentation results of the proposed method were visually compared to the results of the original RANSAC and it can be seen that the extracted components from the proposed method are closer to the actual structure and can also preserve the overall characteristics of the buildings.

Department : Mathematics and Computer Science
Field of Study : Applied Mathematics and Computational Science
Academic Year : 2022

Student's Signature ....
Advisor's Signature ....
Co-advisor's Signature ....

ACKNOWLEDGEMENTS

I would like to deeply express sincere gratitude to Associate Professor Dr. Rajalida Lipikorn, my advisor, for her kind advice and guidance that carried me through all the stages of writing my thesis. I am extremely grateful for her to give me inspiration that drove me into the field of applied mathematics and image processing, it is a great honor to work under her supervision. I would also like to acknowledge Associate Professor Dr. Petarpa Boonserm as my co-advisor, and I am gratefully indebted to her for her encouragement and valuable comments on this thesis.

I would like to thank the scholarship support, the Development and Promotion of Science and Technology Talents Project (DPST), for providing educational opportunities, and also deeply appreciative to the Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University for making it possible for me to complete my master's degree. I would like to express my great appreciation to the International Society for Photogrammetry and Remote Sensing (ISPRS) for providing benchmark on indoor modelling dataset.

Finally, I would like to express my very profound gratitude to my family and my friends for providing me with support and encouragement throughout my years of study and through my rough times during the process of researching and writing this thesis.

CONTENTS

Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Rationale

Designing and planning is a preliminary process of any building renovation. The blueprint that represents the 2D structural components and their measurements are redrawn by the traditional tools called Computer-Aided Design and Drafting, or CAD since the original blueprint might not be up-to-date and inaccurate to use with the reconstruction or renovation of the existing building because the overall structure is usually different from the original blueprint due to modification.

Recently, the introduction of Building Information Modeling (BIM) led us to an alternative way to create 3D models of the building from point cloud data to collaborate with the reconstruction or renovation of the existing buildings. The need to use BIM has increased due to its efficiency and the advance in 3D scanner technology that allows us to collect data in a form of point cloud, which is a set of data points in a space representing 3D objects. This type of data can be used to obtain a 3D model to assemble with the original blueprint for a more precise and up-to-date model that can be processed manually with the building-modeling software. However, the computation time of this process is very high, and highly skilled engineer is needed. This problem led to the release of the more advanced building-modeling techniques that can automatically reconstruct the 3D model from point cloud data and provide more facilities to fit with today's applications.

The overall process of reconstruction of the indoor environment of a building from point cloud usually consists of three steps: data collection, point cloud data segmentation, and BIM creation [1]. Data collection is the process of obtaining point cloud data using a 3D capture technology to construct the data that represent the scene. Next step is the point cloud data segmentation that detects and segments the structural components of a building from point cloud because point cloud data consist of several other components, such as people, tables, chairs, and air conditioning. These segmented point cloud data are then used to create a 3D model of the structural components in the final step.

1.2 Scope of work

As previously mentioned, point cloud data segmentation is an important step of the indoor environment reconstruction, thus several segmentation methods have been proposed. However, methodological studies on segmentation are still ongoing for better results and performance. There are still some challenges in segmentation, for example the accuracy and the computational complexity of structural component segmentation where most of the models are constructed from planar surfaces of a building that are parts of walls, floors, and ceilings that connect into rooms, corridors, etc. These challenges lead to the development of a robust point cloud data segmentation that can automatically detect and segment the structural components.

1.3 Research Objectives

- To propose the point cloud data segmentation method of the structural components that improves the segmentation results.
- To develop the algorithm that can automatically segment structural components from point cloud data.

1.4 Thesis overview

The content is divided into 5 chapters. In chapter II, related work and literatures regarding point cloud reconstruction and segmentation are discussed and followed by background knowledge that are necessary to this thesis. In chapter III, the proposed methodology is explained starting from the preparation of input data to enhance the performance of further operations to the segmentation of floor, ceiling, room, and wall. In chapter IV, the segmentation results of the structural components by the proposed method are discussed, and a comparison of the wall segments using the original RANSAC and the proposed method is given. Finally, chapter V summarizes and discusses the outcomes of this thesis.

CHAPTER II

LITERATURE REVIEW AND BACKGROUND KNOWLEDGE

This chapter is divided into two parts: literature review and background knowledge. Related work and literatures related to this research are reviewed and followed by background knowledge that are used in the proposed method.

2.1 Literature review

The indoor environment of the building generally comprises plane geometric primitives. The 3D model of the indoor environment can be created by detecting and extracting these geometric primitives from point cloud. Tarsha-Kurdi et al. [2] compared the results of roof plane detection from Lidar data between Hough-transform [3] and random sample consensus (RANSAC) algorithm [4], it can be concluded that RANSAC overcomes Hough-transform in terms of successful detection, time, and space complexity. Moreover, this paper also suggested to enhance the RANSAC algorithm in order to increase the percentage of successfully detected results and improve the quality of detected roof planes. By giving more priority to the standard deviation of the detected points and using the binary digital surface model to assess noisy points and small roof planes.

Through the use of localized sampling by the octree space partitioning technique, RANSAC's performance on geometric primitive detection was enhanced in terms of speed and robustness [5]. Testing the model with a subset of points instead of the entire data also reduced the computational complexity of the algorithm.

In addition, a floor plan can also be used to specify the boundary of the room that helps the reconstruction of the building. To generate the floor plan, the cell decomposition technique had been employed after acquiring linear primitive information from the wall [6]. The floor plan space was divided into cells by the intersection of previous information, and then the labeling of the cells was carried out. Budroni et al. [7] used the plane sweep technique to detect the positions of the structural components along with the floor plan generation using cell decomposition to create a 3D model. This technique requires wall information as a priori. Recently, through the use of the Delaunay triangulation technique, Capocchiano et al. [8], [9] proposed an algorithm for recovering the ceiling layout from extracted ceiling sections. After that, the connection between the edges in each generated triangular mesh was taken into account to determine the ceiling's edge. This technique uses only the ceiling information to extract the layout.

Lately, a variety of techniques have been used and developed, which has resulted in more designs for automated processing chains and methodologies. Macher et al. [10], [11] proposed the semi-automatic methods that used a binary image's region growing along the z-axis to identify individual rooms, then segmentations are combined with classification to separate point cloud data into grounds, ceilings and walls. Next, RANSAC algorithm was used to segment point cloud data that are classified as walls into individual walls. Finally, wall, ground, and ceiling point cloud data were used to reconstruct the 3D model.

Another literature proposed to employ graphs in room layout detection [12]. Each wall was represented by a node and graphs were created with respect to their connectivity, followed by finding the cuboid from cycles of four continuous walls, and classifying the connection as the final sequence. For a more automatic method, proposed by Cui et al. [13], walls, floors, and ceilings can be obtained directly using a density histogram along with z coordinates, and RANSAC. Rooms were segmented by visibility analysis based on scanner position, and finally, the model was reconstructed with the help of multi-label graph cuts and energy functions.

However, the performance of structural component segmentation can be improved further, therefore this research proposed a new method to enhance plane segmentation by using a density-based algorithm for discovering clusters in large spatial databases with noise (DBSCAN) [14] as localized sampling and supplementing RANSAC by connectivity and normal deviation conditions.

2.2 Background Knowledge

Depending on the demand, point cloud processing can be understood in a variety of contexts. In general, point cloud processing can be described as the creation of a 3D model. This section mentions point cloud, capture tools, the use of point cloud in a BIM scene, and the techniques used for the segmentation process in this research, which involves preprocessing and segmentation.

2.2.1 Point cloud and Laser scanner technology

Point cloud is a digital three-dimensional representation of a real-world space or object. It composes of several millions of individual measurement points taken from the surface of objects, each with associated x, y, and z coordinates. Each point can additionally have intensity information or even RGB color information, which indicates the return strength of the laser pulse that generated the point, depending on the technique used to capture the cloud and the sensors involved. A digital 3D model that represents an object accurately and in detail can subsequently be created by using these formats. Laser scanners and photogrammetry are the two primary tools employed to capture point cloud.

A laser scanner is a survey-grade system with a variety of sensors and technologies, which takes hundreds of thousands of incredibly accurate measurements per second using laser pulses. The ranges are determined by using a laser to target an object or a surface, then measuring the time it takes for the reflected light to return to the receiver. Most laser scanners also include an RGB camera to provide their color.

Laser scanners are available on the market for a wide range of specialized tasks, such as the capturing of objects, roads, or railways, as well as the creation of extensive topographic maps. To fulfill the needs of any project, you can combine various devices to create final point cloud by joining the point cloud together. In general, data from laser scanners are more accurate than those from photogrammetry.

Rather than being a specific kind of tool, photogrammetry is more of a methodology that uses cameras to capture images of the environment from all angles, then use specialist software to manipulate the images to construct a 3D registration of the scene.

2.2.2 2D drawing or 3D model

The use of blueprints as a design aid for building planning for reconstruction is nothing new. It is a technical drawing that includes lines and curves to illustrate a reconstruction design from survey and measuring the existing building such as a floor plan. It doesn't provide any additional details regarding the building property.

On the other hand, BIM is an integrated workflow that is built from coordinated and reliable information of a project from design through construction and into operation as shown in Figure 2.1. It is composed of virtual objects that represent building components. In a BIM model, components from the real world are represented by actual elements. The components are not only in three dimensions, but they also contain parameters encoded into them to define their elements' properties and functional information, such as the model number, installation date, component connectivity, and any other information that would be important to the construction.

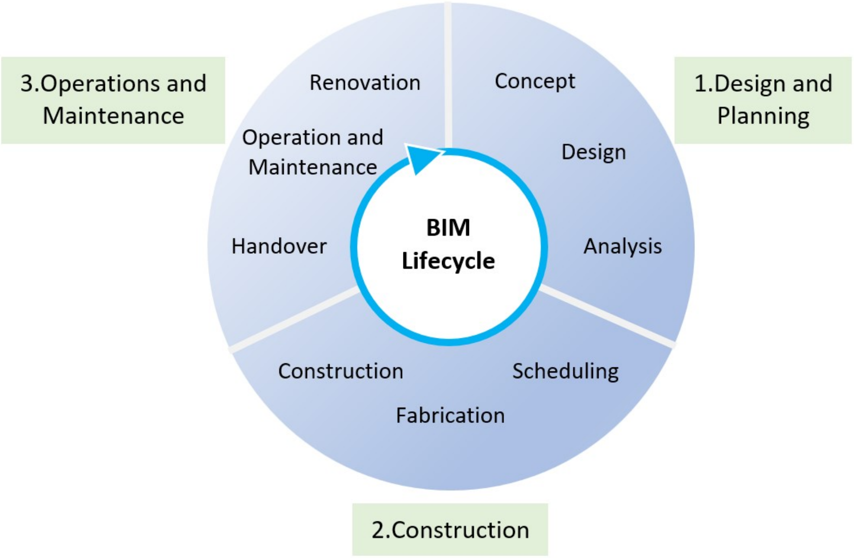


Figure 2.1: BIM lifecycle

In general, a blueprint for construction design contains the drawings from many different points of view while only one model is needed to be created when using BIM and the drawings of this model from any point of view or orientation can be generated, thus this can save time. Each element of a building is drawn once, and it can be displayed in any view whenever their visibility is enabled.

2.2.3 Scan-to-BIM

As previously described, a BIM model is a digital representation of physical and functional properties. Scan-to-BIM is a process that involves capturing high-density point cloud of a real building, building structures, or site and turning it into a digital model that can be utilized for planning.

The flexibility of using BIM allows us to record the measurements and elements of an existing building even after the construction is complete. The most common usage of a data set obtained from BIM is in the context of reconstruction or renovation. The demand for BIM implementation in the Architecture, Engineering, and Construction (AEC) industry is growing, as is the demand for the creation of BIM schematics for existing buildings. Scan-to-BIM is quickly turning into an important step in the BIM process.

The construction process documentation can provide the model with the most critical information when a project involves an existing structure or site, as it will in most situations. As a result, the process known as ”scan-to-BIM,” which involves digitally capturing a physical space or site as laser scan data and using that information to construct, develop, and maintain the BIM model, becomes valuable.

The benefits of BIM are as follows:

- Construction process documentation is often outdated, fragmented, or both. It can waste lots of time elaborately assembling the data to create one cohesive model. Using BIM can update and verify the overall direction of construction which can cooperate with several stakeholders in one model.

- BIM can keep track of what work is completed, when it is finished, and where it is completed at every stage of a project, allowing to compare progress to plan.

- This tool enables checking for errors, conflicts, and clashes in the structure. along with the planning and maintenance.

- Fast and accurate.

Therefore, the creation process is a major stage of the BIM process, and its expansion is directly proportional to the utilization of BIM in building construction, renovation, and maintenance. Another factor is that point cloud technology becomes more commonly accessible and practical, which encourages AEC professionals to consider using scan-to-BIM.

