PERSPECTIVE-INDEPENDENT POINT CLOUD PROCESSING: TOWARDS STREAMLINING 3D COMPUTER VISION WORKFLOWS AND ENHANCING 3D INDOOR SCENE PERCEPTION

by © Ali Ebrahimi

A thesis submitted to the School of Graduate Studies in partial fulfillment
of the requirements for the degree of Doctor of Philosophy

Department of Electrical and Computer Engineering

Faculty of Engineering and Applied Science

Memorial University of Newfoundland

St. John's, Newfoundland, Canada

October 2024

Abstract

The advent of commercially accessible depth sensors, augmented reality headsets, and smartphones equipped with depth sensing technologies has revolutionized the acquisition of 3D data, enabling comprehensive spatial understanding in real-world environments. This advancement has led to the widespread adoption of 3D data, offering significant benefits across a range of applications, from augmented reality to autonomous navigation. However, the complexity of indoor scenes poses significant challenges for 3D computer vision systems, particularly in cluttered environments where background surfaces hinder the detection and analysis of relevant foreground objects. This PhD thesis presents a comprehensive study of perspective-independent point cloud processing techniques tailored to address the challenges posed by cluttered and complex indoor environments. The primary objectives focus on streamlining 3D computer vision workflows by contextually segmenting and subtracting 3D background surfaces while enhancing 3D scene perception through accurate identification of these surfaces and spatial relationships within indoor scenes. Alongside the primary objectives, this thesis also addresses two sub-objectives: size reduction of indoor point clouds and labeling of various elements within complex indoor scenes. To achieve these objectives, four research endeavors are presented. Initially, two techniques were implemented for bounding surface segmentation and removal: Iterative Region-based RANdom SAmple Consensus (IR-RANSAC) and orientation-based M-estimator SAmple Consensus (MSAC). They considerably reduce the size of 3D datasets and the search space of various 3D computer vision applications, resulting in enhanced performance and faster processing times. IR-RANSAC demonstrates robust performance with a mean F_1 score above 94%, while Orientation-based MSAC achieves a mean F_1 score exceeding 98%, showcasing its superior performance and notable computational efficiency. In the subsequent work, PiGPDS, a perspective-independent ground plane detection and segmentation method, was introduced as a method for detecting and segmenting ground planes in 3D complex indoor environments, where the position and orientation of the sensor are unrestricted and unknown. PiGPDS demonstrated exceptional performance, achieving an average F_1 score of 96.01%, accurately segmenting ground surfaces of complex 3D indoor scenes acquired from diverse locations with varying pitches and yaws. Finally, in the concluding endeavor, PiPCS, a Perspective-Independent Point Cloud Simplifier, stands as a significant advancement, building upon the foundational research laid out in earlier studies. PiPCS redefines conventional 3D background subtraction techniques by contextually segmenting and eliminating 3D background components, yielding precisely segmented 3D foreground objects without relying on colour or historical data. PiPCS demonstrates outstanding performance, achieving an average F_1 score of 91.27% and substantial size reductions averaging 74.11% across all dataset's point clouds. PiPCS optimizes 3D computer vision systems by streamlining their workflows, enhancing indoor scene perception, reducing point cloud size, and enabling precise labeling within complex indoor environments.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisor, Dr. Stephen Czarnuch. His unwavering support, guidance, and encouragement have been invaluable throughout my PhD journey. I have learned an immense amount under his mentorship, gaining valuable technical knowledge and research skills. His dedication to excellence and his insightful feedback have contributed significantly to the success of this thesis. I am truly grateful for the opportunity to work with him and for his profound impact on my academic and professional development. Thank you for believing in me.

Second, I would like to extend my gratitude to my supervisory committee members, Dr. Ray Gosine, and Dr. Andrew Vardy. Their constructive feedback, encouragement, and scholarly insights have played a crucial role in shaping this thesis.

Third, I would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for their financial support. Their funding has been crucial for the completion of this thesis.

Last but not least, I extend my heartfelt appreciation to my family for their unwavering support and encouragement throughout my academic journey. I am profoundly grateful for their love, understanding, and patience. Thank you for always being there for me.

Table of Contents

Lists of Abbreviations

2D	Two-Dimensional
3D	Three-Dimensional
AI	Artificial Intelligence
CNN (ConvNet)	Convolutional Neural Network
DECB	Depth-Extended Codebook
FN	False Negative
FP	False Positive
FPM	Feature Pooling Module
GAN	Generative Adversarial Network
GMM	Gaussian Mixture Model
GRASTA	Grassmannian Robust Adaptive Subspace Tracking
GSM	Generic Scene Modeling
HSV	Hue, Saturation, Value
IncPCP	Incremental Principal Component Pursuit
IR	Infrared Radiation
IR-RANSAC	Iterative Region-based Random Sample Consensus
KDE	Kernel Density Estimation
LiDAR	Light Detection and Ranging
MLE	Maximum Likelihood Estimation
MoG	Mixture of Gaussians
MSAC	M-estimator SAmple Consensus

NN	Neural Network
PiGPDS	Perspective-independent Ground Plane Detection and Segmentation
PiPCS	Perspective-independent Point Cloud Simplifier
RAM-NN	Random Access Memory Neural Network
RANSAC	RANdom SAmple Consensus
RGB	Red, Green, Blue
RGBD	Red, Green, Blue, Depth
S3DIS Dataset	Stanford Large-Scale 3D Indoor Spaces Dataset
SOBS	Self-Organizing Background Subtraction
SRPCA	Spatiotemporal Robust Principal Component Analysis
SVM	Support Vector Machine
T2F-MOG	Type-2 Fuzzy Mixture of Gaussians
TCNN	Transposed Convolutional Neural Network
TN	True Negative
ToF	Time-of-Flight
TP	True Positive
WT	Walsh Transform

Lists of Tables

Lists of Figures

Chapter 1. Introduction

The emergence of commercially available depth sensors has enabled a deeper understanding and accurate representation of spatial information in real-world environments. Consequently, 3D data have become widely adopted and provide substantial benefits across diverse applications, ranging from human activity recognition to augmented reality.

In general, indoor bounding surfaces (e.g., surrounding walls, and floor) comprise a large percentage of points within each frame of 3D point clouds, recognized as one of the primary representations of 3D data. These surfaces can decrease the performance of 3D computer vision algorithms (e.g., object segmentation and tracking) by cluttering their search space. Therefore, a robust bounding surface segmentation and removal technique can significantly reduce the search space and bring three main benefits to the computer vision systems: improving downstream results, speeding up downstream processes, and reducing the overall size of the point clouds. Popular plane segmentation methods such as Random Sample Consensus (RANSAC) [1] and its improved variants, are widely used to segment and remove surfaces from a point cloud. However, these estimators easily result in the incorrect association of foreground points to background bounding surfaces because of the stochasticity of randomly sampling, and the limited scene-specific knowledge used by these approaches. To address these challenges, I introduced two bounding surface removal techniques for complex 3D indoor environments. These approaches were developed to rely solely on the raw depth map from unorganized point clouds and support a variety of sensor perspectives. In my first work, IR-RANSAC [2], I proposed a robust bounding surface removal and size reduction technique for complex 3D indoor environments. In my subsequent work [3], I build upon my initial IR-RANSAC method, achieving both a marginal performance boost and a significant reduction in processing time compared to IR-RANSAC.

Several computer vision applications, such as augmented reality and robot navigation, benefit from identifying the location of the ground surface. However, current 3D ground plane detection methods require one or more of these assumptions: the largest plane of the scene is the ground plane; one or multiple depth sensors are placed in specific orientations or positions; a video sequence of data is accessible; the input point cloud is organized; or foreground objects (e.g., a human body) are positioned vertically on the ground plane of the scene. To address these restrictive assumptions, I proposed a perspective-independent ground plane detection and segmentation (PiGPDS) [4] that only requires a single point cloud of an indoor scene and only assumes that the 3D scene contains one surface parallel to its actual ground plane, which is almost always valid for cluttered and complex indoor environments.

The significance of background subtraction in the domains of image processing and computer vision is paramount. This essential preprocessing step simplifies scene information by isolating important foreground objects, enhancing the efficiency and accuracy of a wide range of computer vision applications, such as moving object detection, and video surveillance. Depth-based background subtraction techniques often adopt well-known 2D background modeling techniques, such as Mixture of Gaussians and frame difference. While these methods can address certain limitations of 2D background modeling methods, such as challenges related to illumination changes, they still inherit some of the drawbacks associated with 2D background modeling techniques they are derived from. To overcome these limitations, I developed a Perspective-Independent Point Cloud Simplifier (PiPCS) [5], a native 3D background modeling solution, drawing from the foundation laid by the three preceding studies [2], [3], and [4]. PiPCS expands the conventional concept of background subtraction to encompass the identification, segmentation, and removal of 3D background bounding surfaces, such as walls, windows, curtains, ceilings, and floors, from indoor point clouds. Unlike conventional background subtraction techniques, PiPCS contextually segments and eliminates 3D background components, yielding precisely segmented 3D foreground objects without relying on colour or historical data. PiPCS optimizes 3D computer vision systems by streamlining their workflows, enhancing indoor scene perception, reducing point cloud size, and enabling precise labeling within complex indoor environments.

The motivation for developing these perspective-independent solutions that exclusively utilize 3D coordinates of foreground objects is rooted in the broader objective of creating intelligent, contextually aware assistive technologies tailored for individuals facing a variety of challenges, such as neurodegenerative conditions, cognitive disabilities, and the elderly. These technologies are designed to facilitate monitoring within the privacy of users' homes or private healthcare settings. By relying solely on the depth information, the system mitigates errors in conditions where colour information is limited or absent, such as in dimly lit indoor environments. Furthermore, depth-based computer vision solutions inherently enhance privacy by capturing only the 3D coordinates of foreground objects, such as the human body, without relying on identifying colour data. The overarching goal of this project is to develop plug-and-play monitoring systems, enabling users to install them independently without requiring expert assistance. This user-centric approach acknowledges that sensor placement may be influenced by convenience rather than optimal performance, highlighting the need for robust point cloud processing techniques that can adapt to various challenging environments while supporting varied sensor positions and orientations.

1.1 Research Objectives

The primary objectives of this research are to streamline the intricate workflows of 3D computer vision applications and enhance 3D indoor scene perception through the development and integration of perspective-independent point cloud processing techniques. The detailed breakdown of these objectives is presented below.

- Objective 1. Streamlining 3D Computer Vision Workflows
  The first objective centers on simplifying 3D indoor point clouds, i.e., contextually segmenting and subtracting 3D background bounding surfaces (e.g., walls, curtains, windows, floors, and ceilings) while retaining segmented 3D foreground objects within complex indoor scenes.
- Objective 2. Enhancing 3D Indoor Scene Perception
  The second objective focuses on analyzing the segmented 3D background bounding surfaces to identify their elements like the ceiling, floor, and surrounding walls. The specific emphasis is on accurately identifying the ground plane because once known, it inherently provides information about the ceiling and surrounding walls. This approach enables a comprehensive understanding of the spatial relationships among 3D background elements and 3D foreground objects within indoor scenes.

In addition to the primary objectives, this research also has two sub-objectives: size reduction of indoor point clouds and labeling of various components within complex indoor scenes. Further details regarding these two objectives are outlined below.

- Sub-Objective 1. Size Reduction
  This objective aims to reduce the size of 3D indoor point clouds by removing non-essential large background bounding surfaces, effectively addressing computational and storage challenges. Additionally, it can enhance data transmission efficiency by strategically subtracting unchanging 3D background

from point cloud frames and transmitting a single instance alongside the 3D foreground objects of each frame.

- Sub-Objective 2. Labeling
  This objective serves as a robust labeling tool, accurately segmenting and labeling both 3D background and 3D foreground objects within indoor point clouds, benefiting machine learning-based computer vision applications.

1.2 Research Contributions

This research expands the conventional concept of background modeling and subtraction to contextually segment and subtract 3D background bounding surfaces (e.g., ceiling, floor, and surrounding walls) and retain segmented 3D foreground objects within complex indoor scenes without relying on colour and historical information. This study is fundamentally focused on optimizing 3D computer vision workflows, presenting four comprehensive solutions that contribute substantially to the advancements in the field of 3D point cloud processing.

First, it serves as an essential preprocessing step for a variety of 3D computer vision applications, including 3D object segmentation, human tracking, and human activity recognition. As a preprocessing step, it effectively narrows the search space for downstream processes by segmenting 3D background and foreground objects of indoor scenes, allowing for the easy elimination of only background objects in tasks requiring foreground identification, resulting in notable performance and accuracy improvements of these applications.

Second, it can significantly reduce the size of 3D indoor point clouds by eliminating their large background bounding surfaces, which are non-essential for the majority of 3D computer vision applications, thereby yielding a streamlined and compact dataset. This contribution addresses a critical challenge in the field, where large point cloud datasets can pose significant computational and storage burdens. In addition, it has the potential to optimize data transmission processes by first removing the unchanging and redundant 3D background from all point cloud frames and then transmitting one instance of the 3D background alongside the 3D foreground objects of each frame. This supplementary application can contribute to data transmission efficiency and resource optimization.

Third, it significantly advances 3D indoor scene perception by accurately segmenting all points associated with each boundary of the 3D background, including the ceiling, floor, and surrounding walls within an indoor scene. This capability is particularly beneficial in applications such as virtual reality, where achieving an accurate interpretation of 3D space is crucial for immersive experiences and precise scene representation.

Fourth, it can serve as a labeling tool by accurately segmenting both 3D background and 3D foreground objects within indoor point clouds, providing benefits to machine learning-based computer vision applications.

The research and its derived solutions distinguish themselves through the following key advantages:

- Perspective independence: supporting a wide range of sensor heights relative to the ground, along with different pitches and yaws, adapting to almost all practical sensor perspectives. This is critical to ensuring optimal system performance and consistent results in diverse and non-ideal sensor setups and data collection scenarios.
- Privacy enhancement: prioritizing privacy by exclusively capturing 3D coordinates of foreground objects, such as the human body, without the need for descriptive colour data. This characteristic makes it particularly well-suited for applications in sensitive environments, such as private healthcare settings where monitoring patients or elderly individuals requires privacy protection.

- Universal sensor and data compatibility: relying solely on the raw depth map from unorganized point clouds, confirming its adaptability to a range of sensor types. This key advantage enables its application across a wide range of 3D sensors, making it a versatile solution that seamlessly works with both organized and unorganized point clouds.

1.3 Literature Review

To avoid repetition due to the manuscript-style presentation of this thesis, a summary of the literature review is provided in this section. For a more comprehensive and detailed literature review, readers are encouraged to refer to the following chapters.

1.3.1 RGB, Depth and RGBD Data

RGB data has been highly valuable in computer vision due to its comprehensive colour information, capturing the diverse visual world. Algorithms for analyzing RGB data often rely on colour variations, patterns, and contrasts to extract meaningful information from images or video streams. Recent advancements in Artificial Intelligence (AI), fueled by the availability of large collections of 2D images, have made RGB-based computer vision methods even more effective. These AI-powered approaches utilize deep learning and neural networks to better understand RGB data, enabling accurate analysis and interpretation of visual scenes. On the other hand, RGB-based computer vision applications, such as background modeling, encounter various challenges, including bootstrapping, colour camouflage, illumination changes, intermittent motion, and foreground shadows, as discussed in [6] and [7].

In contrast to RGB sensors, depth sensors such as stereo vision, structured light, time-of-flight (ToF), and LiDAR offer spatial information about the scene, holding the potential to mitigate some of the challenges of the RGB-based methods. Stereo vision sensors (e.g., ZED X [8]) function by analyzing the disparities in perspective between images captured by a pair of cameras, enabling the estimation of depth information. Structured light sensors (e.g., Microsoft Kinect V1) operate by projecting a predefined infrared (IR) light pattern onto a scene. When this pattern interacts with the scene, the pattern becomes distorted. Subsequently, an infrared camera captures this distorted pattern, resulting in a depth map that represents the scene's 3D structure. ToF sensors (e.g., Microsoft Kinect V2, Microsoft Azure Kinect [9], and ToF sensor integrated in Microsoft HoloLens headset [10]) operate by emitting pulses of infrared light and measuring the time it takes for these pulses to travel to an object and bounce back to the camera's sensor. This timing information is then used to calculate the distance from the camera to various points on the object's surface, yielding improved accuracy and performance in depth sensing compared to structured light sensors. Similarly, LiDAR scanners, such as those integrated into devices like the Apple Vision Pro [11] and iPhone 15 Pro [12], also emit light pulses and measure the time it takes for these pulses to reflect off objects. However, LiDAR scanners use laser pulses, enabling them to provide highly detailed 3D representations of the environment with increased precision and range compared to ToF sensors.

While depth data are less affected by the challenges that impact RGB-based computer vision methods, particularly background modeling, relying exclusively on depth data can also present its own set of challenges, including depth shadows, depth camouflage, sensor distance limitations, and specular surface reflections, as discussed in [7] and [13].

Hence, many 3D computer vision applications aim to leverage the complementary relationship between colour and depth data acquired from RGBD sensors. These sensors integrate RGB and depth data to construct a 3D point cloud, capturing colour through RGB cameras while simultaneously measuring object distances using depth sensors. The registration of these data generates a detailed 3D representation of the environment, with each voxel in the point cloud containing both colour and 3D coordinates.

Point cloud data can be categorized into two main types: organized and unorganized datasets. Organized datasets are structured in a matrix-like format, allowing easy access to voxels based on their spatial or geometric relationships using indexing. In contrast, unorganized datasets lack such defined relationships between adjacent voxels, and the voxel coordinates are stored as an unordered one-dimensional array. Converting an organized point cloud to an unorganized one is a straightforward task, whereas the reverse conversion process is notably more intricate and resource-intensive. As spatial relationships between voxels are maintained in organized point clouds, point cloud processing becomes less challenging. Nevertheless, computer vision methods tailored for unorganized datasets are universal [14], meaning they can also be applied to organized datasets.

Although RGBD approaches have shown enhanced performance compared to depth-based techniques in certain contexts, they come with their own set of limitations. RGBD methods demand more computational resources due to two main factors: the fusion of depth and colour data to create a point cloud and the simultaneous processing of both colour and depth information. Additionally, RGBD-based methods are more error-prone in situations where colour information is limited or absent, such as dark indoor environments, compared to their depth-based counterparts. Furthermore, depth-based computer vision solutions, by design, promote privacy as they exclusively capture 3D coordinates of foreground objects (e.g., the human body) without reliance on more identifying colour data. This makes them suitable for applications like human tracking or pose estimation in private settings, such as monitoring patients or elderly individuals within the privacy of their homes.

1.3.2 Plane Segmentation

In general, plane segmentation methods can be categorized into three categories: model fitting-based methods, region growing-based methods, and clustering feature-based methods.

1.3.2.1 Model Fitting-based Methods

Random Sample Consensus (RANSAC) [1] and Hough transform [15] are the most commonly used model fitting-based methods for plane segmentation. The Hough transform is a voting technique for identifying objects that can be modeled parametrically, such as lines, planes, and spheres. Every point is transformed into a unique function (e.g., a sinusoid when modeling lines) in a discretized parameter space. Objects of interest can then be extracted by selecting the maximal intersections between the functions in the discretized parameter space, where the spatial tolerance for model fitting (e.g., to compensate for sensor resolution, noise, and object surface variations) can be accommodated by changing the resolution of the parameter space. The Hough transform has been successfully used for 3D plane segmentation in several publications (e.g., [16] and [17]). Unfortunately, although Hough transform-based methods can robustly segment 3D objects, they necessitate large amounts of memory and significant computational time [18], and their results depend significantly on the proper selection of segmentation parameters [19]. More importantly, Hough transform is unable to discriminate between voxels that lie within a parameterized model (i.e., inliers) and outside the model (i.e., outliers) since spatial relationships are not preserved in the Hough parameter space. The result is that foreground points that belong to objects that are spatially close to the parameterized background model will often be associated with the model, and ultimately the background scene.

The RANSAC algorithm begins with a random selection of data points that estimate the corresponding model parameters (e.g., three points for a plane). Then, the remaining points are examined to determine how many of them are well-approximated by the model by identifying which points are within a fixed threshold (e.g., the orthogonal distance from the planar model). This process is iteratively repeated a user-defined number of times. Terminally, the RANSAC algorithm returns the model with the highest percentage of inliers. Many researchers have proposed RANSAC-based algorithms for 3D plane segmentation, such as [19], [20], and [21].

According to a performance comparison by Tarsha-Kurdi et al. [22], the RANSAC algorithm outperforms the Hough transform approach for 3D roof plane segmentation in terms of both speed and accuracy. However, RANSAC suffers from spurious plane detection in complex 3D indoor environments [21].

1.3.2.2 Region Growing-Based Methods

In general, region growing-based methods have two main stages. First, they pick a seed point and then merge the neighbouring points or voxels that comply with the predefined criteria (e.g., similar normal vector). Several researchers proposed point-based, voxel-based, and hybrid region growing techniques for 3D point cloud segmentation. Tóvári and Pfeifer [23] proposed a point-based region growing algorithm that merges adjacent points to a seed region based on their normal vectors and distance to the adjusting plane. Nurunnabi et al. [24] utilized the same criteria but with a different seed point selection approach and a better normal vectors estimation.

Voxel-based region growing algorithms (e.g., [25] and [26]) improve the speed and robustness of point-based methods by voxel-wise processing of the 3D point clouds. Xiao et al. [27] proposed a 3D plane segmentation method based on a hybrid region growing approach utilizing a subwindow and a single point as growth units. Although their technique is significantly faster than the point-based region growing approach, it was intended for only organized point clouds. Vo et al. [18] proposed a fast plane segmentation technique for urban environments. The method consists of two main stages: first, a coarse segmentation was achieved using an octree-based region growing approach, then a point-based process refined the results by adding unassigned points into incomplete segments.

Region growing-based methods are easy to employ for 3D plane segmentation, primarily for organized point clouds. However, their output results depend on the growing criteria, the seed point selection, and the textures or roughness of planes [28]. Furthermore, they are not robust to occlusion, point density variation, and noise [29].

1.3.2.3 Clustering Feature-Based Methods

Clustering feature-based methods adopt a data clustering approach based on characteristics of planar surfaces, such as normal vector attributes. Filin [30] proposed a clustering method based on an attribute vector comprising a point location, its tangent plane's parameters, and the height difference between the point and its adjacent points. In another work, Filin and Pfeifer [31] computed the point features using a slope adaptive neighborhood system and employed a mode-seeking algorithm to extract clusters. Then, they extended or merged those clusters with their adjacent points or clusters if they share analogous standard deviations and surface parameters. Czerniawski et al. [32] applied a simple density-based clustering algorithm to a normal vector space (i.e., a Gaussian sphere). The dense clusters on the Gaussian sphere represent the directions perpendicular to large planes. Zhou et al. [33] proposed a clustering feature-based method for segmenting planes in terrestrial point clouds. First, they created a 4D parameter space using planes' normal vectors and their distance to the origin. Then, they segmented the planar surfaces by applying the Iso cluster unsupervised classification method.

Despite the efficiency of clustering feature-based methods, employing multi-dimensional features in large point clouds is computationally intensive [18]. Furthermore, they are sensitive to noise and outliers [34]. Moreover, the clustering segmentation approaches cannot reliably segment the edge points as these points may have different feature vectors compared to the surface points.

1.3.3 Ground Plane Detection

Most common conventional 2D methods for estimating the ground plane are homography-based, such as [35], [36], and [37]. These methods rely on certain assumptions; for example, the camera's field of view should be parallel to the ground, and the ground plane should dominate most of the field of view. Compared to conventional 2D methods, deep learning-based techniques such as [38], [39] and [40] can provide more accurate results; however, their ability to provide accurate depth estimation in unknown scenes is limited by the single-view scale ambiguity [41]. Due to the limitations imposed by these restrictive assumptions and the fact that 2D data are not sufficient to fully represent the spatial relationships between the ground plane and other objects in an indoor scene, many researchers have opted for 3D ground plane detection as a more effective solution for cluttered and dynamic indoor environments.

Many conventional 3D techniques for identifying the ground plane use one or more of the following: the Hough transform algorithm [42], the RANSAC algorithm [43], or normal vectors of the scene's surfaces. For instance, Borrmann et al. [44] utilized a 3D Hough transform and a ball-shaped accumulator for ground plane detection in 3D point clouds, while Zeineldin and El-Fishawy [45] proposed an enhanced RANSAC algorithm to identify the ground plane and obstacles for individuals with visual impairments. However, these model fitting-based methods can only work if the ground plane is the largest in the scene; additionally, the RANSAC algorithm is prone to detecting spurious planes in 3D complex indoor environments [46]. Holz et al. [47] proposed a normal-based plane segmentation method for mobile navigation in indoor environments. Although their approach can successfully identify the ground plane of indoor scenes, it is restricted to structured point clouds and only functions properly when the sensor's pitch angle is zero. To overcome these limitations, Zhang and Czarnuch [48] proposed a perspective-independent ground estimation method by processing a video sequence of RGB-D data. Their method can successfully detect the ground plane of dynamic indoor scenes regardless of ground plane size and camera orientation but requires at least one human body to be visible and moving in the RGB-D camera's field of view.

1.3.4 Depth-based Background Subtraction

Depth-based background subtraction techniques center around exploiting the distinct spatial relationships between foreground objects and their surrounding backgrounds within the depth data. These methods often adopt well-known 2D background modeling techniques, including thresholding, single Gaussian [49], Mixture of Gaussians (MoG) [50], and frame difference, and are employed across a spectrum of 3D computer vision applications such as fall detection [51], human tracking [52] and gesture recognition [53].

Thresholding, a fundamental 2D segmentation technique, serves as a basic method for background subtraction in depth maps. For instance, Cinque et al. [54] employed Otsu thresholding to isolate foreground objects in the depth map and subsequently refined the segmented foregrounds by applying region growing and morphological operations.

The single Gaussian and MoG models have been widely utilized for depth-based background subtraction. For example, Rougier et al. [51] employed the single Gaussian model for fall detection among the elderly, and Zhang et al. [55] embraced it as a key initial step in object detection. Frick et al. [56] employed the MoG algorithm to isolate foreground regions from the background for creating 3D-TV contents.

The well-established frame difference method, initially developed for background subtraction of 2D videos, has demonstrated adaptability in scenarios where only depth information is available. This method has been employed in several 3D computer vision applications, including human tracking [52] and gesture recognition [53], to segment foreground objects, such as human bodies.

While the aforementioned depth-based background subtraction techniques can address certain limitations of 2D background modeling methods, such as challenges related to illumination changes, they still inherit some of the drawbacks associated with 2D background modeling techniques they are derived from. For example, choosing the suitable depth threshold can still be challenging when the background itself is dynamic or contains moving objects, often requiring manual tuning based on the depth map's characteristics.

Several depth-based deep learning methods are available for outdoor point cloud processing, particularly for semantic segmentation (e.g., [57], [58], and [59]). These methods have shown significant advancements in accurately classifying and labeling point cloud data, essential for various applications such as autonomous driving and urban scene understanding. These approaches are trained on large labeled datasets, such as the SemanticKITTI dataset [60], which provides extensive outdoor scenes for robust model training. However, most 3D indoor semantic segmentation methods are based on RGBD data. In addition, no depth-based deep learning methods are currently available in the existing literature for 3D background subtraction or simplifying 3D point clouds of indoor environments.

1.4 Thesis Organization

This PhD thesis adopts a manuscript-style format. The current chapter introduces the thesis, detailing its objectives, contributions, literature review and structure. Chapters 2 through 5 each contain a manuscript that has been published to peer-reviewed publications.

Chapter 2 was published as an article titled "Automatic Super-Surface Removal in Complex 3D Indoor Environments Using Iterative Region-Based RANSAC" in the journal Sensors in May 2021.

Chapter 3 was presented as a conference paper titled "Bounding Surface Segmentation and Removal in Complex 3D Indoor Environments Using Orientation-based MSAC" at the 30th Annual Newfoundland Electrical and Computer Engineering Conference (NECEC), St. John's, Canada, in November 2021.

Chapter 4 was presented as a conference paper titled "PiGPDS: Perspective Independent Ground Plane Detection and Segmentation for Complex 3D Indoor Scenes" at the IEEE International Conference on Digital Image Computing: Techniques and Applications (DICTA), Port Macquarie, Australia, in November 2023.

Chapter 5 was published as an article titled "PiPCS: Perspective Independent Point Cloud Simplifier for Complex 3D Indoor Scenes" in the journal IEEE Access in August 2024.

Finally, chapter 6 provides a concise summary of the thesis, along with insights into potential future research paths. Copyright permissions to reproduce manuscripts in this thesis have been obtained as necessary.

1.5 References

[1] M. A. Fischler and R. C. Bolles, "Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[2] A. Ebrahimi and S. Czarnuch, "Automatic super-surface removal in complex 3D indoor environments using iterative region-based RANSAC," Sensors, vol. 21, no. 11, p. 3724, 2021.

[3] A. Ebrahimi and S. Czarnuch, "Bounding Surface Segmentation and Removal in Complex 3D Indoor Environments Using Orientation-based MSAC," in the 30th Annual Newfoundland Electrical and Computer Engineering Conference (NECEC), 2021.

[4] © 2023 IEEE. Reprinted, with permission, from A. Ebrahimi and S. Czarnuch, "PiGPDS: Perspective Independent Ground Plane Detection and Segmentation for Complex 3D Indoor Scenes," in 2023 International Conference on Digital Image Computing: Techniques and Applications (DICTA), IEEE, 2023, pp. 105–112.

[5] A. Ebrahimi and S. Czarnuch, "PiPCS: Perspective Independent Point Cloud Simplifier for Complex 3D Indoor Scenes," in IEEE Access, doi: 10.1109/ACCESS.2024.3452633, 2024.

[6] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "A novel video dataset for change detection benchmarking," IEEE Transactions on Image Processing, vol. 23, no. 11, pp. 4663–4679, 2014.

[7] L. Maddalena and A. Petrosino, "Background subtraction for moving object detection in RGBD data: A survey," J Imaging, vol. 4, no. 5, p. 71, 2018.

[8] StereoLabs, "ZED X Camera." Accessed: Oct. 01, 2023. [Online]. Available: https://www.stereolabs.com/zed-x/

[9] Microsoft, "Azure Kinect." Accessed: Mar. 05, 2023. [Online]. Available: https://azure.microsoft.com/en-us/products/kinect-dk

[10] Microsoft, “HoloLens.” Accessed: Apr. 11, 2024. [Online]. Available: https://www.microsoft.com/en-us/hololens

[11] Apple Inc., “Apple Vision Pro.” Accessed: Apr. 12, 2024. [Online]. Available: https://www.apple.com/apple-vision-pro/

[12] Apple Inc., “iPhone 15 Pro.” Accessed: Apr. 12, 2024. [Online]. Available: https://www.apple.com/ca/iphone-15-pro/

[13] G. Moya-Alcover, A. Elgammal, A. Jaume-i-Capó, and J. Varona, “Modeling depth for nonparametric foreground segmentation using RGBD devices,” Pattern Recognit Lett, vol. 96, pp. 76–85, 2017.

[14] S. Chen, D. Tian, C. Feng, A. Vetro, and J. Kovačević, “Fast resampling of three-dimensional point clouds via graphs,” IEEE Transactions on Signal Processing, vol. 66, no. 3, pp. 666–681, 2018, doi: 10.1109/TSP.2017.2771730.

[15] D. H. Ballard, “Generalizing the Hough transform to detect arbitrary shapes,” Pattern Recognit, vol. 13, no. 2, pp. 111–122, 1981, doi: 10.1016/0031-3203(81)90009-1.

[16] G. Vosselman, G. Sithole, G. Vosselman, B. G. H. Gorte, G. Sithole, and T. Rabbani, “Recognising structure in laser scanner point clouds RECOGNISING STRUCTURE IN LASER SCANNER POINT CLOUDS 1,” 2003.

[17] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter, “The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design,” 3D Research, vol. 2, no. 2, pp. 1–13, Nov. 2011, doi: 10.1007/3DRes.02(2011)3.

[18] A. V. Vo, L. Truong-Hong, D. F. Laefer, and M. Bertolotto, “Octree-based region growing for point cloud segmentation,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 104, pp. 88–100, 2015, doi: 10.1016/j.isprsjprs.2015.01.011.

[19] T. M. Awwad, Q. Zhu, Z. Du, and Y. Zhang, “An improved segmentation approach for planar surfaces from unstructured 3D point clouds,” Photogrammetric Record, vol. 25, no. 129, pp. 5–23, 2010, doi: 10.1111/j.1477-9730.2009.00564.x.

[20] D. Chen, L. Zhang, P. T. Mathiopoulos, and X. Huang, “A methodology for automated segmentation and reconstruction of urban 3-D buildings from ALS point clouds,” IEEE J Sel Top Appl Earth Obs Remote Sens, vol. 7, no. 10, pp. 4199–4217, Oct. 2014, doi: 10.1109/JSTARS.2014.2349003.

[21] L. Li, F. Yang, H. Zhu, D. Li, Y. Li, and L. Tang, “An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells,” Remote Sens (Basel), vol. 9, no. 5, 2017, doi: 10.3390/rs9050433.

[22] 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,” ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007, vol. XXXVI, no. 1, pp. 407–412, 2007.

[23] D. Tóvári and N. Pfeifer, “Segmentation based robust interpolation - A new approach to laser data filtering,” International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, vol. 36, pp. 79–84, 2005.

[24] A. Nurunnabi, D. Belton, and G. West, “Robust segmentation in laser scanning 3D point cloud data,” in 2012 International Conference on Digital Image Computing Techniques and Applications, DICTA 2012, IEEE eXpress Conference Publishing, Nov. 2012, pp. 1–8. doi: 10.1109/DICTA.2012.6411672.

[25] J.-E. Deschaud and F. Goulette, “A Fast and Accurate Plane Detection Algorithm for Large Noisy Point Clouds Using Filtered Normals and Voxel Growing,” 2010.

[26] M. Huang, P. Wei, and X. Liu, “An Efficient Encoding Voxel-Based Segmentation (EVBS) Algorithm Based on Fast Adjacent Voxel Search for Point Cloud Plane Segmentation,” Remote Sens (Basel), vol. 11, no. 23, p. 2727, Nov. 2019, doi: 10.3390/rs11232727.

[27] J. Xiao, J. Zhang, B. Adler, H. Zhang, and J. Zhang, “Three-dimensional point cloud plane segmentation in both structured and unstructured environments,” Rob Auton Syst, vol. 61, no. 12, pp. 1641–1652, Dec. 2013, doi: 10.1016/j.robot.2013.07.001.

[28] X. Leng, J. Xiao, and Y. Wang, “A multi-scale plane-detection method based on the Hough transform and region growing,” Photogrammetric Record, vol. 31, no. 154, pp. 166–192, 2016, doi: 10.1111/phor.12145.

[29] O. Teboul, L. Simon, P. Koutsourakis, and N. Paragios, “Segmentation of building facades using procedural shape priors,” Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3105–3112, 2010, doi: 10.1109/CVPR.2010.5540068.

[30] S. Filin, “Surface clustering from airborne laser scanning data,” International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, vol. 34, 2002.

[31] S. Filin and N. Pfeifer, “Segmentation of airborne laser scanning data using a slope adaptive neighborhood,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 60, no. 2, pp. 71–80, 2006, doi: 10.1016/j.isprsjprs.2005.10.005.

[32] T. Czerniawski, M. Nahangi, S. Walbridge, and C. Haas, “Automated removal of planar clutter from 3D point clouds for improving industrial object recognition,” ISARC 2016 - 33rd International Symposium on Automation and Robotics in Construction, no. Isarc, pp. 357–365, 2016, doi: 10.22260/isarc2016/0044.

[33] G. Zhou, S. Cao, and J. Zhou, “Planar Segmentation Using Range Images from Terrestrial Laser Scanning,” IEEE Geoscience and Remote Sensing Letters, vol. 13, no. 2, pp. 257–261, 2016, doi: 10.1109/LGRS.2015.2508505.

[34] Z. Dong, B. Yang, P. Hu, and S. Scherer, “An efficient global energy optimization approach for robust 3D plane segmentation of point clouds,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 137, pp. 112–133, 2018, doi: 10.1016/j.isprsjprs.2018.01.013.

[35] D. Conrad and G. N. DeSouza, “Homography-based ground plane detection for mobile robot navigation using a modified em algorithm,” in 2010 IEEE International Conference on Robotics and Automation, IEEE, 2010, pp. 910–915.

[36] J. Arróspide, L. Salgado, M. Nieto, and R. Mohedano, “Homography-based ground plane detection using a single on-board camera,” IET Intelligent Transport Systems, vol. 4, no. 2, pp. 149–160, 2010.

[37] S. Kumar, A. Dewan, and K. M. Krishna, “A bayes filter based adaptive floor segmentation with homography and appearance cues,” in Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing, 2012, pp. 1–8.

[38] B. Tan, N. Xue, S. Bai, T. Wu, and G.-S. Xia, “Planetr: Structure-guided transformers for 3d plane recovery,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 4186–4195.

[39] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa, “Planenet: Piece-wise planar reconstruction from a single rgb image,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2579–2588.

[40] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz, “Planercnn: 3d plane detection and reconstruction from a single image,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4450–4459.

[41] Y. Xie, M. Gadelha, F. Yang, X. Zhou, and H. Jiang, “PlanarRecon: Real-time 3D Plane Detection and Reconstruction from Posed Monocular Videos,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 6219–6228.

[42] R. O. Duda and P. E. Hart, “Use of the Hough transformation to detect lines and curves in pictures,” Commun ACM, vol. 15, no. 1, pp. 11–15, 1972.

[43] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[44] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter, “The 3d hough transform for plane detection in point clouds: A review and a new accumulator design,” 3D Research, vol. 2, no. 2, pp. 1–13, 2011.

[45] R. A. Zeineldin and N. A. El-Fishawy, “Fast and accurate ground plane detection for the visually impaired from 3D organized point clouds,” in 2016 SAI computing conference (SAI), IEEE, 2016, pp. 373–379.

[46] L. Li, F. Yang, H. Zhu, D. Li, Y. Li, and L. Tang, “An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells,” Remote Sens (Basel), vol. 9, no. 5, p. 433, 2017.

[47] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, “Real-Time Plane Segmentation Using RGB-D Cameras.,” RoboCup, vol. 7416, pp. 306–317, 2011.

[48] C. Zhang and S. Czarnuch, “Perspective independent ground plane estimation by 2D and 3D data analysis,” IEEE Access, vol. 8, pp. 82024–82034, 2020.

[49] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, “Pfinder: Real-time tracking of the human body,” IEEE Trans Pattern Anal Mach Intell, vol. 19, no. 7, pp. 780–785, 1997.

[50] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in Proceedings. 1999 IEEE computer society conference on computer vision and pattern recognition (Cat. No PR00149), IEEE, 1999, pp. 246–252.

[51] C. Rougier, E. Auvinet, J. Rousseau, M. Mignotte, and J. Meunier, “Fall detection from depth map video sequences,” in Toward Useful Services for Elderly and People with Disabilities: 9th International Conference on Smart Homes and Health Telematics, ICOST 2011, Montreal, Canada, June 20-22, 2011. Proceedings 9, Springer, 2011, pp. 121–128.

[52] J. Han, E. J. Pauwels, P. M. de Zeeuw, and P. H. N. de With, “Employing a RGB-D sensor for real-time tracking of humans across multiple re-entries in a smart environment,” IEEE Transactions on Consumer Electronics, vol. 58, no. 2, pp. 255–263, 2012.

[53] U. Mahbub, H. Imtiaz, T. Roy, M. S. Rahman, and M. A. R. Ahad, “A template matching approach of one-shot-learning gesture recognition,” Pattern Recognit Lett, vol. 34, no. 15, pp. 1780–1788, 2013.

[54] L. Cinque, A. Danani, P. Dondi, and L. Lombardi, “Real-time foreground segmentation with Kinect sensor,” in Image Analysis and Processing—ICIAP 2015: 18th International Conference, Genoa, Italy, September 7-11, 2015, Proceedings, Part II 18, Springer, 2015, pp. 56–65.

[55] X. Zhang, X. Wang, and Y. Jia, “The visual internet of things system based on depth camera,” in Proceedings of 2013 Chinese Intelligent Automation Conference: Intelligent Automation & Intelligent Technology and Systems, Springer, 2013, pp. 447–455.

[56] A. Frick, F. Kellner, B. Bartczak, and R. Koch, “Generation of 3d-tv ldv-content with time-of-flight camera,” in 2009 3DTV conference: the true vision-capture, transmission and display of 3d video, IEEE, 2009, pp. 1–4.

[57] E. Li, S. Casas, and R. Urtasun, “Memoryseg: Online lidar semantic segmentation with a latent memory,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 745–754.

[58] J. Liu, C. Chang, J. Liu, X. Wu, L. Ma, and X. Qi, “Mars3d: A plug-and-play motion-aware model for semantic segmentation on multi-scan 3d point clouds,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 9372–9381.

[59] S. Wang, J. Zhu, and R. Zhang, “Meta-rangeseg: Lidar sequence semantic segmentation using multiple feature aggregation,” IEEE Robot Autom Lett, vol. 7, no. 4, pp. 9739–9746, 2022.

[60] J. Behley et al., “Semantickitti: A dataset for semantic scene understanding of lidar sequences,” in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 9297–9307.

Chapter 2 . Automatic Super-Surface Removal in Complex 3D Indoor Environments Using Iterative Region-Based RANSAC

Co-authorship statement – This chapter was published as an article titled "Automatic Super-Surface Removal in Complex 3D Indoor Environments Using Iterative Region-Based RANSAC" in the journal Sensors in May 2021. Ali Ebrahimi, as the primary author, conducted the literature review, designed the methodology, developed the method, collected data, analyzed the results, created visualizations, and both drafted and revised the manuscript. The co-author, Dr. Stephen Czarnuch, conceived the work, provided essential guidance and supervision throughout the research process, and meticulously reviewed and revised the manuscript. Both authors have read and agreed to the final version of the manuscript.