Due to the demand for BIM applications, studies, research, and development of processing chains have been increased to facilitate automated access to BIM models.

2.2.4 Voxel downsampling

The point cloud's volume and density have a direct impact on how much detail the data represent. However, if the volume of point cloud data is too big, it can affect the computation time. Thus, if it is not necessary to create a BIM model with fine detail, it is possible to reduce the volume of data without significant loss of detail using one of the methods called voxel downsampling.

In this algorithm, the input point cloud is uniformly downsampled using a regular voxel grid. The algorithm operates in two steps as follows:

1. Point cloud data are bucketed into voxels, which create a 3D voxel grid over the input point cloud data.

2. For each voxel, the coordinates of all point cloud data that are located within it will be replaced with their centroid as shown in Figure 2.2.

The sampling size can be adjusted by setting the voxel size along each dimension, generally with a fixed size, depending on the demand on the level of detail to

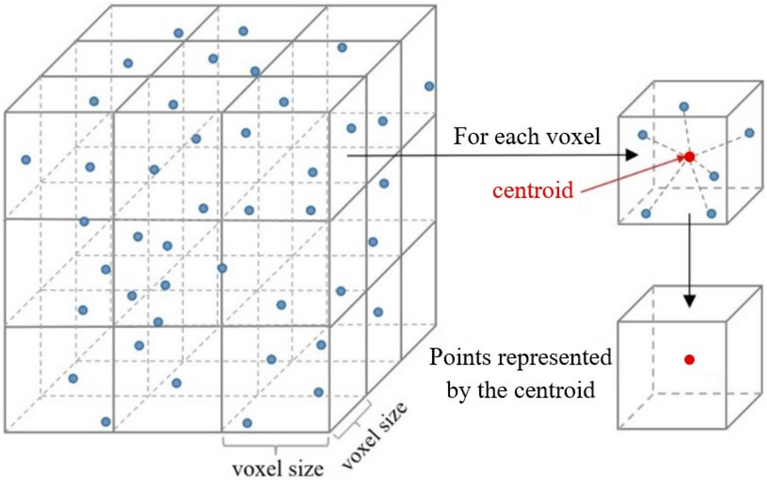


Figure 2.2: Voxel downsampling

be preserved while minimizing processing time. A smaller voxel size retains more information about the original point cloud data but requires more computation time than a larger voxel size.

Although it is a little slower than replacing the coordinates of point cloud data with the voxel's center, this technique represents the underlying surface more accurately and also has a decent cloud distribution.

Note that when we employ a detection method that takes into account the neighborhood or the density of data in a particular location, the usage of voxel downsampling can help reduce the density bias.

2.2.5 Normal estimation

The orientation of surfaces, which can be utilized to determine whether surfaces in the nearby region are morphologically related, is the information that could be used later. This portion of data is typically not given. Consequently, it is necessary to estimate the normal which will be used as one of the features for further segmentation.

One of the methods for estimating the normal at any point on a surface is to use its neighbors to predict its behavior or orientation. In this thesis, the covariance analysis is used to assess the principal axis of each point. This estimation requires the sphere radius of neighbors and the maximum number of nearest neighbors. The estimated normal is then obtained from equations (2.1) – (2.4):

$$C = \frac{1}{k} \sum_{i=1}^{k} (p_i - \bar{p}) \cdot (p_i - \bar{p})^T , \tag{2.1}$$

$$C \cdot \vec{v_j} = \lambda_j \cdot \vec{v_j}, \tag{2.2}$$

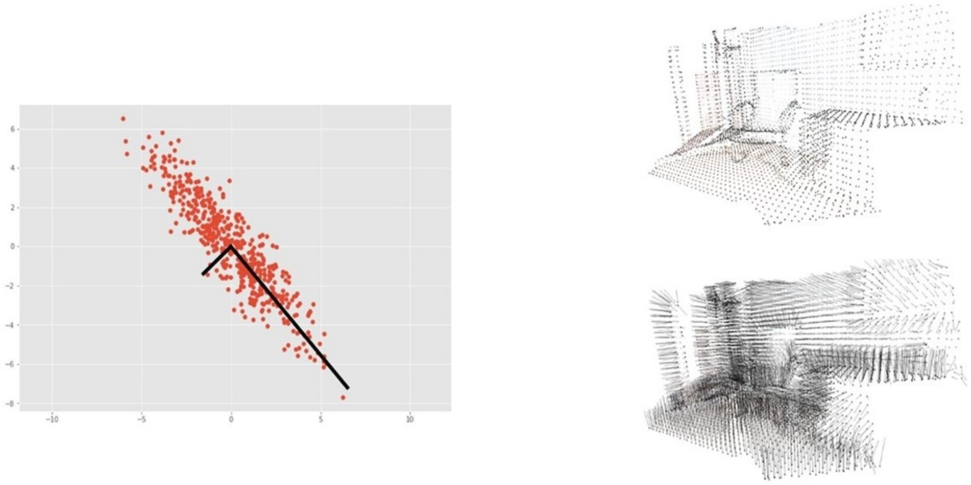
where C is the covariance matrix, k is the number of neighboring points, $\bar{p}$ is the centroid of the nearest neighbors, p_i is the point within the neighborhood of $\bar{p}$, the superscript T denotes the transpose of a matrix, $\vec{v_j}$ is the j-th eigenvector of the covariance matrix, and λ_j is the j-th eigenvalue of the covariance matrix. The eigenvalues can be solved from the characteristic polynomial in equation (2.3):

$$F(\lambda) = det\,(C - \lambda I) = 0, \tag{2.3}$$

where I is the identity matrix. After obtaining the eigenvalues, each eigenvalue is used to identify the corresponding eigenvector from equation (2.2) to get

$$(C - \lambda_j I) \cdot \vec{v_j} = 0. \tag{2.4}$$

After the eigenvalue λ_j and the eigenvector $\vec{v_j}$ are obtained through principal component analysis, the normal for each point is represented by its eigenvector that belongs to the smallest eigenvalue as shown in Figure 2.3.



(a) The eigenvectors of the covariance matrix **(b)** Point cloud data and their normal

Figure 2.3: Normal estimation

2.2.6 RANSAC

Random sample consensus algorithm (RANSAC) is an algorithm used to detect common geometric primitives, such as straight lines in two dimensions, and the plane in three dimensions by determining the score over the data that contains inliers and outliers. The algorithm is iterative and uses random samples to estimate the parameters and generate the candidate plane respectively. It is simple but works even when observing data are associated with noise or outliers.

The score is determined by counting how many point cloud data belong to each candidate plane using a specific distance threshold value. In general, the plane should only contain inliers, but since the random sample might be contaminated, the resample of the data is required for some specified number of trials to ensure that there are no outliers. The model with the highest score will finally be used to estimate the plane's parameters that represent these data as shown in Figure 2.4.

The algorithm is based on the probability that the random sample will cause

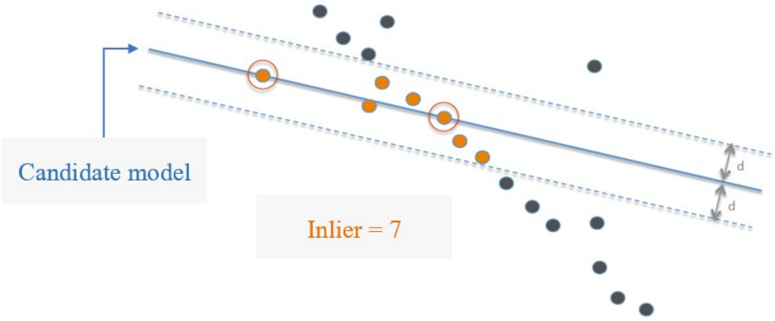


Figure 2.4: RANSAC algorithm line fitting

the sample containing only inliers be found, usually setting the probability to greater than 0.90. The number of random samples or trials is calculated by

$$T = \frac{1 - log\left(1 - \alpha\right)}{log\left(1 - \left(1 - e\right)^{s}\right)}, \tag{2.5}$$

by deriving equation (2.6)

$$\alpha = 1 - \left(1 - \left(1 - e\right)^{s}\right)^{T}, \tag{2.6}$$

where e is the probability that a point is an outlier,

s is the number of points in a sample,

T is the number of trials,

α is the desired probability that a sample contain only inliers.

In this thesis, the number of points in a sample, s, is set to 3 points for the detection of planar structure. The desired probability, α, is set to 0.90, and $1-e$ is determined by the number of points in the smallest plane divided by the number of points in a data set. After that, the number of trials T can be calculated.

Equation (2.5) states that the number of trials strongly depends on a variable e which comes from the proportion of the outlier and the whole data. For the planar structure detection, the RANSAC algorithm can be performed according to Figure 2.5.

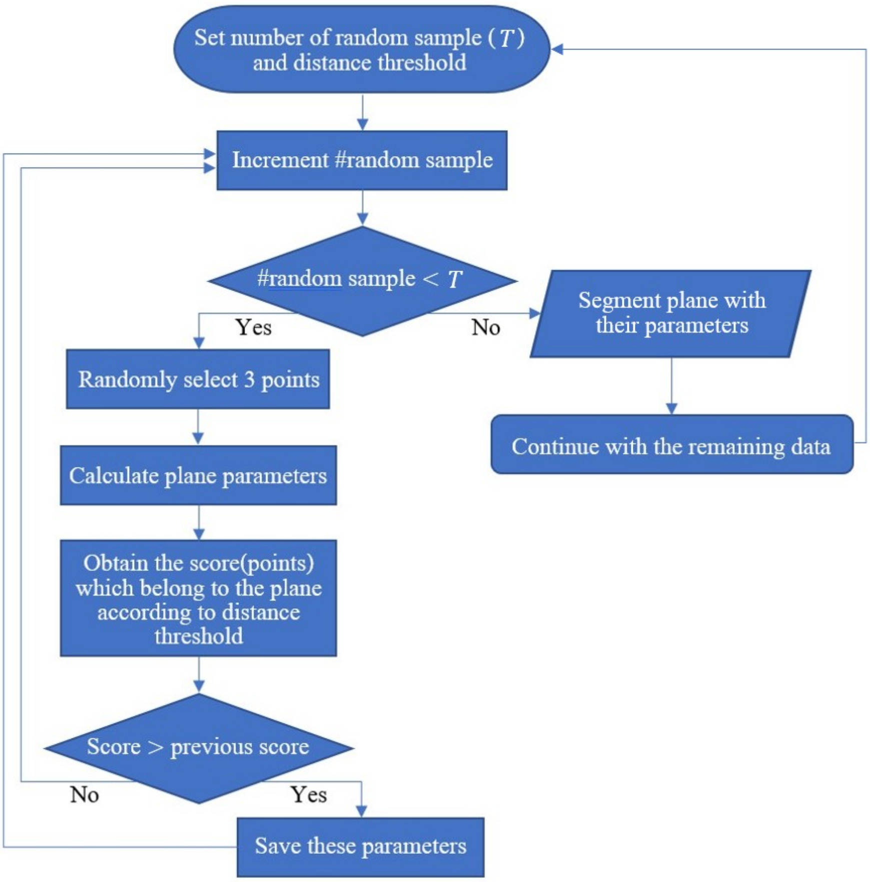

Set number of random sample (T) and distance threshold
Increment #random sample
#random sample < T
Yes
No
Segment plane with their parameters
Continue with the remaining data
Randomly select 3 points
Calculate plane parameters
Obtain the score(points) which belong to the plane according to distance threshold
Score > previous score
No
Yes
Save these parameters

Figure 2.5: RANSAC algorithm flow chart

The algorithm starts by randomly choosing 3 points as the initial, then generating a plane model from those points, and determining which points belong to a model using the distance from the plane model to those points, repeating until the trials meet the value we have set or until no more plane segment are left.

2.2.7 DBSCAN

A density-based algorithm for discovering clusters in large spatial databases with noise (DBSCAN) is a density-based clustering algorithm that can detect clusters of different sizes and shapes from a large volume of data with noise. It works by detecting areas where data points gather densely and separated by areas that are sparse.

There are some definitions that are related to DBSCAN as follows:

Definition 2.1. (The core point condition). The *Eps*-neighborhood of point p in a database D is written by $N_{Eps}(p)$, defined by

1) $N_{Eps}(p) = \{q \in D \mid dist\,(p, q) \leq Eps\}$,

2) $|N_{Eps}(p)| \geq MinPts.$

A point p must have a minimum number of neighbor points ($MinPts$) in an *Eps*-neighborhood in order to be considered as having a sufficient density. The $dist\,(p, q)$ denotes any distance function between two points p and q, such as Euclidean distance.

Definition 2.2. (Directly density-reachable). Point q is directly density-reachable from point p if

1) $q \in N_{Eps}(p)$ and

2) $|N_{Eps}(p)| \geq MinPts$; p is core point.

Definition 2.3. (Density reachable). Point q is density reachable from point p if there exists a sequence of $p_1, ..., p_n$ where $p_1 = p, p_n = q$ such that p_{i+1} is directly density reachable from $p_i; i = 1, ..., n - 1$.

An expansion of the points by a sequence of points with sufficient density is explained by Definitions 2.2 and 2.3. If there is a point that is density-reachable to any two points, we also refer to those points as being density-connected.

A cluster refers to a non-empty subset of points that are density-connected to each other. And border points are points that are part of a cluster, but not a core point.

Definition 2.4. (Noise). Given $C_1, ..., C_k$ denote the clusters of data. Then, noise is the set of points in data that do not belong to any cluster $C_i; i = 1, ..., k$, written by

$$\text{noise} = \{p \in D \mid \forall i : p \notin C_i\}.$$

Then, the concept of DBSCAN clustering is illustrated in Figure 2.6.

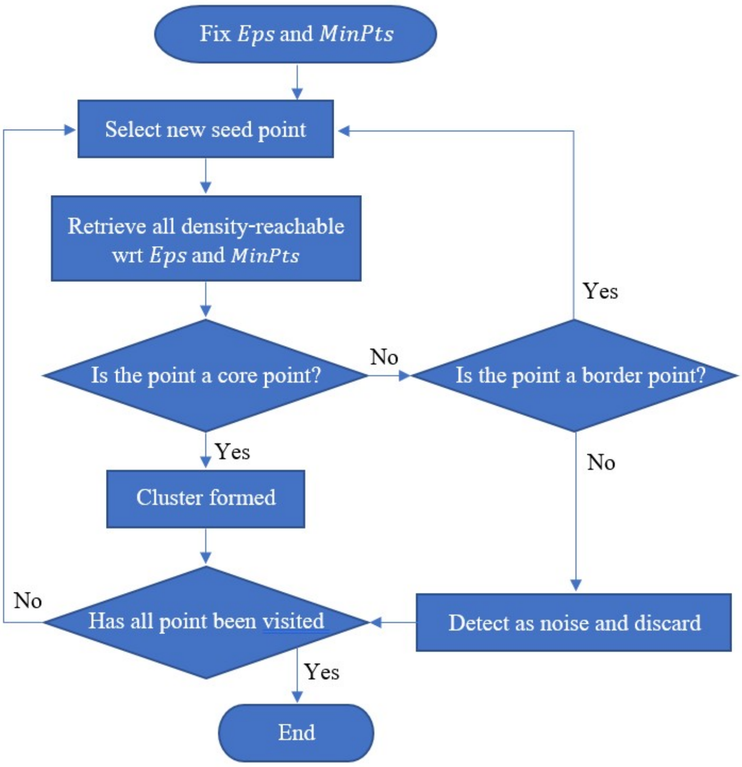

Fix *Eps* and *MinPts*
Select new seed point
Retrieve all density-reachable wrt *Eps* and *MinPts*
Is the point a core point?
No
Is the point a border point?
Yes
Yes
No
Cluster formed
Has all point been visited
No
Detect as noise and discard
Yes
End

Figure 2.6: DBSCAN algorithm flow chart

As previously mentioned, the algorithm has the ability to segment data in a variety of shapes, with the required parameters including Eps and $MinPts$. The algorithm starts by randomly select an initial point as a seed, then retrieving all density-reachable points from the seed with respect to Eps and $MinPts$. If the seed is identified as a core point, then a cluster is formed. On the other hand, if the seed is identified as a border point, the algorithm return to randomly select a new seed that has not been visited. These procedures repeat until all points have been processed.

2.2.8 Normal deviation condition

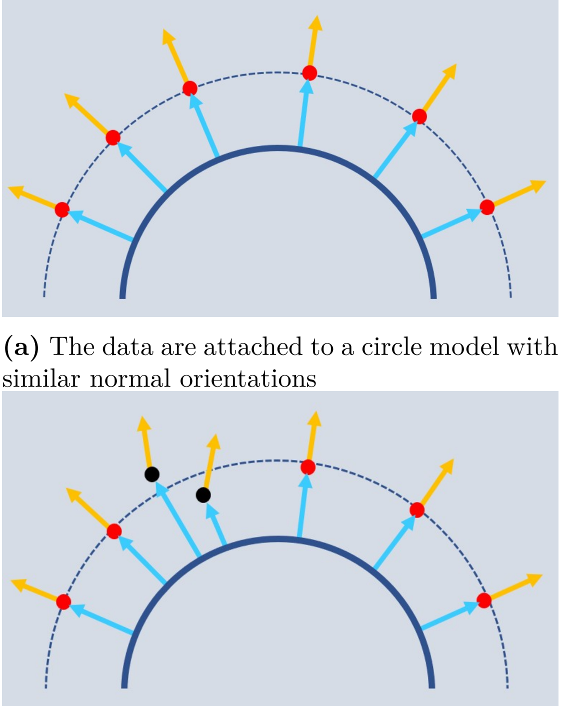
In segmentation tasks, shape detection can be improved through the use of normal features or the orientation of data to detect the shape we are interested in. For example, the normal of a pipe system can indicate its direction [15]. Moreover, it can be seen that in the nearby area of the same object, there should be similar orientations of surfaces, thus the normal pattern is the information that can be helpful in selecting a set of points that belong to a given primitive.

The normal of points that have the potential to be part of an object represented by geometric primitive, should not deviate from each other by some angle, i.e., any point p that follows the normal pattern, is denoted by

$$arccos\left(\left|n\left(p\right)\cdot n\left(\omega,p\right)\right|\right)<\alpha, \tag{2.7}$$

where $n(p)$ is the normal of point, p, and $n(\omega, p)$ is the normal of shape, ω, in the same perspective or from p projection onto the shape, and α is a specified angle threshold.

Figure 2.7 shows an example of normal patterns that consists of a circle model with the candidate model represented in dash line, the red points represent the data together with their normal, and the normal from the model at the projection.



(a) The data are attached to a circle model with similar normal orientations

(b) The normal of black points deviate from the model which may consider as not belonging to the model

Figure 2.7: Circle primitive, data points and their normal

This chapter discussed the related research in the field of point cloud reconstruction, and also described background knowledge regarding point cloud, capture tools, the use of point cloud in BIM scene, and the techniques used in this research such as voxel downsampling, normal estimation, RANSAC, and DBSCAN. The next chapter will explain the proposed methodology, which involves the preparation of input data and the segmentation of floor, ceiling, room, and wall.

CHAPTER III

METHODOLOGY

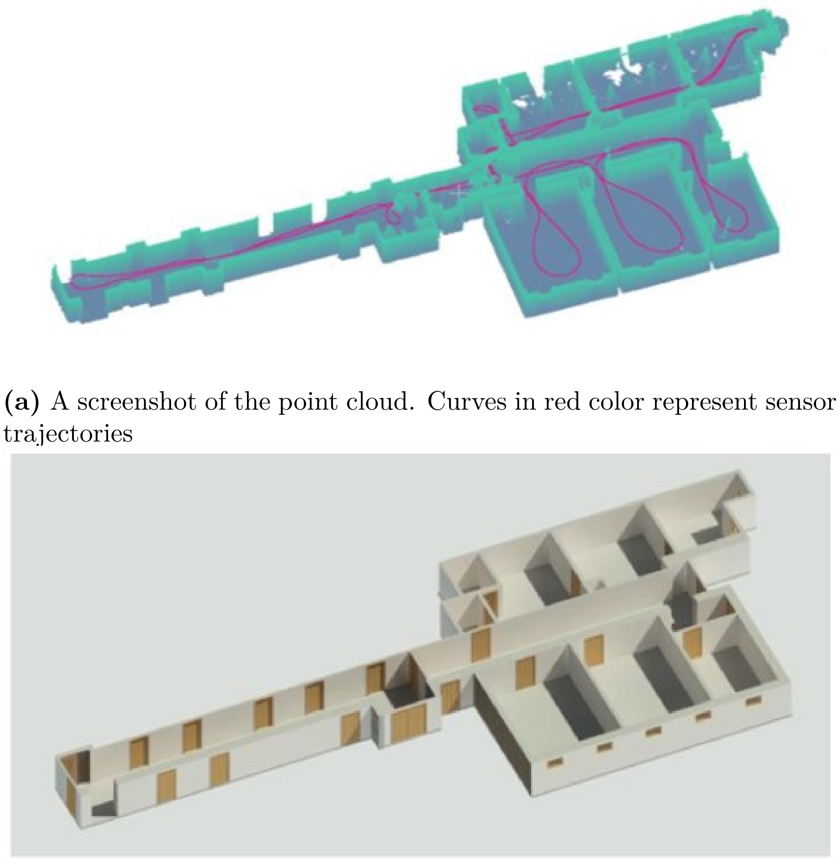
The input data and the proposed method are explained in this chapter. The proposed method for structural component segmentation is divided into two parts consisting of preprocessing, and segmentation.

3.1 Input data

Using the International Society for Photogrammetry and Remote Sensing (ISPRS) [16] benchmark on indoor modelling point cloud dataset, the proposed method was assessed and compared with the original RANSAC method using the data from one of the buildings of the Technische Universität Braunschweig in Germany captured by the Viametris iMS3D system that can be seen in Figure 3.1(a). The data are stored in two distinct files: the point cloud data and the trajectory. The scene, which has a size of approximately 33.6×10^6 points with no color, is composed of several rooms on one floor that are enclosed by walls of different thicknesses. There is a corridor that links the rooms, but no stairs. Several doors and windows, either open or closed, are also included in the scene. The level of clutter, which refers to elements other than structural components, is low and mostly corresponds to the presence of people during the survey. The average point spacing in this dataset is 0.005 meters (m). The reference model of the same dataset was manually created by Autodesk RevitTM software as shown in Figure 3.1(b).

3.2 The proposed method

The proposed method for structural component segmentation is divided into two parts: preprocessing and segmentation as shown in Figure 3.2.



(a) A screenshot of the point cloud. Curves in red color represent sensor trajectories

(b) The reference model

Figure 3.1: ISPRS benchmark dataset

3.2.1 Preprocessing

In order to enhance the performance of subsequent operations, this preliminary process is required to manipulate because the quality or characteristics of data vary depending on the type of capture devices.

3.2.1.1 Downsampling

The first step of preprocessing is downsampling. This step can reduce redundant computation caused by a large volume of point cloud data. Data must first be downsampled using the voxel downsampling technique. To reduce the number of points from the input, the downsampling size is set to 0.05 m because the ma-

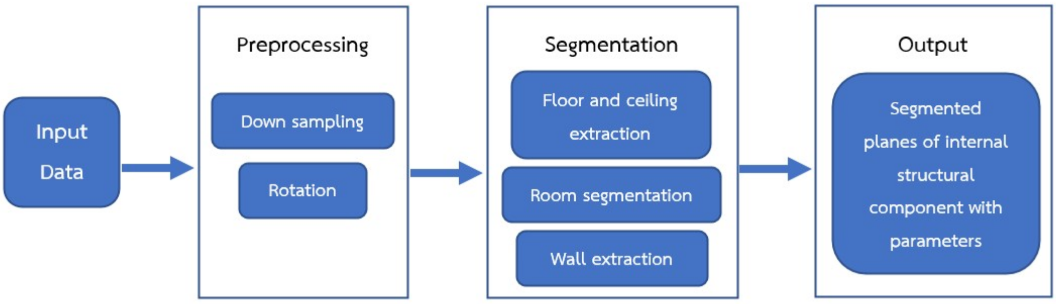
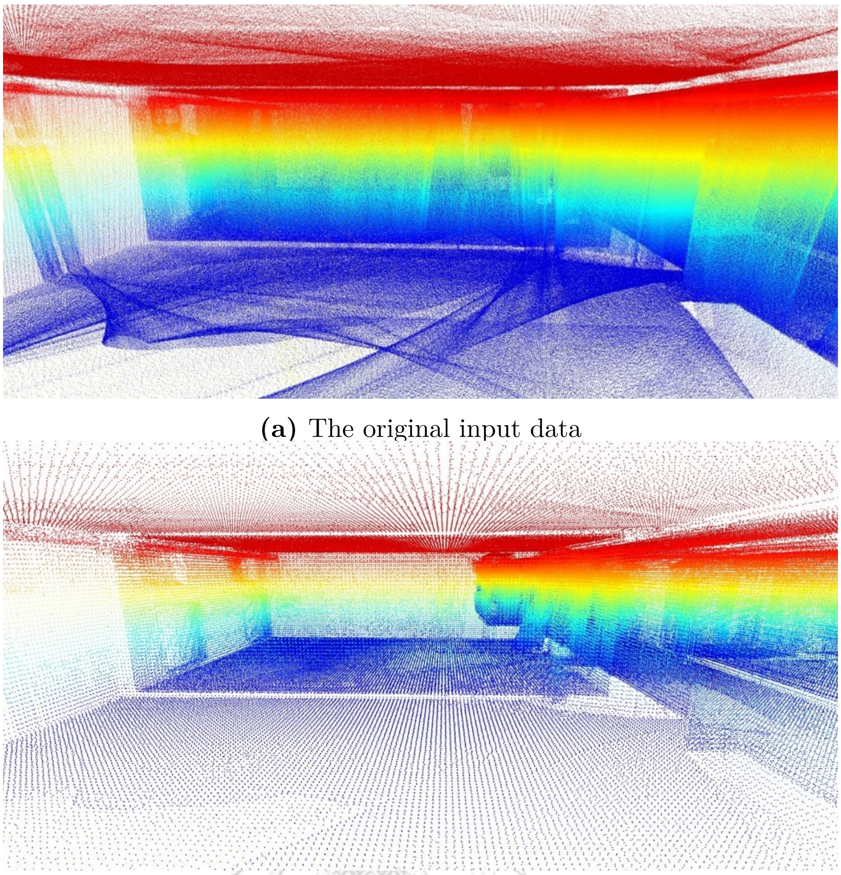


Figure 3.2: Overview of the proposed method

jority of structural components such as walls, ceilings, and floors are thicker than this value. Although a bigger downsampling size can yield faster processing, it can also cause a wall that is thinner than the downsampling size to disappear. This technique can maintain the details of data while reducing unnecessary computation and the density bias resulting in data collection and registration process. An example of downsampling result is shown in Figure 3.3.