2.1 Abstract

Removing bounding surfaces such as walls, windows, curtains, and floor (i.e., super-surfaces) from a point cloud is a common task in a wide variety of computer vision applications (e.g., object recognition and human tracking). Popular plane segmentation methods such as Random Sample Consensus (RANSAC), are widely used to segment and remove surfaces from a point cloud. However, these estimators easily result in the incorrect association of foreground points to background bounding surfaces because of the stochasticity of randomly sampling, and the limited scene-specific knowledge used by these approaches. Additionally, identical approaches are generally used to detect bounding surfaces and surfaces that belong to foreground objects. Detecting and removing bounding surfaces in challenging (i.e., cluttered and dynamic) real-world scenes can easily result in the erroneous removal of points belonging to desired foreground objects such as human bodies. To address these challenges, we introduce a novel super-surface removal technique for 3D complex indoor environments. Our method was developed to work with unorganized data captured from commercial depth sensors and supports varied sensor perspectives. We begin with preprocessing steps and dividing the input point cloud into four overlapped local regions. Then, we apply an iterative surface removal approach to all four regions to segment and remove the bounding surfaces. We evaluate the performance of our proposed method in terms of four conventional metrics: specificity, precision, recall, and F_1 score, on three generated datasets representing different indoor environments. Our experimental results demonstrate that our proposed method is a robust super-surface removal and size reduction approach for complex 3D indoor environments while scoring the four evaluation metrics between 90% and 99%.

Keywords – RANSAC, point cloud, bounding surface removal, wall removal, 3D background subtraction, 3D plane segmentation, 3D preprocessing technique, 3D size reduction

2.2 Introduction

In image processing, background subtraction is widely used in object detection and tracking approaches [1–3] with broad applications such as human tracking [4, 5], face recognition [6], traffic management [7], and surveillance systems [8,9]. Background subtraction is typically a preprocessing phase used to identify and differentiate the foreground pixels (representing objects of interest) from the background pixels (representing uninteresting information). The background pixels can then be subtracted or removed from the original image, leaving only the foreground pixels, which reduces the storage space requirements, reduces the computational complexity, and improves the overall algorithm performance of downstream image processing techniques. Established 2D background subtraction approaches are based on static background segmentation methods [10], adaptive Gaussian mixture models [11,12], real-time codebook models [13,14], and independent component analysis-based techniques [15,16]. Although advanced 2D background subtraction techniques can handle gradual illumination changes and repetitive movements in the background, they perform poorly in the presence of shadows or foreground regions with colours similar to the background [17]. The release of commercially available and inexpensive depth sensors such as the Microsoft Kinect opened new doors for improved background subtraction techniques because of the availability of additional depth data associated with each pixel of colour data. Depth sensor (RGB-D) systems are more robust for background detection problems compared to classic colour-based systems because depth data are largely invariant to colour, texture, shape, and lighting [18,19]. As a result of the advantages of combined depth and colour information, data from RGB-D sensors have also been widely used in background segmentation methods (e.g., [17,20,21]) which have been thoroughly reviewed (see [22] for a comprehensive review of different 3D background subtraction methods and their capabilities in solving common background segmentation challenges).

Background subtraction methods are generally used to analyze individual images and identify the foreground by first estimating a reference background that is developed from historical information obtained from videos or sequences of images. Therefore, the classic application of 2D background subtraction is separating dynamic or moving objects from a relatively static or slow-changing background scene. However, RGB images only contain intensity information and spatial information that is largely restricted to the two dimensions of the image that are perpendicular to the camera's perspective. Accordingly, identifying the boundaries between objects, or conversely identifying object interactions or contact, is limited mainly to detectable intensity differences. In applications that utilize RGB-D data, interactions or contact between objects and object spatial relationships can be more directly measured.

Furthermore, registration between depth and RGB data allows traditional 2D background subtraction approaches to be supplemented with additional depth-based approaches (e.g., [23] and [24]), and further allows RGB-D data to be represented as 3D point clouds [25]. For example, reliably identifying and removing static background components such as roads and walls before modeling the background can result in both improved background subtraction and improved foreground segmentation using both 2D and RGB-D data; improvements that to our knowledge have only been realized through approaches that require additional data and computation, such as motion estimation between consecutive frames (e.g., [26]). Identifying static background components suffers from the same limitations as modeling the entire background using 2D data, suggesting that little benefit is afforded by first removing these background objects, then modeling the background. However, with RGB-D data, parametrically modeled objects (e.g., planes, spheres, cones, cylinders, and cubes) are far more reliably detectable. As a result, researchers have attempted to segment or remove large planar surfaces (e.g., walls, ceiling, and floor surfaces) as a preprocessing or fundamental step before all other algorithms (e.g., [27–29]).

In general, large planar surfaces comprise a large percentage of points within each frame of RGB-D data captured in indoor environments. However, outside specific applications that seek to identify significant surfaces (e.g., ground plane detection [30]), large planar surfaces are not often the objects of interest in 3D computer vision applications. Notably, smaller planar surfaces (e.g., tabletops, chair seats and backs, desks) are more likely to be of interest than larger surfaces at the boundaries of the scene. Furthermore, the large bounding surfaces can decrease the performance of 3D computer vision algorithms (e.g., object segmentation and tracking) by cluttering their search space. Therefore, a robust removal technique for points that belong to surfaces at the outer boundaries of the RGB-D data can significantly reduce the search space and bring three main benefits to the computer vision systems: improving downstream results, speeding up downstream processes, and reducing the overall size of the point clouds. We refer to these large surfaces, which may include points from multiple planes or objects (e.g., points that represent a wall, window, and curtains) at the extremes of the point clouds as super-surfaces in the context of background subtraction applications. Our objective is to develop a robust technique of removing super-surfaces from RGB-D data captured from indoor environments and represented as point clouds. Our intention is that our super-surface removal technique will function as a pre-processing step, improving existing computer vision techniques, and reducing storage requirements, without removing any foreground data.

2.2.1 Related Work

Point cloud data are classified as either organized or unorganized datasets. Organized datasets are represented by a matrix-like structure, where the data (i.e., voxels) are accessible by index, usually according to their spatial or geometric relationship. Unlike organized point clouds, the relationships between adjacent voxels of unorganized datasets are unknown, and the data are simply stored as a one-dimensional, unsorted array. Data from RGB-D sensors are typically stored as organized point clouds, where indexes are referenced according to the spatial resolution of the sensor. However, point cloud pre-processing steps such as down-sampling often produce unorganized point clouds. While it is trivial to convert an organized to an unorganized point cloud, the converse is much more complicated and costly. Since spatial relationships between voxels are preserved, plane detection is less challenging for organized point clouds. However, computer vision approaches designed for unorganized datasets are universal [31] (i.e., they can also be used for organized datasets), so a robust plane detection approach must work with unorganized datasets and not rely on the spatial relationship of voxels as derived from their storage indices. In general, plane segmentation methods can be categorized into three categories: model fitting-based methods, region growing-based methods, and clustering feature-based methods.

2.2.1.1 Model Fitting-based Methods

Random Sample Consensus (RANSAC) [32] and Hough transform [33] are the most commonly used model fitting-based methods for plane segmentation. The Hough trans-form is a voting technique for identifying objects that can be modeled parametrically, such as lines, planes, and spheres. Every point is transformed into a unique function (e.g., a sinusoid when modeling lines) in a discretized parameter space. Objects of interest can then be extracted by selecting the maximal intersections between the functions in the discretized parameter space, where the spatial tolerance for model fitting (e.g., to compensate for sensor resolution, noise, and object surface variations) can be accommodated by changing the resolution of the parameter space. The Hough transform has been successfully used for 3D plane segmentation in several publications (e.g., [34,35]). Unfortunately, although Hough transform-based methods can robustly segment 3D objects, they necessitate large amounts of memory and significant computational time [36], and their results depend significantly on the proper selection of segmentation parameters [37]. More importantly, Hough transform is unable to discriminate between voxels that lie within a parameterized model (i.e., inliers) and outside the model (i.e., outliers) since spatial relationships are not preserved in the Hough parameter space. The result is that foreground points that belong to objects that are spatially close to the parameterized background model will often be associated with the model, and ultimately the background scene.

The RANSAC algorithm begins with a random selection of data points that estimate the corresponding model parameters (e.g., three points for a plane). Then, the remaining points are examined to determine how many of them are well-approximated by the model. Terminally, the RANSAC algorithm returns the model with the highest percentage of inliers that are within a fixed threshold (e.g., the orthogonal distance from the planar model). Many researchers have proposed RANSAC-based algorithms for 3D plane segmentation, such as [37–39].

Awwad et al. [37] proposed a RANSAC-based segmentation algorithm that first clusters the data points into small sections based on their normal vectors and then segments the planar surfaces. This implementation of RANSAC prevents the segmentation of spurious surfaces in the presence of parallel-gradual planes such as stairs. Chen et al. [38] developed an improved RANSAC algorithm through a novel localized sampling technique and a region growing based approach. Their proposed method, intended to segment polyhedral rooftops from noisy Airborne

Laser Scanning (ALS) point clouds, is based on the assumption that rooftops comprise only planar primitives. Li et al. [39] proposed an enhanced RANSAC algorithm based on Normal Distribution Transformation (NDT) cells to prevent segmenting spurious planes. The algorithm considers each NDT cell rather than each point. After dividing the data points into a grid of NDT cells, a combination of the RANSAC algorithm and an iterative reweighted least-square approach fit a plane in each cell. Finally, a connected-component approach extracts large planes and eliminates points that do not belong to planes. Although the proposed method can detect 3D planes more reliable and faster than the standard RANSAC, it requires cell size tuning for different datasets.

According to a performance comparison by Tarsha-Kurdi et al. [40], the RANSAC algorithm outperforms the Hough transform approach for 3D roof plane segmentation in terms of both speed and accuracy. However, RANSAC suffers from spurious plane detection in complex 3D indoor environments [39].

2.2.1.2 Region Growing-Based Methods

In general, region growing-based methods have two main stages. First, they pick a seed point and then merge the neighbouring points or voxels that comply with the predefined criteria (e.g., similar normal vector). Several researchers proposed point-based, voxel-based, and hybrid region growing techniques for 3D point cloud segmentation. Tóvári and Pfeifer [41] proposed a point-based region growing algorithm that merges adjacent points to a seed region based on their normal vectors and distance to the adjusting plane. Nurunnabi et al. [42] utilized the same criteria but with a different seed point selection approach and a better normal vectors estimation.

Voxel-based region growing algorithms (e.g., [43,44]) improve the speed and robustness of point-based methods by voxel-wise processing of the 3D point clouds. Xiao et al. [45] proposed a 3D plane segmentation method based on a hybrid region growing approach utilizing a subwindow and a single point as growth units. Although their technique is significantly faster than the point-based region growing approach, it was intended for only organized point clouds. Vo et al. [36] proposed a fast plane segmentation technique for urban environments. The method consists of two main stages: first, a coarse segmentation was achieved using an octree-based region growing approach, then a point-based process refined the results by adding unassigned points into incomplete segments.

Region growing-based methods are easy to employ for 3D plane segmentation, primarily for organized point clouds. However, their output results depend on the growing criteria, the seed point selection, and the textures or roughness of planes [46]. Furthermore, they are not robust to occlusion, point density variation, and noise [47].

2.2.1.3 Clustering Feature-Based Methods

Clustering feature-based methods adopt a data clustering approach based on characteristics of planar surfaces, such as normal vector attributes. Filin [48] proposed a clustering method based on an attribute vector comprising a point location, its tangent plane's parameters, and the height difference between the point and its adjacent points. In another work, Filin and Pfeifer [49] computed the point features using a slope adaptive neighborhood system and employed a mode-seeking algorithm to extract clusters. Then, they extended or merged those clusters with their adjacent points or clusters if they share analogous standard deviations and surface parameters. Czerniawski et al. [28] applied a simple density-based clustering algorithm to a normal vector space (i.e., a Gaussian sphere). The dense clusters on the Gaussian sphere represent the directions perpendicular to large planes. Zhou et al. [50] proposed a clustering feature-based method for segmenting planes in terrestrial point clouds. First, they created a 4D parameter space using planes' normal vectors and their distance to the origin. Then, they segmented the planar surfaces by applying the Iso cluster unsupervised classification method.

Despite the efficiency of clustering feature-based methods, employing multi-dimensional features in large point clouds is computationally intensive [36]. Furthermore, they are sensitive to noise and outliers [51]. Moreover, the clustering segmentation approaches cannot reliably segment the edge points as these points may have different feature vectors compare to the surface points.

2.2.2 Contributions

Several 3D plane segmentation methods can satisfactorily detect different planar surfaces for various computer vision applications. However, to our knowledge, no approaches have been developed specifically for bounding surface removal, particularly in complex environments: environments that are cluttered, and where the placement of a depth sensor is not ideal. Additionally, existing segmentation approaches generally segment foreground points that belong to parametrically modeled objects of interest (e.g., planes, spheres, cones, cylinders, and cubes), rather than with the intention of removing background points belonging to the bounding surfaces. Therefore, existing approaches can easily remove critical foreground objects (or portions of foreground objects), significantly impacting the segmentation accuracy of semantic information. To overcome these limitations, we propose a method of removing background bounding surfaces (i.e., super-surfaces, such as walls, windows, curtains, and floor). Our novel method is particularly suited to more challenging and cluttered indoor environments, where differentiating between foreground and background points is complicated. Accordingly, our objective is to develop a robust background super-surface removal method that can support a wide range of sensor heights relative to the ground (i.e., support varied sensor perspectives) for organized and unorganized point clouds. Additionally, our approach must ensure that foreground objects, and points belonging to those objects, are preserved during super-surface removal.

Our method significantly reduces the search space, and it can considerably reduce the size of 3D datasets, depending on the number and size of the super-surfaces in each point cloud. Furthermore, when used as a preprocessing step, our approach can improve the results and the running time of different 3D computer vision methods such as object recognition and tracking algorithms. The remainder of this paper is organized as follows. In the next section, we describe our proposed 3D super-surface removal method. In Section 2.4, we provide our experimental results and the evaluation of our proposed method. In Section 2.5, we present our discussion and future work, followed by conclusions in Section 2.6.

2.3 The Iterative Region-Based RANSAC

Our Iterative Region-based RANSAC (IR-RANSAC) has five main steps, as illustrated in Fig. 2.1. We begin with two preprocessing techniques, first down-sampling the raw point cloud and then removing noisy or outlying points in the depth map. Second, we divide the point cloud space into four overlapped local regions based on the current view of the sensor. Third, we segment a base plane in each of the four local regions. Fourth, we implement an iterative plane removal technique to all four local regions, segmenting and removing the super-surfaces. Finally, we cluster the remaining point cloud using the geometric relationship between groups of points, resulting in a final point cloud comprised only of clustered objects of interest.

2.3.1 Downsampling and Denoising

Since input point clouds are generally large in size due to the significant number of 3D points and associated colour information, a downsampling method with low computational complexity can significantly reduce the running time of point cloud processing algorithms. Downsampling is typically achieved using either a random downsample method [52] or a voxelized grid approach [53]. Although the former is more efficient, the latter preserves the shape of the point cloud better and exploits the geometric relationship of the underlying samples. Since we are predominantly concerned with preserving the underlying points that represent the true geometry of objects in the scene, we utilize a voxelized grid approach [53] that returns the centroid of all the points in each 3D voxel grid with a leaf size of 0.1cm. In this way, the downsampled point clouds will still reflect the structure and maintain the geometric properties of the original point cloud while reducing the total amount of points that will need to be processed and stored.

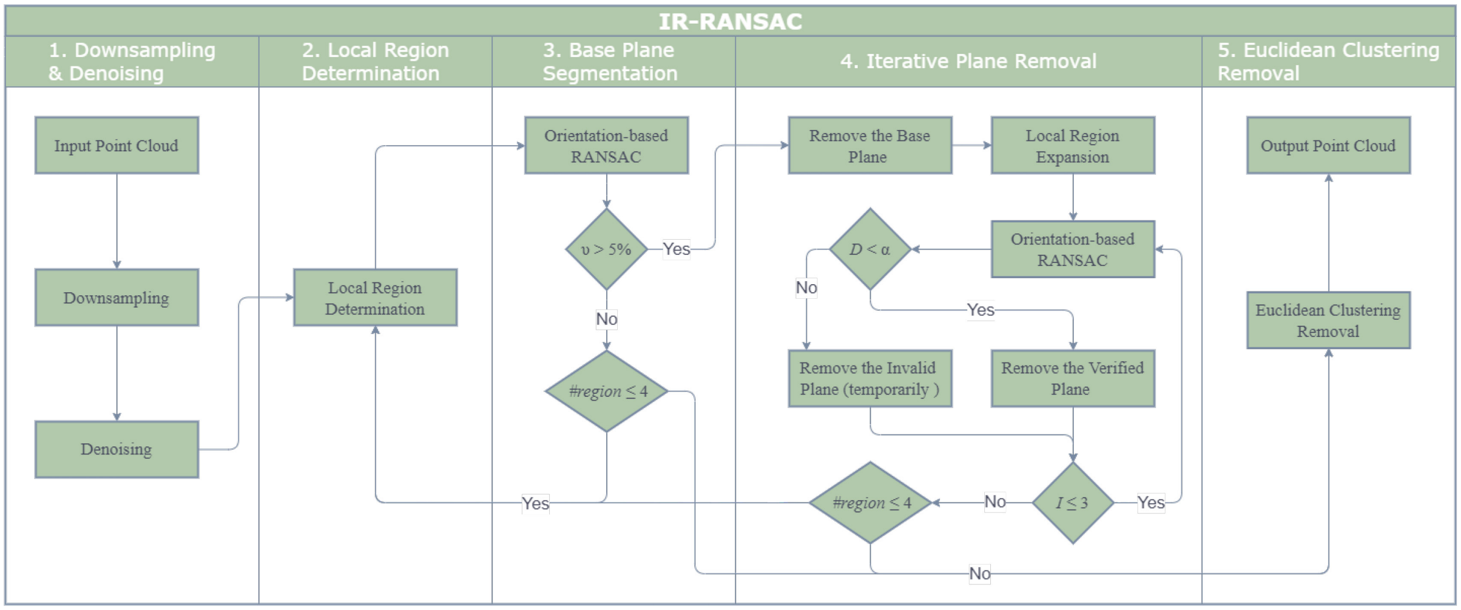

IR-RANSAC
1. Downsampling & Denoising
2. Local Region Determination
3. Base Plane Segmentation
4. Iterative Plane Removal
5. Euclidean Clustering Removal
Input Point Cloud
Downsampling
Denoising
Local Region Determination
Orientation-based RANSAC
υ > 5%
Yes
No
#region ≤ 4
Remove the Base Plane
Local Region Expansion
Orientation-based RANSAC
D < α
No
Yes
Remove the Invalid Plane (temporarily)
Remove the Verified Plane
#region ≤ 4
I ≤ 3
No
Yes
Yes
No
Output Point Cloud
Euclidean Clustering Removal

Figure 2.1 The flowchart of IR-RANSAC

Removing noisy points is a critical point cloud preprocessing task. Noisy or spurious points have two significant impacts on our approach. A noisy point cloud with false or spurious data points, including points outside of a scene's real boundaries (see Fig. 2.2 for example) can lead to a wrong measurement of the overall bounding box containing the point cloud, resulting in the definition of incorrect local regions in our subsequent processing steps. Furthermore, noisy points within the point cloud itself will effectively skew or change the geometry of the true objects. We utilize a statistical outlier removal approach [54] by examining the k-nearest neighbours ($K = 4$) of each point, and removing all points with a distance (σ) of more than one standard deviation of the mean distance to the query point to remove outliers of each captured point cloud. If the average distance of a point to its k-nearest neighbours is above the threshold (σ), it is considered as an outlier. In this way, we remove points that are dissimilar from other points in their neighborhood. Together, these approaches decrease the number of points in the point cloud, reducing downstream processing time and increasing the accuracy of our process.

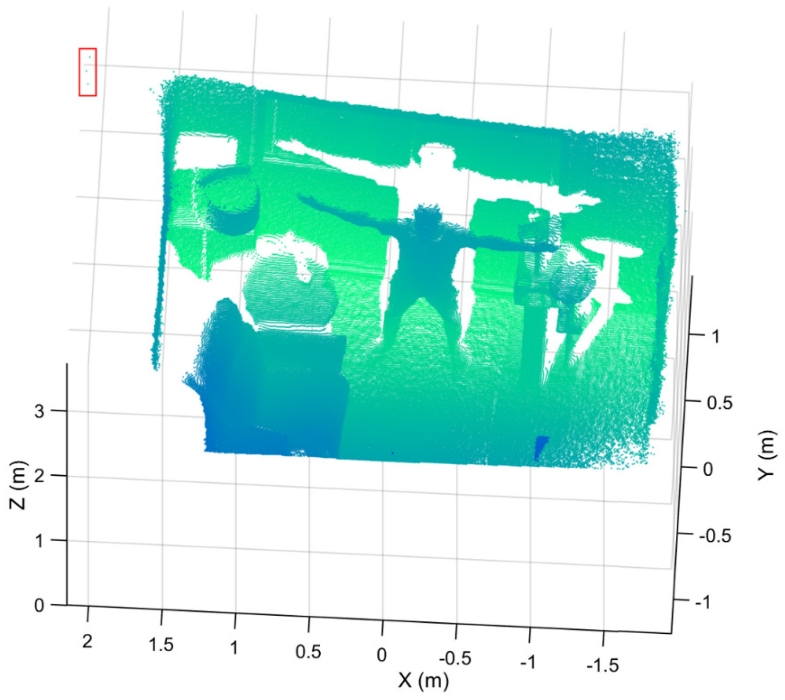


Figure 2.2 A sample point cloud with false data points (surrounded by a red rectangle), detected as the noise outside the true boundaries of the room. These noise artifacts artificially expand the overall outer dimensions of the point cloud.

2.3.2 Local Region Determination

Dividing our captured point clouds into four local regions of interests, based on the properties of our indoor environments, reduces the possibility of detecting foreground planes, increases computational efficiency, and leverages the likely spatial location of potential bounding surfaces. In this way, we exploit knowledge of the scene based on the known sensor perspective, while allowing for surface locations to vary relative to each other in different rooms. Further, these regions help ensure that foreground objects that may appear planar in composition (e.g., tables, beds) are preserved and differentiated from background bounding surfaces.

We partition the point cloud space into four overlapped local regions based on the current view of the sensor. First, we find the bounding values of the downsampled and denoised point cloud, where the values x_{min}, x_{max}, y_{min}, y_{max}, z_{min} and z_{max} are the Euclidean extrema of the bounding box enclosing the point cloud, and $(x_{range}, y_{range}, z_{range}) = (x_{max}, y_{max}, z_{max}) - (x_{min}, y_{min}, z_{min})$ are the Euclidean dimensions of the bounding box. Using the ranges defined in Table 2.1, we then determine the four local regions (see Fig. 2.3 for a sample visualization of the local regions). We will use these four regions to identify, segment, and remove potential super-surfaces in each region.

Table 2.1 Ranges for each local region

Region	**Range**		
	X-Axis	**Y-Axis**	**Z-Axis**
Back	$(-\infty, \infty)$	$(-\infty, \infty)$	$\left[z_{max} - \left(\frac{z_{range}}{4}\right), z_{max}\right]$
Left	$\left[x_{max} - \left(\frac{x_{range}}{4}\right), x_{max}\right]$	$(-\infty, \infty)$	$(-\infty, \infty)$
Right	$\left[x_{min}, x_{min} + \left(\frac{x_{range}}{4}\right)\right]$	$(-\infty, \infty)$	$(-\infty, \infty)$
Bottom	$(-\infty, \infty)$	$\left[y_{min}, y_{min} + \left(\frac{y_{range}}{4}\right)\right]$	$(-\infty, \infty)$

Since our approach must be independent of any prior knowledge about the geometry of the indoor environment and both the location and perspective of the sensor, the four initial local regions may not include all the points that are actually part of the super-surfaces (e.g., Fig. 2.3d, where parts of the floor are not included within the local region).

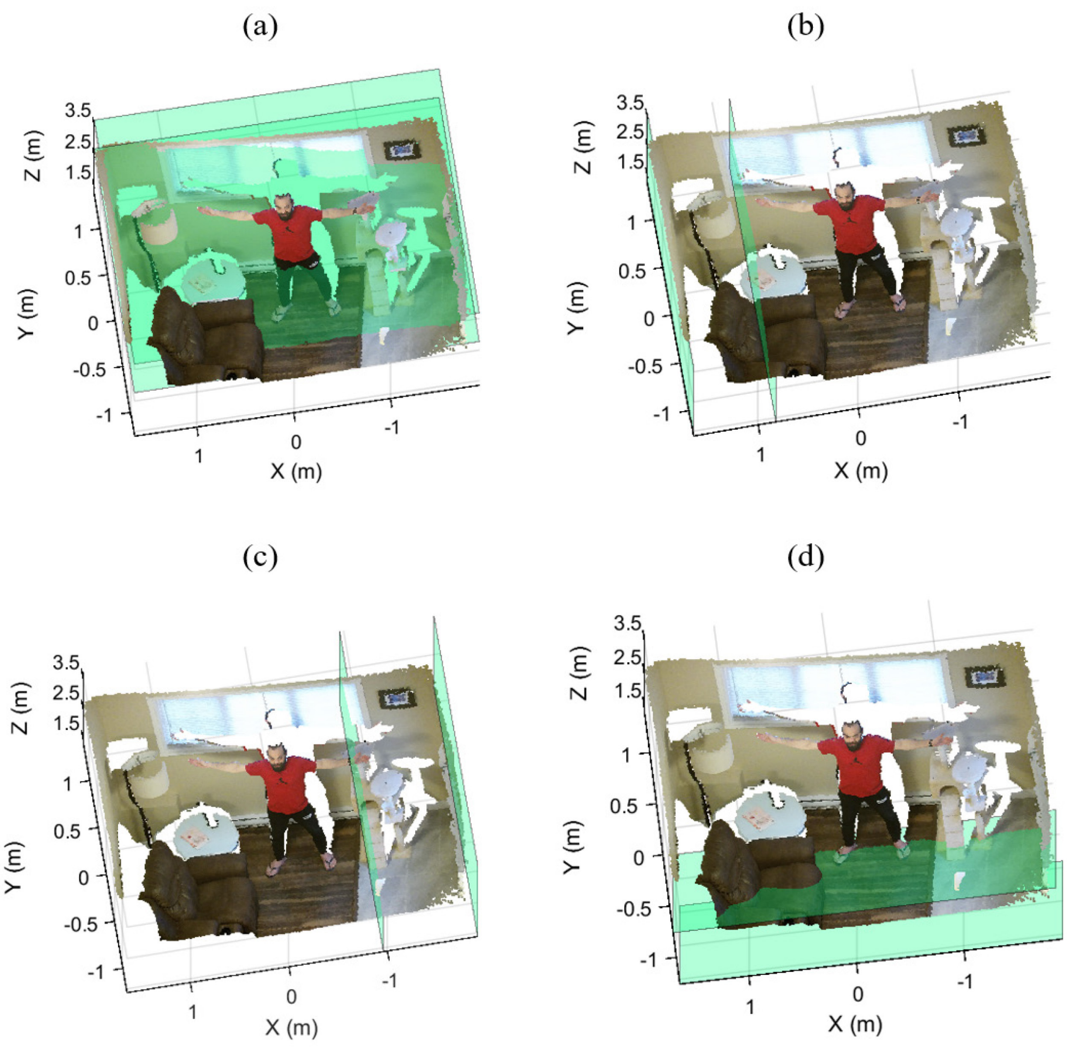


Figure 2.3 The boundaries of the four local regions highlighted in green: (a) the back, (b) left, (c) right, and (d) bottom regions.

Selecting larger initial regions will increase the likelihood that all true points are within the regions but will also increase the likelihood of including points belonging to foreground objects near the super-surfaces (e.g., beds and sofas). To resolve this issue, we implement conservative local regions and extend these four regions after base plane segmentation (see Section 2.3.4).

2.3.3 Base Plane Segmentation

We utilize the RANSAC algorithm [32] to segment the largest planes with a specific orientation in each of the local regions. All segmented plane candidates with more points than a learned value $\upsilon = 5\%$ of the total number of points in the point cloud, are verified as base planes and stored for use in the next step (Section 2.3.4). Planes containing fewer than υ points may be associated with key objects or small bounding planes and are dealt with in subsequent processing steps. Furthermore, υ is set as a proportion of the total points such that it is adaptive to the size of the point cloud.

The RANSAC algorithm iteratively and randomly samples three voxels, A, B and C as a minimum subset to generate a hypothesis plane. These three points represent two vectors $\overrightarrow{AB}$ and $\overrightarrow{AC}$, and their cross product is the normal vector $\vec{N} = (a, b, c)$ to the plane $ax + by + cz + d = 0$. Therefore, the three parameters of the plane $(a, b, and\ c)$ are computed, and d can be solved. In each iteration, the algorithm computes the distance $D = |\frac{ax+by+cz+d}{\sqrt{a^2+b^2+c^2}}|$ between all the remaining data points and the plane and then counts the number of points within a distance threshold ($\delta =$ 4 cm) of the plane. Finally, RANSAC returns the plane with the highest percentage of inliers.

We add an orientation constraint to the standard RANSAC (orientation-based RANSAC) so that we assign priority to segmented planes with the highest percentage of inliers that have an expected orientation relative to the local regions. To do this, we defined an initial reference vector for each of the local regions, aligned with the sensor axes as $[0,0,-1]$, $[-1,0,0]$, $[1,0,0]$ and $[0,1,0]$ for the back, left, right, and bottom regions, respectively (Fig. 2.4). Further, we define a maximum allowance angular variation ($\omega = 45$ degrees) between the normal vector of the planes and our reference vectors to allow for sensor perspective variations.

The maximum number of iterations T required for convergence by the RANSAC algorithm can be approximated as (2.1) [55]. Convergence depends on the number of samples s ($s = 3$ for the plane fitting), the target success probability p (e.g., $p = 99\%$), and the outlier ratio e.

Considering there is no prior knowledge about the underlying outlier ratio, it is difficult to approximate the number of RANSAC iterations. Based on our experimental results and due to the iterative design of IR-RANSAC, the algorithm works appropriately with the number of trials adhering to $1\% \leq T \leq 2\%$ of the data points within each local region. In this study, we set T to 2% of the data points (e.g., if the back region contains 60,000 points, the maximum number of trials will be set to 1200). Increasing p and T improve the robustness of the output at the expense of additional computation:

$$T = \frac{\log(1-p)}{\log(1-(1-e)^s)} \tag{2.1}$$

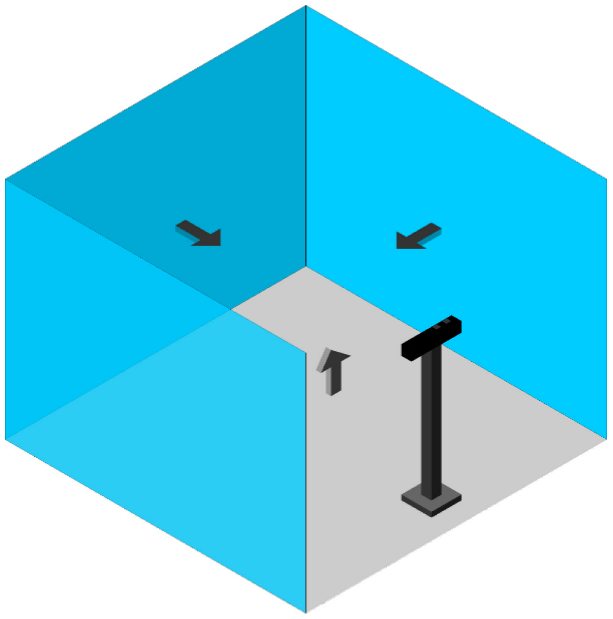


Figure 2.4 The initial reference vectors

2.3.4 Iterative Plane Removal

In a complex indoor environment, bounding surfaces such as walls, windows, and curtains are difficult to fit to a single plane. Increasing the distance threshold (δ) includes more points near the bounding planes, but simultaneously increases the chance of including data from important objects (e.g., the human body) within the extended threshold. Furthermore, the input point cloud can be unorganized, which means the nearest neighbour operations, such as region growing, are not very efficient for segmenting the rest of the super-surfaces. We introduce a novel iterative plane removal technique to segment and remove super-surfaces from a point cloud while minimizing the likelihood of including points that belong to foreground objects.

First, we remove the verified base planes associated with each local region. Then, we expand the local regions according to the ranges in Table 2.2 to completely encompass the areas containing the super-surfaces. Next, we apply the orientation-based RANSAC in each of the extended regions iteratively. The number of iterations depends on the complexity of the indoor environment; based on our experimental results, three iterations are adequate for a challenging indoor environment. In each iteration, segmented planes must be parallel to the base plane of the current region. Hence, we utilize the normal vectors of the base planes as the reference vectors, and we set the maximum allowance angular variation (θ) to 5°. Finally, because employing the orientation-based RANSAC in a larger region increases the probability of a false segmentation, we validate the segmented planes in each iteration.

Table 2.2 Extended ranges for each local region

Region	**Range**		
	X-Axis	**Y-Axis**	**Z-Axis**
Back	$(-\infty, \infty)$	$(-\infty, \infty)$	$\left[z_{max} - \left(\frac{z_{range} \times 3}{4}\right), z_{max}\right]$
Left	$\left[x_{max} - \left(\frac{x_{range}}{2}\right), x_{max}\right]$	$(-\infty, \infty)$	$(-\infty, \infty)$
Right	$\left[x_{min}, x_{min} + \left(\frac{x_{range}}{2}\right)\right]$	$(-\infty, \infty)$	$(-\infty, \infty)$
Bottom	$(-\infty, \infty)$	$\left[y_{min}, y_{min} + \left(\frac{y_{range}}{2}\right)\right]$	$(-\infty, \infty)$

The segmented planes are validated based on their distances, D, from their base planes, where $a, b, c, and\ d$ are parameters of the base plane, and x, y and z are coordinates of a point on the segmented plane. To make the technique robust to high levels of noise, we substitute the distance of a point to the base plane with the mean of all the segmented points' distances from the base plane.

If the distances are less than a threshold (e.g., $\alpha = 10$ cm), those planes will be removed from their regions. Otherwise, they are not part of the super-surfaces and will be temporarily removed from the remaining point cloud. There are two advantages to temporarily removing a false segmented plane. First, it prevents RANSAC from segmenting the false plane once again. Second, it reduces the current region for the next iteration. Figure 2.5 illustrates the output of the iterative plane removal in each iteration when applied to the back region of a point cloud. The green planes are verified and eliminated from the back region, as shown in Fig. 2.5b, 2.5c, and 2.5e. However, the segmented red plane is not verified and temporarily removed from the point cloud, as shown in Fig. 2.5d.

2.3.5 Euclidean Clustering Removal

In this step, we cluster the remaining point cloud based on Euclidean distance to remove the irrelevant small segments and keep the objects of interest. First, we compute the Euclidean distance between each point and its neighbours. Then, we group neighbouring points as a cluster if the distance between any point in an object and an adjacent point is less than a threshold $\varepsilon = 5$ cm, finishing when all the clusters are determined. Finally, we remove all small clusters with fewer than a threshold $\mu = 500$ points. Figure 2.6 illustrates an example of Euclidean clustering removal following the iterative plane removal.

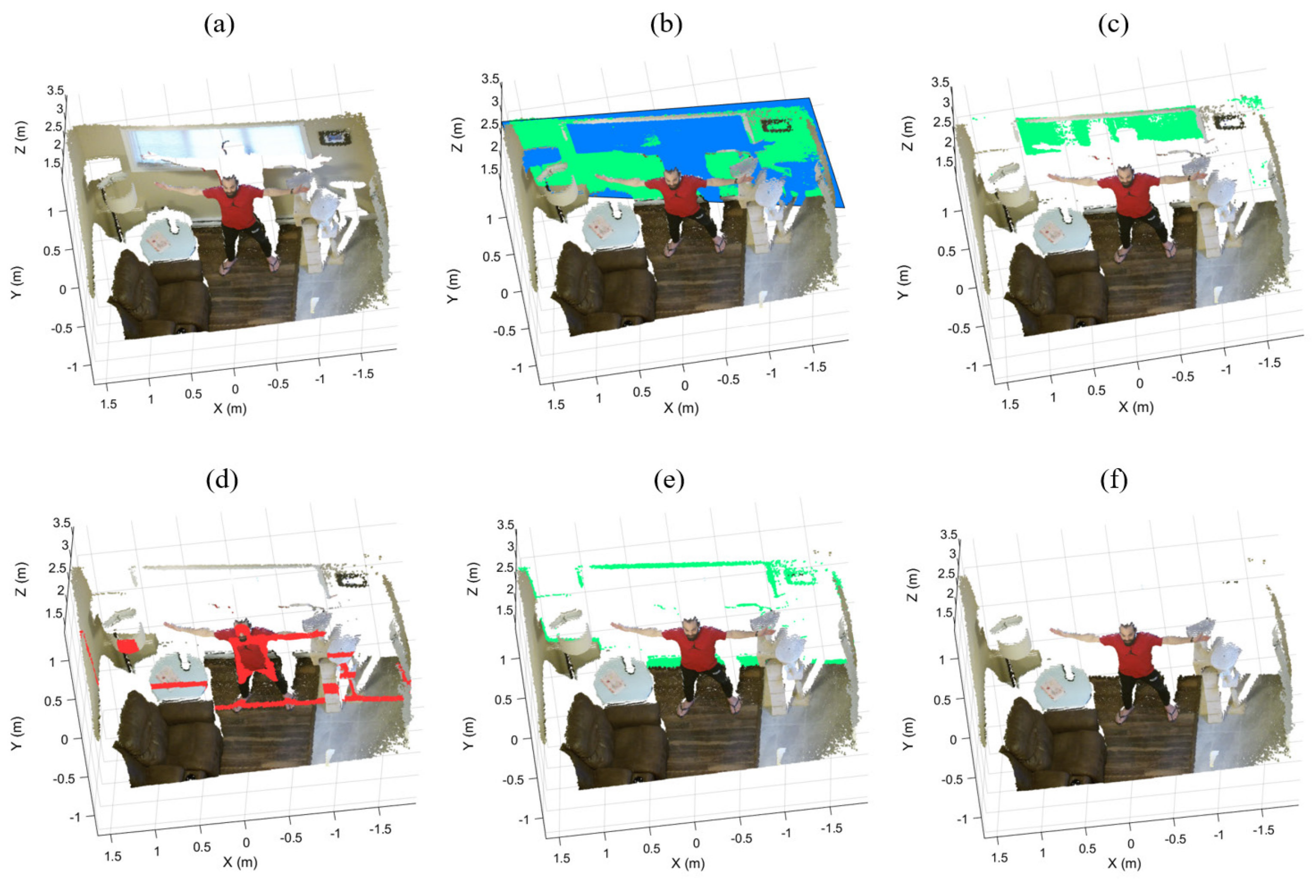


Figure 2.5 The iterative plane removal of the back region: (a) the sample point cloud, (b) the verified base plane, (c, e) the verified segmented planes, (d) the invalid segmented plane, and (f) the remaining point cloud after the back wall removal.

2.4 Experiments and Evaluation

2.4.1 Experimental Setup

We evaluated our method on three generated datasets representing three different complex indoor environments, Room-1, Room-2, and Room-3, as shown in Fig. 2.7a, 2.7c, and 2.7e respectively. Each dataset contains different objects, such as furniture, planar objects, and human bodies. To measure the performance of IR-RANSAC in our challenging environments, we acquired each point cloud from an arbitrary oblique-view location using the Microsoft Kinect V2 sensor. The details of each dataset after the preprocessing steps are listed in Table 2.3.

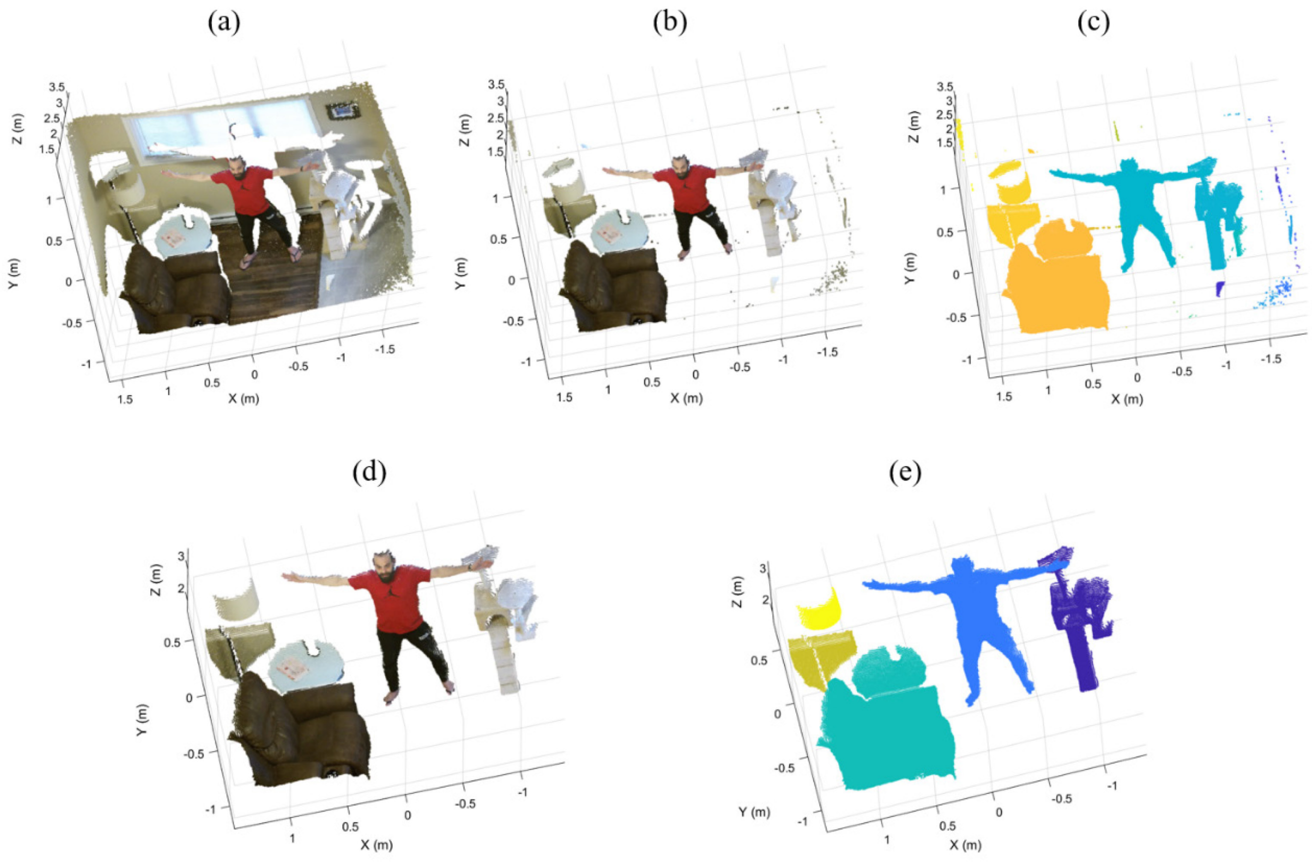


Figure 2.6 Euclidean clustering on the sample point cloud (a) following the iterative plane removal (b) results in properly segmented foreground object clusters as well as residual clusters (c). Removing object clusters with fewer than μ points leaves only foreground objects (d) which can be visualized as coloured objects (e).