3.2.1.2 Normal Estimation and Rotation

The rotation to align the orientation of point cloud data is performed to facilitate further process for some building datasets that do not follow the law of gravity. The Manhattan World assumption [17] describes that most of the real-world buildings' structure position can be approximated by planar surfaces that are parallel to one of the three principal planes as shown in Figure 3.4(a). To align the orientation of point cloud data, first, the angle of rotation can be estimated from the normal surface of a floor or a ceiling. Since a ceiling or a floor covers the largest area of the building's surface, it must be represented by a big group of points with a similar orientation. Therefore, the original orientation of the building can be estimated based on the dominant direction of the surface normal of these points. The normal represented by the principal axis of each point on a surface is calculated using covariance analysis from its neighbors in a sphere with



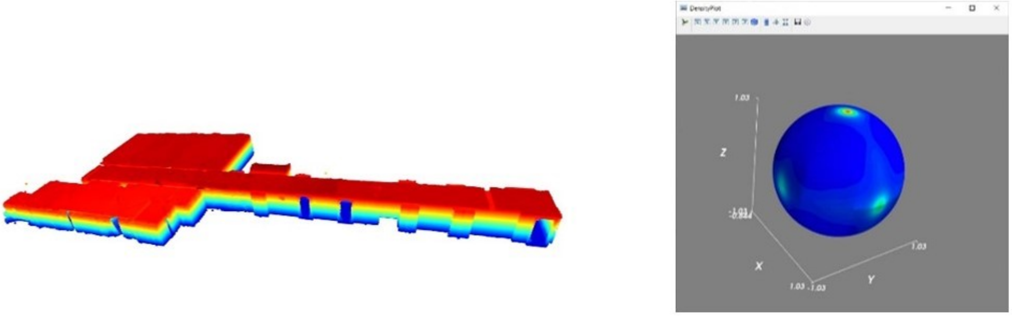
(a) The original input data

(b) Downsampled data

Figure 3.3: Downsampling

a radius set to 0.10 m and a maximum number of nearest neighbors set to 12 points. The sphere size should not be too large because it can make the normal to be imprecise. The sphere size is set equal to two times the downsampling size, since the previous step guaranteed that there would be at least one point in a voxel of size 0.05 m. The number of neighbors is limited to 12 points if the sphere size considering the planar structure is 0.10 m.

Following the estimation of the normal for each data point, the normal is collected, the k-means algorithm [18] is then applied to divide the collection of normal into clusters, and the orientation of the building is assumed to be in the same direction as the dominant direction of the biggest cluster as shown in Figure 3.4(b). The rotation is then performed corresponding to the angle between the standard orientation and the building orientation to obtain data considered as the Manhattan frame.



(a) Point cloud data of the indoor environment of the building from ISPRS Benchmark datasets

(b) Density plot of their normal vectors

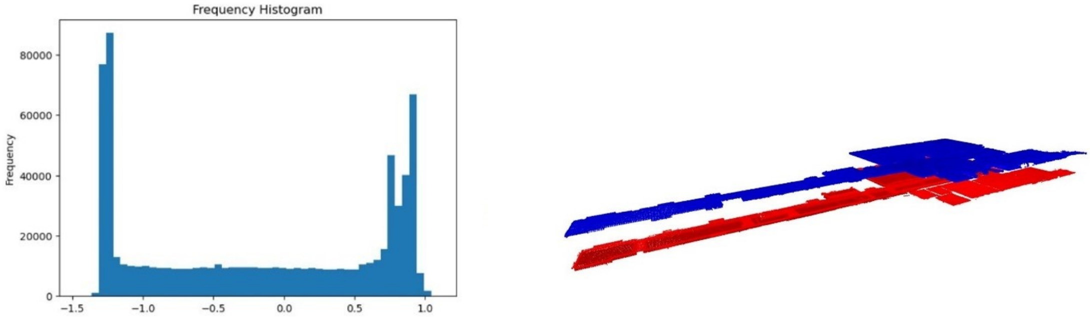
Figure 3.4: Point cloud data and their estimated normal vectors

3.2.2 Segmentation

The segmentation is divided into floor and ceiling extraction, room segmentation, and wall extraction, respectively.

3.2.2.1 Floor and ceiling extraction

The characteristic of the aligned data obtained from previous step allows us to consider frequency histogram of the z-axis. The peak of a histogram at low elevation tends to represent the floor and the peak at high elevation tends to represent the ceiling which are used to extract the floor and the ceiling. The z-coordinate histogram's bin size is 0.05 m as illustrated in Figure 3.5(a). The segmentation can accurately extract the floor and the ceiling for this dataset by slicing the z-values around the peak at intervals of 0.30 m as shown in Figure 3.5(b).



(a) Frequency histogram of the z-values

(b) Extracted floor and ceiling

Figure 3.5: Extracted floor and ceiling obtained from slicing the frequency histogram

3.2.2.2 Room segmentation

Since the complexity of the RANSAC algorithm significantly depend on the size of the data in a random sampling process, the difference between the RANSAC and the proposed random sampling strategy is described below.

Consider point cloud of size N and a target candidate model, ω, consisting of n points. The probability of detecting ω in a single pass with minimal subset of s points is

$$P(n) = \binom{n}{s} / \binom{N}{s} \tag{3.1}$$

and the probability that at least one sample among k samples of s points composing of only inliers of the model is

$$P(n,k) = 1 - (1 - P(n))^k \tag{3.2}$$

Solving equation (3.2) for k to estimate the number of samples T required to detect ω of size n with a probability $P(n,T) \geq \alpha$:

$$1-(1-P(n))^{T}\geq\alpha, \tag{3.3}$$

$$1-\alpha\geq(1-P(n))^{T}, \tag{3.4}$$

$$ln(1-\alpha)\leq T\cdot ln\left(1-P(n)\right), \tag{3.5}$$

$$T\geq\frac{ln(1-\alpha)}{ln(1-P(n))} \tag{3.6}$$

where the Maclaurin series of $ln(1-P(n))$ is

$$\begin{aligned}-\sum_{i=1}^{\infty}\frac{P(n)^{n}}{n}&=-P(n)-\frac{P(n)^{2}}{2}-\frac{P(n)^{3}}{3}-\cdots, &(3.7)\\ &=-P(n)-O\left(P(n)^{2}\right). &(3.8)\end{aligned}$$

so

$$T\approx\frac{-ln(1-\alpha)}{P(n)}. \tag{3.9}$$

The complexity of the RANSAC random sampling is $O(T)=O\left(\frac{1}{P(n)}\right)$.

Considering $P(n)$

$$\begin{aligned}P(n)&=\binom{n}{s}/\binom{N}{s}, &(3.10)\\ &=\frac{n!}{(n-s)!s!}/\frac{N!}{(N-s)!s!}, &(3.11)\end{aligned}$$

since $(n-s)!=\frac{n!}{n(n-1)(n-2)\cdots(n-s+1)}$, as well as $(N-s)!$

$$\begin{aligned}&=\frac{n!n(n-1)\cdots(n-s+1)}{n!s!}/\frac{N!N(N-1)\cdots(N-s+1)}{N!s!}, &(3.12)\\ &=\frac{n(n-1)(n-2)\cdots(n-s+1)}{N(N-1)(N-2)\cdots(N-s+1)}, &(3.13)\\ &\approx\frac{n^{s}}{N^{s}}. &(3.14)\end{aligned}$$

thus

$$\frac{1}{P(n)} \approx \frac{N^s}{n^s} \tag{3.15}$$

Next, consider applying the idea of a localized sampling to the RANSAC algorithm in order to reduce unnecessary computation. Since the data is partitioned into subspaces, then

$$P_{local}(n) = \binom{n}{s} / \binom{\beta N}{s}, \tag{3.16}$$

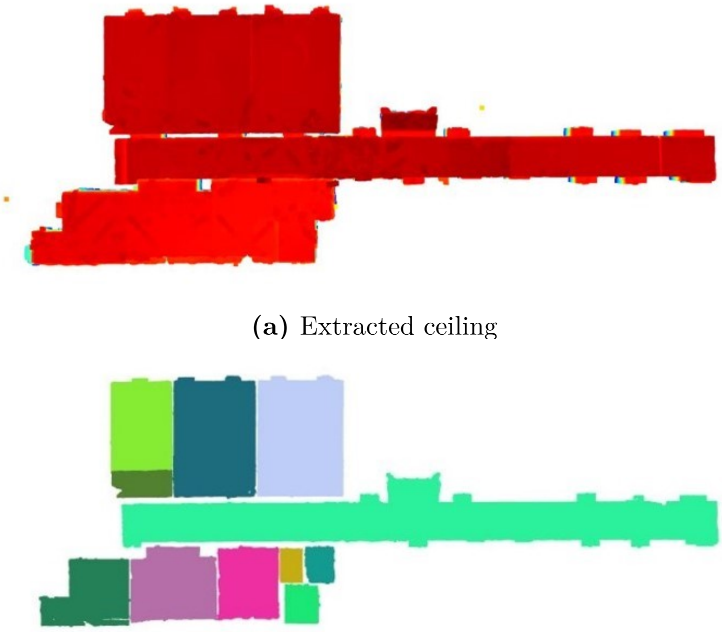
$$\approx \frac{n^s}{(\beta N)^s}, \qquad 0 < \beta \leq 1 \tag{3.17}$$

thus

$$\frac{1}{P_{local}(n)} \approx \frac{(\beta N)^s}{n^s}. \tag{3.18}$$

The difference between $\frac{1}{P(n)}$ and $\frac{1}{P_{local}(n)}$ makes the proposed method with the implementation of the localized sampling faster than the original algorithm.

As a ceiling commonly demonstrates the boundaries of each room, it is noticeable that each room is separated from one another by slight spaces as seen in Figure 3.6(a). Using the DBSCAN algorithm, the ceiling data are divided into individual rooms based on their coordinates. A cluster is an adjoining region of points with high density that is separated from other clusters by regions of points with low density. For the same reason as a normal estimation when looking at planar structures, the DBSCAN's radius is set to 0.08 m, and its minimum number of neighbors is 6. As demonstrated in Figure 3.6(b), the algorithm uses spatial information to analyze each data point and assign the room's label. Note that the connected regions must be handled manually in the case that the ceilings are still connected, which could occur if there was an error made during the data collection step.



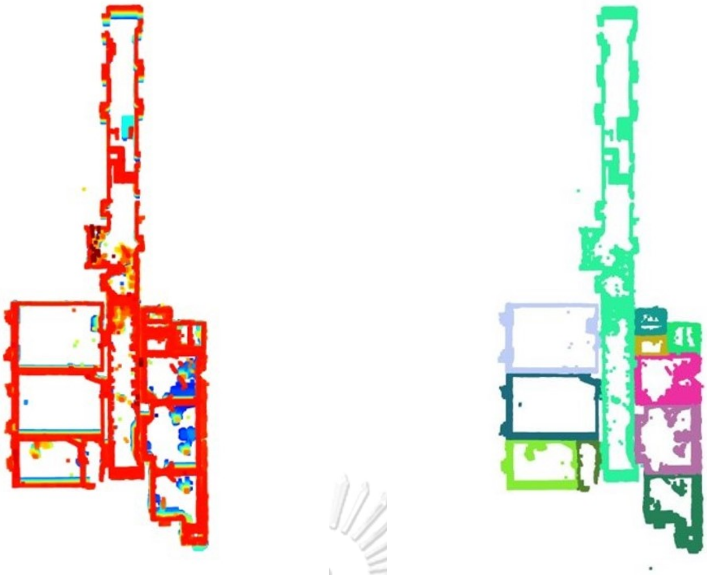
(a) Extracted ceiling

(b) The labeled ceilings as shown in different colors

Figure 3.6: The top views of the ceiling

The remaining points are then assigned the labels using information from the labeled ceiling once the ceiling, which serves as the boundary of each room, has been retrieved. Every remaining point is projected onto the ceiling plane and labeled with respect to the corresponding labeled ceiling. As a result, this process enables the utilization of localized sampling to reduce the burden on the computation. The remaining points are sampled from the room-by-room subsets as shown in Figure 3.7 rather than from all of the remaining points. This resulted in the reduction of the computational complexity by several factors.