To define the ground truth, we manually labeled the points belonging to each super-surface in every dataset by deploying the MATLAB Data tips tool based on their co-ordinates, colours, and the super-surface definition. We employed this lengthy and precise procedure by first using a semi-automated labeling approach using the orientation-based RANSAC and MATLAB Data tips tool. First, we segmented as many super-surface points as possible by running the orientation-based RANSAC in manually selected local regions known to contain super-surfaces. Then, we modified and validated the previous labeled points resulting in our final manually labeled super-surfaces shown in Fig. 2.7b, 2.7d, and 2.7f.

Table 2.3 Parameters of the datasets

Description	Width (m)	Height (m)	Depth (m)	Number of Points
Room-1	3.56	2.65	3.68	179,572
Room-2	3.51	2.60	3.46	179,374
Room-3	4.11	3.27	4.71	181,822

We evaluated the efficiency of our IR-RANSAC and RANSAC (as a baseline) in terms of four pixel-based metrics, precision, recall, F_1 score, and specificity. The first three parameters have been widely utilized for appraising the effectiveness of plane segmentation (e.g., [36], [51], and [56]). To compute these metrics, we defined true positive (TP) as bounding surface points correctly identified, true negative (TN) as foreground points correctly identified, false positive (FP) as foreground points incorrectly identified as bounding surface points, and false negative (FN) as bounding surface points incorrectly identified as foreground points. Precision, measured as $\frac{TP}{TP+FP}$, is the number of correctly removed points (i.e., true positives) with respect to the total number of removed points. Recall, measured as $\frac{TP}{TP+FN}$, is the fraction of true positive among the manually labeled points (ground truth). F_1 score, measured as $2 \times \frac{precision \times recall}{precision+recall}$, or the harmonic mean of the precision and recall, represents the overall performance of our proposed method. The specificity, measured as $\frac{TN}{TN+FP}$, reflects the true negative rate of our algorithm, providing a measure of how well our method distinguishes between foreground points and bounding surface points.

We computed the size reduction of IR-RANSAC as $1 - \frac{S_{OUT}}{S_{IN}}$ where S_{OUT} and S_{IN} are the size (i.e., the number of points) of the output point cloud and the input point cloud, respectively. We implemented our proposed algorithm running MATLAB on an Intel i5-4300M CPU @ 2.60

GHz and with 6.00 GB RAM. The full parameters of IR-RANSAC, determined through experimentation, are listed in Table 2.4 and used for all our experiments.

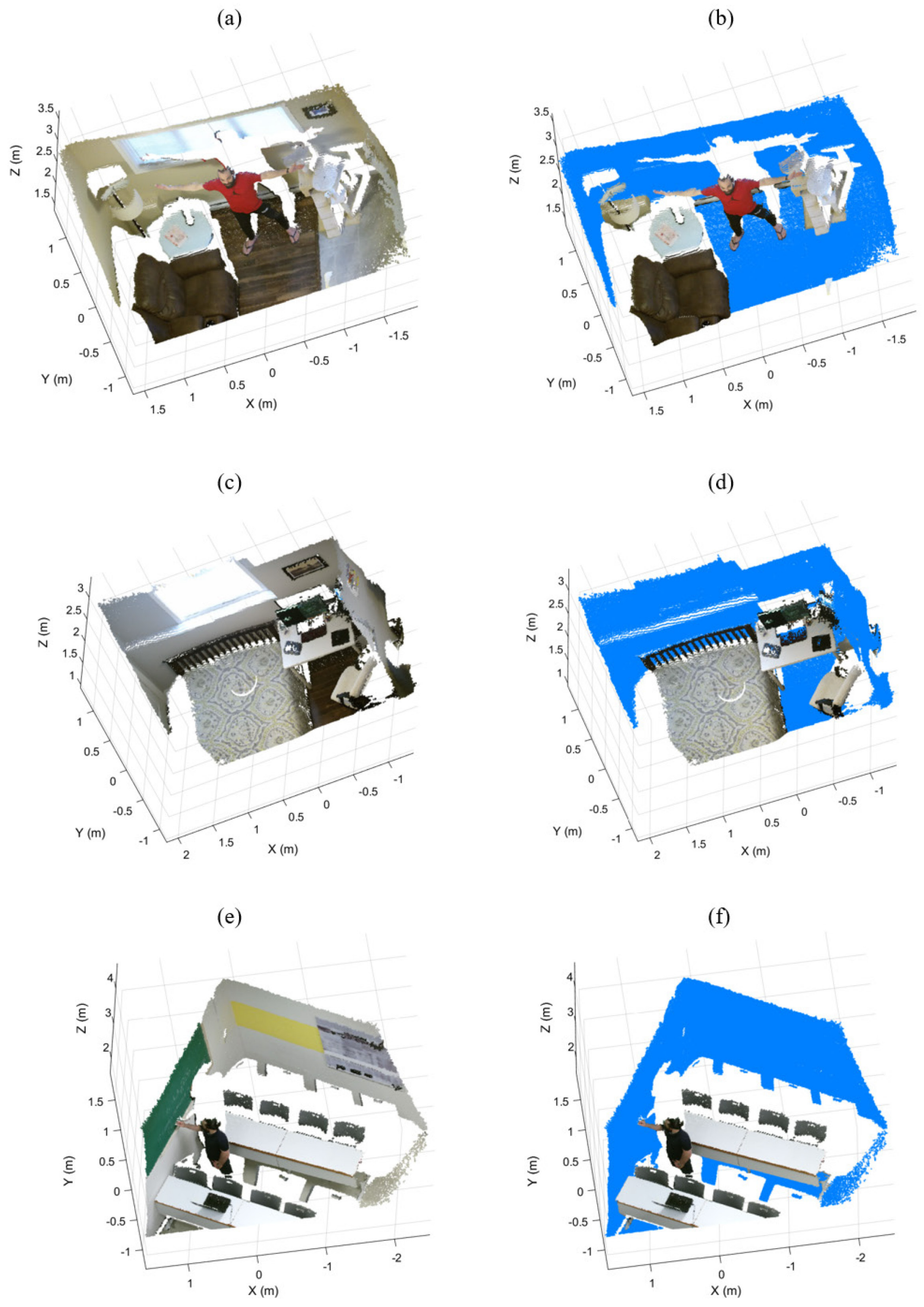
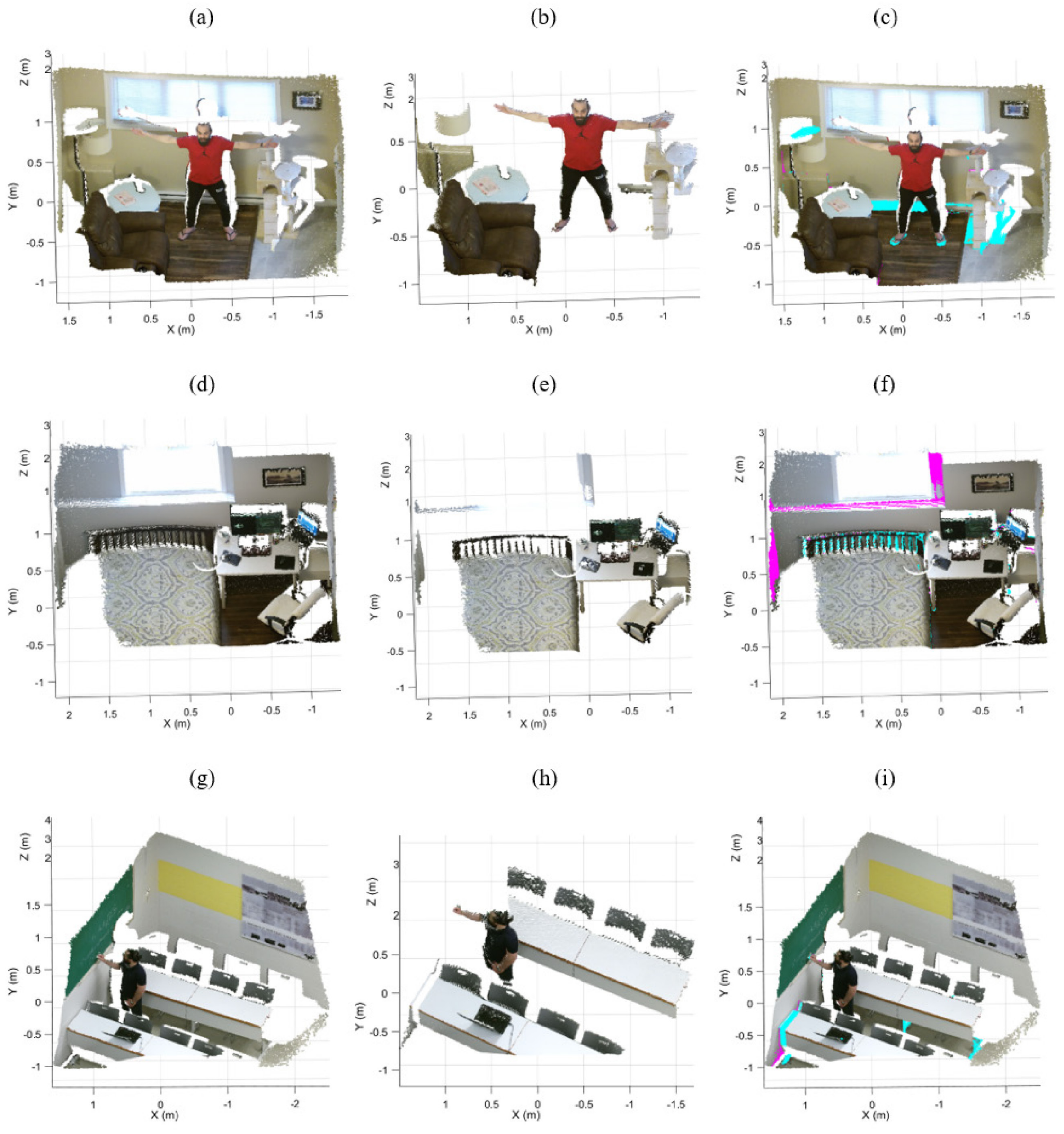


Figure 2.7 The generated datasets, Room-1 (a), Room-2 (c), and Room-3 (e), and their manually labeled super-surfaces in blue colour, (b), (d), and (f) respectively.

2.4.2 Experimental Results

Figure 2.8 shows the output results of our algorithm and erroneously classified points for the three datasets.



False Positive	🟦
False Negative	🟪

Figure 2.8 Output results of IR-RANSAC. The original point clouds, Room-1 (a), Room-2 (d), and Room-3 (g), are visualized with all super-surfaces removed (b), (e), and (h) respectively. The false positive (cyan) and false negative (magenta) points are highlighted over the original point clouds for the Room-1 (c), Room-2 (f) and Room-3 (i) datasets.

Table 2.4 Parameters of IR-RANSAC

Procedure	Descriptor	Parameter	Value
Denoising algorithm	Nearest neighbours	K	4
	Outlier threshold	σ	1 standard deviation
Base plane segmentation	RANSAC distance threshold	δ	4 cm
	Maximum allowance angular variation	ω	45°
	RANSAC maximum iterations	T	2% of region points
	Verification threshold	υ	5% of point cloud points
Iterative plane removal	Iterations	I	3
	Maximum allowance angular variation	θ	5°
	Distance threshold between two planes	α	10 cm
Euclidean clustering removal	Euclidean clustering threshold	μ	500 points
	Euclidean distance threshold	ε	5 cm

Incorrectly classified points are almost always associated with the points belonging to foreground objects that are contacting a super-surface (i.e., objects within the RANSAC distance threshold), such as the baseboard heater, the bed headboard, and the desk legs (Fig. 2.8c, 2.8f, and 2.8i respectively). Notably, as a result of our local region definitions and bounding surface criteria, we successfully prevented consideration of foreground objects that could be misidentified as bounding surfaces (e.g., beds, desks). Furthermore, foreground objects comprised of fewer points than the Euclidean clustering threshold ($\mu = 500$ points) were erroneously removed from the point cloud (e.g., the cyan portion of the lamp in Fig. 2.8c).

Our IR-RANSAC method eliminated nearly all points belonging to the super-surfaces from the Room-1 and Room-3 datasets. However, IR-RANSAC failed to remove two small challenging regions belonging to the back and left super-surfaces of Room-2 (the magenta regions around the window and the bottom left corner of the point cloud in Fig. 2.8f). This is because the planes surrounding the window are perpendicular to the back super-surface, and the other small plane in the bottom left corner is smaller than the verification threshold ($\upsilon = 5\%$ of the total number of points in the point cloud). Notably, both of these two miss-classified regions are greater than the Euclidean clustering threshold ($\mu = 500$ points). Additionally, IR-RANSAC incorrectly assigned a small number of points belonging to foreground objects to a bounding surface. In all cases, these incorrect assignment of foreground points to super-surfaces happened when clustered foreground objects physically contacted super-surfaces (e.g., bed headboard in Fig. 2.8f and desk legs in Fig. 2.8i).

Figure 2.9 shows the output results of the standard RANSAC plane removal for the three datasets. The cyan and magenta colours represent the incorrectly removed regions (i.e., false positive) and undetected regions (i.e., false negative), respectively. The standard RANSAC approach failed to remove many points belonging to the super-surfaces (e.g., the back and right walls of Room-1 and the floors of the other two datasets). Moreover, RANSAC erroneously removed some parts of the human body and the furniture (e.g., couch, bed, and desks). These false segmentations have three main reasons: RANSAC sensitivity to clutter and occlusion, the uncertainty of RANSAC in randomly sampling three points as a minimum subset, and the lack of an orientation constraint.

Our experimental results suggest that IR-RANSAC supports varied sensor locations and removes background boundary surfaces more effectively without removing foreground data in complex 3D indoor environments.

2.4.3 Evaluation

Similar to RANSAC, our IR-RANSAC is stochastic, and accordingly its results can vary depending on the selection of the random subsample. To account for this stochasticity, we conducted 30 experiments on each dataset, similar to the work of Li et al. [39], and computed our four evaluation metrics for IR-RANSAC and standard RANSAC plane removal as a baseline comparator. We illustrate the evaluation results in Fig. 2.10, with the left column visualizing our

IR-RANSAC results and the right column representing our standard RANSAC plane removal results, and the rows corresponding to the three rooms. We implemented the standard RANSAC plane removal traditionally to segment and eliminate the four largest planes in each dataset. We set the parameters of our benchmark RANSAC method to be the same as those used in IR-RANSAC. The mean (M) and standard deviation (SD) of both approaches are shown in Table 2.5 for specificity, precision, recall, and F_1 score, along with execution times.

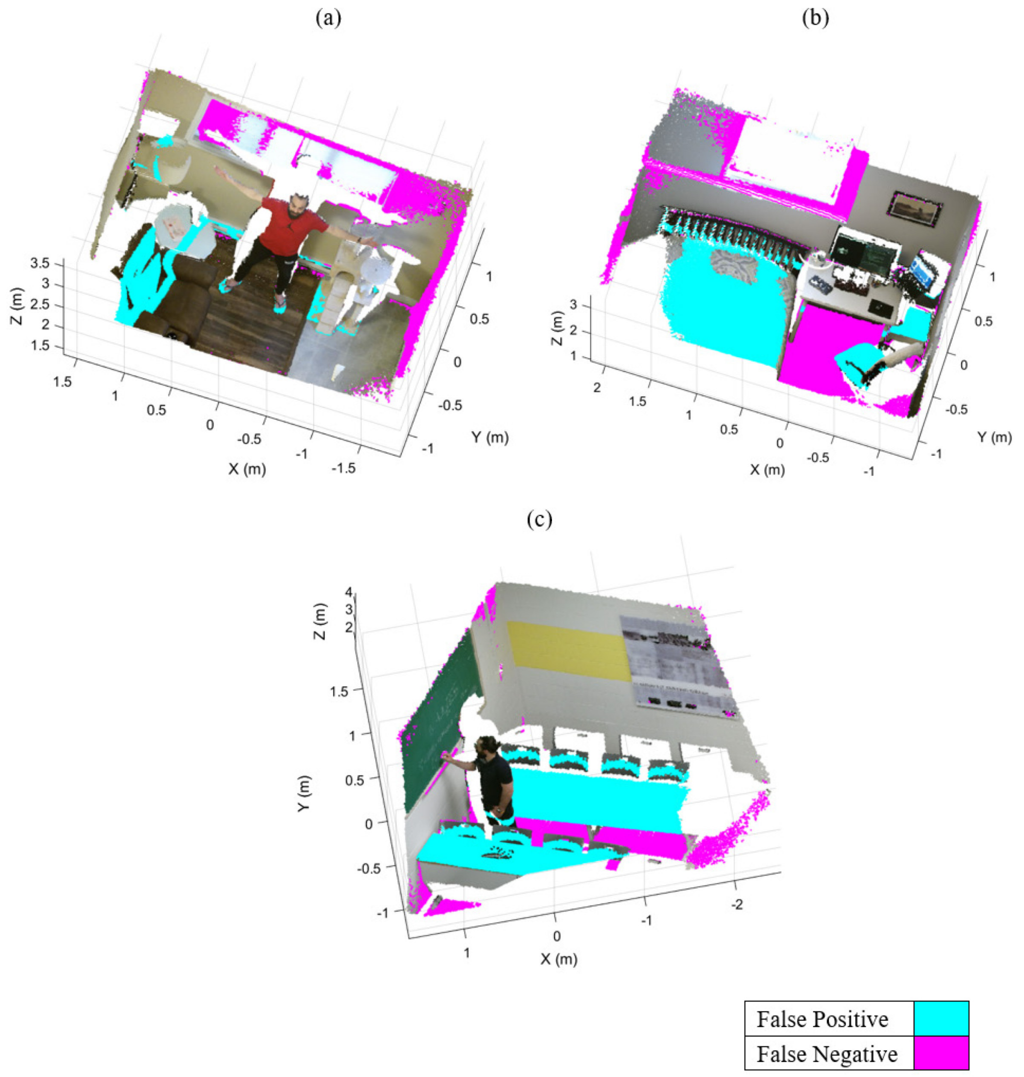


Figure 2.9 The output results of standard RANSAC plane removal and its false positives and false negatives in cyan and magenta, respectively: (a) the Room-1, (b) the Room-2, and (c) the Room-3.

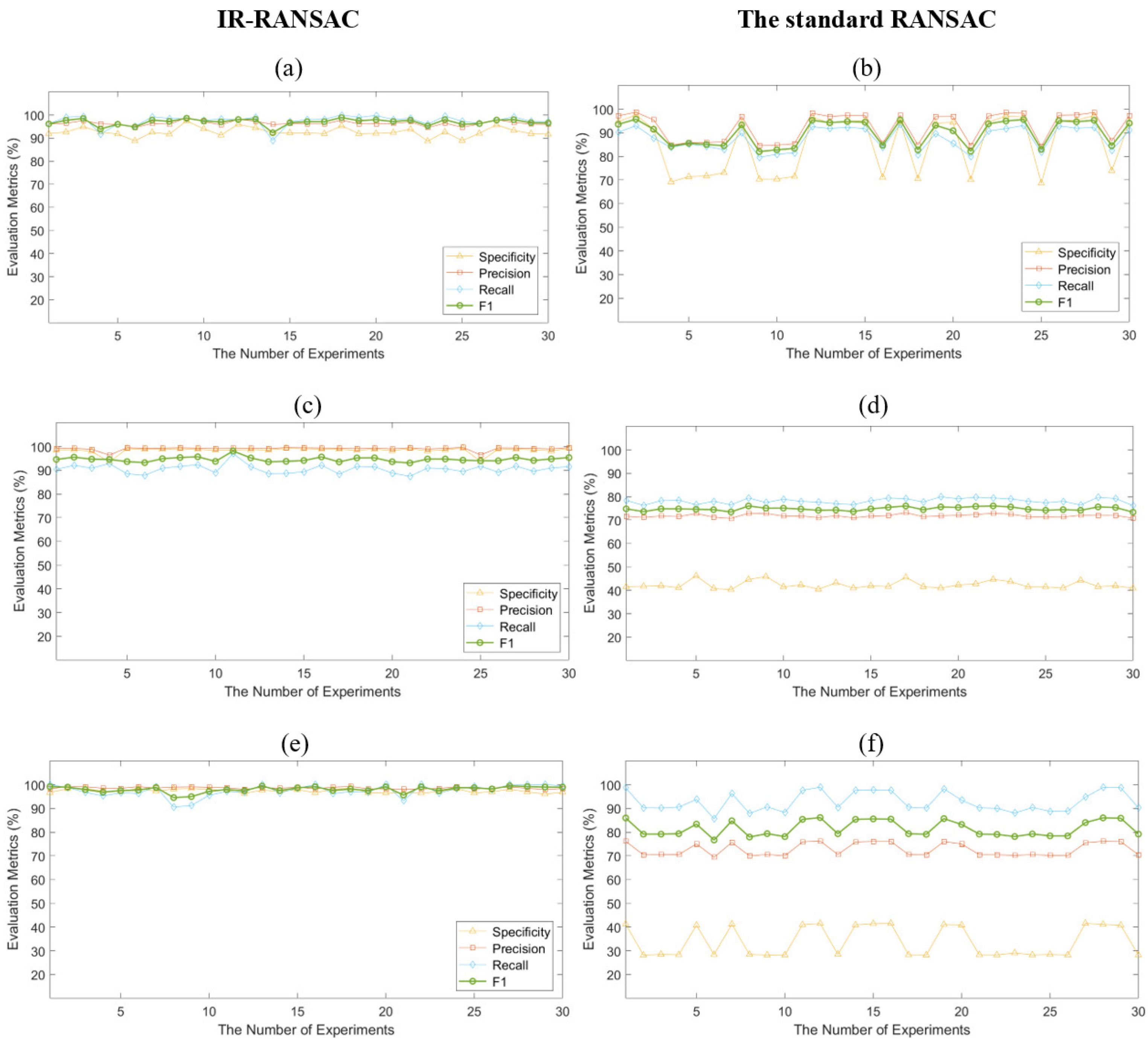


Figure 2.10 The evaluation results of IR-RANSAC and the standard RANSAC plane removal for the three datasets: (a) and (b) the Room-1, (c) and (d) the Room-2, (e) and (f) the Room-3.

Given the original size of the point clouds (see Table 2.3), and the size of the output point clouds for Room-1 (55,705 points), Room-2 (69,044 points), and Room-3 (66,149 points), IR-RANSAC yielded a size reduction of 0.68, 0.62, and 0.64 for Room-1, Room-2, and Room-3, respectively.

Table 2.5 Execution times, mean (M), and standard deviation (SD) of IR-RANSAC and standard RANSAC for specificity, precision, recall, and F1 score.

Dataset	Method	Specificity		Precision		Recall		F_1		Runtime (s)
		M (%)	SD	M (%)	SD	M (%)	SD	M (%)	SD	
Room-1	IR-RANSAC	92.60	0.0200	96.41	0.0097	97.42	0.0230	96.90	0.0142	8.83
	Standard RANSAC	85.47	0.1210	92.56	0.0612	87.60	0.0484	90.00	0.0538	3.20
Room-2	IR-RANSAC	98.38	0.0135	99.06	0.0074	90.52	0.0193	94.59	0.0101	7.50
	Standard RANSAC	42.37	0.0164	71.79	0.0067	78.10	0.0114	74.81	0.0077	2.80
Room-3	IR-RANSAC	97.39	0.0075	98.61	0.0038	97.47	0.0253	98.02	0.0126	6.37
	Standard RANSAC	33.91	0.0644	72.72	0.0278	92.78	0.0419	81.53	0.0333	2.40

2.5 Discussion

Our evaluation results achieved with IR-RANSAC (the first column of Fig. 2.10) are higher and much more consistent in all the four evaluation metrics than those obtained using standard RANSAC (the second column of Fig. 2.10). Additionally, almost all four scores have a lower standard deviation with IR-RANSAC compared to standard RANSAC. Overall, all of our evaluation results were statistically significantly better ($p < 0.05$) with IR-RANSAC than standard RANSAC using a two-sample t-test. Most notably, the F_1 score, which represents the overall performance of the approaches, was statistically higher with IR-RANSAC than standard RANSAC.

In all experiments, our proposed IR-RANSAC method obtained average values above 92% for specificity, 96% for precision, 90% for recall, and 94% for F_1 score. Comparably, the standard RANSAC achieved average values between 33% and 85%, 71% and 92%, 78% and 92%, 74%

and 90% for specificity, precision, recall, and F_1 score, respectively. As illustrated in the second column of Fig. 2.10, there are also many sharp fluctuations in the standard RANSAC evaluation results. The F_1 score fluctuated from 82% to 95% and 76% to 86% for Room-1 and Room-3, respectively. However, it almost remained steady at 74% for the Room-2 dataset. The standard RANSAC approach demonstrated a very low specificity for Room-2 and Room-3, containing many planar furniture.

IR-RANSAC takes about three times as long to execute when compared to the standard RANSAC approach on the same datasets. This is expected though, since our IR-RANSAC method invokes the RANSAC algorithm four times more than the standard RANSAC plane removal. Theoretically, a faster version of IR-RANSAC, which has only one iteration in each local region, can be implemented that would be faster than the standard RANSAC approach because it runs the RANSAC algorithm in smaller regions.

Our evaluation results support that IR-RANSAC is a robust and reliable method for removing the bounding super-surfaces of a complex 3D indoor environment with better performance over traditional RANSAC in all ways except execution time. Our results suggest that IR-RANSAC removes background boundary surfaces effectively without removing foreground data, and can considerably reduce size of 3D point clouds.

The subjects of future research are speeding up the IR-RANSAC algorithm and improving its results in much more complex 3D indoor environments. To improve our algorithm results, we need to reduce its reliance on the Euclidean clustering technique, eliminate the small challenging regions belonging to a super-surface but with a different normal vector (e.g., the highlighted regions around the window in Fig. 2.8f), and implement self-adaptive parameters to be robust to different indoor environments and sensor data.

2.6 Conclusions

We have presented a 3D bounding surface removal technique, IR-RANSAC, that is particularly suited to more challenging and cluttered indoor environments. IR-RANSAC supports varied sensor perspectives for organized and unorganized point clouds, and it considerably reduces the size of 3D datasets. Moreover, IR-RANSAC can improve the results and the running time of different 3D computer vision methods by reducing their search space. After downsampling and denoising a point cloud captured from an oblique view, we divide the point cloud space into four overlapped local regions, exploiting knowledge of the current view of the sensor, and segment a base plane in each of the four regions. We then expand our search space around the base plane in each region, and iteratively segment and remove the remaining points belonging to each super-surface. Finally, we cluster the remaining point cloud using the geometric relationship between groups of points, resulting in a final point cloud comprised only of clustered objects of interest. We evaluated the performance of IR-RANSAC in terms of four metrics: specificity, precision, recall, and F_1 score, on the three generated datasets acquired from an arbitrary oblique-view location and representing different indoor environments. Our experiments demonstrated that our proposed method is a robust super-surface removal and size reduction technique for complex 3D indoor environments. Experimentally, IR-RANSAC outperformed traditional RANSAC segmentation in all categories, supporting our efforts to prioritize the inclusion of all bounding points in each super-surface, while minimizing inclusion of points that belong to foreground objects.

Our intention was to develop a robust method of bounding surface segmentation, maximizing inclusion of bounding surface points and minimizing inclusion of foreground points. Our experimental data suggest that by conceptualizing bounding surfaces (e.g., walls and floor) as unique and different than other large surfaces that belong to foreground objects, it is possible to improve on methods of segmenting and removing these unwanted bounding surfaces specifically. By removing these bounding surfaces and preserving foreground objects, we considerably reduce the size of the resulting dataset, substantially improving downstream storage and processing.

2.7 References

[1] S. H. Shaikh, K. Saeed, and N. Chaki, “Moving object detection using background subtraction,” in Springer Briefs in Computer Science, no. 9783319073859, Springer, 2014, pp. 15–23. doi: 10.1007/978-3-319-07386-6_3.

[2] S. Kumar and J. Sen Yadav, “Video object extraction and its tracking using background subtraction in complex environments,” Perspectives in Science, vol. 8, pp. 317–322, Sep. 2016, doi: 10.1016/j.pisc.2016.04.064.

[3] A. Aggarwal, S. Biswas, S. Singh, S. Sural, and A. K. Majumdar, “Object tracking using background subtraction and motion estimation in MPEG videos,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer, Berlin, Heidelberg, 2006, pp. 121–130. doi: 10.1007/11612704_13.

[4] R. Manikandan, R. Ramakrishnan, and R. Scholar, “Human Object Detection and Tracking using Background Subtraction for Sports Applications,” International Journal of Advanced Research in Computer and Communication Engineering, vol. 2, 2013.

[5] S. Czarnuch, S. Cohen, V. Parameswaran, and A. Mihailidis, “A real-world deployment of the COACH prompting system,” Journal of Ambient Intelligence and Smart Environments, vol. 5, no. 5, pp. 463–478, Jan. 2013, doi: 10.3233/AIS-130221.

[6] W. Zou, Y. Lu, M. Chen, and F. Lv, “Rapid face detection in static video using background subtraction,” in Proceedings - 2014 10th International Conference on Computational Intelligence and Security, CIS 2014, Institute of Electrical and Electronics Engineers Inc., Jan. 2015, pp. 252–255. doi: 10.1109/CIS.2014.146.

[7] H. Yang and S. Qu, “Real-time vehicle detection and counting in complex traffic scenes using background subtraction model with low-rank decomposition,” IET Intelligent Transport Systems, vol. 12, no. 1, pp. 75–85, Feb. 2018, doi: 10.1049/iet-its.2017.0047.

[8] A. Kumar Sahu and A. Choubey, “Motion Detection Surveillance System Using Background Subtraction Algorithm,” International Journal of Advance Research in Computer Science and Management Studies, vol. 1, no. 6, 2013.

[9] S. Hargude and S. R. Idate, “I-Surveillance : Intelligent Surveillance System Using Background Subtraction Technique,” in Proceedings - 2nd International Conference on Computing,

Communication, Control and Automation, ICCUBEA 2016, Institute of Electrical and Electronics Engineers Inc., Feb. 2017. doi: 10.1109/ICCUBEA.2016.7860046.

[10] M. Karaman, L. Goldmann, D. Yu, and T. Sikora, "Comparison of static background segmentation methods," in Visual Communications and Image Processing 2005, SPIE, 2005, pp. 2140–2151.

[11] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 246–252, 1999, doi: 10.1109/cvpr.1999.784637.

[12] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in Proceedings - International Conference on Pattern Recognition, Institute of Electrical and Electronics Engineers Inc., 2004, pp. 28–31. doi: 10.1109/icpr.2004.1333992.

[13] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," Real-Time Imaging, vol. 11, no. 3, pp. 172–185, Jun. 2005, doi: 10.1016/j.rti.2004.12.004.

[14] J. M. Guo, Y. F. Liu, C. H. Hsia, M. H. Shih, and C. S. Hsu, "Hierarchical method for foreground detection using codebook model," IEEE Transactions on Circuits and Systems for Video Technology, vol. 21, no. 6, pp. 804–815, Jun. 2011, doi: 10.1109/TCSVT.2011.2133270.

[15] D. M. Tsai and S. C. Lai, "Independent component analysis-based background subtraction for indoor surveillance," IEEE Transactions on Image Processing, vol. 18, no. 1, pp. 158–167, 2009, doi: 10.1109/TIP.2008.2007558.

[16] H. Jiménez-Hernández, "Background Subtraction Approach Based on Independent Component Analysis," Sensors, vol. 10, no. 6, pp. 6092–6114, Jun. 2010, doi: 10.3390/s100606092.

[17] G. Gordon, T. Darrell, M. Harville, and J. Woodfill, "Background estimation and removal based on range and color," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 459–464, 1999, doi: 10.1109/cvpr.1999.784721.

[18] S. Czarnuch and A. Mihailidis, "Development and evaluation of a hand tracker using depth images captured from an overhead perspective," Disability and Rehabilitation: Assistive Technology, vol. 11, no. 2. Taylor and Francis Ltd, pp. 150–157, Feb. 17, 2016. doi: 10.3109/17483107.2015.1027304.

[19] J. Shotton et al., "Real-time human pose recognition in parts from single depth images," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, 2011, pp. 1297–1304. doi: 10.1109/CVPR.2011.5995316.

[20] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother, "Bi-layer segmentation of binocular stereo video," in Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, IEEE Computer Society, 2005, pp. 407–414. doi: 10.1109/CVPR.2005.91.

[21] E. J. Fernandez-Sanchez, J. Diaz, and E. Ros, "Background subtraction based on color and depth using active sensors," Sensors (Basel), vol. 13, no. 7, pp. 8895–8915, 2013, doi: 10.3390/s130708895.

[22] M. Cristani, M. Farenzena, D. Bloisi, and V. Murino, “Background subtraction for automated multisensor surveillance: A comprehensive review,” EURASIP J Adv Signal Process, vol. 2010, 2010, doi: 10.1155/2010/343057.

[23] W. Zhou, J. Yuan, J. Lei, and T. Luo, “TSNet: Three-stream Self-attention Network for RGB-D Indoor Semantic Segmentation,” IEEE Intell Syst, vol. 1672, no. c, 2020, doi: 10.1109/MIS.2020.2999462.

[24] S. Ottonelli, P. Spagnolo, P. L. Mazzeo, and M. Leo, “Improved video segmentation with color and depth using a stereo camera,” Proceedings of the IEEE International Conference on Industrial Technology, pp. 1134–1139, 2013, doi: 10.1109/ICIT.2013.6505832.

[25] “The Point Cloud Libraryle.” [Online]. Available: https://pointclouds.org/

[26] S. Muthu, R. Tennakoon, T. Rathnayake, R. Hoseinnezhad, D. Suter, and A. Bab-Hadiashar, “Motion Segmentation of RGB-D Sequences: Combining Semantic and Motion Information Using Statistical Inference,” IEEE Transactions on Image Processing, vol. 29, pp. 5557–5570, 2020, doi: 10.1109/TIP.2020.2984893.

[27] N. Vaskevicius, A. Birk, K. Pathak, and S. Schwertfeger, “Efficient representation in three-dimensional environment modeling for planetary robotic exploration,” Advanced Robotics, vol. 24, no. 8–9, pp. 1169–1197, 2010, doi: 10.1163/016918610X501291.

[28] T. Czerniawski, M. Nahangi, S. Walbridge, and C. Haas, “Automated removal of planar clutter from 3D point clouds for improving industrial object recognition,” ISARC 2016 - 33rd International Symposium on Automation and Robotics in Construction, no. Isarc, pp. 357–365, 2016, doi: 10.22260/isarc2016/0044.

[29] R. Kaushik and J. Xiao, “Accelerated patch-based planar clustering of noisy range images in indoor environments for robot mapping,” Rob Auton Syst, vol. 60, no. 4, pp. 584–598, 2012, doi: 10.1016/j.robot.2011.12.001.

[30] C. Zhang, S. C.-I. Access, and undefined 2020, “Perspective Independent Ground Plane Estimation by 2D and 3D Data Analysis,” ieeexplore.ieee.org.

[31] S. Chen, D. Tian, C. Feng, A. Vetro, and J. Kovačević, “Fast resampling of three-dimensional point clouds via graphs,” IEEE Transactions on Signal Processing, vol. 66, no. 3, pp. 666–681, 2018, doi: 10.1109/TSP.2017.2771730.

[32] M. A. Fischler and R. C. Bolles, “Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,” Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[33] D. H. Ballard, “Generalizing the Hough transform to detect arbitrary shapes,” Pattern Recognit, vol. 13, no. 2, pp. 111–122, 1981, doi: 10.1016/0031-3203(81)90009-1.

[34] G. Vosselman, G. Sithole, G. Vosselman, B. G. H. Gorte, G. Sithole, and T. Rabbani, “Recognising structure in laser scanner point clouds RECOGNISING STRUCTURE IN LASER SCANNER POINT CLOUDS 1,” 2003.

[35] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter, “The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design,” 3D Research, vol. 2, no. 2, pp. 1–13, Nov. 2011, doi: 10.1007/3DRes.02(2011)3.

[36] A. V. Vo, L. Truong-Hong, D. F. Laefer, and M. Bertolotto, “Octree-based region growing for point cloud segmentation,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 104, pp. 88–100, 2015, doi: 10.1016/j.isprsjprs.2015.01.011.

[37] T. M. Awwad, Q. Zhu, Z. Du, and Y. Zhang, “An improved segmentation approach for planar surfaces from unstructured 3D point clouds,” Photogrammetric Record, vol. 25, no. 129, pp. 5–23, 2010, doi: 10.1111/j.1477-9730.2009.00564.x.

[38] D. Chen, L. Zhang, P. T. Mathiopoulos, and X. Huang, “A methodology for automated segmentation and reconstruction of urban 3-D buildings from ALS point clouds,” IEEE J Sel Top Appl Earth Obs Remote Sens, vol. 7, no. 10, pp. 4199–4217, Oct. 2014, doi: 10.1109/JSTARS.2014.2349003.

[39] L. Li, F. Yang, H. Zhu, D. Li, Y. Li, and L. Tang, “An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells,” Remote Sens (Basel), vol. 9, no. 5, 2017, doi: 10.3390/rs9050433.

[40] 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,” ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007, vol. XXXVI, no. 1, pp. 407–412, 2007.

[41] D. Tóvári and N. Pfeifer, “Segmentation based robust interpolation - A newapproach to laser data filtering,” International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, vol. 36, pp. 79–84, 2005.

[42] A. Nurunnabi, D. Belton, and G. West, “Robust segmentation in laser scanning 3D point cloud data,” in 2012 International Conference on Digital Image Computing Techniques and Applications, DICTA 2012, IEEE eXpress Conference Publishing, Nov. 2012, pp. 1–8. doi: 10.1109/DICTA.2012.6411672.

[43] J.-E. Deschaud and F. Goulette, “A Fast and Accurate Plane Detection Algorithm for Large Noisy Point Clouds Using Filtered Normals and Voxel Growing,” 2010.

[44] M. Huang, P. Wei, and X. Liu, “An Efficient Encoding Voxel-Based Segmentation (EVBS) Algorithm Based on Fast Adjacent Voxel Search for Point Cloud Plane Segmentation,” Remote Sens (Basel), vol. 11, no. 23, p. 2727, Nov. 2019, doi: 10.3390/rs11232727.

[45] J. Xiao, J. Zhang, B. Adler, H. Zhang, and J. Zhang, “Three-dimensional point cloud plane segmentation in both structured and unstructured environments,” Rob Auton Syst, vol. 61, no. 12, pp. 1641–1652, Dec. 2013, doi: 10.1016/j.robot.2013.07.001.

[46] X. Leng, J. Xiao, and Y. Wang, “A multi-scale plane-detection method based on the Hough transform and region growing,” Photogrammetric Record, vol. 31, no. 154, pp. 166–192, 2016, doi: 10.1111/phor.12145.

[47] O. Teboul, L. Simon, P. Koutsourakis, and N. Paragios, "Segmentation of building facades using procedural shape priors," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3105–3112, 2010, doi: 10.1109/CVPR.2010.5540068.

[48] S. Filin, "Surface clustering from airborne laser scanning data," International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, vol. 34, 2002.

[49] S. Filin and N. Pfeifer, "Segmentation of airborne laser scanning data using a slope adaptive neighborhood," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 60, no. 2, pp. 71–80, 2006, doi: 10.1016/j.isprsjprs.2005.10.005.

[50] G. Zhou, S. Cao, and J. Zhou, "Planar Segmentation Using Range Images from Terrestrial Laser Scanning," IEEE Geoscience and Remote Sensing Letters, vol. 13, no. 2, pp. 257–261, 2016, doi: 10.1109/LGRS.2015.2508505.

[51] Z. Dong, B. Yang, P. Hu, and S. Scherer, "An efficient global energy optimization approach for robust 3D plane segmentation of point clouds," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 137, pp. 112–133, 2018, doi: 10.1016/j.isprsjprs.2018.01.013.

[52] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets," Auton Robots, 2013, doi: 10.1007/s10514-013-9327-2.

[53] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in Proceedings - IEEE International Conference on Robotics and Automation, 2011. doi: 10.1109/ICRA.2011.5980567.

[54] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D Point cloud based object maps for household environments," Rob Auton Syst, vol. 56, no. 11, pp. 927–941, 2008, doi: 10.1016/j.robot.2008.08.005.