For example, when using the probability of finding the good sample, mentioned in equation (2.5), equal to 0.9 according to the standard setting, which is typically between 0.90-0.99, a set of 350,793 points needs to be re-sampled 1,553,060,809 times, whereas a segmented room containing 59,201 points only requires 7,464,880 with the difference of more than 200 times.



(a) Remaining points (b) Labeled remaining points

Figure 3.7: Remaining point cloud after the floor and ceiling are extracted

3.2.2.3 Wall extraction

The original algorithm worked by first obtaining an initial solution using the smallest set of data before enlarging the solution with data from the entire dataset that were consistent with the initial solution. According to the implementation, the original algorithm did not yield satisfactory results for the building of indoor environment datasets. The problems are the segmentation accuracy of the extracted planes, and connectivity. RANSAC's estimation were based on scores, which may be too coarse, resulting in low plane quality. Due to lack of connectivity and high computation time on large data that consists of many objects when RANSAC random sampling was done through the entire data, as a consequence, this research proposes to modify RANSAC by using localized sampling to reduce the computational complexity and applying connected components and normal deviation conditions into score calculation part to improve the accuracy and smoothness of the detected plane which can be seen in Figure 3.8. The modified RANSAC was used to extract vertical planes representing the wall segments from the remaining data points that contain parts of the wall segments and clutter.

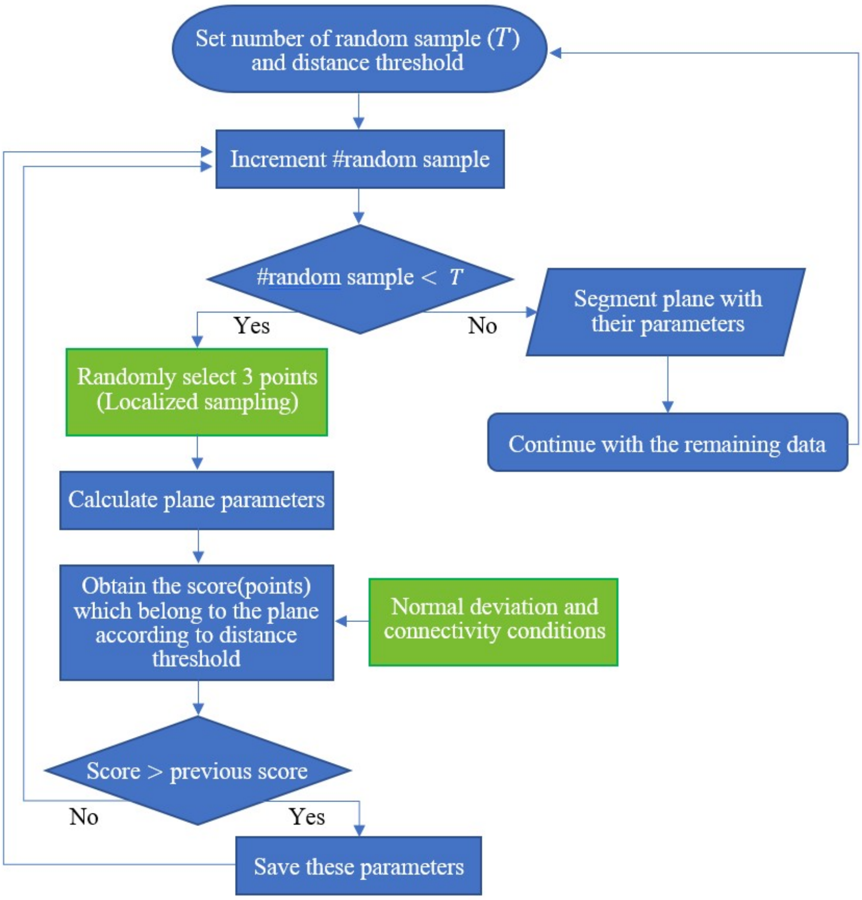

Set number of random sample (T) and distance threshold
Increment #random sample
#random sample < T
Yes
No
Segment plane with their parameters
Randomly select 3 points (Localized sampling)
Continue with the remaining data
Calculate plane parameters
Obtain the score(points) which belong to the plane according to distance threshold
Normal deviation and connectivity conditions
Score > previous score
No
Yes
Save these parameters

Figure 3.8: Algorithm flow chart

Once the threshold distance defining the inliers was set to 0.05 m and the angle between the detected plane and dominant axis was set to be less than 10 degrees, the algorithm iteratively re-sampled the minimal subset to generate the candidate walls until there was no more data point on the wall. The algorithm repeated from one room to another.

Besides using the modified RANSAC, the DBSCAN algorithm was used to analyze the connectivity of data for connected components. The algorithm used a distance radius of 0.08 m and a minimum number of neighbors of 6 points to assess if the detected plane segment was large enough to be recognized as a wall.

Another parameter is the segment size, the standard wall specification in this research requires that a wall segment's size must be at least 0.25 m × 2 m, or 200 points corresponding to the downsampling process. The normal deviation between the detected plane and the inlier point should be constrained by the specified angle threshold, which was set to 10 degrees, in order to maintain the smoothness of the detected plane.

In this chapter, the input data, the ISPRS benchmark on the indoor modelling point cloud dataset, was described. The proposed methodology is explained, starting from preprocessing, the input data is prepared by performing voxel downsampling and normal estimation with rotation. Segmentation was then described that is, floor and ceiling are extracted by considering the frequency histogram along the z-axis, the room is segmented by the projection of the remaining point onto the labeled ceiling obtained by using the DBSCAN algorithm, and the wall is extracted using the modified RANSAC. After that, the proposed method can be performed structural components segmentation to obtain floor, ceiling, and wall segments. The proposed method is applied with the default threshold values setting as shown in Table 3.1.

Description	Value
Voxel downsampling size	0.05 m
Normals search radius	0.10 m
Normals maximum nearest neighbor	12 points
Bin size of z-coordinate frequency histogram	0.05 m
Ceiling and floor slicing interval	0.30 m
RANSAC distance threshold defining the inliers	0.05 m
RANSAC minimum number of points required	200
Maximum angle between detected plane and dominant axis	10
Maximum angle between detected plane and inlier points	10
DBSCAN distance radius	0.08 m
DBSCAN minimum number of neighbors	6 points

Table 3.1: Setting thresholds involved in this thesis

To evaluate the performance of the proposed method, the original RANSAC and the proposed method were tested on the same dataset and the visual aspect of the results were compared in the next chapter.

CHAPTER IV

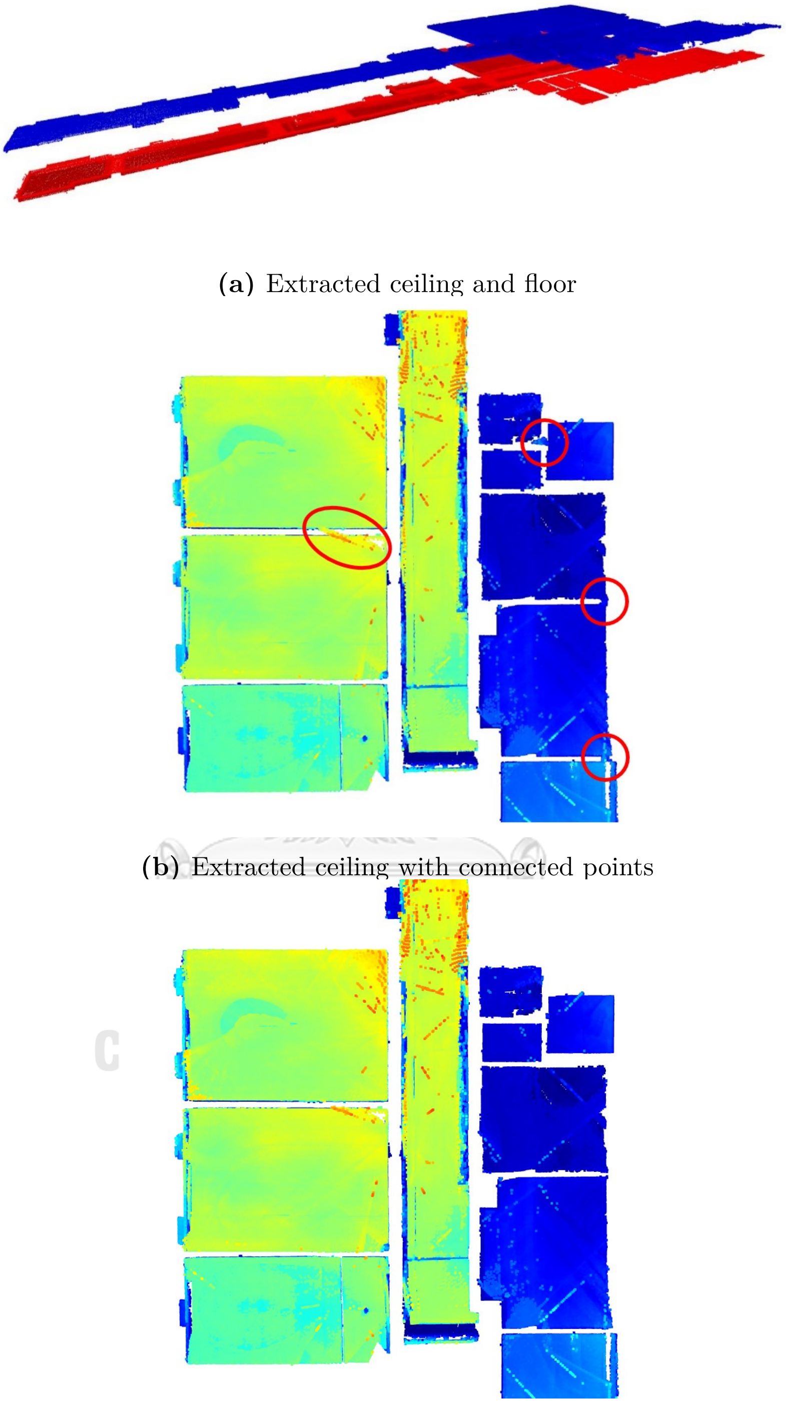
SEGMENTATION RESULTS

The experiment and results are discussed in this chapter. Results from floor and ceiling extraction are presented in the first section, followed by results from room segmentation, and results with a comparison of the wall segments from wall extraction using the original RANSAC and the proposed method are given in the last section.

The experiments were performed on Jupyter Notebook version 6.0.3. The proposed method and visualization were implemented using Open3D, an open-source library that supports rapid development of software that works with 3D data in both C++ and Python. The input data from ISPRS is a text file (.txt), and the point cloud data contain x, y, and z coordinates.

4.1 Floor and ceiling extraction results

The extracted ceiling and floor are displayed in Figure 4.1(a). However, the implementation has a defect, that is the ceilings that were subsequently used to determine the boundary for each room were still connected, which are enclosed by the red circles shown in Figure 4.1(b). These data points were caused by an error made during the data collection step, which will result in DBSCAN malfunction. Therefore, the connected points in those areas were manually removed and the result of the extracted ceiling after removing connected points is shown in Figure 4.1(c).