[55] W. Förstner and B. P. Wrobel, Photogrammetric Computer Vision. 2016. doi: 10.1016/S0076-5392(09)60371-4.

[56] J. Yan, J. Shan, and W. Jiang, "A global optimization approach to roof segmentation from airborne lidar point clouds," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 94, pp. 183–193, 2014, doi: 10.1016/j.isprsjprs.2014.04.022.

Chapter 3. Bounding Surface Segmentation and Removal in Complex 3D Indoor Environments Using Orientation-based MSAC

Co-authorship statement – This chapter was accepted as a full conference paper titled "Bounding Surface Segmentation and Removal in Complex 3D Indoor Environments Using Orientation-based MSAC" and presented at the 30th Annual Newfoundland Electrical and Computer Engineering Conference (NECEC), St. John's, Canada, in November 2021. Ali Ebrahimi, as the primary author, conducted the literature review, conceptualized the study, designed the methodology, developed the method, collected data, analyzed the results, created visualizations, presented the paper, and both drafted and revised the manuscript. The co-author, Dr. Stephen Czarnuch, provided essential guidance and supervision throughout the research process, and meticulously reviewed and revised the manuscript. Both authors have read and agreed to the final version of the manuscript.

3.1 Abstract

Indoor bounding surfaces (e.g., walls, windows, curtains, and floor) can decrease the performance of 3D computer vision algorithms (e.g., object segmentation and tracking) by cluttering their search space. Therefore, a robust removal technique for points that belong to surfaces at the outer boundaries of the data can significantly reduce the search space and improve downstream results. In this paper, we introduce an orientation-based bounding surface removal technique using the M-estimator SAmple Consensus (MSAC) algorithm. Our method removes background bounding surfaces of challenging (i.e., cluttered and dynamic) real-world scenes while minimizing the inclusion of points that belong to foreground objects such as human bodies. We developed our method to work with unorganized data captured from commercial depth sensors with varied perspectives. First, we preprocess the input point cloud and divide it into four overlapped local regions based on the current view of the sensor. Then, we apply an orientation-based surface removal approach to all four regions to segment and remove the bounding surfaces. Next, we remove irrelevant small segments, keeping only objects of interest. Finally, we evaluate the performance of our proposed method using four conventional metrics: specificity, precision, recall, and F_1 score, on three generated datasets representing different indoor environments. Our experimental results demonstrate that our proposed method robustly removes bounding surfaces for complex 3D indoor environments while scoring all the evaluation metrics above 93%.

Keywords – MSAC, point cloud, bounding surface removal, wall removal, 3D background subtraction, 3D plane segmentation, 3D preprocessing technique, 3D size reduction

3.2 Introduction

Background subtraction is widely used in object detection, and tracking approaches [1-3] with broad applications such as human tracking [4,5], face recognition [6], traffic management [7], and surveillance systems [8,9]. Established 2D background subtraction approaches are based on static background segmentation methods [10], adaptive Gaussian mixture models [11,12], real-time codebook models [13,14], and independent component analysis-based techniques [15,16]. Although advanced 2D background subtraction techniques can handle gradual illumination changes and repetitive movements in the background, they perform poorly in the presence of shadows or foreground regions with colours similar to the background [17]. The release of commercially available and inexpensive depth sensors such as Microsoft Kinect opened new doors for improved background subtraction techniques because of the availability of additional depth data associated with each pixel of colour data. Depth sensor (RGB-D) systems are more robust for background detection problems compared to classic colour-based systems because depth data are largely invariant to colour, texture, shape, and lighting [18,19].

Background subtraction methods are generally used to analyze individual images and identify the foreground by first estimating a reference background that is developed from historical information obtained from videos or sequences of images. Therefore, the classic application of 2D background subtraction is separating dynamic or moving objects from a relatively static or slow-changing background scene. However, RGB images only contain intensity information and spatial information that is largely restricted to the two dimensions of the image that are perpendicular to the camera's perspective. Accordingly, identifying the boundaries between objects, or conversely identifying object interactions or contact, is limited mainly to detectable intensity differences. In applications that utilize RGB-D data, interactions or contact between objects and object spatial relationships can be more directly measured.

Reliably identifying and removing static background components (e.g., roads and walls) before modeling the background can result in both improved background subtraction and improved foreground segmentation using both 2D and RGB-D data. Identifying static background components suffers from the same limitations as modeling the entire background using 2D data, suggesting that little benefit is afforded by first removing these background objects, then modeling the background. However, with RGB-D data, parametrically modeled objects (e.g., planes, spheres, cones, cylinders, and cubes) are far more reliably detectable. As a result, researchers have attempted to segment or remove large planar surfaces (e.g., walls, ceiling, and floor surfaces) as a preprocessing or fundamental step before all other algorithms (e.g., [20-22]).

In general, large planar surfaces comprise a large percentage of points within each frame of RGB-D data captured in indoor environments. However, outside specific applications that seek to identify significant surfaces (e.g., ground plane detection), large planar surfaces are not often the objects of interest in 3D computer vision applications. Notably, smaller planar surfaces (e.g., tabletops, chair seats and backs, desks) are more likely to be of interest than larger surfaces at the boundaries of the scene. Furthermore, the large bounding surfaces can decrease the performance of 3D computer vision algorithms (e.g., object segmentation and tracking) by cluttering their search space. Therefore, a robust removal technique for points that belong to surfaces at the outer boundaries of the RGB-D data can significantly reduce the search space and bring three main benefits to the computer vision systems: improving downstream results, speeding up downstream processes, and reducing the overall size of the point clouds.

In our previous work [23], we proposed a robust bounding surface removal and size reduction technique, Iterative Region-based RANSAC (IR-RANSAC), for complex 3D indoor environments. In this paper, we extend our previous method and evaluate the performance of the extended version against IR-RANSAC.

3.3 Methodology

Our proposed method has four main steps. We begin with two preprocessing techniques, first downsampling the raw point cloud and then removing noisy or outlying points in the depth map. Second, we implement an MSAC based plane removal technique, segmenting and removing the bounding planes. Third, we introduce a distance-based point removal technique to remove the remaining parts of the bounding surfaces. Finally, we cluster the remaining point cloud using the geometric relationship between groups of points, resulting in a final point cloud comprised only of clustered objects of interest.

3.3.1 Preprocessing

Since input point clouds are generally large in size due to the significant number of 3D points and associated colour information, a downsampling method with low computational complexity can significantly reduce the running time of point cloud processing algorithms. We utilize a voxelized grid approach [24] that returns the centroid of all the points in each 3D voxel grid with a leaf size of 0.1cm. In this way, the downsampled point clouds will still reflect the structure and maintain the geometric properties of the original point cloud while reducing the total amount of points that will need to be processed and stored.

Removing noisy points is a critical point cloud preprocessing task. Noisy or spurious points have two significant impacts on our approach. A noisy point cloud with false or spurious data points, including points outside of a scene's real boundaries can lead to a wrong measurement of the overall bounding box containing the point cloud, resulting in the definition of incorrect local regions in our subsequent processing steps. Furthermore, noisy points within the point cloud itself will effectively skew or change the geometry of the true objects. We utilize a statistical outlier removal approach [25] by examining the k-nearest neighbours (K = 4) of each point, and removing all points with a distance (σ) of more than one standard deviation of the mean distance to the query point to remove outliers of each captured point cloud. If the average distance of a point to its k-nearest neighbours is above the threshold (σ), it is considered as an outlier. In this way, we remove points that are dissimilar from other points in their neighborhood. Together, these approaches decrease the number of points in the point cloud, reducing downstream processing time and increasing the accuracy of our process.

3.3.2 MSAC Plane Removal

First, we employ our point cloud partitioning approach [23] to divide the point cloud space into four overlapped local regions based on the current view of the sensor. Dividing our captured point clouds into four local regions of interests, based on the properties of our indoor environments, reduces the possibility of detecting foreground planes, increases computational efficiency, and leverages the likely spatial location of potential bounding surfaces. In this way, we exploit knowledge of the scene based on the known sensor perspective, while allowing for surface locations to vary relative to each other in different rooms. Further, these regions help ensure that foreground objects that may appear planar in composition (e.g., tables, beds) are preserved and differentiated from background bounding surfaces.

Then, we utilize the M-estimator SAmple Consensus (MSAC) algorithm [26], an improved variant of the RANdom SAmple Consensus (RANSAC) algorithm [27], to segment the largest planes with a specific orientation in each of the local regions. We add an orientation constraint to the MSAC algorithm (orientation-based MSAC) so that we assign priority to segmented planes with the highest percentage of inliers that have an expected orientation relative to the local regions. To do this, we use our defined initial reference vectors [23] for each of the local regions, aligned with the sensor axes as [0,0,-1], [-1,0,0], [1,0,0] and [0,1,0] for the back, left, right, and bottom regions, respectively. Further, we define a maximum allowance angular variation ($\omega = 45$ degrees) between the normal vector of the planes and our reference vectors to allow for sensor perspective variations. All segmented plane candidates with more points than a learned value ($\upsilon = 5\%$ of the total number of points in the point cloud) are verified as integral parts of bounding surfaces and removed from the point cloud, as shown in Fig. 3.1b. Planes containing fewer than υ points may be associated with key objects or small bounding planes and are dealt with in a subsequent processing step (Euclidean clustering removal). Furthermore, υ is set as a proportion of the total points such that it is adaptive to the size of the point cloud.

3.3.3 Distance-based Removal

In a complex indoor environment, bounding surfaces such as walls, windows, and curtains are difficult to fit into a single plane. Furthermore, the input point cloud can be unorganized, which means the nearest neighbour operations, such as region growing, are not very efficient for segmenting the rest of the bounding surfaces. We introduce a distance-based point removal technique to segment and remove the remaining parts of the bounding surfaces from the input point cloud and minimize the likelihood of including points that belong to foreground objects.

Since the verified segmented planes belong to the background surfaces at the outer boundaries of the point cloud, all points behind these planes are also associated with the background bounding surfaces and should be removed from the point cloud. To do this, we use

(3.1) to compute distance D between all points (x, y, z) of the point cloud and the verified planes $(ax + by + cz + d = 0)$ and remove any points associated with a negative distance.

$$D = \frac{ax + by + cz + d}{\sqrt{a^2 + b^2 + c^2}} \tag{3.1}$$

Figure 3.1c illustrates the output of the distance-based removal following the MSAC plane removal when applied to the back region of a point cloud.

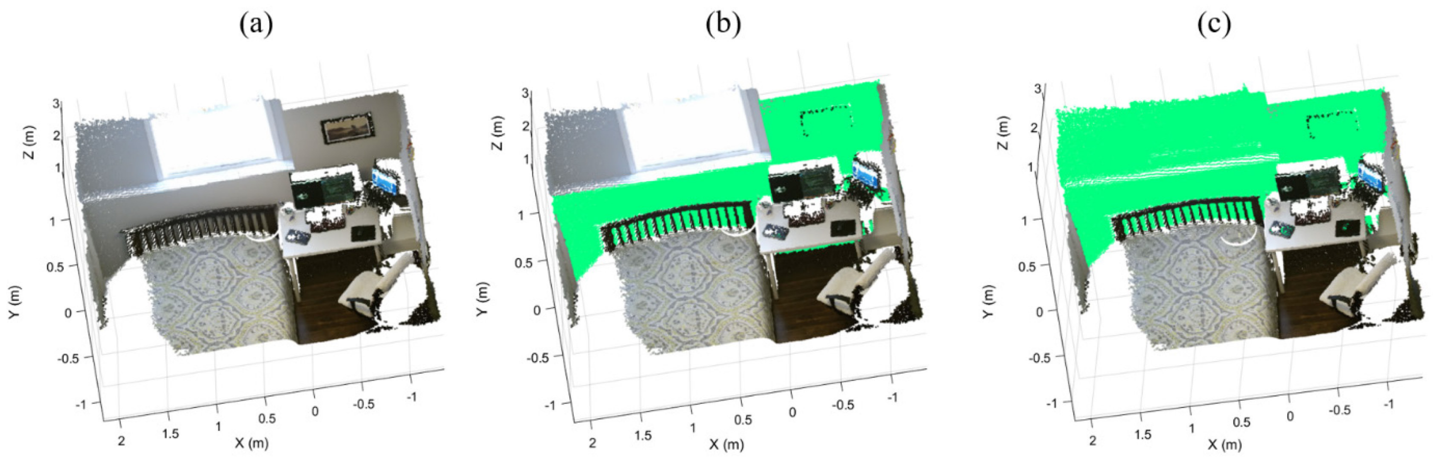


Figure 3.1 The bounding surface segmentation: (a) the sample point cloud, (b) the verified segmented plane, and (c) the bounding surface of the back region highlighted in green.

3.3.4 Euclidean Clustering Removal

In this step, we cluster the remaining point cloud based on Euclidean distance to remove the irrelevant small segments and keep the objects of interest. First, we compute the Euclidean distance between each point and its neighbours. Then, we group neighbouring points as a cluster if the distance between any point in an object and an adjacent point is less than a threshold ε = 5 cm, finishing when all the clusters are determined. Finally, we remove all small clusters with fewer than a threshold μ = 300 points, as shown in Fig 3.2.

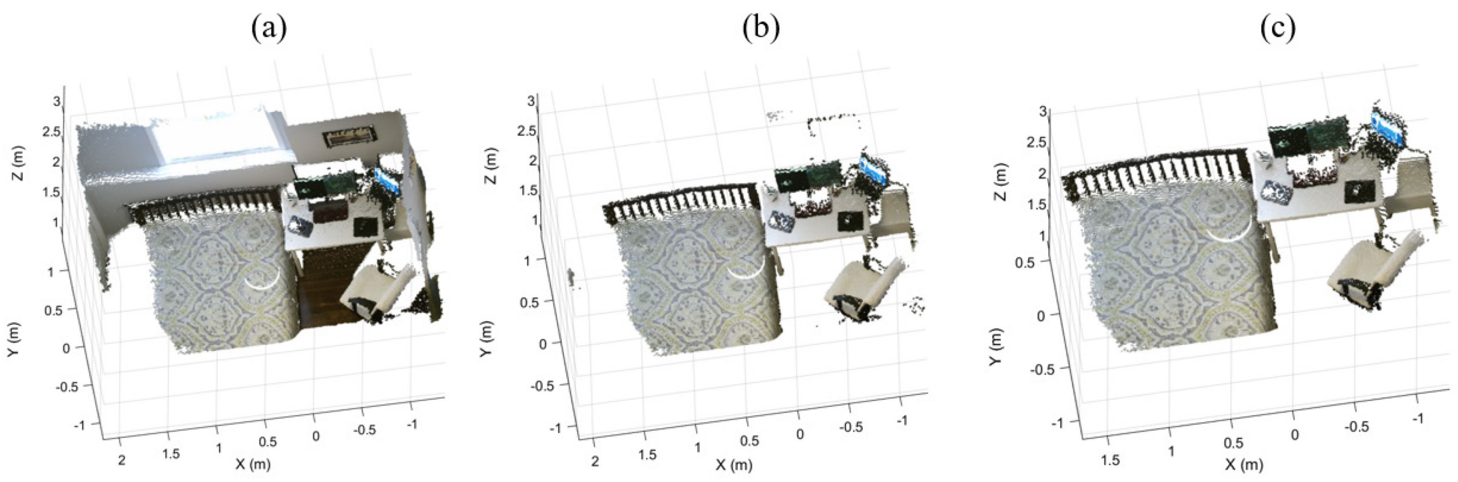


Figure 3.2 Euclidean clustering removal: (a) the sample point cloud, (b) the remaining point cloud after applying the MSAC plane removal and the distance-based removal, (c) the foreground objects after employing the Euclidean clustering removal technique.

3.4 Experiments and Evaluation

We evaluated our method on three generated datasets [23] representing three different complex indoor environments, Room-1, Room-2, and Room-3, as shown in Fig. 3.3a, 3.3d, and 3.3g, respectively. Each dataset contains different objects, such as furniture, planar objects, and human bodies, and is acquired from an arbitrary oblique-view location using the Microsoft Kinect V2 sensor. We implemented our proposed algorithm running MATLAB on an Intel i5-4300M CPU @ 2.60 GHz and with 6.00 GB RAM.

3.4.1 Experimental Results

Figures 3.3b, 3.3e, and 3.3h show the output results of our algorithm and erroneously classified points for the three datasets. The cyan and magenta colours represent the incorrectly removed regions (i.e., false positive) and undetected regions (i.e., false negative), respectively. Our method eliminated nearly all points belonging to the bounding surfaces of the three point clouds. Incorrectly classified points are almost always associated with the points belonging to foreground objects that are contacting a bounding surface, such as the baseboard heater, the bed headboard, and the desk legs (Fig. 3.3c, 3.3f, and 3.3i, respectively). Notably, as a result of our local region definitions and bounding surface criteria, we successfully prevented consideration of foreground objects that could be misidentified as bounding surfaces (e.g., beds, desks).

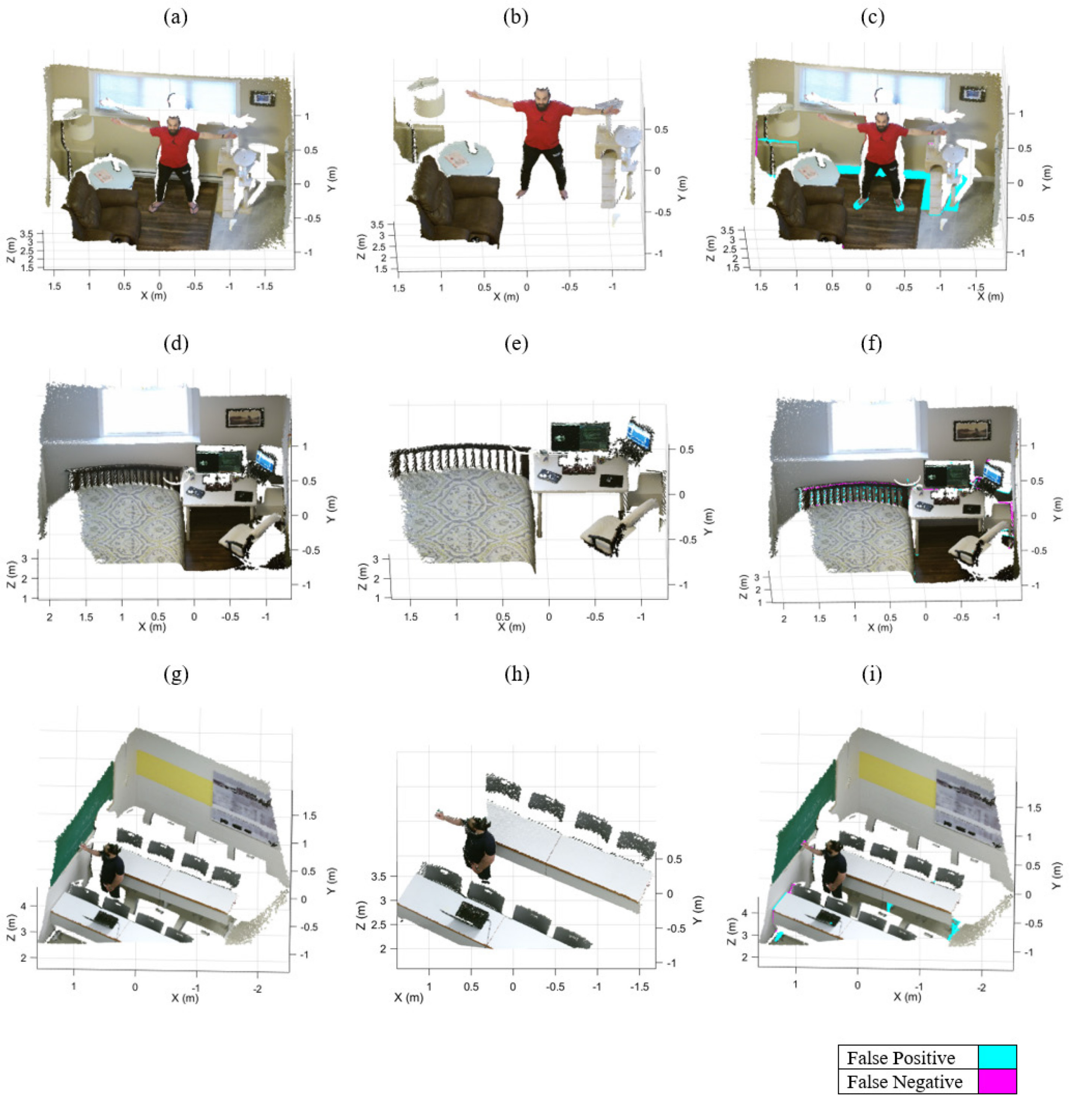


Figure 3.3 Output results of the proposed method. The original point clouds, Room-1 (a), Room-2 (d), and Room-3 (g), are visualized with all bounding surfaces removed (b), (e), and (h), respectively. The false positive (cyan) and false negative (magenta) points are highlighted over the original point clouds for the Room-1 (c), Room-2 (f), and Room-3 (i) datasets.

Figure 3.4 shows the output results of IR- RANSAC for the three datasets. IR-RANSAC failed to remove two small challenging regions belonging to the back and left bounding surfaces of Room-2 (the magenta regions around the window and the bottom left corner of the point cloud in Fig. 3.4b). Furthermore, IR-RANSAC incorrectly removed foreground objects (e.g., the cyan portion of the lamp in Fig. 3.4a) comprised of fewer points than the Euclidean clustering threshold.

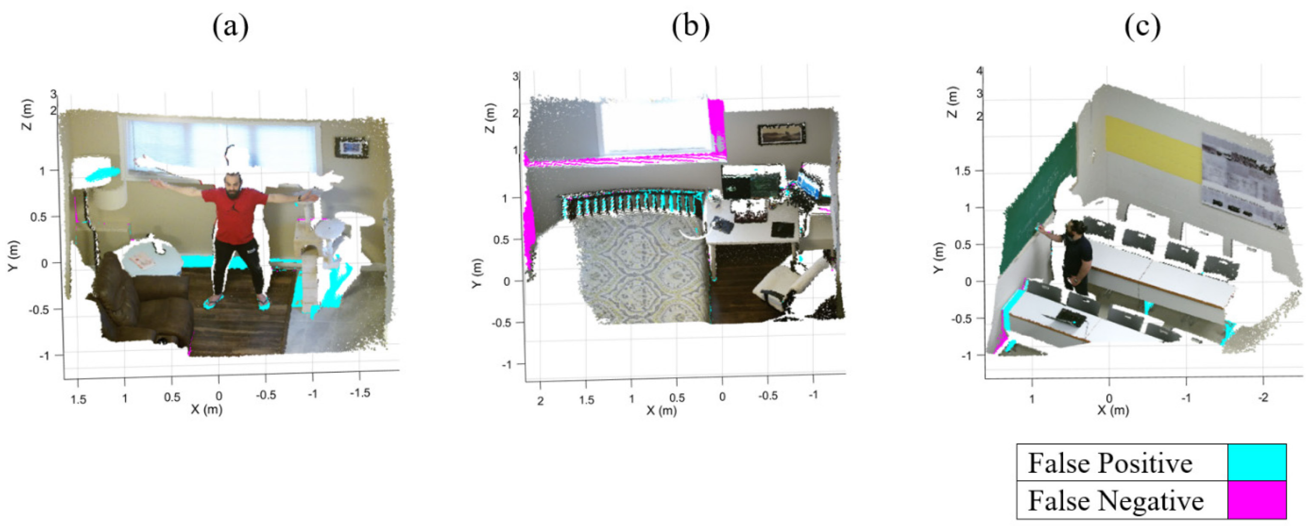


Figure 3.4 The output results of IR-RANSAC and its false positives and false negatives in cyan and magenta, respectively: (a) the Room-1, (b) the Room-2, and (c) the Room-3.

3.4.2 Evaluation

We evaluated the efficiency of our proposed method and IR-RANSAC [23] in terms of four pixel-based metrics, precision, recall, F_1 score, and specificity. Precision is the number of correctly removed points (i.e., true positives) with respect to the total number of removed points. Recall is the fraction of true positive among the ground truth points. F_1 score is the harmonic mean of the precision and recall, and represents the overall performance of our proposed method. The specificity reflects the true negative rate of our algorithm, providing a measure of how well our method distinguishes between foreground points and bounding surface points.

Similar to MSAC, our algorithm is stochastic, and accordingly its results can vary depending on the selection of the random subsample. To account for this stochasticity, we conducted 30 experiments on each dataset, similar to the work of A. Ebrahimi and S. Czarnuch [23], and computed the four evaluation metrics for our proposed method (i.e., Orientation-based MSAC), and IR-RANSAC as a benchmark method. The mean (M) and standard deviation (SD) of both approaches are shown in Table 3.1 for specificity, precision, recall, and F_1 score, along with execution times.

We computed the size reduction of both methods as $1 - \frac{S_{OUT}}{S_{IN}}$ where S_{OUT} and S_{IN} are the size (i.e., the number of points) of the output point cloud and the input point cloud, respectively. Our proposed method yielded a size reduction of 0.70, 0.66, and 0.64, where IR-RANSAC yielded a size reduction of 0.68, 0.62, and 0.64 for Room-1, Room-2, and Room-3, respectively.

3.5 Discussion and Conclusions

Our experimental results suggest that our proposed method supports varied sensor locations and removes background boundary surfaces effectively without removing foreground data in complex 3D indoor environments. Unlike IR-RANSAC, our new method can eliminate the small challenging regions belonging to a bounding surface but with a different normal vector (e.g., the highlighted regions around the window in Fig. 3.4b). Furthermore, unlike IR-RANSAC, our proposed method does not remove foreground objects (e.g., the cyan portion of the lamp in Fig. 3.4a). This is because we reduced the reliance of our new method on the Euclidean clustering technique (i.e., only using Euclidean clustering removal with a minimal threshold to remove the irrelevant small segments).

Table 3.1 Execution times, mean (M), and standard deviation (SD) of our proposed method (Orientation-based MSAC) and IR-RANSAC for specificity, precision, recall, and F1 score.

Dataset	Method	Specificity		Precision		Recall		F_1		Runtime (s)
		M (%)	SD	M (%)	SD	M (%)	SD	M (%)	SD	
Room-1	Orientation-based MSAC	93.44	0.0154	96.87	0.0072	99.24	0.0088	98.03	0.0051	4.03
	IR-RANSAC	92.60	0.0200	96.41	0.0097	97.42	0.0230	96.90	0.0142	8.83
Room-2	Orientation-based MSAC	98.08	0.0168	98.97	0.0087	97.59	0.0051	98.27	0.0034	3.73
	IR-RANSAC	98.38	0.0135	99.06	0.0074	90.52	0.0193	94.59	0.0101	7.50
Room-3	Orientation-based MSAC	97.29	0.0089	98.59	0.0046	99.74	0.0045	99.16	0.0028	3.23
	IR-RANSAC	97.39	0.0075	98.61	0.0038	97.47	0.0253	98.02	0.0126	6.37

In all experiments, our proposed method obtained average values above 93% for specificity, 97% for precision, 97% for recall, and 98% for F_1 score. The evaluation results support that our bounding surface removal is a robust and reliable method for removing the background boundary surfaces of a complex 3D indoor environment. Our new method slightly outperformed IR-RANSAC in all four evaluation metrics, and is significantly faster than IR-RANSAC (i.e., almost twice as fast as IR-RANSAC). As shown in Table 3.1, almost all four scores have a lower standard deviation with our new approach compared to IR-RANSAC.

We have presented a 3D bounding surface removal technique that is particularly suited to more challenging and cluttered indoor environments. Our method supports varied sensor perspectives for organized and unorganized point clouds, and it considerably reduces the size of 3D datasets. Moreover, it can improve the results and the running time of different 3D computer vision methods by reducing their search space. In future work, we will further extend the method to simultaneously remove the background bounding surfaces and detect bounding surface types (e.g., ceiling and floor) in an entirely perspective-independent setup.

3.6 References

[1] S. H. Shaikh, K. Saeed, and N. Chaki, “Moving object detection using background subtraction,” in Springer Briefs in Computer Science, no. 9783319073859, Springer, 2014, pp. 15–23. doi: 10.1007/978-3-319-07386-6_3.

[2] S. Kumar and J. Sen Yadav, “Video object extraction and its tracking using background subtraction in complex environments,” Perspectives in Science, vol. 8, pp. 317–322, Sep. 2016, doi: 10.1016/j.pisc.2016.04.064.

[3] A. Aggarwal, S. Biswas, S. Singh, S. Sural, and A. K. Majumdar, “Object tracking using background subtraction and motion estimation in MPEG videos,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer, Berlin, Heidelberg, 2006, pp. 121–130. doi: 10.1007/11612704_13.

[4] R. Manikandan, R. Ramakrishnan, and R. Scholar, “Human Object Detection and Tracking using Background Subtraction for Sports Applications,” International Journal of Advanced Research in Computer and Communication Engineering, vol. 2, 2013.

[5] S. Czarnuch, S. Cohen, V. Parameswaran, and A. Mihailidis, “A real-world deployment of the COACH prompting system,” Journal of Ambient Intelligence and Smart Environments, vol. 5, no. 5, pp. 463–478, Jan. 2013, doi: 10.3233/AIS-130221.

[6] W. Zou, Y. Lu, M. Chen, and F. Lv, “Rapid face detection in static video using background subtraction,” in Proceedings - 2014 10th International Conference on Computational Intelligence and Security, CIS 2014, Institute of Electrical and Electronics Engineers Inc., Jan. 2015, pp. 252–255. doi: 10.1109/CIS.2014.146.

[7] H. Yang and S. Qu, “Real-time vehicle detection and counting in complex traffic scenes using background subtraction model with low-rank decomposition,” IET Intelligent Transport Systems, vol. 12, no. 1, pp. 75–85, Feb. 2018, doi: 10.1049/iet-its.2017.0047.

[8] A. Kumar Sahu and A. Choubey, “Motion Detection Surveillance System Using Background Subtraction Algorithm,” International Journal of Advance Research in Computer Science and Management Studies, vol. 1, no. 6, 2013.

[9] S. Hargude and S. R. Idate, “I-Surveillance : Intelligent Surveillance System Using Background Subtraction Technique,” in Proceedings - 2nd International Conference on Computing, Communication, Control and Automation, ICCUBEA 2016, Institute of Electrical and Electronics Engineers Inc., Feb. 2017. doi: 10.1109/ICCUBEA.2016.7860046.

[10] M. Karaman, L. Goldmann, D. Yu, and T. Sikora, “Comparison of static background segmentation methods,” in Visual Communications and Image Processing 2005, SPIE, 2005, pp. 2140–2151.

[11] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 246–252, 1999, doi: 10.1109/cvpr.1999.784637.

[12] Z. Zivkovic, “Improved adaptive Gaussian mixture model for background subtraction,” in Proceedings - International Conference on Pattern Recognition, 2004, vol. 2, pp. 28–31.

[13] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, “Real-time foreground-background segmentation using codebook model,” Real-Time Imaging, vol. 11, no. 3, pp. 172–185, Jun. 2005.

[14] J. M. Guo, Y. F. Liu, C. H. Hsia, M. H. Shih, and C. S. Hsu, “Hierarchical method for foreground detection using codebook model,” IEEE Trans. Circuits Syst. Video Technol., vol. 21, no. 6, pp. 804–815, Jun. 2011.

[15] D. M. Tsai and S. C. Lai, “Independent component analysis-based background subtraction for indoor surveillance,” IEEE Trans. Image Process., vol. 18, no. 1, pp. 158–167, 2009.

[16] H. Jiménez-Hernández, “Background Subtraction Approach Based on Independent Component Analysis,” Sensors, vol. 10, no. 6, pp. 6092–6114, Jun. 2010.

[17] G. Gordon, T. Darrell, M. Harville, and J. Woodfill, “Background estimation and removal based on range and color,” Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 459–464, 1999, doi: 10.1109/cvpr.1999.784721.

[18] S. Czarnuch and A. Mihailidis, “Development and evaluation of a hand tracker using depth images captured from an overhead perspective,” Disability and Rehabilitation: Assistive Technology, vol. 11, no. 2. Taylor and Francis Ltd, pp. 150–157, Feb. 17, 2016.

[19] J. Shotton et al., “Real-time human pose recognition in parts from single depth images,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2011, pp. 1297–1304.

[20] N. Vaskevicius, A. Birk, K. Pathak, and S. Schwertfeger, “Efficient representation in three-dimensional environment modeling for planetary robotic exploration,” Adv. Robot., vol. 24, no. 8–9, pp. 1169–1197, 2010.

[21] T. Czerniawski, M. Nahangi, S. Walbridge, and C. Haas, “Automated removal of planar clutter from 3D point clouds for improving industrial object recognition,” ISARC 2016 - 33rd Int. Symp. Autom. Robot. Constr., no. Isarc, pp. 357–365, 2016.

[22] R. Kaushik and J. Xiao, “Accelerated patch-based planar clustering of noisy range images in indoor environments for robot mapping,” Rob. Auton. Syst., vol. 60, no. 4, pp. 584–598, 2012.

[23] A. Ebrahimi and S. Czarnuch, “Automatic Super-Surface Removal in Complex 3D Indoor Environments Using Iterative Region-Based RANSAC,” Sensors, vol. 21, no. 11, p. 3724, May 2021.

[24] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," 2011 IEEE International Conference on Robotics and Automation, 2011, pp. 1-4.

[25] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, “Towards 3D Point cloud based object maps for household environments,” Rob. Auton. Syst., vol. 56, no. 11, pp. 927–941, 2008.

[26] P. H. S. Torr and A. Zisserman, “MLESAC: A new robust estimator with application to estimating image geometry,” Computer vision and image understanding, vol. 78, no. 1, pp. 138–156, 2000.

[27] M. A. Fischler and R. C. Bolles, “Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,” Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

Chapter 4. PiGPDS: Perspective Independent Ground Plane Detection and Segmentation for Complex 3D Indoor Scenes

Co-authorship statement – This chapter was accepted as a full conference paper titled "PiGPDS: Perspective Independent Ground Plane Detection and Segmentation for Complex 3D Indoor Scenes" and presented at the IEEE International Conference on Digital Image Computing: Techniques and Applications (DICTA), Port Macquarie, Australia, in November 2023. Ali Ebrahimi, as the primary author, conducted the literature review, conceptualized the study, designed the methodology, developed the method, collected data, analyzed the results, created visualizations, presented the paper, and both drafted and revised the manuscript. The co-author, Dr. Stephen Czarnuch, provided essential guidance and supervision throughout the research process, and meticulously reviewed and revised the manuscript. Both authors have read and agreed to the final version of the manuscript.

4.1 Abstract

Ground plane detection and segmentation techniques can benefit and help improve the accuracy and robustness of a wide range of computer vision applications, from 3D object segmentation and autonomous navigation to mixed and augmented reality. Existing approaches often rely on restrictive assumptions to simplify the problem, such as the ground plane being the largest plane in the scene or the camera location or orientation being ideal. We present a ground plane segmentation technique for real-world 3D indoor scenes where the position and orientation of the sensor are unrestricted and unknown. Our method only requires one 3D point cloud of an indoor scene and assumes that the scene contains at least one surface parallel to the actual ground plane, which is generally true for 3D indoor scenes. We begin by utilizing a voxelized grid downsampling method to enhance the speed of the algorithm. Subsequently, we use K-medoids clustering and an angular-based zone determination technique to identify the ground zone. Next, we divide the ground zone into several clusters using the Euclidean clustering algorithm, and we employ the M-estimator Sample Consensus (MSAC) algorithm to fit the largest plane in each cluster with a specific orientation. Finally, based on the geometric relationship between the fitted planes of the ground zone, we estimate the ground plane, verify it and segment all its associated points using a distance-based approach. We evaluated our method on public and self-generated datasets, in which we positioned a depth sensor at various locations, pitches, and yaws. Our experimental results demonstrate that our proposed method can robustly and efficiently detect and segment the ground plane of complex 3D indoor scenes and supports varied sensor locations and orientations. We evaluate the performance of our proposed method in terms of four conventional metrics: specificity, precision, recall, and F_1 score, with average experimental results of 98.28, 95.48, 96.64, and 96.01, respectively.

Keywords – 3D ground plane detection, 3D ground surface segmentation, 3D ground labeling, Perspective-independent point cloud processing

4.2 Introduction

The release of commercially available depth sensors such as Microsoft Azure Kinect [1], mixed reality devices with integrated depth sensors like Microsoft HoloLens [2], and mobile devices with LiDAR or depth sensors like iPhone Pro [3] has enabled a more comprehensive understanding and accurate representation of spatial information in real-world environments. As a result, 3D data have become widely adopted in recent years and provide significant advantages in various applications, including but not limited to augmented reality [4], autonomous navigation [5], simulation [6], and gaming [7]. Many computer vision systems, such as transitional navigation techniques for AR scene interaction [8], robot navigation [9], and 3D object tracking [10], benefited from identifying the location of the ground surface.

Most common conventional 2D methods for estimating the ground plane are homography-based, such as [11], [12], and [13]. These methods rely on certain assumptions; for example, the camera's field of view should be parallel to the ground, and the ground plane should dominate most of the field of view. Compared to conventional 2D methods, deep learning-based techniques such as [14], [15] and [16] can provide more accurate results; however, their ability to provide accurate depth estimation in unknown scenes is limited by the single-view scale ambiguity [17]. Due to the limitations imposed by these restrictive assumptions and the fact that 2D data are not sufficient to fully represent the spatial relationships between the ground plane and other objects in an indoor scene, many researchers have opted for 3D ground plane detection as a more effective solution for cluttered and dynamic indoor environments.

Many conventional 3D techniques for identifying the ground plane use one or more of the following: the Hough transform algorithm [18], the RANdom SAmple Consensus (RANSAC) algorithm [19], or normal vectors of the scene's surfaces. For instance, Borrmann et al. [20] utilized a 3D Hough transform and a ball-shaped accumulator for ground plane detection in 3D point clouds, while Zeineldin and El-Fishawy [21] proposed an enhanced RANSAC algorithm to identify the ground plane and obstacles for individuals with visual impairments. However, these model fitting-based methods can only work if the ground plane is the largest in the scene; additionally, the RANSAC algorithm is prone to detecting spurious planes in 3D complex indoor environments [22]. Holz et al. [23] proposed a normal-based plane segmentation method for mobile navigation in indoor environments. Although their approach can successfully identify the ground plane of indoor scenes, it is restricted to structured point clouds and only functions properly when the sensor's pitch angle is zero. To overcome these limitations, Zhang and Czarnuch [24] proposed a perspective-independent ground estimation method by processing a video sequence of RGB-D data. Their method can successfully detect the ground plane of dynamic indoor scenes regardless of ground plane size and camera orientation but requires at least one human body to be visible and moving in the RGB-D camera's field of view.

Therefore, current 3D ground plane detection methods require one or more of these assumptions: the largest plane of the scene is the ground plane; one or multiped depth sensors are placed in specific orientations or positions; a video sequence of data is accessible; the input point cloud is organized; or foreground objects (e.g., a human body) are positioned vertically on the ground plane of the scene. To address these restrictive assumptions, we propose a perspective-independent ground plane detection and segmentation (PiGPDS) that only requires a single point cloud of an indoor scene and only assumes that the 3D scene contains one surface parallel to its actual ground plane, which is almost always valid for cluttered and complex indoor environments.

4.3 Methodology

Our PiGPDS has five major steps. First, we downsample the input point cloud using a voxelized grid approach. Second, we determine the ground zone of the downsampled point cloud based on its voxels' normal vectors. Third, we divide the ground zone into several parallel clusters based on the Euclidean distance between the zone's voxels. Fourth, we segment the largest plane with a specific orientation in every cluster using the M-estimator Sample Consensus (MSAC) algorithm [25]. Finally, using the geometric relationship between the segmented planes, we estimate the ground plane, verify it and segment all its associated voxels.

4.3.1 Downsampling

A widely adopted technique to speed up 3D computer vision algorithms is the downsampling of point clouds. Random downsampling [26] and voxelized grid [27] approaches are two commonly used techniques for reducing the size of point clouds. We conducted experiments on both techniques and discovered that the voxelized grid approach better maintains the shape of the input point cloud; however, it can be slightly slower than the random downsampling technique. Given that the indoor point clouds are relatively smaller in size compared to outdoor point clouds and our primary objective is to preserve the underlying points representing the true geometry of the ground plane in the scene, we opted for a voxelized grid approach that computes the centroid of all points in each 3D voxel grid with a leaf size of 0.2 cm. Based on our experiments, this approach offers a suitable balance between speed and accuracy and significantly reduces the running time of our PiGPDS. It should be noted that the downsampled point cloud is used as the primary input for all of the algorithm's processing steps except for the final step (4.3.5.3 Distance-based Segmentation), which specifically targets the ground surface points in the original point cloud.

4.3.2 Ground Zone Determination

In this step, first, we utilize K-medoids clustering [28] to divide points of the downsampled point cloud into distinct zones with similar normal vectors. Then, we determine the ground zone of the point cloud by using an angular-based approach.

4.3.2.1 K-medoids Clustering

Using K-medoids clustering, we group the normal vectors of the point cloud's points into K clusters. Similar to the K-means clustering algorithm [29], the K-medoids algorithm clusters a set of observations into K subsets so that the subsets minimize the sum of distances between an observation and the center of the observation's cluster. Unlike the K-means algorithm, the center of each cluster in K-medoids clustering is always a member of the cluster itself, called a medoid. This difference is the main reason for using K-medoids clustering rather than K-means clustering in this work, such that we can also take advantage of the normal vector of each medoid as the reference vector of each zone. In addition, K-medoids clustering is more robust and less sensitive to noise and outliers than K-means [30].

The computation of a surface point's normal can be approximated by the problem of estimating its tangent plane's normal, which in turn becomes a least-square plane fitting estimation problem [31]. We estimate the normal vector of each point of the downsampled point cloud by using its neighbouring points to fit a local plane and determine its normal vector. Since the sign of a given point's surface normal cannot be solved mathematically [32], the orientations of the estimated normals are initially vague and not consistently oriented over the entire point cloud. To solve this problem, we flip all normals with regard to the viewpoint (i.e., the sensor center).

Prior to clustering the normal vectors, we need to estimate their optimal K within an unknown indoor environment. Various data properties, such as cluster size, density, and separability, can be analyzed to evaluate the optimal number of clusters. Generally, the goal of clustering analysis is to minimize the variance within clusters and maximize the separation between clusters. Using the MATLAB Evalclusters function [33], we could evaluate the optimal K for clustering the normal vectors of different point clouds by employing different validation techniques such as Calinski-Harabasz [34], Davis-Bouldin [35], and Silhouette [28][36]. Our initial experimentation showed that the Calinski-Harabasz technique works best for different types of indoor environments. Therefore, we evaluate the optimal number of clusters using the Calinski-Harabasz validation technique in this study. Fig. 4.1b illustrates the result of K-medoids clustering for a sample point cloud using a 3D scatter plot. As shown in the 3D plot, K-medoids clustering split the normal vectors of the sample point cloud into three individual clusters. Each cluster represents one point cloud zone where every point has a similar normal vector (i.e., surfaces of each zone are parallel).

4.3.2.2 Angular-based Zone Determination

To identify which zone represents the ground zone of the point cloud, first, we compute the angle between each zone's reference vector (i.e., the normal vector of the zone's medoid) and our proposed ground reference vector. The ground reference vector is chosen to be perpendicular to the Z axis of a sensor placed ideally parallel to the ground plane of an indoor scene (i.e., reference vector $= [0,1,0]$), as shown in Fig. 4.1c.

After computing the angles between each zone's reference vector and the ground reference vector, we choose the zone associated with the minimum angle as the ground zone; for instance, Fig. 4.1d shows the ground zone points of the sample point cloud highlighted in blue. By using this angular-based approach, we integrate the perspective-independent feature into our ground zone determination technique. The only assumption of our angular-based approach is that the input point cloud contains a section of the ground plane or at least a surface parallel to the ground plane (excluding the ceiling surface), which is almost always valid for 3D indoor point clouds.

4.3.3 Euclidean Clustering

In this step, we segment the ground zone into several clusters based on the Euclidean distance between the zone's points. As illustrated in Fig. 4.1d, the ground zone consists of several parallel surfaces (i.e., surfaces with the same normal vector) that are at different distances along the surface normal from each other.

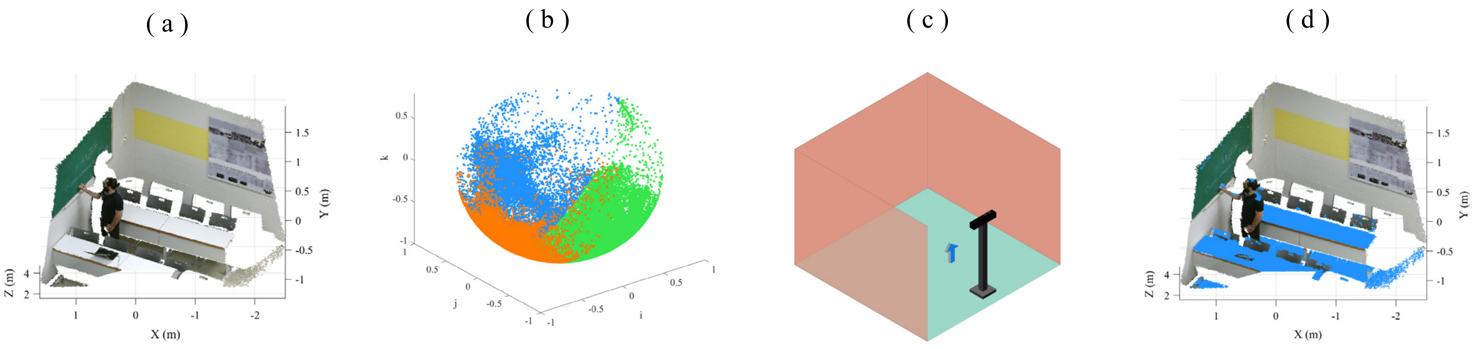


Figure 4.1 Ground zone determination, (a) the sample point cloud, (b) K-medoids clustering of the normal vectors of the sample point cloud represented as a 3D scatter plot, (c) the ground reference vector of a 3D indoor scene illustrated as a blue arrow, and (d) the ground zone points highlighted in blue over the sample point cloud.

The primary motivation for dividing the ground zone into distinct individual clusters is facilitating the MSAC algorithm to efficiently segment the optimal plane of each of these clusters in the next step, Orientation-based MSAC. Moreover, cluster or cell-based RANSAC (the same applied to cluster-based MSAC) can increase the accuracy of the algorithm by preventing the segmentation of spurious planes [22].

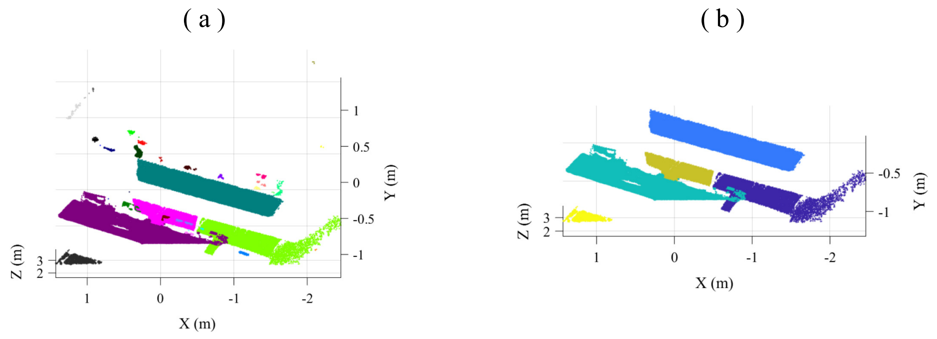


Figure 4.2 Euclidean clustering, (a) the initial clusters and (b) the large clusters of the ground zone in Fig. 4.1d.

To cluster the ground zone's points, first, we compute the Euclidean distance between every point and its neighbours. Then, we group neighbouring points as a cluster if the distance between any point and an adjacent point is less than a threshold ($\varepsilon = 5\ cm$), finishing when all the clusters are determined. For example, the Euclidean clustering of the ground zone shown in Fig. 4.1d results in 37 distinct clusters, as illustrated in Fig. 4.2a. The tiny clusters in Fig. 4.2a usually belong to small foreground objects and may not have enough points for robustly fitting planes using the MSAC algorithm in the next step. Therefore, we remove any cluster with fewer than a threshold, adaptive to the size of the point cloud ($\mu = 0.1\%$ of point cloud points).

For example, by removing the tiny clusters in Fig. 4.2a, we could decrease the number of clusters from 37 to 5, as shown in Fig. 4.2b. In addition, eliminating these tiny clusters from the ground zone can speed up our PiGPDS algorithm by employing the MSAC algorithm in fewer clusters.

4.3.4 Orientation-based MSAC

In this step, we utilize the MSAC algorithm to segment the largest plane with a specific orientation in each of the clusters of the ground zone. The MSAC algorithm, an improved variant of the RANSAC algorithm, generates a hypothesis plane by randomly and iteratively sampling three voxels as a minimum subset within each cluster. In each iteration, it computes the distance between the plane and the remaining voxels of the cluster and then counts the number of inliers within a distance threshold ($\delta = 2$ cm) of the plane. Finally, it returns the plane with the highest percentage of inliers.

Given that all the points within the ground zone have almost the same normal vector and points within a cluster nearly lie on a flat surface, the MSAC algorithm might be capable of segmenting the desirable planes of the ground zone. However, similar to the work in [37], we add an orientation-based approach to the MSAC to increase the accuracy of the fitted planes and ensure that all the segmented planes within the ground zone are parallel. The orientation-based MSAC prioritizes planes with the highest percentage of inliers with an expected orientation relative to the ground zone surfaces. To do this, we use the ground zone's reference vector (i.e., the normal vector of the ground zone's medoid) and assign a maximum allowable angular variation ($\beta = 5°$), defined as the angle between this reference vector and the normal vectors of the ground zone's fitted planes. For example, employing orientation-based MSAC in the ground zone's clusters (illustrated in Fig. 4.2b) results in the segmented planes highlighted over the sample point cloud, as shown in Fig. 4.3a.

According to Förstner and Wrobel [38], the maximum number of iterations I required for convergence by the RANSAC algorithm (the same applied to our orientation-based MSAC algorithm) depends on the number of samples s ($s = 3$ for plane fitting), the target success probability p ($p = 99\%$), and the outlier ratio o and can be approximated as (4.1). At the expense of additional computation, a higher success probability increases I and can improve the robustness of the output.

$$I = \frac{\log(1-p)}{\log(1-(1-o)^s)} \tag{4.1}$$

4.3.5 Ground Surface Detection and Segmentation

In the final step of our PiGPDS algorithm, first, we estimate the ground plane based on the geometric relationship between the segmented planes of the ground zone, then verify it and segment all its associated points using a distance-based approach.

4.3.5.1 Ground Plane Estimation

To find the ground plane, we calculate and analyze the distances between all segmented planes of the ground zone. That is, first, we randomly initialize one of the planes as the ground plane ($ax + by + cz + d = 0$), then we iterate through the other planes of the zone, and we find the distance $\overline{D}$ between them and the initial ground plane. Since planes within the zone may not be completely parallel (i.e., two non-parallel planes intersect at some straight line in a 3D space), we compute the mean of distances between all points of each plane $\{(x_i,\ y_i,\ z_i): 1 \leq i \leq n\}$ and the initial ground plane as (4.2), where n is the number of points in each plane. In every iteration, if the computed distance is a negative number (i.e., the new plane is located underneath the initial ground plane), we update the initial ground plane with the new plane.

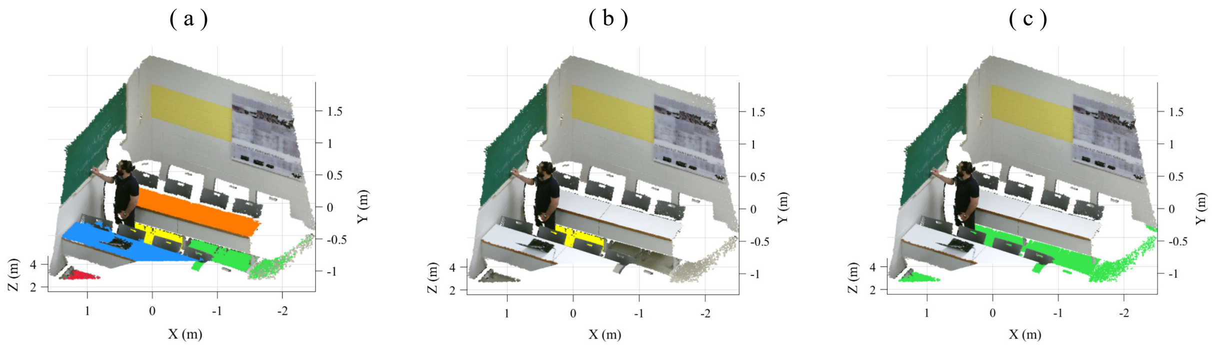


Figure 4.3 All the segmented planes of the ground zone (a), the verified ground plane (b), and the segmented ground surface containing all the visible ground points (c) highlighted over the sample point cloud.

$$\overline{D} = \frac{\sum_{i=1}^{n} \frac{ax_i + by_i + cz_i + d}{\sqrt{a^2 + b^2 + c^2}}}{n} \tag{4.2}$$

4.3.5.2 Distance-based Verification

When there is no visible ground plane in the input point cloud, the estimated ground plane does not reflect the actual ground plane of the point cloud. Therefore, we verify the estimated ground plane based on the simple fact that there are almost no data points beneath a true ground plane of a point cloud, as follows. First, we compute the distance D between the estimated ground plane ($ax + by + cz + d = 0$) and all points of the downsampled point cloud (x, y, z), except points corresponding to the estimated ground plane itself, using (4.3). Then, we verify the plane as the actual ground plane if almost all computed distances are positive (i.e., less than negative of MSAC threshold, $-\delta$, to exclude ground plane points not segmented by the MSAC algorithm). Even if the verified ground plane does not include all points of the visible ground surface of a point cloud (e.g., see the verified ground plane of the sample point cloud in Fig. 4.3b), its parametric plane model can still be used for various computer vision applications to represent the ground plane of the scene.

$$D = \frac{ax + by + cz + d}{\sqrt{a^2 + b^2 + c^2}} \tag{4.3}$$

4.3.5.3 Distance-based Segmentation

Since the verified ground plane is a part of the visible ground surface, it could be possible to segment all its associated points using nearest-neighbour operations, such as region-growing algorithms. However, these operations are inefficient when the point cloud is unorganized. In addition, they can lead to incorrect segmentation of the visible ground surface where a foreground object touches the surface, dividing the visible ground plane into multiple smaller but visible planes. Hence, we use a distance-based technique to segment the remaining parts of the visible ground surface and then we refine the ground surface to minimize the likelihood of erroneously including points associated with foreground objects.

First, we compute the distance D between the verified plane and all points of the original point cloud (unlike the previous step, where the downsampled point cloud is used). In order to segment the ground plane points located in other clusters of the ground zone (e.g., see the green and red planes' points of the sample point cloud in Fig. 4.3a) and also ground plane points not segmented by the MSAC algorithm (e.g., see the non-highlighted ground plane points in Fig. 4.3a) that are slightly distanced further from the verified plane, we include any data points with a distance less than two times the MSAC threshold, 2δ. Then, we refine the ground surface by excluding segmented points whose normal vectors differ from the verified ground plane's normal vector, effectively eliminating points associated with foreground objects in contact with the ground surface. As shown in Fig. 4.3c, our PiGPDS algorithm can segment the entire ground surface visible in the original sample point cloud, as highlighted in green over the point cloud.

4.4 Experiments

We evaluated our method on our own generated dataset [39] and NYU depth dataset V2 [40]. Our dataset includes scenes acquired using both the Microsoft Azure Kinect (i.e., Kinect V4) and Kinect V2 sensors by placing the sensors in several different locations with different pitches and yaws, while the NYU depth dataset V2 was acquired with the Kinect V2 sensor. The two datasets showcase unique indoor settings with a diverse array of objects, including furniture, pets, and human bodies, almost all in contact with the ground plane, as shown in Fig. 4.4 and Fig. 4.5.

We implemented our proposed algorithm running MATLAB on an Intel i7-12700H CPU @ 2.30 GHz and with 16 GB RAM.

For our own dataset, where ground truth labels were unavailable, we relied on visual inspection (i.e., qualitative evaluation) as the primary means to assess the results. This approach not only allowed us to thoroughly examine and analyze the outcomes of our PiGPDS but also served as a way to evaluate the perspective independence of our approach. Furthermore, to quantify the performance of PiGPDS, we assessed its efficiency on NYU depth dataset V2 in terms of four pixel-based metrics, specificity, precision, recall, and F_1 score. The last three parameters have been commonly used to assess the effectiveness of plane segmentation (e.g., [41], [42], [43]). To compute these metrics, we defined true positives (TP) as correctly identified ground points, true negatives (TN) as correctly identified non-ground points, false positives (FP) as non-ground points incorrectly identified as ground, and false negatives (FN) as ground points incorrectly identified as non-ground points. The specificity, measured as $TN/(TN + FP)$, assesses our method's ability to differentiate ground points from the non-ground points. Precision, measured as $TP/(TP + FP)$, represents the number of correctly segmented points relative to the total number of segmented points. Recall, measured as $TP/(TP + FN)$, represents the ratio of true positives among the ground truth points. F_1 score, the harmonic mean of the precision and recall, measured as $2 \times (precision \times recall)/(precision + recall)$, represents the overall performance of PiGPDS.

Figure 4.4 demonstrates the output results of the PiGPDS algorithm for fifteen unorganized point clouds derived from our dataset (see Table 4.1 for point clouds' details). PiGPDS successfully detected the ground plane and segmented nearly all points associated with the ground surface in almost all point clouds (e.g., see the surfaces highlighted in green over the point clouds in Fig.

4.4). Notably, our PiGPDS correctly recognized the lack of a visible ground surface in point clouds if they contain at least a surface parallel to the actual ground plane, excluding the ceiling surface, (e.g., see the point cloud surrounded by a blue rectangle, Bedroom3_013_v4, in Fig. 4.4). The PiGPDS algorithm fails only when a point cloud contains no visible surface parallel to the ground plane (e.g., see the point cloud surrounded by a red rectangle, Bedroom3_018_v4, in Fig. 4.4).

Figure 4.5 shows the output results of the PiGPDS algorithm and erroneously classified points for ten point clouds obtained from the public dataset, NYU depth dataset V2 (see Table 4.2 for point clouds' details). PiGPDS accurately identified the ground plane and segmented almost all points related to the ground surface across all ten point clouds (see the green-highlighted surfaces in Fig. 4.5). The false positives, incorrectly segmented points, and the false negatives, undetected ground points are highlighted in blue and red, respectively, over the original point clouds in Fig. 4.5. Undetected ground points (see the red-highlighted points in Fig. 4.5) are almost always associated with the noisy points close to the ground surface that are incorrectly labeled as the ground truth. These points have different normal vectors compared to the normal vector of the ground plane; as a result, the ground surface refiner (see 4.3.5.3 Distance-based Segmentation) removes them from the ground surface. Incorrectly segmented points (see the blue-highlighted areas or gaps between carpets in Fig. 4.5) are actually correctly identified by PiGPDS as the ground surface; however, they are not labeled as the ground truth. Note, since the NYU dataset only provides object labels, in cases where carpets are on the floor, we labeled the ground truth by combining the visible ground surface and the carpet surfaces. We could not include the gap areas in the ground truth because they are not labeled as objects in the NYU depth dataset.

Table 4.2 presents the quantitative evaluation results of our PiGPDS on the NYU depth dataset V2 in terms of the four pixel-based metrics: specificity, precision, recall, and F_1 score.

PiGPDS achieved a specificity above 97% for all point clouds, except for the 282_bedroom_0131 point cloud, which scored 93.72%. The precision was above 93% for all point clouds, except for the 282_bedroom_0131 and 580_living_room_0081 point clouds, which scored 91.08% and 88.20%, respectively. Additionally, the recall was above 94% for all point clouds, and the F_1 score was above 94% for all point clouds, except for the 580_living_room_0081 point cloud, which scored 91.84%. Note, the lower scores are attributed to the incorrectly labeled ground truth of these two point clouds.

Table 4.1 Details of point clouds derived from our generated dataset and PiGPDS execution time for each point cloud

File name	Points	Runtime (s)	File name	Points	Runtime (s)	File name	Points	Runtime (s)
Classroom1_001_v2	181893	2.8	Bedroom3_001_v2	179360	3.3	Bedroom3_015_v4	523011	4.0
Bedroom1_001_v2	179759	2.9	Bedroom3_011_v4	538587	3.0	Bedroom3_016_v4	526919	3.8
Bedroom2_001_v2	175026	3.0	Bedroom3_012_v4	515014	3.2	Bedroom3_017_v4	540567	3.6
Bedroom2_011_v4	556635	4.5	Bedroom3_013_v4	530972	2.4	Bedroom3_018_v4	539761	3.6
Bedroom2_012_v4	544473	4.6	Bedroom3_014_v4	499193	3.7	Bedroom3_019_v4	541507	3.9

Table 4.2 Details of point clouds derived from NYU depth dataset V2, PiGPDS evaluation results and execution time for each point cloud

Index	File name	Specificity	Precision	Recall	F_1	Runtimes	Index	File name	Specificity	Precision	Recall	F_1	Runtimes
150	living_room_0004	99.41	96.70	94.40	95.58	3.6	580	living_room_0081	97.96	88.20	95.79	91.84	3.7
151	living_room_0005	100	100	94.41	97.12	3.5	909	bedroom_0025	97.32	93.74	96.55	95.12	4.1
282	bedroom_0131	93.72	91.08	98.83	94.80	3.3	1330	living_room_0075	98.20	95.31	97.37	96.33	3.1
358	home_office_0001	99.85	99.45	94.81	97.08	4.6	1331	living_room_0075	98.43	97.21	96.67	96.94	4.8
450	printer_room_0001	99.13	98.43	98.42	98.42	3.2	1435	dining_room_0033	98.74	94.70	99.10	96.85	2.5

* All point clouds have 307200 points.

4.5 Discussion and Conclusions

In this paper, we proposed a robust ground plane detection and segmentation technique for unorganized point clouds. The existing methods for detecting the ground plane of a 3D indoor scene rely on one or more of the following: 2D RGB data processing, video sequence data processing, organized point clouds, or ideal locations and orientations of the sensor. Additionally, certain essential assumptions, such as the ground plane being the largest plane in the scene and being perpendicular to or touching visible foreground objects (such as a human body), are generally required. However, our PiGPDS can robustly detect the ground plane and segment almost all voxels associated with the ground surface of both organized and unorganized point clouds without these restrictive assumptions. Moreover, PiGPDS supports varied sensor locations and only requires a single point cloud of an indoor scene. Notably, it does not rely on point cloud colour or 2D data (e.g., colour or texture).

First, we utilize a voxelized grid downsampling approach to speed up the downstream processes of our algorithm. Next, we group the downsampled point cloud into different zones using the K-medoids clustering algorithm and identify the ground zone using our proposed angular-based approach. Then, in order to improve the accuracy and efficiency of the subsequent plane fitting algorithm, we employ the Euclidean clustering algorithm to divide the ground zone into multiple parallel clusters. After that, using the Orientation-based MSAC algorithm, we fit the largest plane with a specific orientation within each cluster. Finally, we utilize a three-step ground surface segmentation technique consisting of ground plane estimation, distance-based verification, and distance-based segmentation to accurately detect the ground plane and segment all ground surface points of the original point cloud.

We evaluated our PiGPDS algorithm on twenty-five point clouds derived from our generated dataset and the public NYU depth dataset V2. Our algorithm accurately identified the ground plane and segmented the visible ground surface of twenty-three of the point clouds. For the other two point clouds without the visible ground surface (surrounded by the blue and red rectangles in Fig. 4.4), PiGPDS correctly identified the absence of a visible ground surface in one of them (Bedroom3_013_v4) and failed to do the same for the other point cloud (Bedroom3_018_v4) that contains only the ceiling and two wall surfaces of the 3D scene. Therefore, it does not meet our only assumption that input point clouds should contain at least one

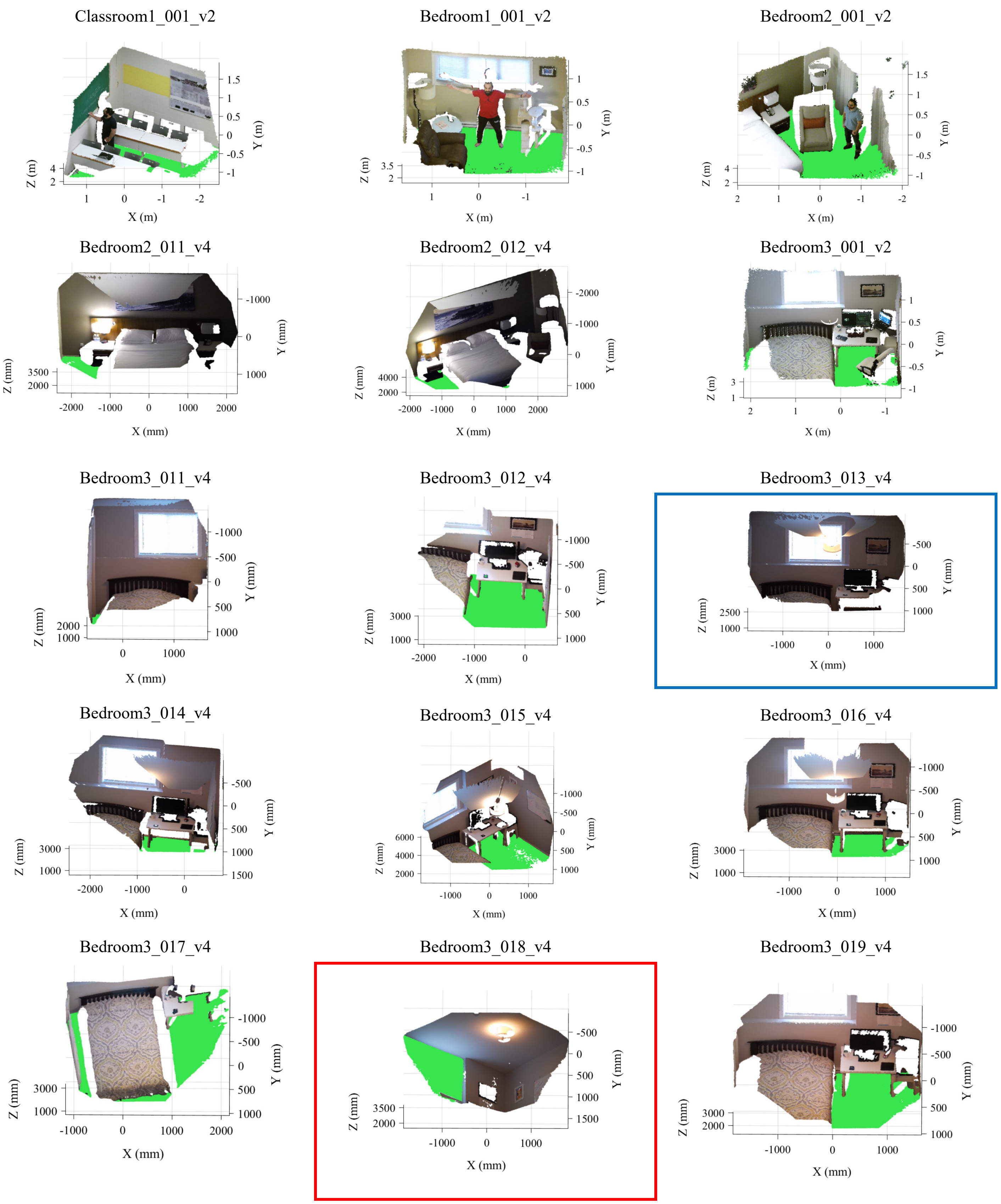


Figure 4.4 PiGPDS ground segmentation results on our dataset's point clouds; the segmented ground planes are highlighted in green over the original point clouds. The blue rectangle represents a point cloud without a visible ground surface, and the red rectangle shows an example of a point cloud without any visible surfaces parallel to the actual ground plane where PiGPDS fails to detect the ground plane correctly.

surface parallel to the actual ground plane, excluding the ceiling surface, even if the ground plane itself is not visible. As a result, the algorithm erroneously detected the ground zone, leading to incorrect segmentation of the window as the scene's ground plane.

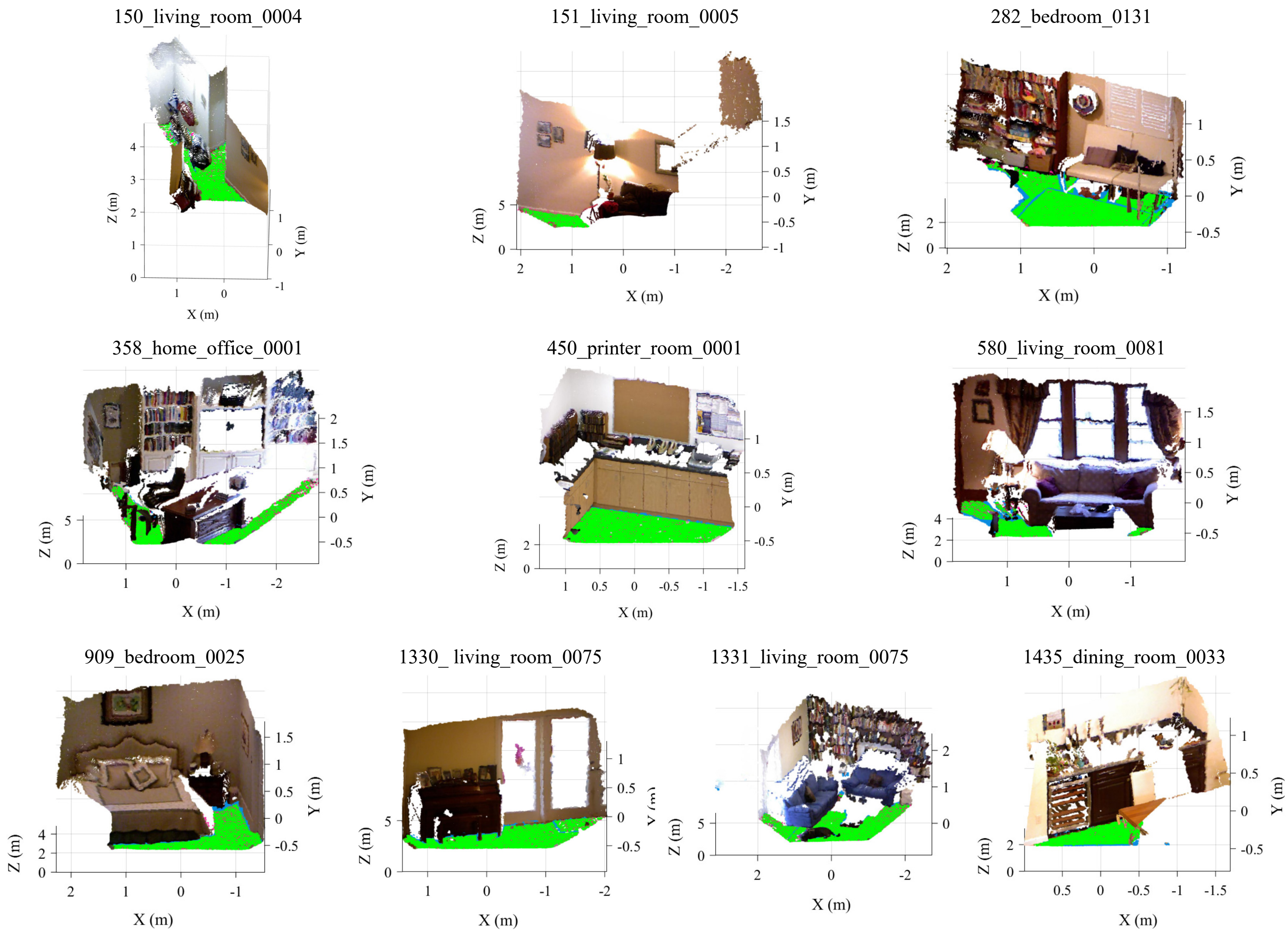


Figure 4.5 PiGPDS ground segmentation results on the public NYU dataset; the segmented ground planes (green), the false positive (blue) and the false negative (red) points are highlighted over the original point clouds.

Our experimental and evaluation results suggest that PiGPDS can successfully segment all points of the ground surface in 3D complex indoor scenes without erroneously including points associated with foreground objects in contact with the ground surface. For example, see human feet, a pet body, and furniture legs in the point clouds represented in Fig. 4.4 and Fig. 4.5 that are in contact with the ground surfaces but are not segmented by PiGPDS. In addition, our results demonstrate that PiGPDS is a robust and perspective-independent ground plane detection, accurately segmenting ground surfaces of complex 3D indoor scenes acquired from different locations with varying pitches and yaws (e.g., see the segmented ground surfaces of point clouds derived from our generated challenging dataset shown in Fig. 4.4). Furthermore, based on our experimentation, PiGPDS is fast regarding execution times, as evidenced by the algorithm run times for both datasets listed in Tables 4.1 and 4.2. These all make our PiGPDS algorithm highly suitable for a wide range of real-world applications where fast and reliable ground plane detection or segmentation is critical such as augmented reality, 3D object segmentation, robot navigation, and accurately labeling ground surfaces of 3D point clouds, providing benefits to machine learning-based computer vision applications.

In future work, we will expand our generated dataset by incorporating more complex 3D point clouds acquired from a diverse range of indoor environments to evaluate the robustness and accuracy of PiGPDS under more challenging scenarios and improve its applicability to real-world applications. We will also explore eliminating the only assumption of PiGPDS that at least one surface is parallel to the actual ground plane of a 3D scene.

4.6 References

[1] Microsoft, “Azure Kinect.” Accessed: Mar. 05, 2023. [Online]. Available: https://azure.microsoft.com/en-us/products/kinect-dk

[2] Microsoft, “HoloLens.” Accessed: Jun. 04, 2023. [Online]. Available: https://www.microsoft.com/en-us/hololens

[3] Apple Inc., “iPhone Pro.” Accessed: Jun. 06, 2023. [Online]. Available: https://www.apple.com/ca/iphone-14-pro/specs/

[4] M. Billinghurst, A. Clark, and G. Lee, “A survey of augmented reality,” Foundations and Trends® in Human–Computer Interaction, vol. 8, no. 2–3, pp. 73–272, 2015.

[5] F. Gao, W. Wu, W. Gao, and S. Shen, “Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments,” J Field Robot, vol. 36, no. 4, pp. 710–733, 2019.

[6] A. G. Bruzzone and F. Longo, “3D simulation as training tool in container terminals: The TRAINPORTS simulator,” J Manuf Syst, vol. 32, no. 1, pp. 85–98, 2013.

[7] A. Kulshreshth, K. Pfeil, and J. J. LaViola, “Enhancing the gaming experience using 3D spatial user interface technologies,” IEEE Comput Graph Appl, vol. 37, no. 3, pp. 16–23, 2017.

[8] M. Tatzgern, R. Grasset, D. Kalkofen, and D. Schmalstieg, “Transitional augmented reality navigation for live captured scenes,” in 2014 IEEE Virtual Reality (VR), IEEE, 2014, pp. 21–26.

[9] S. Oßwald, J.-S. Gutmann, A. Hornung, and M. Bennewitz, “From 3D point clouds to climbing stairs: A comparison of plane segmentation approaches for humanoids,” in 2011 11th IEEE-RAS International Conference on Humanoid Robots, IEEE, 2011, pp. 93–98.

[10] T. Liu, Y. Liu, Z. Tang, and J.-N. Hwang, “Adaptive ground plane estimation for moving camera-based 3D object tracking,” in 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP), IEEE, 2017, pp. 1–6.

[11] D. Conrad and G. N. DeSouza, “Homography-based ground plane detection for mobile robot navigation using a modified em algorithm,” in 2010 IEEE International Conference on Robotics and Automation, IEEE, 2010, pp. 910–915.

[12] J. Arróspide, L. Salgado, M. Nieto, and R. Mohedano, “Homography-based ground plane detection using a single on-board camera,” IET Intelligent Transport Systems, vol. 4, no. 2, pp. 149–160, 2010.

[13] S. Kumar, A. Dewan, and K. M. Krishna, “A bayes filter based adaptive floor segmentation with homography and appearance cues,” in Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing, 2012, pp. 1–8.

[14] B. Tan, N. Xue, S. Bai, T. Wu, and G.-S. Xia, “PlaneTR: Structure-guided transformers for 3d plane recovery,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 4186–4195.

[15] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa, “PlaneNet: Piece-wise planar reconstruction from a single rgb image,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2579–2588.

[16] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz, “PlaneRCNN: 3d plane detection and reconstruction from a single image,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4450–4459.

[17] Y. Xie, M. Gadelha, F. Yang, X. Zhou, and H. Jiang, “PlanarRecon: Real-time 3D Plane Detection and Reconstruction from Posed Monocular Videos,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 6219–6228.

[18] R. O. Duda and P. E. Hart, “Use of the Hough transformation to detect lines and curves in pictures,” Commun ACM, vol. 15, no. 1, pp. 11–15, 1972.

[19] M. A. Fischler and R. C. Bolles, “Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,” Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[20] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter, “The 3d hough transform for plane detection in point clouds: A review and a new accumulator design,” 3D Research, vol. 2, no. 2, pp. 1–13, 2011.

[21] R. A. Zeineldin and N. A. El-Fishawy, “Fast and accurate ground plane detection for the visually impaired from 3D organized point clouds,” in 2016 SAI computing conference (SAI), IEEE, 2016, pp. 373–379.

[22] L. Li, F. Yang, H. Zhu, D. Li, Y. Li, and L. Tang, “An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells,” Remote Sens (Basel), vol. 9, no. 5, p. 433, 2017.

[23] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, “Real-Time Plane Segmentation Using RGB-D Cameras.,” RoboCup, vol. 7416, pp. 306–317, 2011.

[24] C. Zhang and S. Czarnuch, “Perspective independent ground plane estimation by 2D and 3D data analysis,” IEEE Access, vol. 8, pp. 82024–82034, 2020.

[25] P. H. S. Torr and A. Zisserman, “MLESAC: A new robust estimator with application to estimating image geometry,” Computer vision and image understanding, vol. 78, no. 1, pp. 138–156, 2000.

[26] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, “Comparing ICP variants on real-world data sets,” Auton Robots, vol. 34, no. 3, pp. 133–148, 2013.

[27] R. B. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” in 2011 IEEE international conference on robotics and automation, IEEE, 2011, pp. 1–4.

[28] L. Kaufman and P. J. Rousseeuw, Finding groups in data: an introduction to cluster analysis. John Wiley & Sons, 2009.

[29] J. MacQueen, “Classification and analysis of multivariate observations,” in 5th Berkeley Symp. Math. Statist. Probability, 1967, pp. 281–297.

[30] E. Herman, K.-E. Zsido, and V. Fenyves, “Cluster Analysis with K-Mean versus K-Medoid in Financial Performance Evaluation,” Applied Sciences, vol. 12, no. 16, p. 7985, 2022.

[31] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, “Surface reconstruction from unorganized points,” in Proceedings of the 19th annual conference on computer graphics and interactive techniques, 1992, pp. 71–78.

[32] R. B. Rusu, “Semantic 3D object maps for everyday manipulation in human living environments,” KI-Künstliche Intelligenz, vol. 24, no. 4, pp. 345–348, 2010.

[33] MATLAB, “MATLAB.” The MathWorks Inc., Natick, Massachusetts, 2022.

[34] T. Caliński, J. Harabasz, and T. Caliliski, “A dendrite method for cluster analysis,” COMMUNICATIONS IN STATISTICS, vol. 3, no. 1, pp. 1–27, 1974.

[35] D. L. Davies and D. W. Bouldin, “A cluster separation measure,” IEEE Trans Pattern Anal Mach Intell, no. 2, pp. 224–227, 1979.

[36] P. J. Rousseeuw, “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis,” J Comput Appl Math, vol. 20, pp. 53–65, 1987.

[37] A. Ebrahimi and S. Czarnuch, “Automatic super-surface removal in complex 3D indoor environments using iterative region-based RANSAC,” Sensors, vol. 21, no. 11, p. 3724, 2021.

[38] W. Förstner and B. P. Wrobel, Photogrammetric computer vision. Springer, 2016.

[39] A. Ebrahimi and S. Czarnuch, “VISOR Lab 3D-Dataset.” Accessed: Jul. 14, 2023. [Online]. Available: https://github.com/visorlab/3D-Dataset

[40] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from rgbd images.,” ECCV (5), vol. 7576, pp. 746–760, 2012.

[41] A. V. Vo, L. Truong-Hong, D. F. Laefer, and M. Bertolotto, “Octree-based region growing for point cloud segmentation,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 104, pp. 88–100, 2015, doi: 10.1016/j.isprsjprs.2015.01.011.

[42] J. Yan, J. Shan, and W. Jiang, “A global optimization approach to roof segmentation from airborne lidar point clouds,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 94, pp. 183–193, 2014, doi: 10.1016/j.isprsjprs.2014.04.022.

[43] Z. Dong, B. Yang, P. Hu, and S. Scherer, “An efficient global energy optimization approach for robust 3D plane segmentation of point clouds,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 137, pp. 112–133, 2018, doi: 10.1016/j.isprsjprs.2018.01.013.

Chapter 5. PiPCS: Perspective Independent Point Cloud Simplifier for Complex 3D Indoor Scenes

Co-authorship statement – This chapter was published as an article titled "PiPCS: Perspective Independent Point Cloud Simplifier for Complex 3D Indoor Scenes" in the journal IEEE Access in August 2024. Ali Ebrahimi, as the primary author, conducted the literature review, conceptualized the study, designed the methodology, developed the method, collected data, analyzed the results, created visualizations, and both drafted and revised the manuscript. The co-author, Dr. Stephen Czarnuch, provided essential guidance and supervision throughout the research process, and meticulously reviewed and revised the manuscript. Both authors have read and agreed to the final version of the manuscript.

5.1 Abstract

The emergence of commercially accessible depth sensors has driven the widespread adoption of 3D data, offering substantial benefits across diverse applications, ranging from human activity recognition to augmented reality. However, indoor environments present significant challenges for 3D computer vision applications, particularly in cluttered and dynamic scenes where background bounding surfaces hinder the detection and analysis of foreground objects. We introduce a novel perspective-independent point cloud simplifier (PiPCS) for complex 3D indoor scenes. PiPCS streamlines 3D computer vision workflows by contextually segmenting and subtracting background bounding surfaces and preserving segmented foreground objects within indoor scenes, effectively reducing the size of indoor point clouds and enhancing 3D indoor scene perception. Methodologically, we use estimated surface normals to intelligently divide an input point cloud into distinct zones, which are then split into multiple distinct parallel clusters. Next, we find the largest plane in each cluster and sort the fitted planes within each zone based on their distance along the zone's normal vector to identify the bounding surfaces. Finally, we segment the 3D background, simplify the point cloud by employing a voxel-based background subtraction technique, and segment 3D foreground objects via a cluster-based segmentation approach. We evaluated PiPCS on the Stanford S3DIS dataset and our own challenging dataset, achieving average values of 97.08% for specificity and 91.27% for F_1 score on the S3DIS dataset and size reductions averaging 74.11% overall. Our experimental and evaluation results demonstrate that PiPCS robustly simplifies and reduces the size of unorganized indoor point clouds.

Index Terms – 3D background subtraction, 3D foreground segmentation, 3D indoor scene perception, indoor point cloud simplification, indoor point cloud preprocessing, point cloud size reduction

5.2 Introduction

Background subtraction in the domains of image processing and computer vision is of paramount significance. This essential preprocessing step simplifies scene information by isolating important foreground objects, enhancing the efficiency and accuracy of a wide range of computer vision applications, such as moving object detection [1], video surveillance [2], video conferencing [3], face recognition [4], and human tracking [5]. For instance, in video surveillance, background subtraction improves security threat detection by differentiating normal changes from the unauthorized presence of people. Similarly, in video conferencing, background identification can increase the privacy and focus of the participants by, for example, blurring or changing the background.