(a) Extracted ceiling and floor

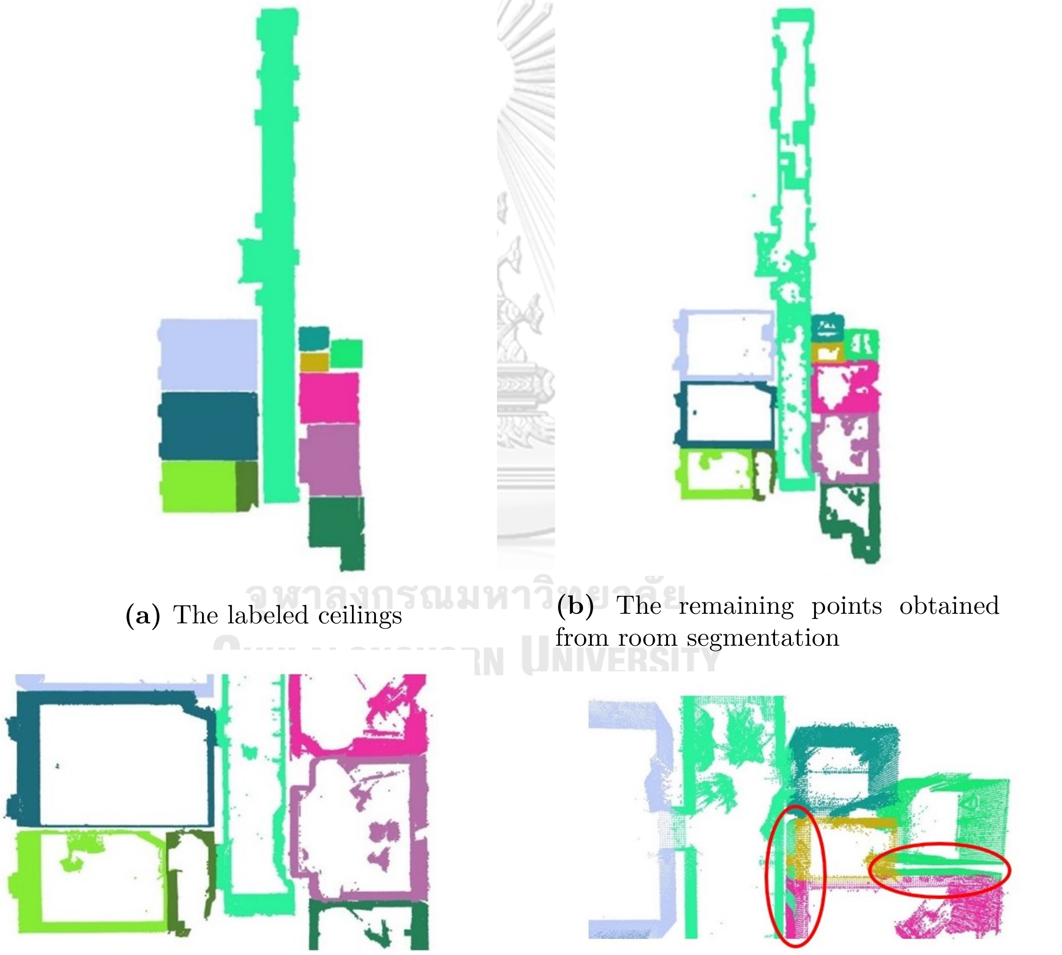
(b) Extracted ceiling with connected points

(c) Extracted ceiling after removing connected points

Figure 4.1: Extracted ceiling before and after removing connected points

4.2 Room segmentation results

The DBSCAN algorithm was used to cluster the ceiling from the extracted ceiling after removing connected points obtained from section 4.1. The room segmentation results are displayed in Figure 4.2 from the top view and the close-up. After the points on the ceiling were clustered into individual rooms with labels by color as shown in Figure 4.2(a), the room segmentation result that were obtained from the projection of the remaining points is shown in Figure 4.2(b).



(a) The labeled ceilings

(b) The remaining points obtained from room segmentation

(c) Close-up result

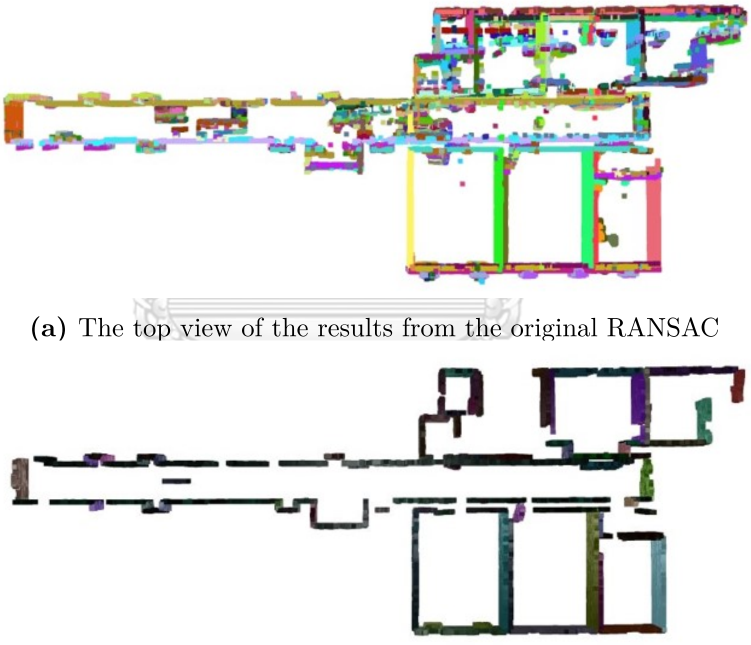
(d) Incorrect segmentation in the red circles

Figure 4.2: Screenshots of the point cloud on room segmentation

Considering the details in the close-up results, the quality of the room segmentation result is quite reliable as shown in Figure 4.2(c); however, there are still some errors from segmentation as shown in Figure 4.2(d), where some parts of the extracted ceilings did not completely cover the rooms, leading to mislabeling in that section.

4.3 Wall extraction results and the comparison

The modified RANSAC algorithm with localized sampling was then applied to extract the walls from the remaining points after they had been partitioned into subspaces. Figures 4.3 - 4.5 illustrate the results and the comparison between the original RANSAC and the proposed method from the same perspective, where different colors represent different detected segments.

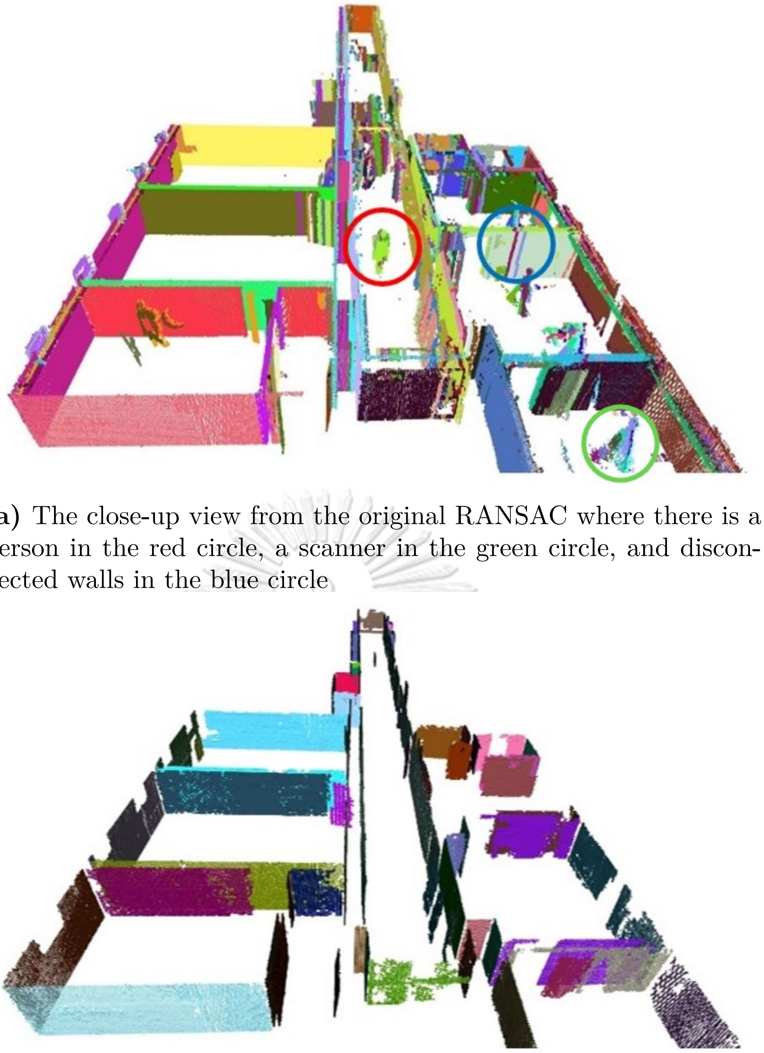


(a) The top view of the results from the original RANSAC

(b) The top view of the results from the proposed method

Figure 4.3: Results of wall extraction from the original RANSAC and the proposed method

The results from Figures 4.3(a), 4.4(a), and 4.5(a) were extracted using the original RANSAC, and the results from Figures 4.3(b), 4.4(b), and 4.5(b) were obtained from extraction using the modified RANSAC. Overall, it can be



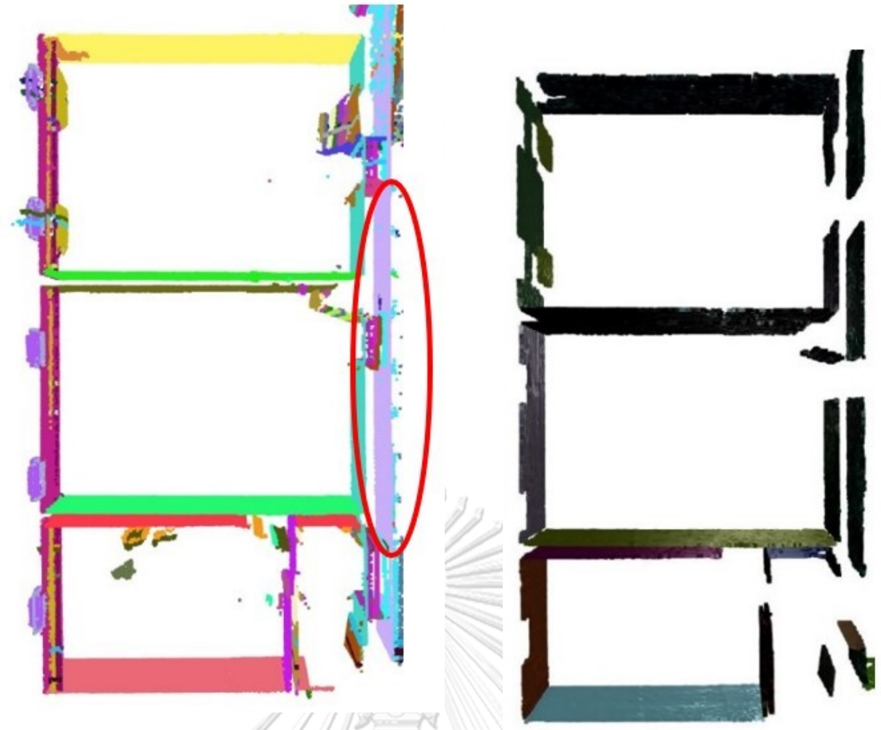
(a) The close-up view from the original RANSAC where there is a person in the red circle, a scanner in the green circle, and disconnected walls in the blue circle

(b) The close-up view from the proposed method

Figure 4.4: Results of wall extraction from the original RANSAC and the proposed method

noticed that the original RANSAC algorithm broke up the walls into fragments and could not clearly distinguished the walls from the clutter. On the other hand, the proposed method can accurately segment the walls.

The results of the original RANSAC demonstrate that RANSAC used only scores to determine the walls in Figures 4.3(a), 4.4(a), and 4.5(a). This has an effect on segmentation of the indoor scene because the scene usually contains non-structural objects. Thus, the segmented walls, as illustrated in Figure 4.4(a),



(a) The close-up view from the original RANSAC where the red circle shows the detected wall with no connectivity

(b) The close-up view from the proposed method

Figure 4.5: Results of wall extraction from the original RANSAC and the proposed method

also associate the clutter of a scanner and a person. Moreover, due to a lack of connectivity constraint, the segmented walls may compose of small planes or various irrelevant plane components. Moreover, Figure 4.4(a) shows that the walls were broken into fragments as each wall is displayed in many different colors. For instance, the original RANSAC determined that the purple wall strip in the blue circle located in-between the grey wall is connected to another purple wall even though they are not actually connected, but instead it should belong to the grey wall. On the other hand, the modified RANSAC can solve this problem as shown in Figure 4.4(b) where the wall is displayed by only one color in purple. Figure 4.5(b) shows the walls that were acquired using the modified RANSAC, which eliminated all disconnected components from the result, in contrast to Figure 4.5(a), which shows the segmented walls that are not connected in the red circle.

The results from modified RANSAC obviously demonstrate that the method can distinguish the walls from the clutters, which makes the results more trustworthy. Additionally, it can handle the problem of disconnected walls having a similar orientation as seen in Figure 4.4(a). Including the normal deviation and connectivity constraints to the original algorithm make the proposed method to segment the walls appropriately.

CHAPTER V

CONCLUSIONS AND FUTURE WORK

This thesis presents a robust segmentation method using DBSCAN and modified RANSAC with normal deviation conditions to detect planar structural components for the indoor environment of a building from point cloud data. This chapter discusses the conclusion of the thesis and provide some possibilities about future work.