Historically, RGB data has been highly valuable in background modeling due to its comprehensive colour information, capturing the diverse visual world. Algorithms for identifying the background and ultimately isolating foreground objects using RGB data often rely on analyzing colour variations, patterns, and contrasts, along with employing frame difference techniques. In addition, recent advancements in Artificial Intelligence (AI), fueled by the availability of large collections of 2D images, have made RGB-based methods even more effective for background modeling. These AI-powered approaches use deep learning and neural networks to understand RGB data better, helping them accurately identify and separate foreground objects in complex scenes.

On the other hand, background modeling using 2D RGB data involves several challenges, such as bootstrapping, colour camouflage, illumination changes, intermittent motion, moving backgrounds, and foreground shadows, as elaborated in [6] and [7]. Bootstrapping is the task of learning a scene's background model even when there are no empty training frames. Colour camouflage makes segmentation difficult when foreground objects closely match the background's colour. Illumination changes require adapting to varying lighting conditions (and as a result, varying colours and intensities) for accurate detection, while intermittent motion focuses on detecting objects that may stop or start moving. Moving backgrounds can be along a continuum from extreme (e.g., entire scene changes) to more subtle (e.g., waving trees), and foreground shadows can significantly impact both the background and foreground objects.

Compared to RGB sensors, depth sensors, including stereo vision, structured light, time-of-flight (ToF) and LiDAR, provide spatial information about the scene that holds the potential to address some of the aforementioned challenges. Stereo vision sensors (e.g., ZED X [8]) function by analyzing the disparities in perspective between images captured by a pair of cameras, enabling the estimation of depth information. Structured light sensors (e.g., Microsoft Kinect V1) operate by projecting a predefined infrared (IR) light pattern onto a scene. When this pattern interacts with the scene, the pattern becomes distorted. Subsequently, an infrared camera captures this distorted pattern, resulting in a depth map that represents the scene's 3D structure. ToF sensors (e.g., Microsoft Kinect V2, Microsoft Azure Kinect [9], and ToF sensor integrated in Microsoft HoloLens headset [10]) operate by emitting pulses of infrared light and measuring the time it takes for these pulses to travel to an object and bounce back to the camera's sensor. This timing information is then used to calculate the distance from the camera to various points on the object's surface, yielding improved accuracy and performance in depth sensing compared to structured light sensors. Similarly, LiDAR scanners, such as those integrated into devices like the Apple Vision Pro [11] and iPhone 15 Pro [12], also emit light pulses and measure the time it takes for these pulses to reflect off objects. However, LiDAR scanners use laser pulses, enabling them to provide highly detailed 3D representations of the environment with increased precision and range compared to ToF sensors.

Although depth data are less prone to the aforementioned challenging issues that impact RGB-based background modeling algorithms, relying solely on depth data can also introduce several challenges, as discussed in [7] and [13]: depth shadows, depth camouflage, sensor distance limitations, and specular surface reflections. Depth shadows occur when foreground objects obstruct the passage of the sensor's infrared light, whereas depth camouflage arises when foreground objects situated near the background share similar depth values. Sensor distance limitations, when objects are either too close or too far from the sensor, along with specular surface reflections common in shiny materials such as glass windows and mirrors, both contribute to depth measurement challenges.

As elaborated in [14] regarding the state-of-the-art Azure Kinect sensor, the depth cameras may provide invalid values, typically represented as a depth value of 0, for some 3D pixels (i.e., voxels) in certain situations. The invalidated voxels typically result from oversaturated IR signals, insufficient IR signal strength for depth generation, and pixels containing mixed signals from both the foreground and background, particularly noticeable around object edges. Moreover, voxels may be invalidated when they receive signals from multiple objects within the scene. This phenomenon is often observed in corners due to IR light reflecting off one wall onto another, introducing uncertainty into depth measurements [14]. Notably, our experiments indicate that the Azure Kinect sensor demonstrates greater sensitivity in such scenarios compared to its predecessor (the Kinect V2 sensor), as noticeable in point clouds illustrated in Fig. 5.11 and Fig. 5.10, respectively.

Thus, many recent methods aim to leverage the complementary relationship between colour and depth data acquired from RGBD sensors. Typically, these approaches either extend existing background models initially designed for the RGB data or depth data or create two separate background models for the scene using one based on RGB data and the other based on the depth data, followed by the integration of results using various methods and criteria.

RGBD sensors (e.g., the Microsoft Kinect lineup) combine RGB and depth data to create a 3D point cloud. These sensors capture colour through RGB cameras and simultaneously measure object distances using depth sensors. The registration of these data generates a detailed 3D representation of the environment, with each voxel in the point cloud containing both colour and 3D coordinates. Point cloud data can be categorized into two main types: organized and unorganized datasets. Organized datasets are structured in a matrix-like format, allowing easy access to voxels based on their spatial or geometric relationships using indexing. In contrast, unorganized datasets lack such defined relationships between adjacent voxels, and the voxel coordinates are stored as an unordered one-dimensional array.

Converting an organized point cloud to an unorganized one is a straightforward task, whereas the reverse conversion process is notably more intricate and resource-intensive. As spatial relationships between voxels are maintained in organized point clouds, point cloud processing becomes less challenging. Nevertheless, computer vision methods tailored for unorganized datasets are universal [15], meaning they can also be applied to organized datasets. Hence, a robust 3D background modeling or foreground segmentation approach should be developed based on unorganized datasets without relying on the spatial relationships derived from the organized point clouds' indices.

While RGBD methods have demonstrated improved performance over depth-based techniques in certain contexts, they come with their own set of limitations. RGBD methods demand more computational resources due to two main factors: the fusion of depth and colour data to create a point cloud and the simultaneous processing of both colour and depth information. Additionally, RGBD-based methods are more error-prone in situations where colour information is limited or absent, such as dark indoor environments, compared to their depth-based counterparts. Furthermore, depth-based computer vision solutions, by design, promote privacy as they exclusively capture 3D coordinates of foreground objects (e.g., the human body) without reliance on more identifying colour data. This makes them suitable for applications like human tracking or pose estimation in private settings, such as monitoring patients or elderly individuals within the privacy of their homes.

5.2.1 Related Work

In the context of computer vision applications, various methods have been employed to subtract background or isolate foreground objects within 2D or 3D video sequences. These methods can be generally grouped into three categories based on the type of data they utilize: RGB-based, depth-based, and RGBD-based methods. Each of these categories comes with its own set of advantages and limitations.

5.2.1.1 RGB-based Background Subtraction Methods

RGB-based background subtraction methods have evolved significantly, integrating mathematical, signal processing, and machine learning principles. These methods encompass techniques that rely on both temporal changes in pixel values and colour information in the RGB colour space.

The most straightforward mathematical-based techniques for background modeling involve computing the temporal average, temporal median, or histogram over time. Despite their widespread adoption in earlier traffic surveillance systems, these methods are vulnerable to video surveillance challenges such as dynamic backgrounds and illumination changes [16].

In contrast, Wren *et al.* [17] introduced a statistical background subtraction model using the univariate Gaussian distribution. However, this method's reliance on a single Gaussian model limits its effectiveness to static backgrounds, leading to false detections in scenarios with moving elements like water fountains and swaying trees. In response, Stauffer and Grimson [18] presented a novel approach to background modeling, suggesting the representation of each pixel with an independent mixture model known as the Mixture of Gaussians (MOG). ViBe technique [19] is another noteworthy algorithm among statistical background subtraction methods. It creates and maintains a background model for each pixel by selecting a random subset of historical pixel values from the video frame. This model is continuously updated by comparing new pixel values with the stored historical data. When a pixel's value significantly deviates from the model, it is classified as part of the foreground, indicating potential motion or the presence of an object.

Fuzzy theory was introduced to the field of background subtraction by researchers seeking to address inaccuracies in complex scenarios. El Baf *et al.* [20] proposed an algorithm to handle uncertainties from dynamic backgrounds, employing the Type-2 Fuzzy Mixture of Gaussians model (T2F-MOG). Furthermore, in their subsequent study [21], they utilized the T2F-MOG model to extract moving objects in infrared videos. Additionally, Azab *et al.* [22] introduced a new technique for background modeling and subtraction for motion detection in real-time videos, incorporating a combination of colour, texture, and edge features alongside fuzzy concepts. This approach shows promise in effectively mitigating challenges arising from illumination variations and shadows.

Background modeling techniques frequently employ signal processing filters such as the Wiener filter [23] and Kalman filter [24] to model the background, as proposed in methods [25] and [26], respectively. While these methods are robust in handling gradual illumination changes, they are susceptible to challenges posed by highly dynamic backgrounds. In another study utilizing signal processing principles, Tezuka and Nishitani [27] introduced a foreground segmentation technique, leveraging Gaussian mixture models (GMMs) across various block sizes with Walsh transform (WT), where the WT's spectral nature significantly reduces computational steps. Their method shows robustness in challenging conditions such as heavy snow and changes in global lighting.

Background modeling using machine learning methods has been explored extensively, employing various techniques, including clustering, subspace learning, support vector machine (SVM), and neural network (NN) modeling. Several scholarly works have explored various clustering approaches for background modeling, such as employing k-means clustering, as detailed in [28] and the Codebook model, as discussed in [29]. These clustering-based models excel at producing accurate foreground objects, especially in scenarios with video noise and dynamic backgrounds.

Grassmannian Robust Adaptive Subspace Tracking (GRASTA) [30] and Incremental Principal Component Pursuit (incPCP) [31] are acknowledged as two pioneering achievements in subspace learning methods for background modeling. GRASTA excels in foreground-background separation by employing an advanced algorithm within the Grassmannian space, dynamically tracking and updating the background subspace to ensure robustness and adaptability to temporal changes. On the other hand, incPCP takes a unique approach to video background modeling by analyzing each frame individually, enabling real-time adaptation to background changes. This incremental approach not only makes incPCP ideal for streaming video applications but also minimizes memory usage and computational complexity.

SVM-based methods have been introduced to enhance the robustness of background modeling, particularly in dynamic environments. These models can effectively learn and adapt to complex environmental variations, making them well-suited when background elements change over time. For instance, Han and Davis [32] introduced an SVM-based method for background modeling and subtraction in video sequences, integrating various features like colour, gradient, and Haar-like features. Their method combines generative background modeling with discriminative classification using SVMs, resulting in robustness against shadows, illumination changes, and spatial background variations. Similarly, Lin *et al.* [33] proposed an automatic background initialization method for visual tracking systems, employing probabilistic SVM to overcome the unrealistic assumption of an absence of moving objects during initialization. Their approach formulates the problem as an online classification task, potentially enabling real-time performance. It evaluates all elements of each image frame using SVM classification, with a particular emphasis on the optical flow value and inter-frame difference as the two most crucial features for SVM-based classification.

One of the earliest approaches to use neural networks for background modeling and foreground detection was introduced by Schofield *et al.* [34] focusing on counting people in video images, employing a Random Access Memory (RAM) neural network. However, this method had two main limitations: it required the images to be highly representative of the scene's background and lacked a mechanism for background maintenance, as the information in the RAM-NN could not be modified once trained with a single pass of background images. Subsequently, several researchers have proposed neural network-driven techniques for background modeling in 2D videos. In the following, we briefly discuss some of the most widely recognized methods and refer readers to two exhaustive and detailed surveys, [35] and [36], for further exploration.

In 2008, Maddalena and Petrosino [37] proposed the Self Organizing Background Subtraction (SOBS) method, featuring a 2D self-organizing neural network architecture that preserves pixel spatial relations. This method automatically models the background through the network's neuron weights, initializing each pixel with a neural map consisting of $n \times n$ weight vectors using HSV colour values. New pixel information from subsequent video frames is then compared to the current model to determine whether the pixel corresponds to the background or foreground. In subsequent research, various enhancements and adaptations of the SOBS method have been developed, such as the Fuzzy and Coherence-based SOBS (SOBS_CF) algorithm [38], Spatially Coherent SOBS (SC-SOBS) algorithm [39], and Multi Independent-Layered Self-Balanced SOBS (MILSBSOBS) algorithm [40], each offering unique advantages in specific scenarios. However, despite their effectiveness, a significant drawback of SOBS-based approaches is the requirement for manual adjustment of at least four parameters [41].

In 2016, Braham and Van Droogenbroeck [42] were the first to apply Convolutional Neural Networks (ConvNets) to the task of background subtraction. Their background subtraction algorithm, inspired by the architecture of the LeNet-5 model [43], comprises four main stages: extracting the background model, generating a specific-scene dataset, training the network, and performing a ConvNet-based background subtraction. The main drawback of this method is its limited applicability to specific scenes, requiring retraining for different video contexts.

In 2018, Lim and Keles [44] introduced two robust encoder-decoder neural networks, Foreground Segmentation Network with Multiple Inputs (FgSegNet_M) and Foreground Segmentation Network with Single Input (FgSegNet_S), tailored for moving object segmentation. FgSegNet_M adapts the first four blocks of the VGG-16 Net [45] and utilizes a triplet CNN alongside a Transposed Convolutional Neural Network (TCNN). On the other hand, FgSegNet_S employs a single-input CNN followed by a Feature Pooling Module (FPM) and the TCNN. In 2020, the authors introduced an enhanced architecture known as FgSegNet-V2 [46]. FgSegNet-based models achieved remarkable success by securing the top five positions on the CDnet2014 [41] dataset, a benchmark widely used for evaluating moving object segmentation algorithms.

In 2020, Zheng *et al.* [47] proposed a novel background subtraction algorithm based on parallel vision and Bayesian generative adversarial networks (GANs). The algorithm begins by employing the median filtering algorithm for background image extraction. Subsequently, it constructs the background subtraction model using Bayesian GANs to classify pixels into foreground and background, while leveraging parallel vision theory to enhance results in complex scenes. This method ranked sixth on the CDnet2014 dataset, whereas the author's initial background subtraction algorithm [48] using Bayesian GANs secured the eighth position.

In 2021, Rahmon *et al.* [49] introduced MU-Net1 and MU-Net2, variants of Motion U-Net (MU-Net), a hybrid moving object detection system comprising a single-stream encoder module followed by a decoder module. MU-Net integrates motion, change, and appearance cues using an autoencoder deep convolutional neural network. MU-Net1 uses only a single RGB frame without temporal information, while MU-Net2 employs a three-channel input stream including grayscale, flux motion, and change cues. The encoder-generated feature maps are processed through the decoder stage, resulting in a robust, multi-cue system for detecting moving objects. MU-Net2 and MU-Net1 achieved the seventh and ninth positions, respectively, on the CDnet2014 dataset.

Machine learning-based methods, particularly those utilizing neural networks, have significantly advanced the detection of moving objects against static backgrounds, often outperforming traditional methods. Nevertheless, they face challenges such as sensitivity to dynamic backgrounds, camera jitter, and camouflage [35]. Furthermore, recent comparisons presented in [50] among several state-of-the-art background modeling methods have revealed that utilizing depth information leads to notably superior performance compared to relying solely on colour. In response, researchers have explored depth-based and RGBD-based background subtraction methods, leveraging depth information to enhance detection accuracy.

5.2.1.2 Depth-based Background Subtraction Methods

Depth-based background subtraction techniques center around exploiting the distinct spatial relationships between foreground objects and their surrounding backgrounds within the depth data. These methods often adopt well-known 2D background modeling techniques, including thresholding, single Gaussian [17], MoG [18], and frame difference, and are employed across a spectrum of 3D computer vision applications such as fall detection [51], human tracking [52] and gesture recognition [53].

Thresholding, a fundamental 2D segmentation technique, serves as a basic method for background subtraction in depth maps. For instance, Cinque *et al.* [54] employed Otsu thresholding to isolate foreground objects in the depth map and subsequently refined the segmented foregrounds by applying region growing and morphological operations. Czarnuch and Mihailidis [55] calculated the average historical value for each pixel and utilized a pixel-based threshold to segment the foreground image but required a sequence of background images.

The single Gaussian and MoG models have been widely utilized for depth-based background subtraction. For example, Rougier *et al.* [51] employed the single Gaussian model for fall detection among the elderly, and Zhang *et al.* [56] embraced it as a key initial step in object detection. Frick *et al.* [57] employed the MoG algorithm to isolate foreground regions from the background for creating 3D-TV content.

The well-established frame difference method, initially developed for background subtraction of 2D videos, has demonstrated adaptability in scenarios where only depth information is available. This method has been employed in several 3D computer vision applications, including human tracking [52] and gesture recognition [53], to segment foreground objects such as human bodies.

While the aforementioned depth-based background subtraction techniques can address certain limitations of 2D background modeling methods, such as challenges related to illumination changes, they still inherit some of the drawbacks associated with 2D background modeling techniques from which they are derived. For example, choosing the suitable depth threshold can still be challenging when the background itself is dynamic or contains moving objects, often requiring manual tuning based on the depth map's characteristics.

To overcome some of these limitations, particularly in the context of indoor scenes with static backgrounds, Ebrahimi and Czarnuch [58] explored parametric modeling of the background surfaces (e.g., walls, windows, and floor) using the well-known plane segmentation algorithm RANdom SAmple Consensus (RANSAC) [59]. The proposed method isolates foreground objects by iteratively segmenting and removing large background surfaces within four local regions. However, it falls short in accurately segmenting all the surfaces that belong to a non-planar background, containing surfaces with differing normal vectors, such as those between a window and a wall. Additionally, it is unsuitable for 3D scenes with more than four static background surfaces, such as when the point cloud includes the ceiling surface along with three walls and a floor, or when the indoor scenes contain more than four walls.

5.2.1.3 RGBD-based Background Subtraction Methods

Leveraging both RGB colour and depth information, RGBD-based background subtraction methods offer enhanced accuracy and robustness. By integrating colour data with valuable 3D spatial information, these techniques overcome the limitations of RGB or depth-based approaches, especially in challenging scenarios such as colour camouflage and depth camouflage. These methods typically involve either refining existing background models designed for RGB or depth data or alternatively, constructing two distinct background models for the scene, one relying on RGB data and the other on depth data, and subsequently integrating the results using various criteria.

One of the early contributions to RGB-D background estimation is the method proposed by Gordon *et al.* [60]. This method, an adaptation of the MoG algorithm, integrates both colour and depth data captured using a stereo imaging system. It models each background pixel using a mixture of four-dimensional Gaussian distributions, with three components representing the colour data in the YUV colour space and the fourth component representing the depth data. The method treats colour and depth independently, adjusting colour-matching criteria based on the reliability of depth data. When depth data is reliable, decisions prioritize depth over colour, reducing colour camouflage errors. Conversely, an unreliable stereo-matching algorithm tightens colour-matching criteria to address issues like shadows or local illumination changes. Similarly, Clapés *et al.* [61] proposed a surveillance system employing a per-pixel background subtraction technique, utilizing a four-dimensional Gaussian distribution to integrate colour and depth information for object recognition and user identification. However, unlike the method by Gordon *et al.*, they used a single Gaussian distribution.

Another noteworthy contribution to the RGB-D background modeling, leveraging the MoG model, is the research conducted by Camplani and Salgado [62]. They introduced an approach that combines two per-pixel statistical classifiers, one based on depth features and the other on colour features. The likelihood function of the background utilized in both classifiers relies on a mixture of Gaussians model. Additionally, the combination of classifiers relies on a weighted average, which dynamically adjusts each classifier's support within the ensemble by taking into account foreground detections in preceding frames, as well as depth and colour edges. The objective is to mitigate false detections resulting from challenging issues that individual classifiers cannot effectively address, such as shadows, illumination changes, colour camouflage, and depth camouflage.

Several RGBD-based methods leverage the ViBe algorithm for background subtraction. Leens *et al.* [63] introduced a multi-camera setup for video segmentation, integrating colour and depth data from a low-resolution ToF camera. Their algorithm independently applies the ViBe algorithm to the colour and depth data, producing foreground masks that are subsequently combined using logical operations and post-processed with morphological operations. Ottonelli *et al.* [64] enhanced ViBe's colour segmentation by integrating a compensation factor derived from data captured by a stereo camera. Their method combines colour and depth information to improve foreground segmentation, leading to more accurate detection of human silhouettes. Similarly, Zhou *et al.* [65] constructed colour and depth models using the ViBe algorithm and fused them with a weighting system that accounts for the reliability of depth information. By assigning weights to the depth data based on its uncertainty, their approach ensures robust segmentation across different scenes and lighting conditions.

The capabilities of the 2D Codebook background model are enhanced by certain RGBD-based methods, such as [66] and [67]. Fernandez-Sanchez *et al.* [66] employed the Codebook method for RGB-D based background modeling, utilizing data from Kinect cameras. They developed a Depth-Extended Codebook (DECB) approach, combining RGB and depth data directly into the model. Their findings demonstrate that their DECB method outperforms a four-dimensional Codebook approach, where depth serves as an extra channel in the background model. Similarly, Murgia *et al.* [67] extended the Codebook model to address scenarios with varying illumination conditions. Their method involved integrating depth information into the Codebook model, facilitating RGB-D fusion during segmentation. Furthermore, they incorporated colorimetric invariants to ensure colour consistency across diverse lighting conditions.

In recent years, there has been a surge in research focused on RGB-D background modeling and subtraction techniques. In the following, we spotlight some of the latest recognized methods and direct readers to a detailed survey [68] for further exploration.

Moya-Alcover *et al.* [69] introduced the Generic Scene Modeling (GSM) approach, which integrates depth and colour information for scene modeling. They utilized Kernel Density Estimation (KDE) with a three-dimensional Gaussian Kernel to construct background models for each pixel. The model incorporates one dimension for depth information and two for normalized chromaticity coordinates. Through a probabilistic strategy, the method distinguishes between background and foreground objects and detects changes in background objects within frames based on pixel model distributions. Additionally, GSM incorporates a probabilistic depth data model to handle inaccuracies, improving foreground segmentation and ensuring robust segmentation across different scenarios. However, one limitation of KDE-based approaches like GSM are their relatively high computational complexity, which poses challenges for real-time applications.

Despite the inherent slowness of KDE-based methods, Trabelsi *et al.* [70] improved the computational efficiency of their proposed RGBD-KDE algorithm by adapting the Fast Gaussian Transform. Similar to GSM, their approach constructs a scene background model using KDE, albeit with a two-dimensional Gaussian kernel. Depth information is integrated into one kernel dimension, while intensity, calculated as the average of RGB components, is represented in the other dimension.

Javed *et al.* [71] presented the Spatiotemporal Robust Principal Component Analysis (SRPCA) algorithm for detecting moving objects in RGBD videos. Their method comprises three primary stages: first, identifying dynamic images to create a dynamic input sequence by filtering out static video frames; second, computing spatiotemporal graph Laplacians; and finally, employing RPCA to integrate the previous two steps for separating background and foreground components. They evaluated their algorithm on the SBM-RGBD dataset [72] tailored for moving object detection, achieving a notable third-place ranking in performance.

Maddalena and Petrosino introduced the RGB-D Self-Organizing Background Subtraction (RGBD-SOBS) method [73], which initially constructs separate background models for colour and depth data before merging them. Leveraging their previous work on RGB videos, specifically the SC-SOBS algorithm [39], they developed distinct background models for both colour and depth using a self-organizing neural background model. The resulting colour and depth detection masks are used to guide the selective model update process, which helps refine the background models and ultimately generates the final detection masks. Additionally, in their subsequent study

[50], they provided further insights into the RGBD-SOBS algorithm and extensively evaluated its robustness to various challenges in maintaining colour and depth backgrounds, offering comparisons with state-of-the-art methods. RGBD-SOBS and SC-SOBS (i.e., RGB-SOBS) demonstrated outstanding performance, securing the first and second positions on the SBM-RGBD dataset, respectively.

Although RGBD-based background subtraction methods have shown better performance compared to colour-based and depth-based techniques in specific scenarios, they have inherent limitations. One common drawback is their computational complexity, particularly in real-time applications dealing with high-resolution videos. This complexity arises from the integration of depth and colour data to create a point cloud and the simultaneous processing of both types of information, requiring substantial computational resources. Furthermore, RGBD-based approaches are more prone to errors in environments with limited or absent colour information, such as dark indoor settings. Finally, almost all these methods are not well-suited for complex and cluttered 3D indoor environments or scenarios where the positions and orientations of sensors are not known or ideal.

5.2.2 Contributions

Our perspective-independent point cloud simplifier (PiPCS) expands the conventional concept of background subtraction to encompass the identification, segmentation, and removal of 3D background bounding surfaces, such as walls, windows, curtains, ceilings, and floors, from indoor point clouds. Conventional background subtraction techniques are commonly used to identify foreground objects (e.g., moving objects in video or image sequences) without any particular consideration for specific background content. Our work leverages the unique opportunity to contextually segment and remove 3D background components and retain segmented

3D foreground objects within indoor point cloud scenes without relying on colour and historical information. This distinction makes it particularly well-suited to applications in complex indoor environments, where distinguishing between foreground and background points can be a challenging task. Our point cloud simplifier contributes to the field of 3D point cloud processing by offering four comprehensive solutions.

1) PiPCS serves as an essential preprocessing step for a variety of 3D computer vision applications, including 3D object segmentation, human tracking, and human activity recognition. As a preprocessing step, PiPCS effectively narrows the search space for downstream processes by segmenting 3D background and foreground objects of indoor scenes, allowing for the easy elimination of only background objects in tasks requiring foreground identification, resulting in notable performance and accuracy improvements of these applications.
2) PiPCS can significantly reduce the size of 3D indoor point clouds by eliminating their large background bounding surfaces, which are non-essential for the majority of 3D computer vision applications, thereby yielding a streamlined and compact dataset. This contribution addresses a critical challenge in the field, where large point cloud datasets can pose significant computational and storage burdens. In addition, PiPCS has the potential to optimize data transmission processes by first subtracting the unchanging and redundant 3D background from all point cloud frames and then transmitting one instance of the 3D background alongside the 3D foreground objects of each frame. This supplementary application can contribute to data transmission efficiency and resource optimization.
3) PiPCS significantly advances 3D indoor scene perception by accurately segmenting and identifying all points associated with each boundary of the 3D background, including the ceiling, floor, and surrounding walls within an indoor scene. The specific emphasis is on accurately identifying the ground plane because once known, it inherently provides information about the ceiling and surrounding walls.

This approach enables a comprehensive understanding of the spatial relationships among 3D background elements and 3D foreground objects within indoor scenes. This capability is particularly beneficial in applications such as virtual reality, where achieving an accurate interpretation of 3D space is crucial for immersive experiences and precise scene representation.

4) PiPCS can serve as a labeling or annotation tool by accurately segmenting both 3D background and 3D foreground objects within indoor point clouds, providing benefits to machine learning-based computer vision applications.

Our PiPCS distinguishes itself through the following key advantages:

- Perspective independence: PiPCS supports a wide range of sensor heights relative to the ground, along with different pitches and yaws, enabling it to adapt to almost all practical sensor perspectives. This is critical to ensure optimal system performance and consistent results in diverse and non-ideal sensor setups and data collection scenarios.
- Privacy enhancement: Our approach prioritizes privacy by exclusively utilizing 3D coordinates of foreground objects, such as the human body, without the need for descriptive colour data. This characteristic makes it particularly well-suited for applications in sensitive environments, such as private healthcare settings where monitoring patients or elderly individuals requires privacy protection.
- Universal sensor and data compatibility: PiPCS relies solely on the raw depth map from unorganized point clouds, confirming its adaptability to a range of sensor types. This key advantage enables its application across a wide range of 3D sensors, making it a versatile solution that seamlessly works with both organized and unorganized point clouds.

The subsequent sections of this paper are organized as follows. In the next section, we elaborate on our proposed point cloud simplifier. In section 5.4, we present the experimental and evaluation results of our proposed method. In section 5.5, we discuss our findings and future research directions, followed by conclusions in section 5.6.

5.3 Point Cloud Simplifier

Our perspective-independent point cloud simplifier (PiPCS) has four main steps, as illustrated in Fig 5.1 We begin with splitting the input point cloud into different zones based on the normal vectors of the point cloud voxels. Second, we segment each zone into several large plane-like clusters. Third, we find the largest plane in every cluster, and then we sort the planes within each zone by distance along the zone normal. Finally, following the segmentation of 3D background, we derive the simplified point cloud using a voxel-based background subtraction technique and segment 3D foreground objects via a cluster-based segmentation method.

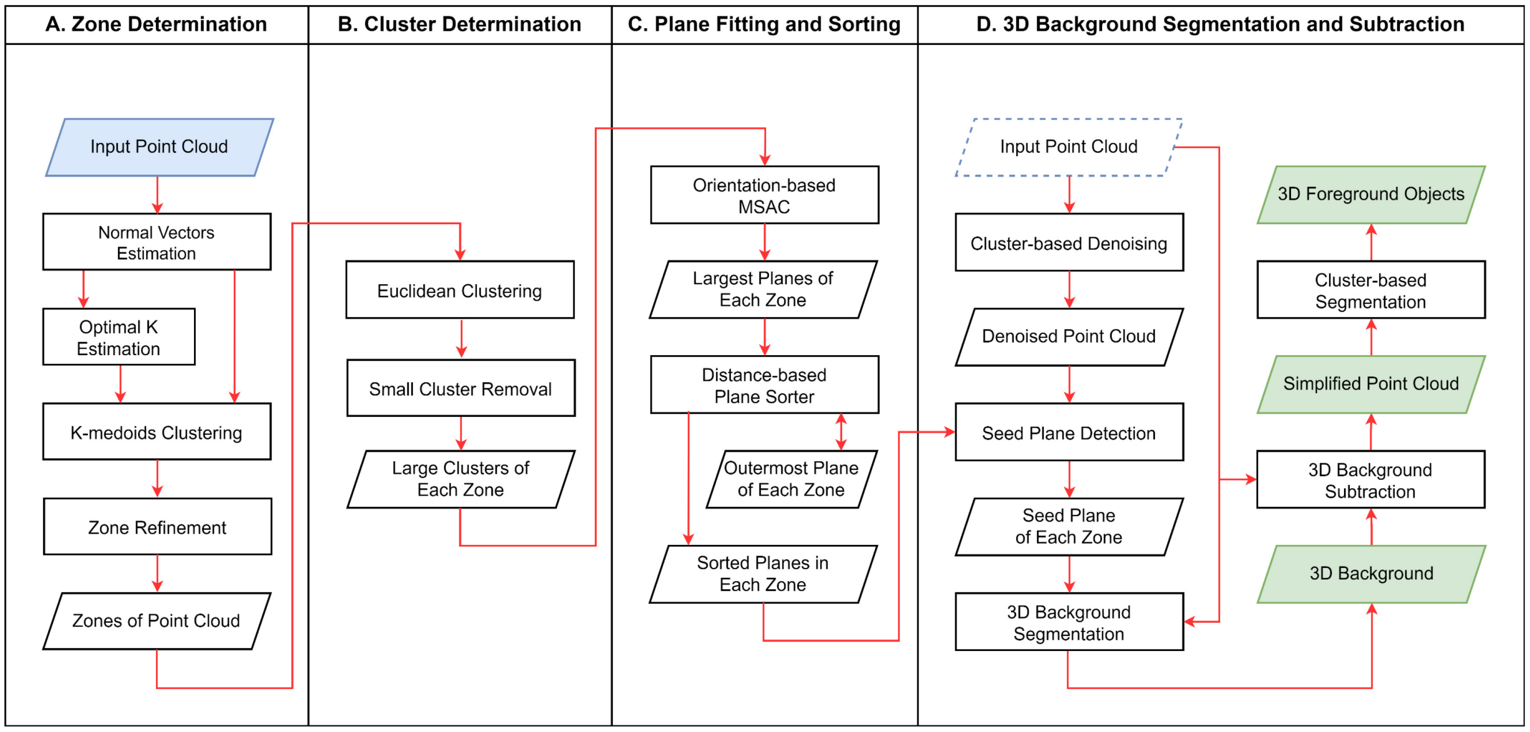


Figure 5.1 The flowchart of PiPCS

5.3.1 Zone Determination

Dividing our captured point clouds into different zones based on the normal vectors of point cloud points, and processing each zone individually can significantly increase the efficiency and accuracy of the MSAC algorithm (see section 5.3.3.1) and, as a result, the accuracy of our point cloud simplifier. Since each zone has fewer points than the point cloud itself and all points in every zone have similar normal vectors (i.e., the number of outlier points of the MSAC algorithm dramatically decreases), the MSAC algorithm runs faster and more accurately. To identify zones automatically in a general point cloud captured from an arbitrary view perspective, we compute the normal vectors, estimate the optimal number of clusters, cluster the point cloud voxels, and refine the zones, producing point cloud subsets for each zone.

5.3.1.1 Normal Vectors Estimation

We can estimate the normal of a surface point by estimating the normal of its tangent plane, which can be achieved through a least-square plane fitting estimation problem [74]. We use the neighbouring points of each point in the input point cloud to fit a local plane and determine its normal vector. Due to the fact that the sign of a point's surface normal cannot be mathematically determined [75], the orientations of the estimated normals are initially vague and lack consistency across the entire point cloud, as shown in Fig. 5.2a. To tackle this issue, we flip the normals with respect to the point cloud's centroid, as shown in Fig. 5.2b.

5.3.1.2 Optimal Cluster Size (k) Estimation

Unsupervised partitioning algorithms like the *k*-means clustering algorithm [76] can be used to group data into a finite number of *k* distinct clusters. However, the outcomes of these algorithms can be different based on chosen parameters, particularly the value of *k* (i.e., the number of clusters). Therefore, before clustering the adjusted normal vectors (see section 5.3.1.3), it is necessary to estimate the optimal number of clusters, *k*, that will represent all distinct normal clusters within an unknown indoor environment.

To evaluate the optimal number of clusters, it is common to analyze data properties such as density, cluster size, and separability. Clustering analysis typically aims to minimize the variance within clusters while maximizing the separation between them. Several cluster validation techniques, such as well-known Calinski-Harabasz [77], Davis-Bouldin [78], and Silhouette [79][80] have been proposed to evaluate the optimal *k*. We employed the most common validation techniques to determine the optimal *k* for clustering the normal vectors of different indoor point clouds. Based on our experimentation, the Calinski-Harabasz technique is most effective across a range of indoor environments.

5.3.1.3 K-medoids Clustering

After estimating the optimal *k* of the input point cloud, we can more accurately group the adjusted normal vectors of the point cloud's points into *k* clusters. We use *k*-medoids clustering [80] to divide the point cloud points into distinctive zones with similar normal vectors.

The *k*-medoids algorithm groups a set of observations into *k* subsets, minimizing the sum of distances between observations and the center of their respective clusters. However, unlike the *k*-means algorithm, *k*-medoids clustering always assigns a member of a cluster, known as a medoid, as the center of the cluster, which allows us to exploit the normal vector of each medoid as the reference vector for each zone in the next section, zones refinement, and in section 5.3.3.1, Orientation-based MSAC. Furthermore, in comparison to *k*-means, *k*-medoids clustering is known for its greater robustness to noise and outliers [81].

Figure 5.3a shows an indoor point cloud obtained using Microsoft Azure Kinect, and Fig. 5.3b demonstrates the result of *k*-medoids clustering of the point cloud as a 3D scatter plot. The 3D plot illustrates how *k*-medoids clustering groups the normal vectors of the sample point cloud into six clusters; each cluster corresponds to a specific zone of the point cloud where every point shares a similar normal vector, and all surfaces are almost parallel. Fig. 5.3c shows the point cloud zones highlighted over the sample point cloud.

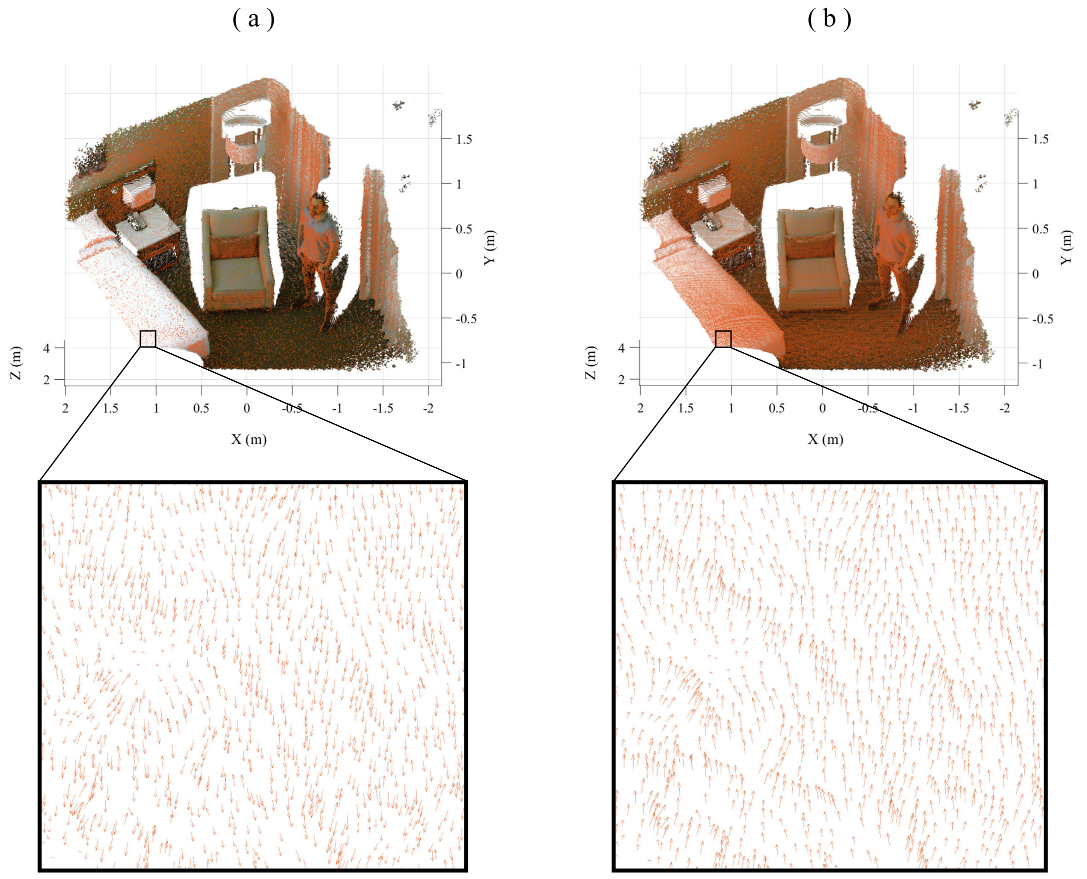


Figure 5.2 Normal vectors estimation: (a) estimated normals, and (b) adjusted normals are shown over a sample point cloud.

5.3.1.4 Zone Refinement

Normal vectors located on the borders of two clusters (e.g., see Fig. 5.3b) may not be clustered accurately. These points usually connect surfaces at different angles in the original point cloud (e.g., borders between adjacent surfaces highlighted with different colours in Fig. 5.3c), or belong to objects that do not have flat surfaces (e.g., cushions).

To ensure all points within a point cloud zone have almost the same normal vectors, we refine each zone based on the angles of its normal vectors with the zone's reference vector (i.e., the normal vector of the zone's medoid). After computing the angles between each zone's reference vector and its normal vectors, we eliminate any point with an angle more than a threshold ($\alpha = 20\text{ degrees}$) from the point cloud zone. Fig. 5.3d demonstrates the result of the zone refinement for the sample point cloud, with points that were removed from zone clusters retaining their original colour (i.e., not being assigned the colour of the zone).

Our zone refinement technique increases the accuracy of the downstream plane fitting approach (i.e., the MSAC algorithm) by preserving only the surface points with the same normal vectors in each zone while incurring minimal computational overhead due to its efficient implementation through vectorization techniques.

5.3.2 Cluster Determination

In this step, we segment each of the refined zones into multiple clusters based on the Euclidean distance between the zone's voxels. As shown in Fig. 5.3d, each refined zone is composed of various parallel surfaces that are situated at different distances along the zone's surface normal; for instance, see the green zone (the ground and other green surfaces parallel to the ground) in Fig. 5.3d. By dividing the refined zones into individual clusters, we enable the MSAC algorithm (see section 5.3.3.1) to efficiently fit the optimal plane within each cluster while preventing the segmentation of spurious planes [82].