5.1 Conclusions

Chapter I discusses the significance of the preliminary process of a building reconstruction that uses point cloud data and BIM to create 3D models for design and planning through the reconstruction. Moreover, this thesis intends to improve and develop the automatic structural component segmentation method for the indoor environment data of the building. Related research and background knowledge are explained in Chapter II. Some explanations on point cloud, scanner technology, and the use of BIM are also included. Point cloud processing techniques used in this thesis, including voxel downsampling, normal estimation, RANSAC and DBSCAN algorithms are explained, along with the concept of normal deviation. The RANSAC algorithm, that is used to detect the planar structure, needs to be modified for better performance in building reconstruction from point cloud. Therefore, the modified RANSAC is proposed for indoor environment data. Chapter III presents the details regarding input data and the proposed method. The proposed method starts with voxel downsampling, normal estimation, and rotation to enhance the input data for segmentation. The proposed segmentation is divided into two main stages: floor and ceiling are extracted by slicing at the peaks of the frequency histogram of the data along the z-axis, the rooms are partitioned into subspaces by using DBSCAN to identify their boundary on the ceiling, and walls are finally extracted by using modified RANSAC. Since segmentation by the original RANSAC yields low quality, lack of connectivity and requires high computation time, the RANSAC algorithm is modified. The computational performance is enhanced through random sampling in subspace. The score calculation of the algorithm is modified to also include connectivity constraints and normal deviation to improve segmentation accuracy. In chapter IV, the segmentation results using the proposed method are discussed and compared with those obtained from using the original RANSAC. The proposed method can reduce the computational complexity by partitioning the whole floor into subspaces, then performing localized sampling, which helps the RANSAC algorithm to enable faster detection. The segmentation is improved with the help of the normal deviation and connectivity constraints added to RANSAC's score calculation. It can separate the clutter from the walls and detect connected walls. However, the algorithm is unable to identify walls that are not smooth or flat, which may be encountered in typical situations where curtains may be included.

The results from this research consist of point cloud of the segmented plane with their parameters which can be used to construct a floor layout by considering the intersection of the extracted walls that are projected onto the horizontal plane and the height can be estimated by the extracted floor and ceiling elevation. Moreover, these data can be used in 3D model generation by BIM extensible software in the future.

5.2 Future work

Since the segmentation contains several processes in the framework that integrate various fields of study, there are some suggestions that may be considered to be future work to complete the automatic algorithm that covers all processes.

- Solving manual processes such as room segmentation that remain in this thesis in order to reduce interaction of a user during the process.
- The further study of related research is needed for the development of the algorithm's limitations.
- The study of translation techniques to automatically transform the results obtained from this thesis into the BIM model.
- An analysis of the performance and the results of the algorithm are needed to clarify the quantitative accuracy.

REFERENCES

[1] N. Hichri, C. Stefani, L. D. Luca, P. Veron, and G. Hamon, “From point cloud to BIM: A survey of existing approaches,” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-5/W2, pp. 343–348, 2013.

[2] F. Tarsha-Kurdi, T. Landes, and P. Grussenmeyer, “Hough-TRANSFORM and extended RANSAC algorithms for automatic detection of 3D building roof planes from LiDAR data,” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 124—132, 2007.

[3] P. V. C. Hough, “Method and means for recognizing complex patterns,” *U.S. Patent 3.069.654*, 1962.

[4] M. A. Fischler and R. C. Bolles, “Random sample consensus,” *Communications of the ACM*, vol. 24, no. 6, pp. 381—395, 1981.

[5] R. Schnabel, R. Wahl, and R. Klein, “Efficient RANSAC for point-cloud shape detection,” *Computer Graphics Forum*, vol. 26, no. 2, pp. 214—226, 2007.

[6] M. Kada and F. Luo, “Generalisation of building ground plans using half-spaces,” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, no. 4, 2008.

[7] A. Budroni and J. Boehm, “Automated 3D reconstruction of interiors from point clouds,” *International Journal of Architectural Computing*, vol. 8, no. 1, pp. 55—73, 2010.

[8] F. Capocchiano, R. Ravanelli, and M. Crespi, “A tool for crowdsourced building information modeling throughlow-cost range camera: Preliminary demonstration and potential,” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-2/W8, pp. 75—81, 2017.

[9] F. Capocchiano and R. Ravanelli, “An original algorithm for BIM generation from indoor survey point clouds,” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-2/ W13, pp. 769—776, 2019.

[10] H. Macher, T. Landes, and P. Grussenmeyer, “Point clouds segmentation as base for as-built BIM creation,” *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. II-5/W3, pp. 191—197, 2015.

[11] H. Macher, T. Landes, and P. Grussenmeyer, “From point clouds to building information models: 3D semi-automatic reconstruction of indoors of existing buildings,” *Applied Sciences*, vol. 7, no. 10, p. 1030, 2017.

[12] S. Murali, P. Speciale, M. R. Oswald, and M. Pollefeys, “Indoor Scan2BIM: Building information models of house interiors,” *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[13] Y. Cui, Q. Li, B. Yang, W. Xiao, C. Chen, and Z. Dong, “Automatic 3-D reconstruction of indoor environment with mobile laser scanning point clouds,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 8, pp. 3117—3130, 2019.

[14] M. Ester, H. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” *Knowledge Discovery and Data Mining*, vol. 96, no. 34, pp. 226—231, 1996.

[15] G. VOSSELMAN, B. GORTE, G. SITHOLE, and T. RABBANI, “Recognising structure in laser scanner point clouds,” *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 46*, vol. 8/ W2, pp. 33—38, 2004.

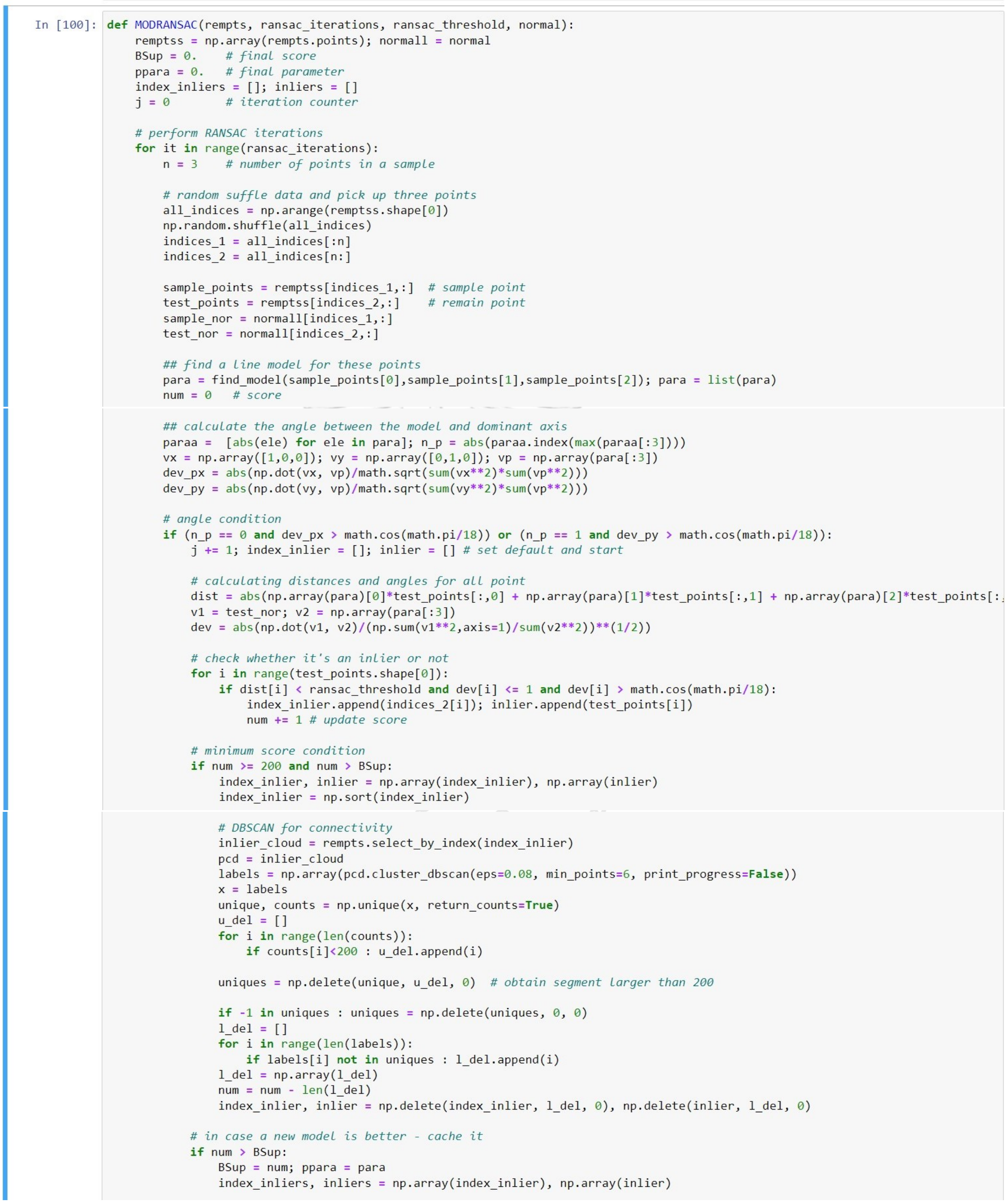
[16] K. Khoshelham, L. D. Vilariño, M. Peter, Z. Kang, and D. Acharya, “The ISPRS benchmark on indoor modelling,” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-2/ W7, pp. 367—372, 2017.

[17] J. M. Coughlan and A. L. Yuille, “Manhattan world: Compass direction from a single image by Bayesian inference,” *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, pp. 941–947, 1999.

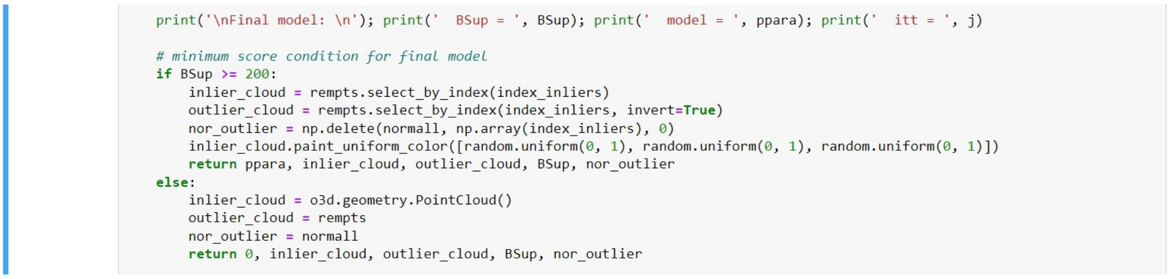
[18] J. B. MacQueen, “Some methods for classification and analysis of multivariate observations,” *Proceedings of the 5th Berkeley Symposium on Mathematics Statistics and Probability*, pp. 281—297, 1967.

APPENDIX

APPENDIX A : The modified RANSAC code for plane segmentation in python.

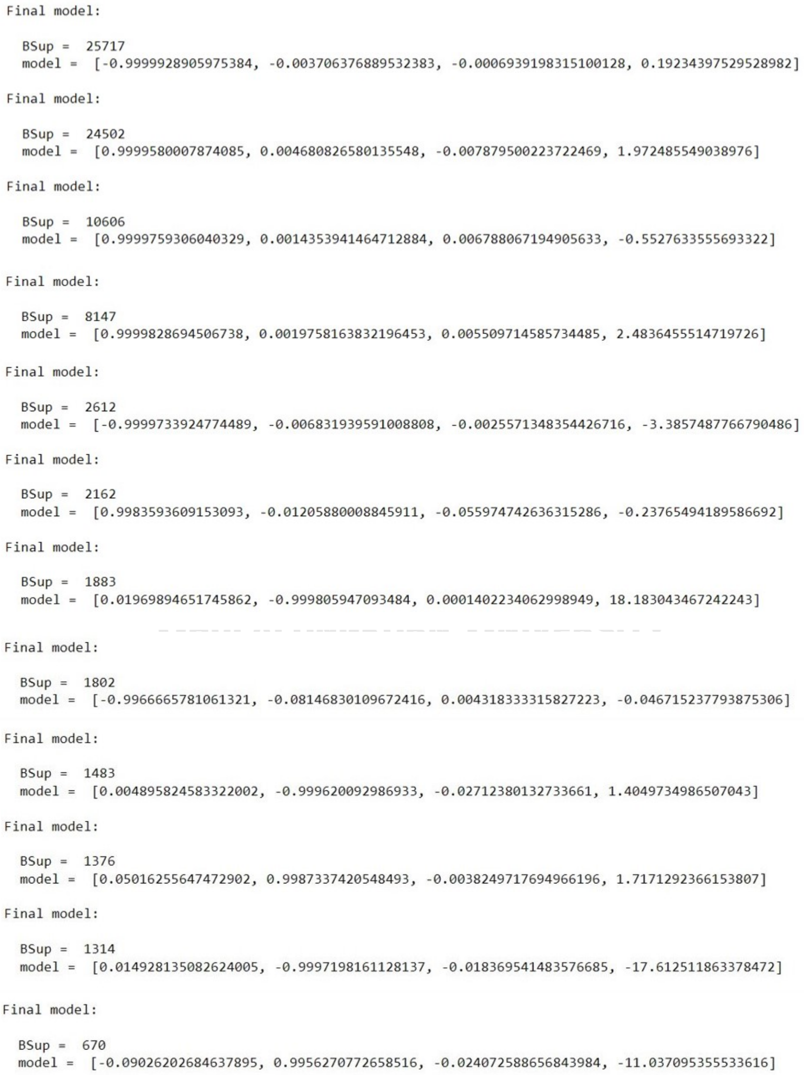


```
In [100]: def MODRANSAC(rempts, ransac_iterations, ransac_threshold, normal):
              remptss = np.array(rempts.points); normall = normal
              BSup = 0.    # final score
              ppara = 0.   # final parameter
              index_inliers = []; inliers = []
              j = 0        # iteration counter

              # perform RANSAC iterations
              for it in range(ransac_iterations):
                  n = 3    # number of points in a sample

                  # random suffle data and pick up three points
                  all_indices = np.arange(remptss.shape[0])
                  np.random.shuffle(all_indices)
                  indices_1 = all_indices[:n]
                  indices_2 = all_indices[n:]

                  sample_points = remptss[indices_1,:]  # sample point
                  test_points = remptss[indices_2,:]    # remain point
                  sample_nor = normall[indices_1,:]
                  test_nor = normall[indices_2,:]

                  ## find a line model for these points
                  para = find_model(sample_points[0],sample_points[1],sample_points[2]); para = list(para)
                  num = 0   # score

                  ## calculate the angle between the model and dominant axis
                  paraa =  [abs(ele) for ele in para]; n_p = abs(paraa.index(max(paraa[:3])))
                  vx = np.array([1,0,0]); vy = np.array([0,1,0]); vp = np.array(para[:3])
                  dev_px = abs(np.dot(vx, vp)/math.sqrt(sum(vx**2)*sum(vp**2)))
                  dev_py = abs(np.dot(vy, vp)/math.sqrt(sum(vy**2)*sum(vp**2)))

                  # angle condition
                  if (n_p == 0 and dev_px > math.cos(math.pi/18)) or (n_p == 1 and dev_py > math.cos(math.pi/18)):
                      j += 1; index_inlier = []; inlier = [] # set default and start

                      # calculating distances and angles for all point
                      dist = abs(np.array(para)[0]*test_points[:,0] + np.array(para)[1]*test_points[:,1] + np.array(para)[2]*test_points[:,
                      v1 = test_nor; v2 = np.array(para[:3])
                      dev = abs(np.dot(v1, v2)/(np.sum(v1**2,axis=1)/sum(v2**2))**(1/2))

                      # check whether it's an inlier or not
                      for i in range(test_points.shape[0]):
                          if dist[i] < ransac_threshold and dev[i] <= 1 and dev[i] > math.cos(math.pi/18):
                              index_inlier.append(indices_2[i]); inlier.append(test_points[i])
                              num += 1 # update score

                      # minimum score condition
                      if num >= 200 and num > BSup:
                          index_inlier, inlier = np.array(index_inlier), np.array(inlier)
                          index_inlier = np.sort(index_inlier)

                          # DBSCAN for connectivity
                          inlier_cloud = rempts.select_by_index(index_inlier)
                          pcd = inlier_cloud
                          labels = np.array(pcd.cluster_dbscan(eps=0.08, min_points=6, print_progress=False))
                          x = labels
                          unique, counts = np.unique(x, return_counts=True)
                          u_del = []
                          for i in range(len(counts)):
                              if counts[i]<200 : u_del.append(i)

                          uniques = np.delete(unique, u_del, 0)  # obtain segment larger than 200

                          if -1 in uniques : uniques = np.delete(uniques, 0, 0)
                          l_del = []
                          for i in range(len(labels)):
                              if labels[i] not in uniques : l_del.append(i)
                          l_del = np.array(l_del)
                          num = num - len(l_del)
                          index_inlier, inlier = np.delete(index_inlier, l_del, 0), np.delete(inlier, l_del, 0)

                      # in case a new model is better - cache it
                      if num > BSup:
                          BSup = num; ppara = para
                          index_inliers, inliers = np.array(index_inlier), np.array(inlier)
```

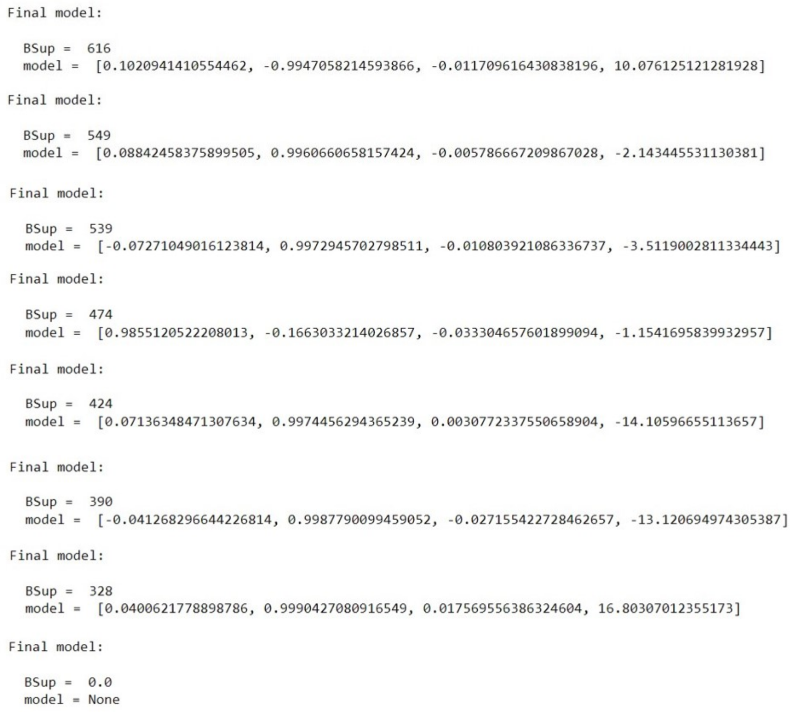


```
    print('\nFinal model: \n'); print('  BSup = ', BSup); print('  model = ', ppara); print('  itt = ', j)

    # minimum score condition for final model
    if BSup >= 200:
        inlier_cloud = rempts.select_by_index(index_inliers)
        outlier_cloud = rempts.select_by_index(index_inliers, invert=True)
        nor_outlier = np.delete(normall, np.array(index_inliers), 0)
        inlier_cloud.paint_uniform_color([random.uniform(0, 1), random.uniform(0, 1), random.uniform(0, 1)])
        return ppara, inlier_cloud, outlier_cloud, BSup, nor_outlier
    else:
        inlier_cloud = o3d.geometry.PointCloud()
        outlier_cloud = rempts
        nor_outlier = normall
        return 0, inlier_cloud, outlier_cloud, BSup, nor_outlier
```

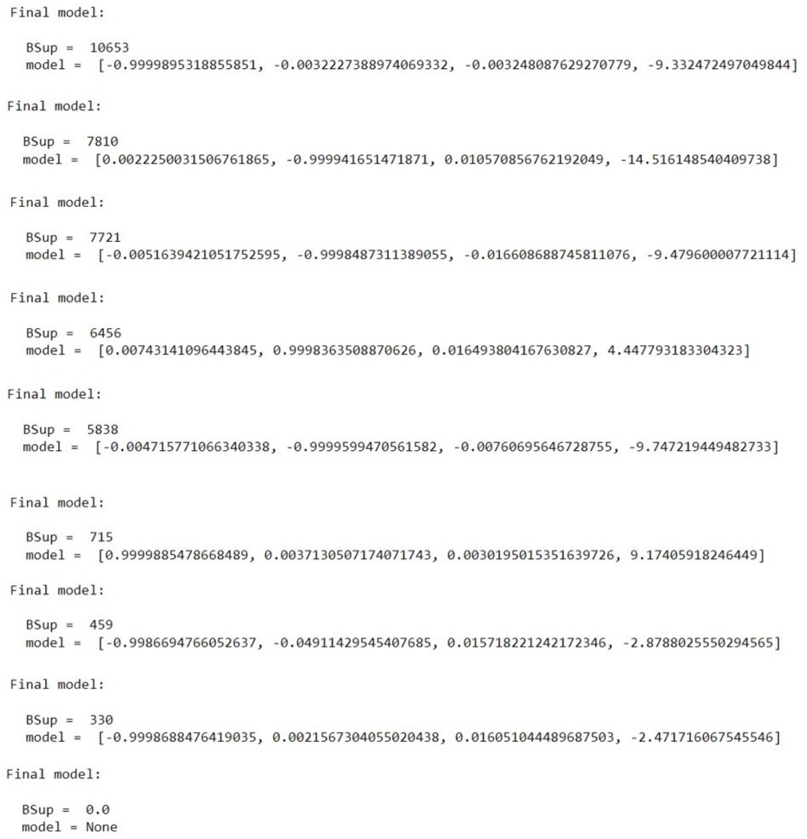
APPENDIX B : Part of the planar parameters detected in form of $Ax + By + Cz + D = 0$ and the number of points belonging to the model (BSup).



```
Final model:

  BSup =  25717
  model =  [-0.9999928905975384, -0.003706376889532383, -0.0006939198315100128, 0.19234397529528982]

Final model:

  BSup =  24502
  model =  [0.9999580007874085, 0.004680826580135548, -0.007879500223722469, 1.972485549038976]

Final model:

  BSup =  10606
  model =  [0.9999759306040329, 0.0014353941464712884, 0.006788067194905633, -0.5527633555693322]

Final model:

  BSup =  8147
  model =  [0.9999828694506738, 0.0019758163832196453, 0.005509714585734485, 2.4836455514719726]

Final model:

  BSup =  2612
  model =  [-0.9999733924774489, -0.006831939591008808, -0.0025571348354426716, -3.3857487766790486]

Final model:

  BSup =  2162
  model =  [0.9983593609153093, -0.01205880008845911, -0.055974742636315286, -0.23765494189586692]

Final model:

  BSup =  1883
  model =  [0.01969894651745862, -0.999805947093484, 0.0001402234062998949, 18.183043467242243]

Final model:

  BSup =  1802
  model =  [-0.9966665781061321, -0.08146830109672416, 0.004318333315827223, -0.046715237793875306]

Final model:

  BSup =  1483
  model =  [0.004895824583322002, -0.999620092986933, -0.02712380132733661, 1.4049734986507043]

Final model:

  BSup =  1376
  model =  [0.05016255647472902, 0.9987337420548493, -0.0038249717694966196, 1.7171292366153807]

Final model:

  BSup =  1314
  model =  [0.014928135082624005, -0.9997198161128137, -0.018369541483576685, -17.612511863378472]

Final model:

  BSup =  670
  model =  [-0.09026202684637895, 0.9956270772658516, -0.024072588656843984, -11.037095355533616]
```



```
Final model:

  BSup =  616
  model =  [0.1020941410554462, -0.9947058214593866, -0.011709616430838196, 10.076125121281928]

Final model:

  BSup =  549
  model =  [0.08842458375899505, 0.9960660658157424, -0.005786667209867028, -2.143445531130381]

Final model:

  BSup =  539
  model =  [-0.07271049016123814, 0.9972945702798511, -0.010803921086336737, -3.5119002811334443]

Final model:

  BSup =  474
  model =  [0.9855120522208013, -0.1663033214026857, -0.033304657601899094, -1.1541695839932957]

Final model:

  BSup =  424
  model =  [0.07136348471307634, 0.9974456294365239, 0.0030772337550658904, -14.10596655113657]

Final model:

  BSup =  390
  model =  [-0.041268296644226814, 0.9987790099459052, -0.027155422728462657, -13.120694974305387]

Final model:

  BSup =  328
  model =  [0.0400621778898786, 0.9990427080916549, 0.017569556386324604, 16.80307012355173]

Final model:

  BSup =  0.0
  model = None
```



```
Final model:

  BSup =  10653
  model =  [-0.9999895318855851, -0.0032227388974069332, -0.003248087629270779, -9.332472497049844]

Final model:

  BSup =  7810
  model =  [0.0022250031506761865, -0.999941651471871, 0.010570856762192049, -14.516148540409738]

Final model:

  BSup =  7721
  model =  [-0.0051639421051752595, -0.9998487311389055, -0.016608688745811076, -9.479600007721114]

Final model:

  BSup =  6456
  model =  [0.00743141096443845, 0.9998363508870626, 0.016493804167630827, 4.447793183304323]

Final model:

  BSup =  5838
  model =  [-0.004715771066340338, -0.9999599470561582, -0.00760695646728755, -9.747219449482733]

Final model:

  BSup =  715
  model =  [0.9999885478668489, 0.0037130507174071743, 0.0030195015351639726, 9.17405918246449]

Final model:

  BSup =  459
  model =  [-0.9986694766052637, -0.04911429545407685, 0.015718221242172346, -2.8788025550294565]

Final model:

  BSup =  330
  model =  [-0.9998688476419035, 0.0021567304055020438, 0.016051044489687503, -2.471716067545546]

Final model:

  BSup =  0.0
  model = None
```

BIOGRAPHY

Name	Mr. Thanapon Doougphummet
Date of Birth	July 14, 1996
Place of Birth	Bangkok, Thailand
Educations	B.Sc (Mathematics), Kasetsart University, 2019
Scholarships	Development and Promotion of Science and Technology Talents Project (DPST)