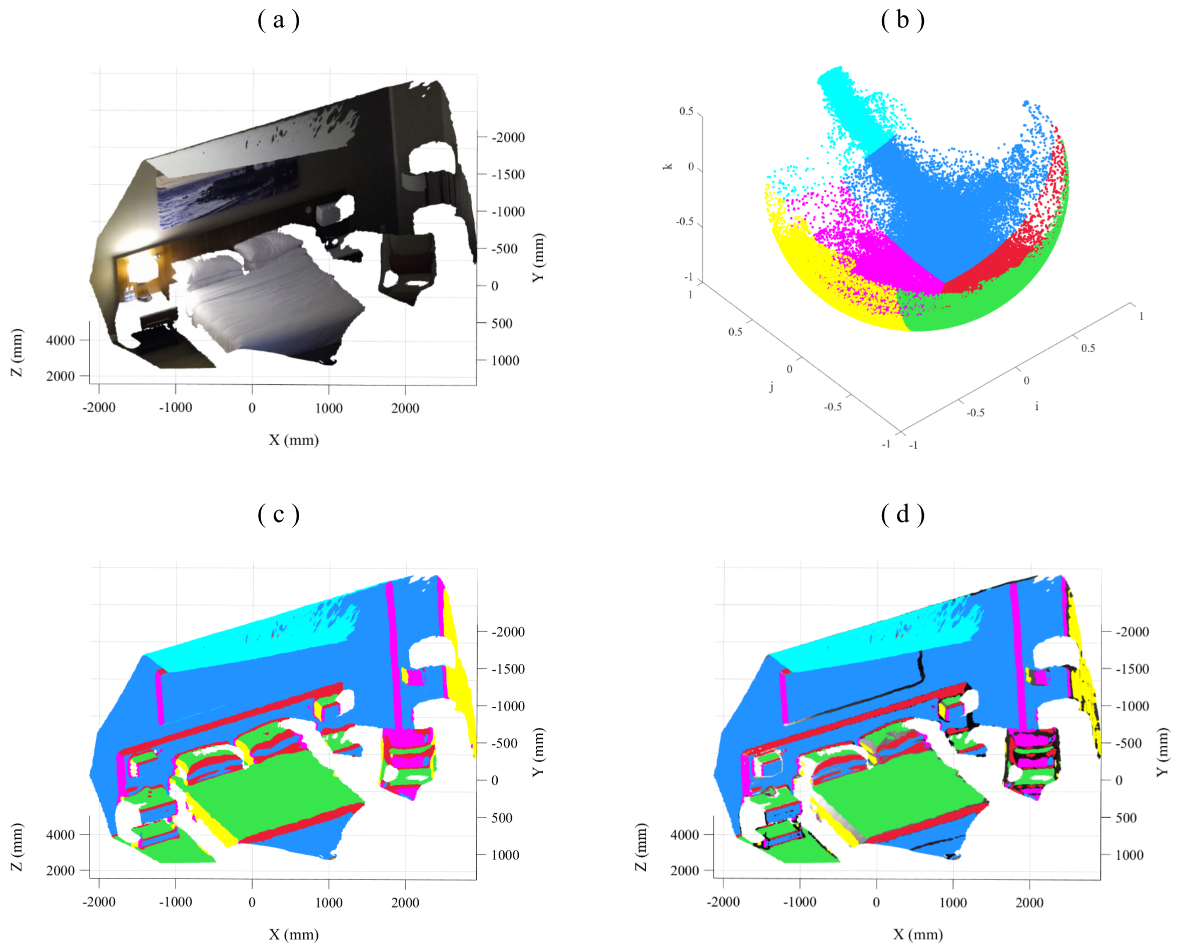


Figure 5.3 *k*-medoids clustering and zone refinement: (a) the sample point cloud, (b) *k*-medoids clustering of the normal vectors of the sample point cloud represented as a 3D scatter plot, (c) the initial zones, and (d) the refined zones of the point cloud highlighted over the sample point cloud. Note, the sensor is located very close to the room ceiling and covers a portion of the ceiling highlighted with cyan colour over the sample point cloud.

5.3.2.1 Euclidean Clustering

To cluster each zone of the point cloud, we first calculate the Euclidean distance between each point and its neighbouring points. Then, we group adjacent points into a cluster if the distance between them is below a threshold ($\varepsilon = 5$ cm). We repeat this process until all the clusters are determined. For instance, the Euclidean clustering of the green zone in Fig. 5.3d results in 20 individual clusters, as demonstrated in Fig. 5.4a.

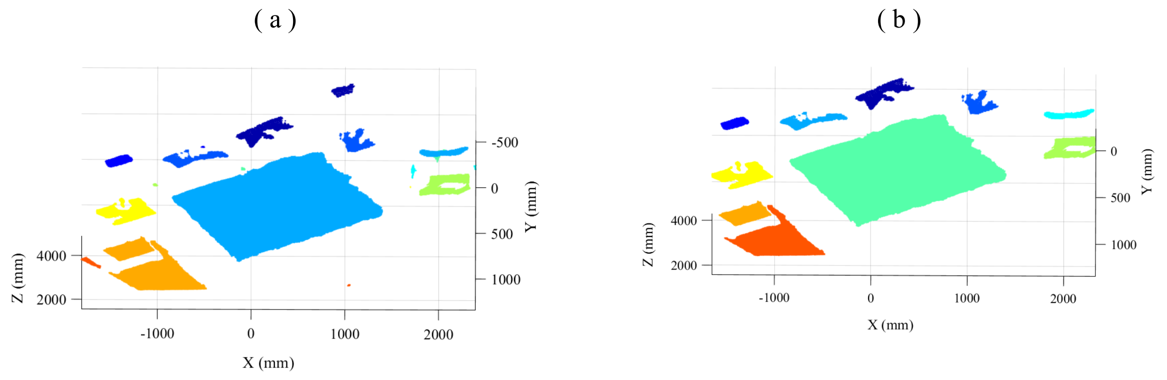


Figure 5.4 Euclidean clustering and small cluster removal: (a) the initial clusters and (b) the large clusters of the green zone in Fig. 5.3d.

5.3.2.2 Small Cluster Removal

As illustrated in Fig. 5.4a, some clusters can be very small, typically associated with small foreground objects or sensor noise. Since our point cloud simplifier aims to segment the 3D background of a point cloud (i.e., the large bounding surfaces) and remove them from the original point cloud, we can preserve potential background clusters by removing small clusters and only preserving the large clusters.

We accomplish this by removing clusters from each zone that are smaller than a threshold, adaptive to the size of the point cloud ($\mu = 0.1\%$ of point cloud points). For example, by removing the small clusters shown in Fig. 5.4a, we decrease the number of clusters from 20 to 10, as illustrated in Fig. 5.4b. Removing the zones' small clusters improves the speed of our PiPCS by preserving and analyzing only the large clusters (i.e., by employing the downstream plane fitting approach in fewer clusters). Moreover, small clusters may not contain sufficient points to fit planes robustly using the MSAC algorithm.

5.3.3 Plane Fitting and Sorting

In this step, we use the M-estimator SAmple Consensus (MSAC) algorithm [83] to segment the largest plane with a predetermined orientation in each of the large clusters of every zone. Then, we sort the segmented planes in each zone based on a distance-based approach.

5.3.3.1 Orientation-based MSAC

The MSAC algorithm, an enhanced version of the RANSAC algorithm [59], iteratively selects three random voxels as a minimum subset within each cluster to generate a hypothesis plane. During each iteration, the algorithm computes the distance between the plane and the remaining voxels of the cluster and identifies the number of inliers within a predefined distance threshold ($\delta = 2$ cm). The algorithm ultimately returns the plane with the highest percentage of inliers.

The main difference between RANSAC and MSAC is how they calculate the model's quality. RANSAC counts only the number of inliers within the threshold, regardless of how close they are to the model. However, MSAC uses the sum of all point-model distances as the quality measure. More specifically, as presented by Torr and Zisserman [83], RANSAC finds the minimum of a cost function C_1, defined as (5.1), where MSAC minimizes a new cost function C_2, defined as (5.2), where ρ_1 and ρ_2 can be calculated using (5.3) and (5.4), respectively, δ is the distance threshold, e_i is the Maximum Likelihood Estimation (MLE) error [83] for the ith point, and λ is a constant penalty.

$$C_1 = \sum_i \rho_1 \left(e_i^2\right) \tag{5.1}$$

$$C_2 = \sum_i \rho_2 \left(e_i^2\right) \tag{5.2}$$

Equation (5.1) conveys that, in RANSAC, each outlier scores a constant penalty, and inliers score nothing. Therefore, a significantly large distance threshold (δ) may cause a poor estimation. MSAC algorithm remedies this issue at no extra cost with a new cost function (5.2), where inliers are now scored on how well they fit the data, and outliers are still given a fixed penalty.

$$\rho_1(e^2) = \begin{cases} 0, & e^2 < \delta^2 \\ \lambda, & e \ \geq \delta^2 \end{cases} \tag{5.3}$$

$$\rho_2(e^2) = \begin{cases} e^2, & e^2 < \delta^2 \\ \delta^2, & e \ \geq \delta^2 \end{cases} \tag{5.4}$$

Since the normal vectors of all points within a refined zone are nearly the same and the points within each large cluster are generally located on a flat surface, the MSAC algorithm is effective in segmenting the planes in each zone. To enhance the accuracy of the plane fitting process and guarantee that all planes within a zone are parallel, we integrate an orientation-based approach into the MSAC algorithm. Our orientation-based MSAC prioritizes planes with the highest percentage of inliers that have an expected orientation relative to the surfaces of each zone. To accomplish this, we utilize the normal vector of each zone's medoid (see section 5.3.1.3, K-medoids Clustering) as the zone's reference vector and define a maximum allowable angular variation ($\beta = 5°$), the angle between the normal vectors of the zone's fitted planes and the reference vector. For instance, employing orientation-based MSAC in the large clusters of the green zone (shown in Fig. 5.4b) and the blue zone in Fig. 5.3d results in the segmented planes, as shown in Fig. 5.5a and Fig. 5.5b, respectively.

According to Förstner and Wrobel [84], the RANSAC algorithm requires a maximum number of iterations I to achieve convergence, and the same holds true for our orientation-based MSAC algorithm. The value of I is dependent on three factors: the outlier ratio o, the target success probability (p = 99%), and the number of samples ($s = 3$ for plane fitting) and can be approximated as (5.5). Increasing the success probability leads to a higher value of I, which results in greater computational costs but also improves the overall robustness of the output.

$$I = \frac{\log(1-p)}{\log(1-(1-o)^s)} \tag{5.5}$$

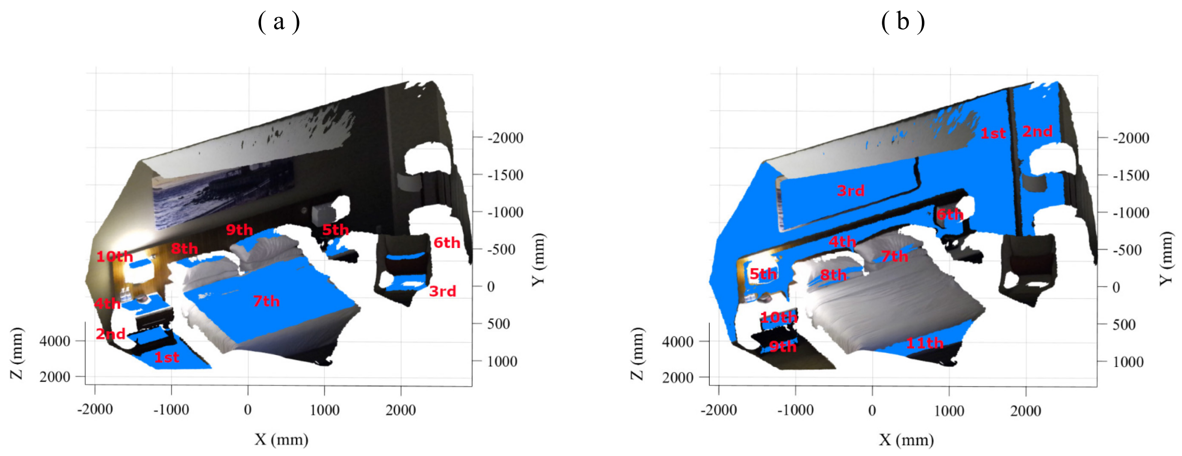


Figure 5.5 Orientation-based MSAC and distance-based plane sorter: the segmented planes of the green zone (a) and the blue zone (b) in Fig. 5.3d and their order numbers demonstrated over the sample point cloud.

5.3.3.2 Distance-based Plane Sorter

In this step, we sort the segmented planes in each zone based on their distances to the outermost plane of their own zone. To find the outermost plane of each zone, we iteratively compute and examine the distances between all the segmented planes of the zone. First, we randomly initialize one of the segmented planes of each zone as its outermost plane candidate, represented as ($ax + by + cz + d = 0$). Then, we proceed by calculating the distance $\overline{D}$ between the outermost plane candidate and the remaining planes of the zone. Given that the planes within a zone may not be entirely parallel and eventually intersect at some straight line in a 3D space, we calculate the average distance between the outermost plane candidate and all points belonging to each plane $\{(x_i,\ y_i,\ z_i): 1 \leq i \leq n\}$ as (5.6), where n represents the total number of points within each plane. During each iteration, if the calculated distance is negative (i.e., when the new plane is situated behind the outermost plane candidate), we replace the current outermost plane candidate with the new plane.

$$\overline{D} = \frac{\sum_{i=1}^{n} \frac{ax_i + by_i + cz_i + d}{\sqrt{a^2 + b^2 + c^2}}}{n} \tag{5.6}$$

After detecting the outermost plane of each zone, we can simply sort the remaining planes of every zone based on their distances to the outermost plane. For example, Fig. 5.5 shows the order of segmented planes of the green and blue zones in Fig. 5.3d, where the outermost planes are indicated as "1st", and the numbers of the other planes show how close they are to the outermost planes (e.g., the "2nd" planes are the closet ones to the outermost planes).

5.3.4 3D Background Segmentation and Subtraction

In this step, after denoising the input point cloud, we detect the seed plane of each zone (i.e., the zone's verified bounding plane closest to the center of the point cloud). Then, we segment the 3D background of the point cloud using the detected seed planes and employing a distance-based segmentation method. After that, we acquire the simplified point cloud using a voxel-based background subtraction method. Finally, we segment foreground objects of the point cloud using a cluster-based segmentation technique.

5.3.4.1 Cluster-based Denoising

Removing noisy or false data points from the original point cloud is a critical preprocessing task for the next step, where we examine the outermost plane of each zone as candidates of the point cloud's bounding planes (see section 5.3.4.2). The spurious data points located outside of a scene's actual boundaries (see Fig. 5.6a, for example) can lead to incorrect detection of seed planes in the next section, resulting in erroneous segmentation of the point cloud's background. These false data points usually occur because of the presence of reflective surfaces in indoor environments, such as glass windows or mirrors.

One of the most common point cloud denoising approaches is to examine the k-nearest neighbours of all data points and remove any point whose average distance to its k-nearest neighbours is above a threshold (i.e., eliminating disparate points from each neighbourhood) [85]. However, it does not work when we have clusters of false points (e.g., spurious points enclosed by a red rectangle in Fig. 5.6a). Therefore, we utilize a cluster-based denoising technique to remove these clusters of false points. First, we group point cloud points into different clusters using the Euclidean clustering introduced in section 5.3.2.1 (as shown in Fig. 5.6b). Then, we remove any cluster smaller than the adaptive threshold μ. Fig. 5.6c illustrates the result of the cluster-based denoising for a sample point cloud.

5.3.4.2 Seed Plane Detection

We use a three-step approach to detect the seed planes of the input point cloud's zones. The seed plane of a zone is its closest verified bounding plane to the center of the point cloud. First, we examine whether the outermost plane of a zone can be a bounding plane of the point cloud. Second, if the outermost plane is verified as a bounding plane, then we attempt to verify other bounding planes by examining the remaining zone's sorted planes. Finally, we select the closest bounding plane to the point cloud's center as the zone's seed plane.

In the first step, we examine the outermost plane of each zone (for example, the "1st" plane of the sample point cloud shown in Fig. 5.5b) to verify it as a bounding plane at the extremes of its zone, associating but not limited to the scene's floor, walls, doors, windows, and curtains. The outermost plane of a zone can be verified as a bounding plane if its zone comprises some dataset points associated with any of the aforementioned bounding planes. Alternatively, an outermost plane can be a bounding plane if no dataset points are behind it, otherwise could be a part of a foreground object. For example, if the sample point cloud shown in Fig. 5.5a does not cover the floor (the "1st" plane) of the room, then the zone's outermost plane (the "2nd" plane) is a piece of a foreground object (the nightstand).

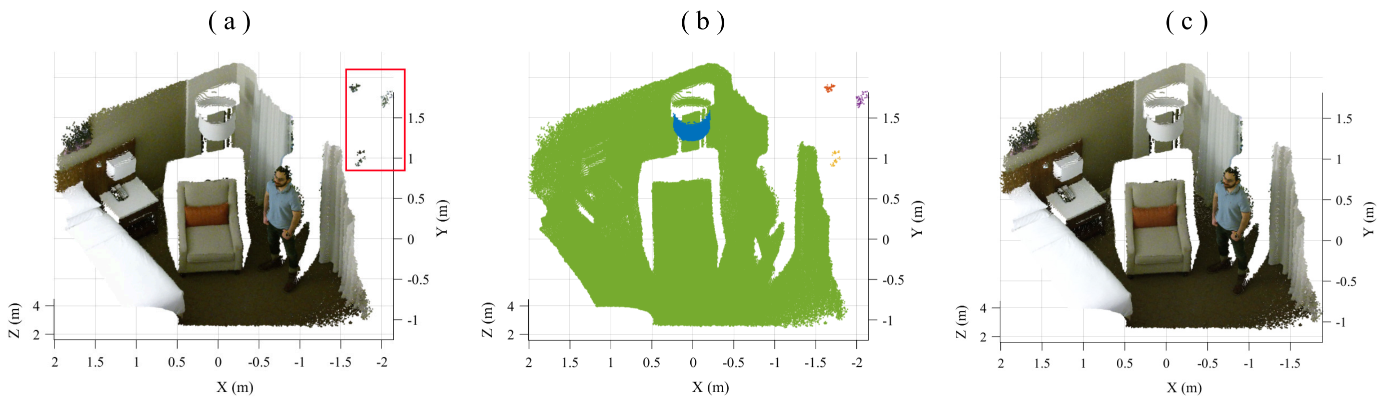


Figure 5.6 Cluster-based denoising: (a) a sample point cloud and its clusters of false points surrounded by a red rectangle, (b) the point cloud clusters highlighted with different colours over the point cloud, and (c) the denoised point cloud.

As discussed in the previous section, a point cloud can consist of spurious data points outside of a scene's actual boundaries. Therefore, we use the denoised point cloud to verify the outermost planes as the bounding planes of the original point cloud. First, we compute the distance D between the outermost plane of each zone ($ax + by + cz + d = 0$) and all points of the denoised point cloud (x, y, z) except points corresponding to the outermost plane itself as (5.7). Then, we verify any outermost plane as a bounding plane if almost all distances are greater than zero (i.e., less than negative of MSAC threshold, $-\delta$, to exclude outer points of the bounding plane not segmented by the MSAC algorithm, see section 5.3.3.1). For example, Fig. 5.7a and Fig. 5.7b show the outermost planes and the verified bounding planes of a sample point cloud, respectively. Note that the tiny red outermost plane above the floor (i.e., green plane) in Fig. 5.7a is not verified as a bounding plane and is removed from Fig. 5.7b.

$$D = \frac{ax + by + cz + d}{\sqrt{a^2 + b^2 + c^2}} \tag{5.7}$$

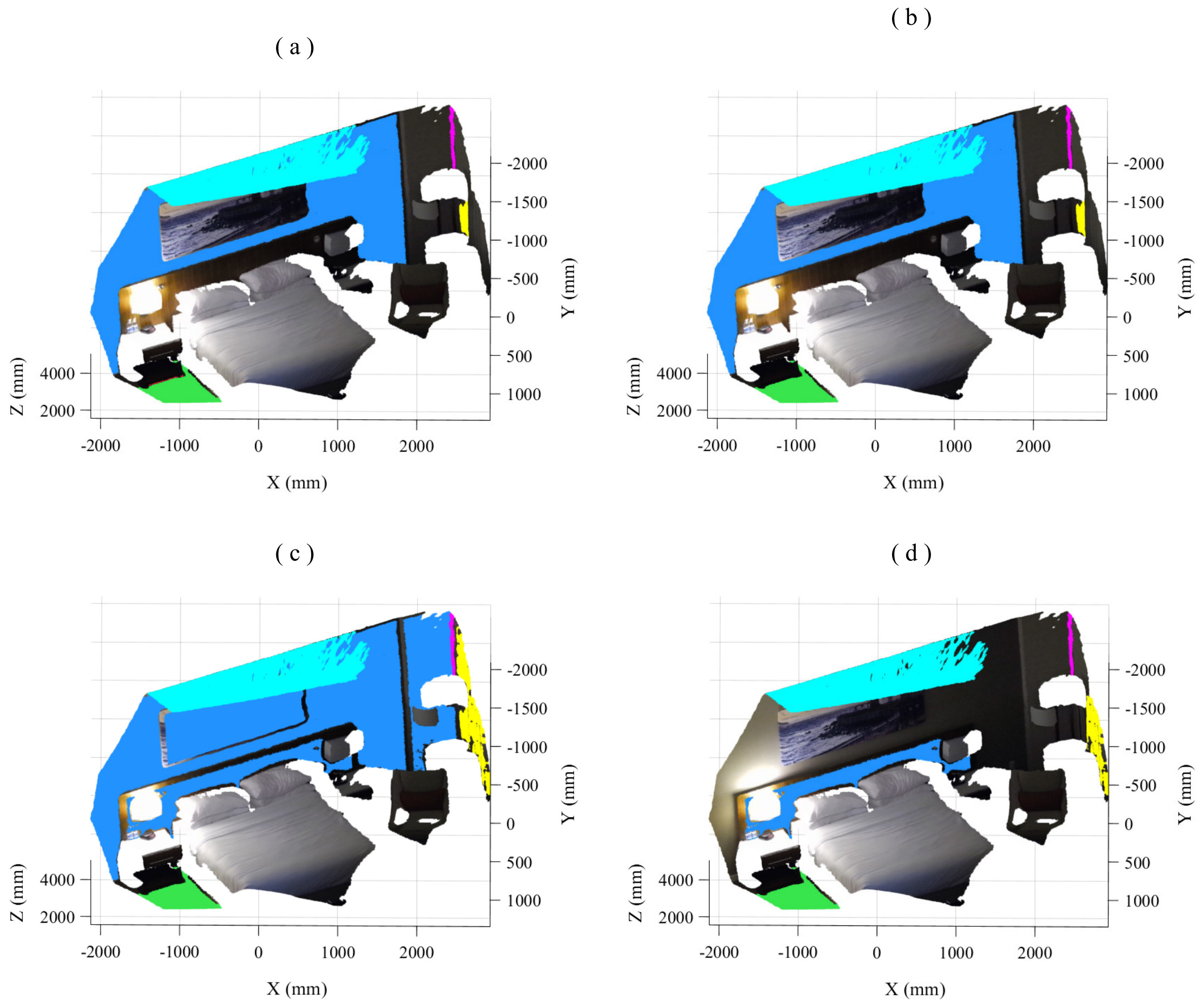


Figure 5.7 Seed plane detection: (a) the outermost planes, (b) the verified outermost planes as the bounding planes, (c) all the bounding planes, and (d) the seed planes of the sample point cloud.

In the second step, we aim to verify the bounding planes of more challenging indoor scenes where these planes are spread over the adjacent planes of the verified outermost planes, excluding the floor. To identify the floor, we utilize our perspective-independent ground plane detection method proposed in our previous work [86]. For example, as shown in Fig. 5.5b, the "2nd" plane (a section of the back wall), the "3rd" plane (a large picture on the back wall), and the "4th" plane (the bed headboard on the back wall) could also be accepted as the bounding planes of the sample point cloud.

If a zone's outermost plane is not verified as a bounding plane, the outermost plane should be associated with a foreground object (e.g., the tiny red outermost plane in Fig. 5.7a), and the zone does not contain any points associated with the 3D background of the point cloud (see the next section, 3D background segmentation). Otherwise, we continue to verify the successive sorted planes of the zone. In order to verify the next plane as a bounding plane, it should satisfy two conditions. First, its distance to the zone's outermost plane should be less than the maximum distance threshold (Δ). The maximum distance threshold between adjacent planes depends on the complexity and geometry of an indoor environment; based on our experimental results and due to the existence of the second condition, the algorithm works appropriately with $30\ cm \leq \Delta \leq 35\text{ cm}$. Second, the plane should also be large enough compared to the sizes of the previously verified bounding planes of its zone (i.e., the number of its points should not be fewer than the average of total points forming the previously verified bounding planes of its zone) to be considered a bounding plane. These two conditions prevent the algorithm from verifying a plane associated with the foreground objects as a bounding plane. Fig. 5.7c illustrates all verified bounding planes of the sample point cloud. Note that some of the point cloud's zones (e.g., the blue and yellow zones in Fig. 5.7c) may have more than one verified bounding plane to be considered as the seed plane of their zones.

In the last step, if a zone contains any bounding plane, we select the seed plane of the zone as follows: if the zone contains more than one bounding plane, the furthest bounding plane from the outermost plane (i.e., the closest one to the point cloud's center) is the seed plane (e.g., the blue and the yellow seed planes in Fig. 5.7d); otherwise, the outermost plane itself is the seed plane of the zone (e.g., cyan, magenta, and green seed planes in Fig. 5.7d).

5.3.4.3 3D Background Segmentation

The detected bounding planes of a complex indoor scene may not cover all the background surfaces at the outer boundaries of the point cloud, such as windows and curtains. In this section, we use the verified seed planes to segment the remaining points of the point cloud's background surfaces. Given that the seed planes constitute a significant part of the background surfaces, the nearest neighbour operations, such as region-growing algorithms, could be used to segment the remaining background surfaces. However, these methods may lead to incorrect segmentation of the 3D background when a foreground object touches one of the bounding surfaces. Additionally, these methods tend to be inefficient when applied to unorganized point clouds. Therefore, we introduce a distance-based segmentation technique to segment the remaining portions of the background surfaces associated with the seed planes from the input point cloud while minimizing the possibility of erroneously incorporating points that belong to foreground objects.

First, we initialize the 3D background of the point cloud with all the seed planes' points, and then we extend the background surfaces by adding all points behind each of these seed planes. To do this, as explained in the previous section, we utilize (5.7) to find the distance between all points of the original point cloud (unlike the previous section, where we used the denoised point cloud) and each seed plane. After that, we add points with a negative distance (i.e., less than MSAC threshold, δ, to include inner bounding plane points not segmented by the MSAC algorithm) to the 3D background. The background points of each zone are highlighted over the sample point cloud, as shown in Fig. 5.8b. Note that the background points associated with the magenta zone in Fig.

5.8b are common to both magenta and yellow zones. The 3D background of the sample point cloud is demonstrated in Fig. 5.8c.

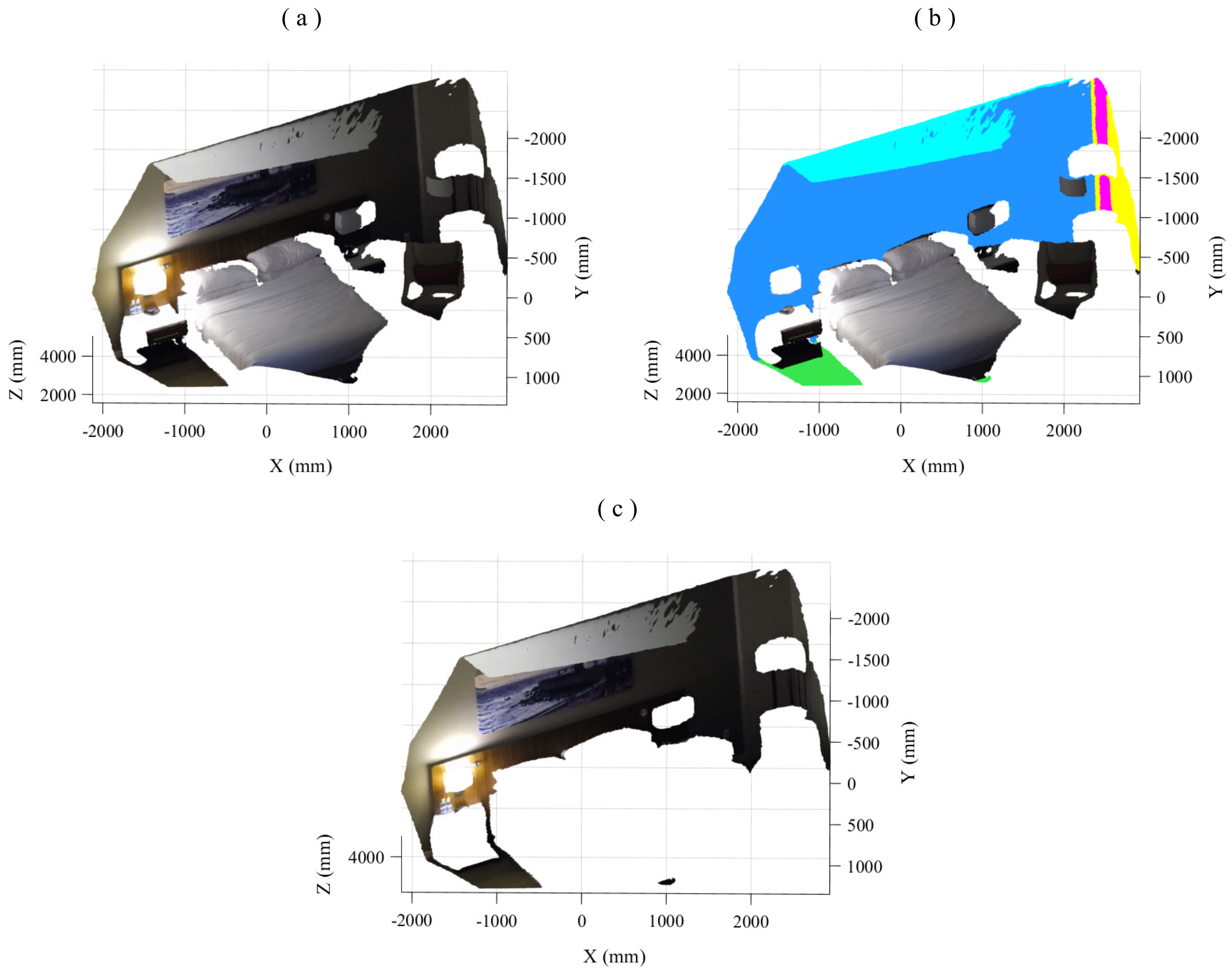


Figure 5.8 3D Background Segmentation: (a) the sample point cloud, (b) the zones' background points highlighted over the point cloud, and (c) the 3D background of the point cloud.

5.3.4.4 3D Background Subtraction

We can acquire the simplified point cloud (i.e., the 3D foreground of a point cloud), including only the foreground objects, such as humans, pets, and furniture, by subtracting the segmented 3D background from the original point cloud. Our 3D background subtraction is voxel-based such that we find the set difference between point cloud voxels and 3D background voxels to obtain the 3D foreground of the point cloud. The 3D foreground of the sample point cloud is shown in Fig. 5.9b.

5.3.4.5 Cluster-based Segmentation

One of the significant benefits of removing the 3D background from a point cloud is splitting groups of points associated with different foreground objects of the point cloud. Hence, we can individually segment foreground objects of a point cloud by clustering its 3D foreground utilizing the Euclidean clustering introduced in section 5.3.2.1. Fig. 5.9c demonstrates foreground objects of the sample point cloud, highlighted with different colours over the point cloud.

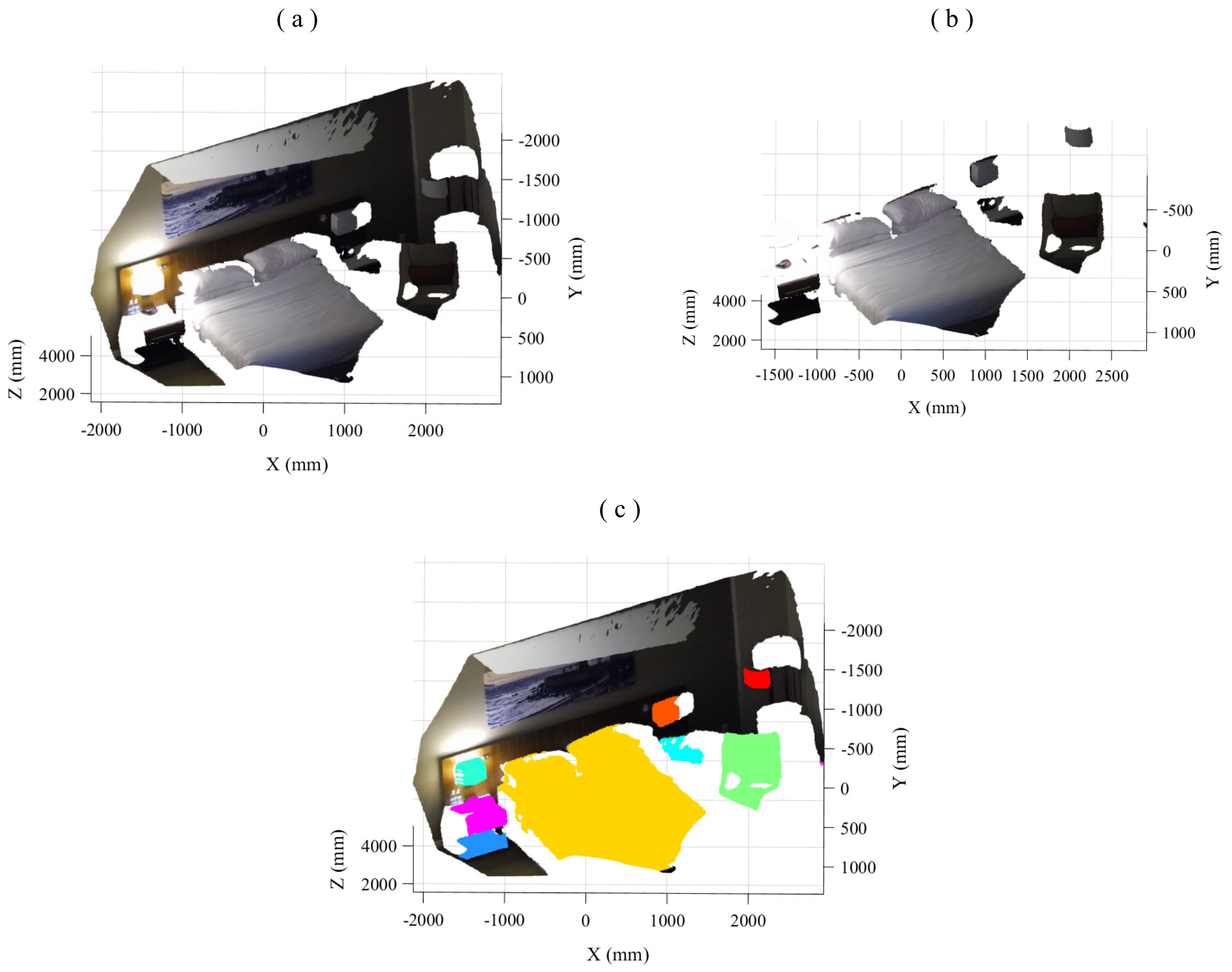


Figure 5.9 3D Background Subtraction and Cluster-based Segmentation: (a) the sample point cloud, (b) the 3D foreground of the point cloud, and (c) the foreground objects highlighted with different colours over the point cloud.

5.4 Experiments and Evaluation

5.4.1 Experimental Setup

We evaluated our PiPCS on our own generated dataset [87] and the Stanford large-scale 3D Indoor Spaces dataset (S3DIS) [88]. Our dataset consists of scenes captured using the Microsoft Kinect V2 and Kinect V4 (i.e., Azure Kinect) sensors placed at different locations with varying pitches and yaws, while the S3DIS depth dataset was acquired with the Matterport Camera [89], a 360° camera consisting of three structured-light sensors at different pitches. The two datasets feature diverse indoor settings, encompassing a wide range of objects such as furniture, planar objects, and human bodies, which are in contact with one of the bounding surfaces such as the ground plane, ceiling, and walls, as shown in Fig. 5.10, 5.11, and 5.12. We implemented our proposed algorithm using MATLAB on a system consisting of an Intel i7-12700H CPU @ 2.30 GHz, an NVIDIA GeForce RTX 3060 Laptop GPU, and 16 GB of RAM.

To conduct a quantitative evaluation of PiPCS, we assessed its efficiency on the labeled S3DIS dataset using four conventional voxel-based metrics: specificity, precision, recall, and F_1 score. These metrics, especially the last three, have been commonly utilized to quantitatively assess the effectiveness of plane segmentation, as evidenced by previous studies (e.g., [90], [91], [92], and [93]). Additionally, all four metrics were used in the SBM-RGBD dataset and challenge [94] as performance metrics for foreground object detection from RGBD videos. To compute these metrics, we classified points as follows: true positives (TP) are foreground points correctly identified, true negatives (TN) are background points correctly identified, false positives (FP) are background points incorrectly identified as foreground, and false negatives (FN) are foreground points incorrectly identified as background. The specificity, calculated as $TN/(TN + FP)$, measures our method's capability to differentiate the foreground points from the background points. Precision, calculated as $TP/(TP + FP)$, indicates the proportion of correctly segmented points compared to the total number of segmented points. Recall, calculated as $TP/(TP + FN)$, signifies the proportion of true positives relative to the total number of ground truth points. The F_1 score measures the overall performance of PiPCS by taking into account both precision and recall, calculated as $2 \times (precision \times recall) / (precision + recall)$.

For our experiments, we extracted point clouds of different indoor spaces by disjointing them from each of the six large areas presented in the S3DIS dataset. Since the dataset is annotated with semantic labels such as foreground objects, walls, floor, and ceiling, we were able to accurately define the ground truth points for the evaluation of our method.

Additionally, we conducted a qualitative evaluation of PiPCS on our challenging dataset, where the acquired point clouds vary significantly in sensor positions and orientations. In the absence of ground truth labels, we adopted visual inspection as the primary method to assess the PiPCS results on our dataset. This approach not only allowed us to examine the perspective independence of our PiPCS, but also enabled us to thoroughly evaluate the performance of our point cloud simplifier for complex 3D indoor scenes.

We quantified the size reduction achieved by PiPCS as (5.8), where OPC_count and SPC_count represent the number of voxels of the Original Point Cloud and the Simplified Point Cloud, respectively. This metric allows us to evaluate the effectiveness of our PiPCS in reducing point cloud size through a voxel-based analysis.

$$Size\ reduction = \frac{OPC_{count} - SPC_{count}}{OPC_{count}} \tag{5.8}$$

5.4.2 Experimental Results

Figures 5.10 and 5.11 show the output results of PiPCS on our dataset's point clouds captured using Microsoft Kinect v2 and Azure Kinect, respectively (see Table 5.1 for point clouds' details). As shown in the second and third columns of Fig. 5.10 and 5.11, our method successfully segments the 3D background and 3D foreground (i.e., the simplified point cloud) of all ten point clouds. The fourth column of each figure demonstrates the 3D foreground objects, highlighted with different colours over each point cloud.

Similarly, PiPCS correctly segmented nearly all points associated with the 3D background and 3D foreground of six different indoor scenes derived from the S3DIS dataset (see Table 5.2 for point clouds' details), as shown in the second and third columns of Fig. 5.12, respectively. However, PiPCS failed to classify challenging L-shaped structures (i.e., unconventional ceiling-wall intersections usually associated with a building's foundation) as the 3D background and misclassified them as foreground objects of the point clouds demonstrated in the first and fifth rows of Fig. 5.12 (see the two L-shaped structures surrounded by red rectangles in the third column of Fig. 5.12). The 3D foreground objects of each point cloud are individually segmented and highlighted with different colours over the original point cloud, as shown in the fourth column of Fig. 5.12.

Our experimental results suggest that PiPCS effectively simplifies unorganized point clouds of complex 3D indoor scenes and supports varied sensor locations. In addition, PiPCS segments 3D foreground, 3D background (i.e., all bounding surfaces, such as ceiling, walls, and floor), and 3D foreground objects of indoor scenes without using the point cloud's 2D data, such as colour and texture.

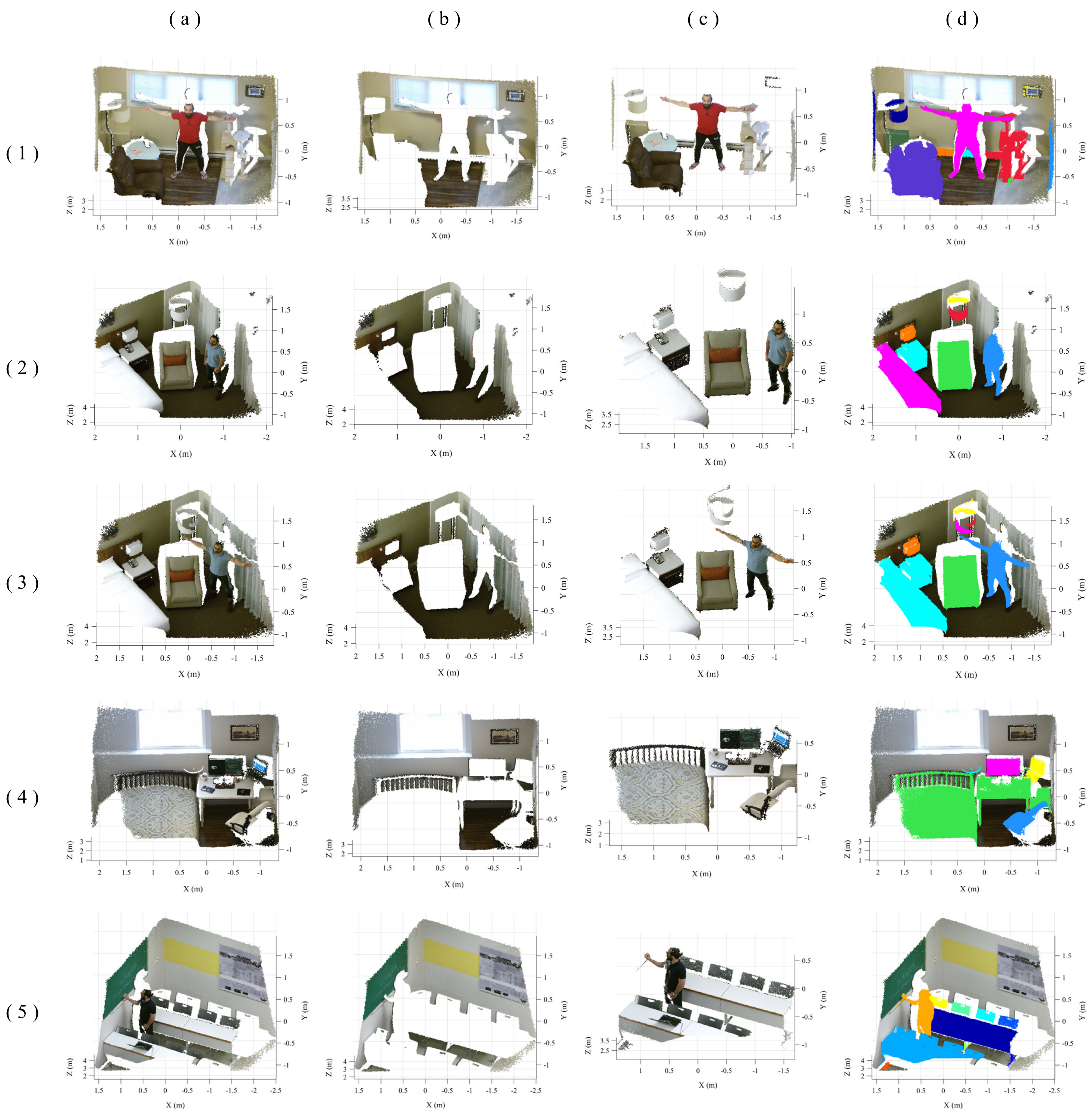


Figure 5.10 PiPCS output results on our dataset's point clouds captured using Microsoft Kinect v2: the first column (a) displays the input point clouds, the second (b) and third (c) columns show the 3D backgrounds and the simplified point clouds, respectively, and the fourth column (d) illustrates the 3D foreground objects highlighted with different colours over the input point clouds.

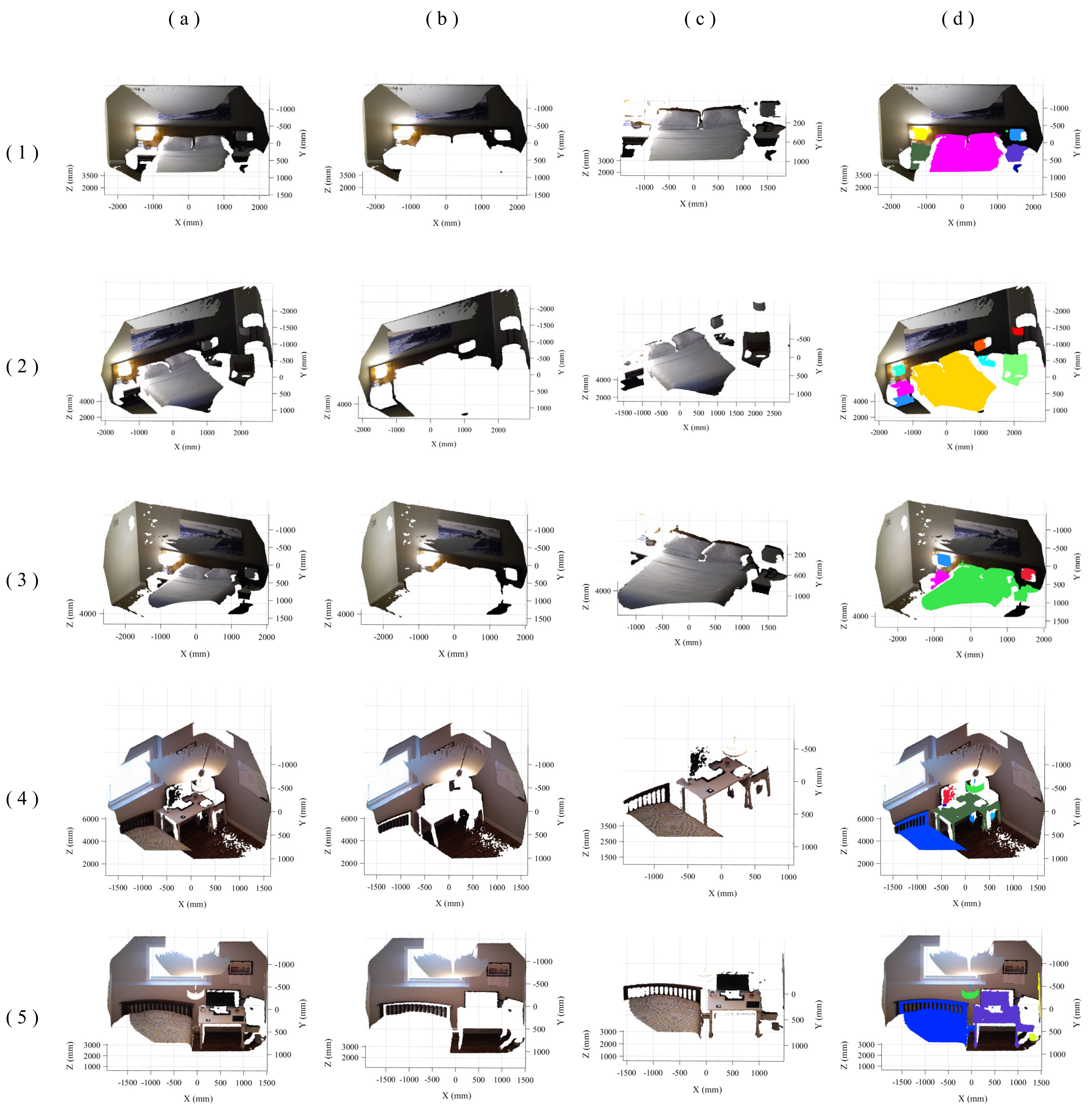


Figure 5.11 PiPCS output results on our dataset's point clouds captured using Microsoft Azure Kinect: the first column (a) displays the input point clouds, the second (b) and third (c) columns show the 3D backgrounds and the simplified point clouds, respectively, and the fourth column (d) illustrates the 3D foreground objects highlighted with different colours over the input point clouds.

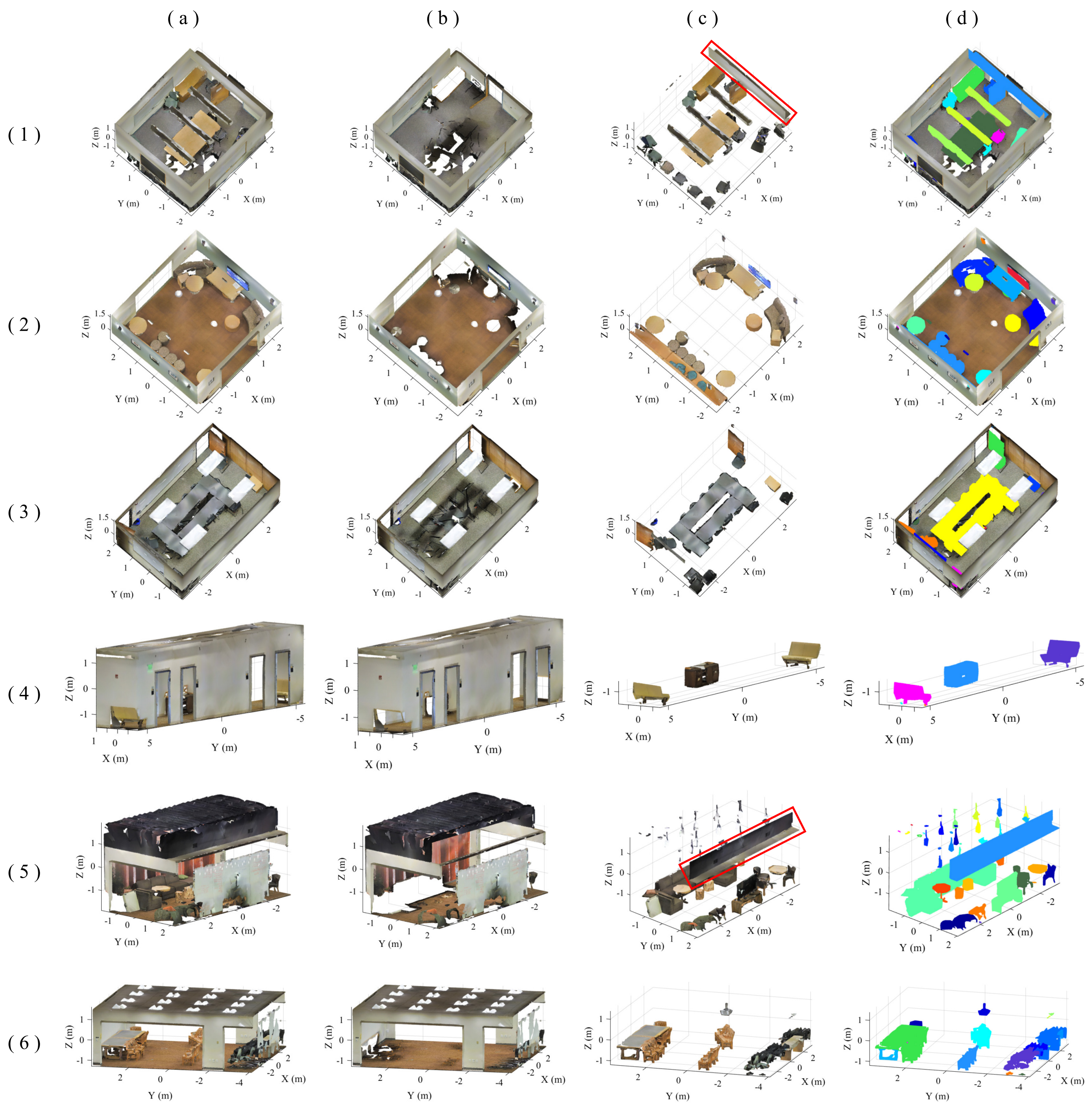


Figure 5.12 PiPCS output results on the S3DIS dataset's point clouds: the first column (a) displays the input point clouds, the second (b) and third (c) columns show the 3D backgrounds and the simplified point clouds, respectively, (misclassified L-shaped structures surrounded by red rectangles in column c), and the fourth column (d) illustrates the 3D foreground objects highlighted with different colours. Note, we excluded points belonging to the ceilings of the first three point clouds to visualize the foreground objects within them. Additionally, this allows us to highlight the segmented 3D foreground objects over these three point clouds in column d.

5.4.3 Evaluation

The size reduction results of PiPCS on our own dataset and the S3DIS dataset are listed in Table 5.1 and Table 5.2, respectively. We also included the original size of the point clouds alongside their corresponding simplified sizes, allowing for a comprehensive comparison of the reduction achieved by our method for both datasets. Furthermore, the execution times of the PiPCS algorithm for both datasets are presented in Tables 5.1 and 5.2.

Table 5.1 PiPCS evaluation results on our dataset's point clouds

Dataset	Figure indices	File name	Original size (Pts)	Simplified size (Pts)	Size reduction (%)	Runtime (s)
Kinect v2 (Fig. 5.10)	1	Bedroom1_001_v2	217088	55111	74.61	7.37
	2	Bedroom2_001_v2	217088	67210	69.04	7.79
	3	Bedroom2_004_v2	217088	65248	69.94	8.27
	4	Bedroom3_001_v2	217088	60737	72.02	7.54
	5	Classroom1_001_v2	217088	61536	71.65	6.52
Azure Kinect (Fig. 5.11)	1	Bedroom2_011_v4	556635	162800	70.75	13.38
	2	Bedroom2_012_v4	544473	214876	60.54	14.40
	3	Bedroom2_013_v4	522375	144318	72.37	12.87
	4	Bedroom3_015_v4	523011	88385	83.10	14.88
	5	Bedroom3_016_v4	526919	163986	68.88	16.45

Table 5.2 PiPCS evaluation results on the S3DIS dataset's point clouds

Area & file name	Original size (Pts)	Simplified size (Pts)	Size reduction (%)	Specificity (%)	Precision (%)	Recall (%)	F_1 (%)	Runtime (s)
Area_1 conferenceRoom_1	919573	262505	71.45	91.63	78.02	89.30	83.28	40.49
Area_3 lounge_1	792952	196484	75.22	99.89	99.69	90.86	95.07	31.05
Area_4 conferenceRoom_1	913148	195585	78.58	99.98	99.94	89.62	94.50	37.74
Area_5 hallway_4	1023041	55719	94.55	100	100	92.16	95.92	42.34
Area_6 lounge_1	1299627	343274	73.59	91.41	74.35	93.42	82.80	72.71
Area_6 openspace_1	1958651	403481	79.40	99.57	98.36	93.80	96.02	120.09

The misclassified points of the PiPCS algorithm for the six point clouds acquired from the S3DIS dataset are depicted in Fig. 5.13. The incorrectly classified foreground points (false positives), and the incorrectly classified background points (false negatives) are highlighted in cyan and magenta, respectively, over the original point clouds in Fig. 5.13. The evaluation results of the PiPCS algorithm, including specificity, precision, recall, and F_1 score, for the S3DIS point clouds are presented in Table 5.2.

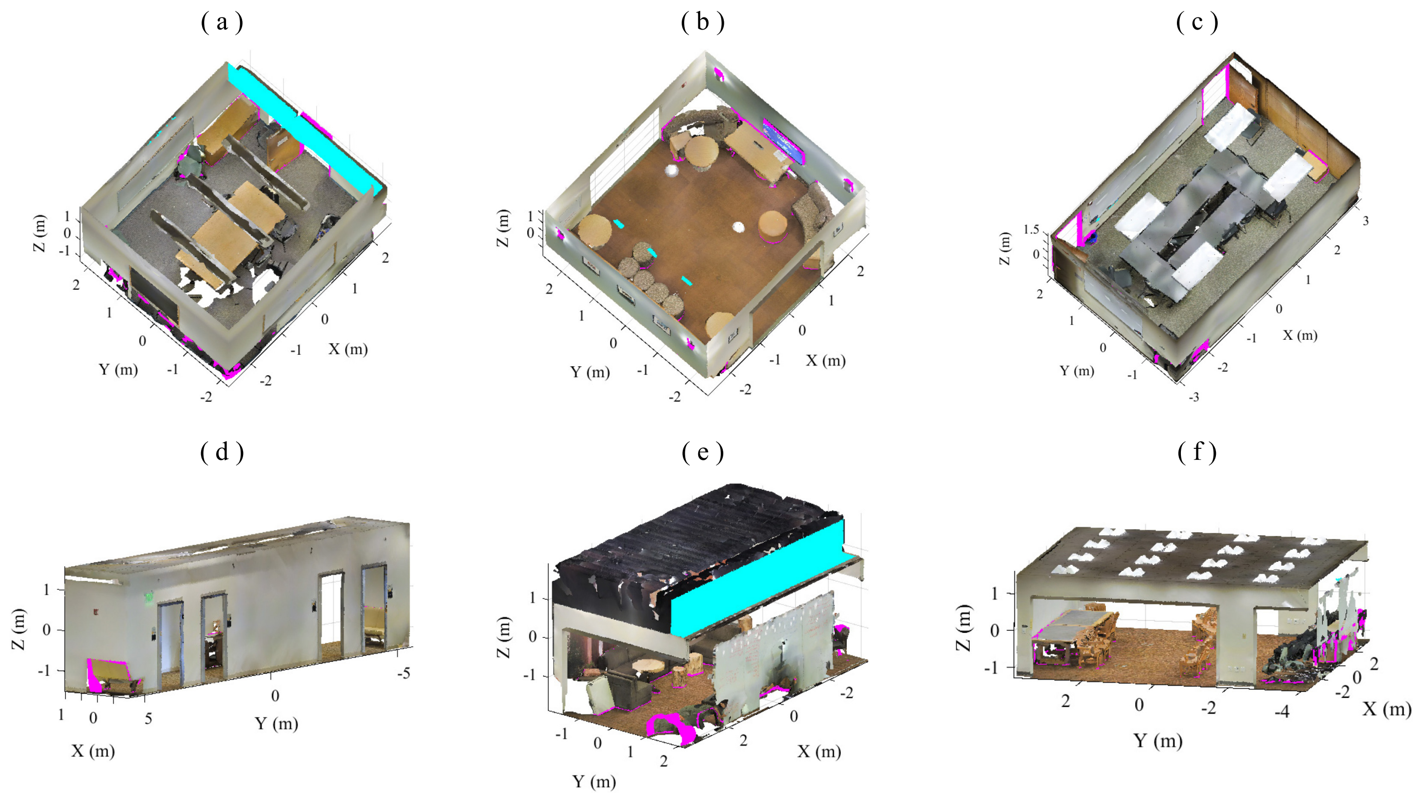


Figure 5.13 PiPCS misclassified points for the S3DIS dataset's point clouds: the false positive (cyan) and false negative (magenta) points are highlighted over the original point clouds, (a) Area_1 conferenceRoom_1, (b) Area_3 lounge_1, (c) Area_4 conferenceRoom_1, (d) Area_5 hallway_4, (e) Area_6 lounge_1, (f) Area_6 openspace_1.

Our evaluation of PiPCS focuses on demonstrating its effectiveness and efficiency in simplifying challenging indoor point clouds. Notably, PiPCS is currently the only point cloud simplifier specifically designed for indoor environments available in the literature, making comparisons with other methods in this category unfeasible. Consequently, PiPCS stands out for its unique contribution and innovative approach to indoor point cloud simplification.

5.5 Discussion

The incorrectly classified foreground points (the cyan-highlighted points in Fig. 5.13) are almost entirely associated with the L-shaped ceiling-wall structures of the two challenging S3DIS point clouds ('Area_1 conferenceRoom_1' and 'Area_6 lounge_1'). The reason for this misclassification is that the distance between the two surfaces of these L-shaped structures and their zone's outermost planes exceeds the maximum distance threshold (Δ), failing to satisfy the first of the two conditions required for verifying the successive sorted planes of each zone (see section 5.3.4.2, Seed Plane Detection). These two conditions serve as safeguards to prevent the verification of any planar foreground objects (e.g., cabinets or cupboards) adjacent to the outermost plane of a zone as the seed plane, which could otherwise lead to erroneous segmentation of any foreground objects under or behind these L-shaped structures as the 3D background of the point cloud (see section 5.3.4.3, 3D Background Segmentation).

The incorrectly classified background points (the magenta-highlighted points in Fig. 5.13) are generally associated with the points of foreground objects that touch the background surfaces (e.g., see the magenta-highlighted points around furniture, pictures, and sconces touching walls or the ground surfaces in Fig. 5.13). Note, the magenta-highlighted door frames in Fig. 5.13C are actually correctly classified as 3D background by PiPCS; however, they are labeled as part of doors in the S3DIS dataset (i.e., open doors and their frames are inevitably included in the ground truth or the 3D foreground of the point cloud).

In experiments on the S3DIS dataset, our proposed PiPCS method obtained average values of 97.08% for specificity, 91.73% for precision, 91.53% for recall, and 91.27% for F_1 score. In more detail, aside from the two point clouds with L-shaped ceiling-wall structures, PiPCS demonstrated remarkable results across all metrics for the point clouds derived from the S3DIS dataset, with specificity, precision, recall, and F_1 score surpassing 99.57%, 98.36%, 89.62%, and 94.50%, respectively.

Our PiPCS algorithm demonstrated significant size reductions across both datasets, as shown in Tables 5.1 and 5.2. It achieved an average size reduction of 73.13% and 69.82% for our dataset's point clouds captured using Microsoft Kinect v2 and Microsoft Azure Kinect, respectively. Moreover, PiPCS yielded an average size reduction of 75.43% for the S3DIS dataset's point clouds.

The efficiency of the PiPCS algorithm across point clouds of various sizes is evident from the runtimes presented in Tables 5.1 and 5.2. PiPCS resulted in an average runtimes of 6.95 and 14.92 seconds for processing the point clouds captured using Microsoft Kinect v2 and Microsoft Azure Kinect, respectively. Furthermore, the algorithm resulted in average runtime of 56.60 seconds for the S3DIS dataset's point clouds, excluding the very large 'Area_6 openspace_1' point cloud, which achieved a runtime of 120.09 seconds for processing nearly two million points. Given that the 3D background (i.e., 3D bounding surfaces, such as walls, ceiling, and floor) remains static in nearly all indoor scenes, and many 3D computer vision applications (e.g., human detection and tracking) only need to segment the 3D foreground objects in the first frame or periodically, PiPCS is well-suited for several real-time 3D computer vision applications. In other words, once the 3D background of an indoor scene is segmented, only the last two steps of PiPCS (i.e., 3D background subtraction and Cluster-based Segmentation), which are computationally inexpensive, need to be employed to accurately segment the 3D foreground objects.

Our experimental and evaluation results substantiate PiPCS as a robust and reliable technique for both simplifying and reducing the size of unorganized indoor point clouds. Furthermore, our results indicate that PiPCS supports varied sensor positions and orientations, accurately segmenting the 3D background and 3D foreground objects of complex 3D indoor scenes without relying on colour or historical information.

In future work, we will expand our dataset by incorporating a larger number of 3D point clouds from diverse indoor environments and label the 3D background elements and foreground objects of each point cloud using our PiPCS algorithm. Subsequently, we will publicly release this annotated dataset, facilitating advancements in 3D machine learning-based applications specifically designed for depth data, such as 3D object recognition and human detection within complex 3D indoor scenes.

5.6 Conclusions

In this paper, we proposed an innovative approach to simplify complex 3D indoor scenes by contextually segmenting and removing 3D background components while retaining segmented 3D foreground objects within indoor point clouds. Our point cloud simplifier, PiPCS, revolutionizes 3D indoor point cloud processing by offering a multifaceted solution. As a preprocessing step, it effectively narrows the search space for downstream processes by segmenting 3D background and foreground objects of indoor scenes, resulting in notable performance and accuracy improvements of these applications. Additionally, it can significantly reduce the size of 3D indoor point clouds by eliminating their 3D background elements, addressing computational and storage burdens, and optimizing data transmission processes. Furthermore, it enhances 3D indoor scene perception by accurately identifying each boundary of the 3D background and segmenting the 3D foreground objects within complex indoor scenes. This functionality also makes it a suitable tool for labeling or annotating indoor point clouds, streamlining the data preparation process for machine learning-based computer vision applications. Finally, PiPCS stands out for its key advantages: perspective independence, privacy enhancement, and universal sensor and data compatibility, making it an essential solution for a wide range of 3D point cloud processing tasks. Our experimental and evaluation results validate PiPCS as a robust and reliable method for both simplifying and reducing the size of unorganized indoor point clouds. Moreover, they demonstrate that PiPCS supports diverse sensor positions and orientations, accurately segmenting the 3D background and 3D foreground objects of complex indoor scenes without relying on the point cloud's 2D data or historical information.

5.7 References

[1] J.-M. Guo, C.-H. Hsia, Y.-F. Liu, M.-H. Shih, C.-H. Chang, and J.-Y. Wu, “Fast background subtraction based on a multilayer codebook model for moving object detection,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 10, pp. 1809–1821, 2013.

[2] L. Li, Z. Wang, Q. Hu, and Y. Dong, “Adaptive nonconvex sparsity based background subtraction for intelligent video surveillance,” *IEEE Trans Industr Inform*, vol. 17, no. 6, pp. 4168–4178, 2020.

[3] Z. Kuang, X. Tie, X. Wu, and L. Ying, “FUNet: Flow Based Conference Video Background Subtraction,” in *International Conference on Smart Multimedia*, Springer, 2022, pp. 18–28.

[4] A. Benlamoudi *et al.*, “Face Presentation Attack Detection Using Deep Background Subtraction,” *Sensors*, vol. 22, no. 10, p. 3760, 2022.

[5] S. Saravanakumar, A. Vadivel, and C. G. S. Ahmed, “Multiple human object tracking using background subtraction and shadow removal techniques,” in *2010 international conference on signal and image processing*, IEEE, 2010, pp. 79–84.

[6] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, “A novel video dataset for change detection benchmarking,” *IEEE Transactions on Image Processing*, vol. 23, no. 11, pp. 4663–4679, 2014.

[7] L. Maddalena and A. Petrosino, “Background subtraction for moving object detection in RGBD data: A survey,” *J Imaging*, vol. 4, no. 5, p. 71, 2018.

[8] StereoLabs, “ZED X Camera.” Accessed: Oct. 01, 2023. [Online]. Available: https://www.stereolabs.com/zed-x/

[9] Microsoft, “Azure Kinect.” Accessed: Mar. 05, 2023. [Online]. Available: https://azure.microsoft.com/en-us/products/kinect-dk

[10] Microsoft, “HoloLens.” Accessed: Apr. 11, 2024. [Online]. Available: https://www.microsoft.com/en-us/hololens

[11] Apple Inc., “Apple Vision Pro.” Accessed: Apr. 12, 2024. [Online]. Available: https://www.apple.com/apple-vision-pro/

[12] Apple Inc., “iPhone 15 Pro.” Accessed: Apr. 12, 2024. [Online]. Available: https://www.apple.com/ca/iphone-15-pro/

[13] G. Moya-Alcover, A. Elgammal, A. Jaume-i-Capó, and J. Varona, “Modeling depth for nonparametric foreground segmentation using RGBD devices,” *Pattern Recognit Lett*, vol. 96, pp. 76–85, 2017.

[14] Microsoft, “Azure Kinect DK depth camera.” Accessed: Oct. 05, 2023. [Online]. Available: https://learn.microsoft.com/en-us/azure/kinect-dk/depth-camera

[15] S. Chen, D. Tian, C. Feng, A. Vetro, and J. Kovačević, “Fast resampling of three-dimensional point clouds via graphs,” *IEEE Transactions on Signal Processing*, vol. 66, no. 3, pp. 666–681, 2018, doi: 10.1109/TSP.2017.2771730.

[16] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung, “Deep neural network concepts for background subtraction: A systematic review and comparative evaluation,” *Neural Networks*, vol. 117, pp. 8–66, 2019.

[17] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, “Pfinder: Real-time tracking of the human body,” *IEEE Trans Pattern Anal Mach Intell*, vol. 19, no. 7, pp. 780–785, 1997.

[18] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proceedings. 1999 IEEE computer society conference on computer vision and pattern recognition (Cat. No PR00149)*, IEEE, 1999, pp. 246–252.

[19] O. Barnich and M. Van Droogenbroeck, “ViBe: A universal background subtraction algorithm for video sequences,” *IEEE Transactions on Image processing*, vol. 20, no. 6, pp. 1709–1724, 2010.

[20] F. El Baf, T. Bouwmans, and B. Vachon, “Type-2 fuzzy mixture of Gaussians model: Application to background modeling,” in *International Symposium on Visual Computing*, Springer, 2008, pp. 772–781.

[21] F. El Baf, T. Bouwmans, and B. Vachon, “Fuzzy statistical modeling of dynamic backgrounds for moving object detection in infrared videos,” in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, 2009, pp. 60–65.

[22] M. M. Azab, H. A. Shedeed, and A. S. Hussein, “A new technique for background modeling and subtraction for motion detection in real-time videos,” in *2010 IEEE International Conference on Image Processing*, IEEE, 2010, pp. 3453–3456.

[23] N. Wiener, *Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications*. The MIT press, 1949.

[24] R. E. Kalman, “A new approach to linear filtering and prediction problems,” 1960.

[25] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, “Wallflower: Principles and practice of background maintenance,” in *Proceedings of the seventh IEEE international conference on computer vision*, IEEE, 1999, pp. 255–261.

[26] J. Zhong, “Segmenting foreground objects from a dynamic textured background via a robust kalman filter,” in *Proceedings ninth IEEE international conference on computer vision*, IEEE, 2003, pp. 44–50.

[27] H. Tezuka and T. Nishitani, “A precise and stable foreground segmentation using fine-to-coarse approach in transform domain,” in *2008 15th IEEE International Conference on Image Processing*, IEEE, 2008, pp. 2732–2735.

[28] W. Chen, C. He, C. Ji, M. Zhang, and S. Chen, “An improved K-means algorithm for underwater image background segmentation,” *Multimed Tools Appl*, vol. 80, pp. 21059–21083, 2021.

[29] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, “Real-time foreground–background segmentation using codebook model,” *Real-time imaging*, vol. 11, no. 3, pp. 172–185, 2005.

[30] J. He, L. Balzano, and A. Szlam, “Incremental gradient on the grassmannian for online foreground and background separation in subsampled video,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2012, pp. 1568–1575.

[31] P. Rodriguez and B. Wohlberg, “Incremental principal component pursuit for video background modeling,” *J Math Imaging Vis*, vol. 55, no. 1, pp. 1–18, 2016.

[32] B. Han and L. S. Davis, “Density-based multifeature background subtraction with support vector machine,” *IEEE Trans Pattern Anal Mach Intell*, vol. 34, no. 5, pp. 1017–1023, 2011.

[33] H.-H. Lin, T.-L. Liu, and J.-H. Chuang, “A probabilistic SVM approach for background scene initialization,” in *Proceedings. International conference on image processing*, IEEE, 2002, pp. 893–896.

[34] A. J. Schofield, P. A. Mehta, and T. J. Stonham, “A system for counting people in video images using neural networks to identify the background scene,” *Pattern Recognit*, vol. 29, no. 8, pp. 1421–1428, 1996.

[35] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung, “Deep neural network concepts for background subtraction: A systematic review and comparative evaluation,” *Neural Networks*, vol. 117, pp. 8–66, 2019.

[36] R. Kalsotra and S. Arora, “Background subtraction for moving object detection: explorations of recent developments and challenges,” *Vis Comput*, vol. 38, no. 12, pp. 4151–4178, 2022.

[37] L. Maddalena and A. Petrosino, “A self-organizing approach to background subtraction for visual surveillance applications,” *IEEE Transactions on image processing*, vol. 17, no. 7, pp. 1168–1177, 2008.

[38] L. Maddalena and A. Petrosino, “A fuzzy spatial coherence-based approach to background/foreground separation for moving object detection,” *Neural Comput Appl*, vol. 19, pp. 179–186, 2010.

[39] L. Maddalena and A. Petrosino, “The SOBS algorithm: What are the limits?,” in *2012 IEEE computer society conference on computer vision and pattern recognition workshops*, IEEE, 2012, pp. 21–26.

[40] G. Gemignani and A. Rozza, “A novel background subtraction approach based on multi layered self-organizing maps,” in *2015 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2015, pp. 462–466.

[41] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung, “Deep neural network concepts for background subtraction: A systematic review and comparative evaluation,” *Neural Networks*, vol. 117, pp. 8–66, 2019.

[42] M. Braham and M. Van Droogenbroeck, “Deep background subtraction with scene-specific convolutional neural networks,” in *2016 international conference on systems, signals and image processing (IWSSIP)*, IEEE, 2016, pp. 1–4.

[43] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[44] L. A. Lim and H. Y. Keles, “Foreground segmentation using convolutional neural networks for multiscale feature encoding,” *Pattern Recognit Lett*, vol. 112, pp. 256–262, 2018.

[45] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[46] L. A. Lim and H. Y. Keles, “Learning multi-scale features for foreground segmentation,” *Pattern Analysis and Applications*, vol. 23, no. 3, pp. 1369–1380, 2020.

[47] W. Zheng, K. Wang, and F.-Y. Wang, “A novel background subtraction algorithm based on parallel vision and Bayesian GANs,” *Neurocomputing*, vol. 394, pp. 178–200, 2020.

[48] W. Zheng, K. Wang, and F. Wang, “Background subtraction algorithm based on Bayesian generative adversarial networks,” *Acta Automatica Sinica*, vol. 44, no. 5, pp. 878–890, 2018.

[49] G. Rahmon, F. Bunyak, G. Seetharaman, and K. Palaniappan, "Motion U-Net: Multi-cue encoder-decoder network for motion segmentation," in *2020 25th international conference on pattern recognition (ICPR)*, IEEE, 2021, pp. 8125–8132.

[50] L. Maddalena and A. Petrosino, "Self-organizing background subtraction using color and depth data," *Multimed Tools Appl*, vol. 78, no. 9, pp. 11927–11948, 2019.

[51] C. Rougier, E. Auvinet, J. Rousseau, M. Mignotte, and J. Meunier, "Fall detection from depth map video sequences," in *Toward Useful Services for Elderly and People with Disabilities: 9th International Conference on Smart Homes and Health Telematics, ICOST 2011, Montreal, Canada, June 20-22, 2011. Proceedings 9*, Springer, 2011, pp. 121–128.

[52] J. Han, E. J. Pauwels, P. M. de Zeeuw, and P. H. N. de With, "Employing a RGB-D sensor for real-time tracking of humans across multiple re-entries in a smart environment," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 2, pp. 255–263, 2012.

[53] U. Mahbub, H. Imtiaz, T. Roy, M. S. Rahman, and M. A. R. Ahad, "A template matching approach of one-shot-learning gesture recognition," *Pattern Recognit Lett*, vol. 34, no. 15, pp. 1780–1788, 2013.

[54] L. Cinque, A. Danani, P. Dondi, and L. Lombardi, "Real-time foreground segmentation with Kinect sensor," in *Image Analysis and Processing—ICIAP 2015: 18th International Conference, Genoa, Italy, September 7-11, 2015, Proceedings, Part II 18*, Springer, 2015, pp. 56–65.

[55] S. Czarnuch and A. Mihailidis, "Development and evaluation of a hand tracker using depth images captured from an overhead perspective," *Disabil Rehabil Assist Technol*, vol. 11, no. 2, pp. 150–157, 2016.

[56] X. Zhang, X. Wang, and Y. Jia, "The visual internet of things system based on depth camera," in *Proceedings of 2013 Chinese Intelligent Automation Conference: Intelligent Automation & Intelligent Technology and Systems*, Springer, 2013, pp. 447–455.

[57] A. Frick, F. Kellner, B. Bartczak, and R. Koch, "Generation of 3d-tv ldv-content with time-of-flight camera," in *2009 3DTV conference: the true vision-capture, transmission and display of 3d video*, IEEE, 2009, pp. 1–4.

[58] A. Ebrahimi and S. Czarnuch, "Automatic super-surface removal in complex 3D indoor environments using iterative region-based RANSAC," *Sensors*, vol. 21, no. 11, p. 3724, 2021.

[59] M. A. Fischler and R. C. Bolles, "Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[60] G. Gordon, T. Darrell, M. Harville, and J. Woodfill, "Background estimation and removal based on range and color," in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, IEEE, 1999, pp. 459–464.

[61] A. Clapés, M. Reyes, and S. Escalera, "Multi-modal user identification and object recognition surveillance system," *Pattern Recognit Lett*, vol. 34, no. 7, pp. 799–808, 2013.

[62] M. Camplani and L. Salgado, "Background foreground segmentation with RGB-D Kinect data: An efficient combination of classifiers," *J Vis Commun Image Represent*, vol. 25, no. 1, pp. 122–136, 2014.

[63] J. Leens, S. Piérard, O. Barnich, M. Van Droogenbroeck, and J.-M. Wagner, "Combining color, depth, and motion for video segmentation," in *Computer Vision Systems: 7th International Conference on Computer Vision Systems, ICVS 2009 Liège, Belgium, October 13-15, 2009. Proceedings 7*, Springer, 2009, pp. 104–113.

[64] S. Ottonelli, P. Spagnolo, P. L. Mazzeo, and M. Leo, "Improved video segmentation with color and depth using a stereo camera," in *2013 IEEE international conference on industrial technology (ICIT)*, IEEE, 2013, pp. 1134–1139.

[65] X. Zhou, X. Liu, A. Jiang, B. Yan, and C. Yang, "Improving video segmentation by fusing depth cues and the visual background extractor (ViBe) algorithm," *Sensors*, vol. 17, no. 5, p. 1177, 2017.

[66] E. J. Fernandez-Sanchez, J. Diaz, and E. Ros, "Background subtraction based on color and depth using active sensors," *Sensors*, vol. 13, no. 7, pp. 8895–8915, 2013.

[67] J. Murgia, C. Meurie, and Y. Ruichek, "An improved colorimetric invariants and RGB-depth-based codebook model for background subtraction using kinect," in *Human-Inspired Computing and Its Applications: 13th Mexican International Conference on Artificial Intelligence, MICAI 2014, Tuxtla Gutiérrez, Mexico, November 16-22, 2014. Proceedings, Part I 13*, Springer, 2014, pp. 380–392.

[68] L. Maddalena and A. Petrosino, "Background subtraction for moving object detection in RGBD data: A survey," *J Imaging*, vol. 4, no. 5, p. 71, 2018.

[69] G. Moya-Alcover, A. Elgammal, A. Jaume-i-Capó, and J. Varona, "Modeling depth for nonparametric foreground segmentation using RGBD devices," *Pattern Recognit Lett*, vol. 96, pp. 76–85, 2017.

[70] R. Trabelsi, I. Jabri, F. Smach, and A. Bouallegue, "Efficient and fast multi-modal foreground-background segmentation using RGBD data," *Pattern Recognit Lett*, vol. 97, pp. 13–20, 2017.

[71] S. Javed, T. Bouwmans, M. Sultana, and S. K. Jung, "Moving object detection on RGB-D videos using graph regularized spatiotemporal RPCA," in *New Trends in Image Analysis and Processing–ICIAP 2017: ICIAP International Workshops, WBICV, SSPandBE, 3AS, RGBD, NIVAR, IWBAAS, and MADiMa 2017, Catania, Italy, September 11-15, 2017, Revised Selected Papers 19*, Springer, 2017, pp. 230–241.

[72] M. Camplani, L. Maddalena, G. Moyá Alcover, A. Petrosino, and L. Salgado, "A benchmarking framework for background subtraction in RGBD videos," in *New Trends in Image Analysis and Processing–ICIAP 2017: ICIAP International Workshops, WBICV, SSPandBE, 3AS, RGBD, NIVAR, IWBAAS, and MADiMa 2017, Catania, Italy, September 11-15, 2017, Revised Selected Papers 19*, Springer, 2017, pp. 219–229.

[73] L. Maddalena and A. Petrosino, "Exploiting color and depth for background subtraction," in *New Trends in Image Analysis and Processing–ICIAP 2017: ICIAP International Workshops, WBICV, SSPandBE, 3AS, RGBD, NIVAR, IWBAAS, and MADiMa 2017, Catania, Italy, September 11-15, 2017, Revised Selected Papers 19*, Springer, 2017, pp. 254–265.

[74] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," in *Proceedings of the 19th annual conference on computer graphics and interactive techniques*, 1992, pp. 71–78.

[75] R. B. Rusu, "Semantic 3D object maps for everyday manipulation in human living environments," *KI-Künstliche Intelligenz*, vol. 24, no. 4, pp. 345–348, 2010.

[76] J. MacQueen, "Classification and analysis of multivariate observations," in *5th Berkeley Symp. Math. Statist. Probability*, 1967, pp. 281–297.

[77] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics-theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[78] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans Pattern Anal Mach Intell*, no. 2, pp. 224–227, 1979.

[79] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *J Comput Appl Math*, vol. 20, pp. 53–65, 1987.

[80] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

[81] E. Herman, K.-E. Zsido, and V. Fenyves, “Cluster Analysis with K-Mean versus K-Medoid in Financial Performance Evaluation,” *Applied Sciences*, vol. 12, no. 16, p. 7985, 2022.

[82] L. Li, F. Yang, H. Zhu, D. Li, Y. Li, and L. Tang, “An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells,” *Remote Sens (Basel)*, vol. 9, no. 5, p. 433, 2017.

[83] P. H. S. Torr and A. Zisserman, “MLESAC: A new robust estimator with application to estimating image geometry,” *Computer vision and image understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[84] W. Förstner and B. P. Wrobel, *Photogrammetric computer vision*. Springer, 2016.

[85] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, “Towards 3D point cloud based object maps for household environments,” *Rob Auton Syst*, vol. 56, no. 11, pp. 927–941, 2008.

[86] © 2023 IEEE. Reprinted, with permission, from A. Ebrahimi and S. Czarnuch, “PiGPDS: Perspective Independent Ground Plane Detection and Segmentation for Complex 3D Indoor Scenes,” in *2023 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, IEEE, 2023, pp. 105–112.

[87] A. Ebrahimi and S. Czarnuch, “VISOR Lab 3D-Dataset.” Accessed: Jul. 14, 2023. [Online]. Available: https://github.com/visorlab/3D-Dataset

[88] I. Armeni *et al.*, “3d semantic parsing of large-scale indoor spaces,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1534–1543.

[89] Matterport, “Matterport Camera.” Accessed: Jul. 14, 2023. [Online]. Available: https://matterport.com/pro2

[90] A. Ebrahimi and S. Czarnuch, “Bounding Surface Segmentation and Removal in Complex 3D Indoor Environments Using Orientation-based MSAC,” in *the 30th Annual Newfoundland Electrical and Computer Engineering Conference (NECEC)*, 2021.

[91] Z. Dong, B. Yang, P. Hu, and S. Scherer, “An efficient global energy optimization approach for robust 3D plane segmentation of point clouds,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 137, pp. 112–133, 2018, doi: 10.1016/j.isprsjprs.2018.01.013.

[92] A. V. Vo, L. Truong-Hong, D. F. Laefer, and M. Bertolotto, “Octree-based region growing for point cloud segmentation,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 104, pp. 88–100, 2015, doi: 10.1016/j.isprsjprs.2015.01.011.

[93] J. Yan, J. Shan, and W. Jiang, “A global optimization approach to roof segmentation from airborne lidar point clouds,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 94, pp. 183–193, 2014, doi: 10.1016/j.isprsjprs.2014.04.022.

[94] M. Camplani, L. Maddalena, G. Moyá Alcover, A. Petrosino, and L. Salgado, “A benchmarking framework for background subtraction in RGBD videos,” in *New Trends in Image Analysis and Processing–ICIAP 2017: ICIAP International Workshops, WBICV, SSPandBE, 3AS, RGBD, NIVAR, IWBAAS, and MADiMa 2017, Catania, Italy, September 11-15, 2017, Revised Selected Papers 19*, Springer, 2017, pp. 219–229.

Chapter 6. Summary

This PhD thesis addresses the challenges faced by 3D computer vision systems in cluttered indoor environments through the implementation of perspective-independent point cloud processing techniques aimed to streamline 3D computer vision workflows and enhance 3D indoor scene perception.

Two bounding surface segmentation and removal techniques, IR-RANSAC [1] and orientation-based MSAC [2], were presented in this research. These methods are particularly suited to more challenging and cluttered indoor environments and support varied sensor perspectives. They considerably reduce the size of 3D datasets and the search space of various 3D computer vision applications, resulting in enhanced performance and faster processing times. In all experiments IR-RANSAC obtained average values above 92% for specificity, 96% for precision, 90% for recall, and 94% for F_1 score. In contrast, Orientation-based MSAC slightly outperformed IR-RANSAC across all evaluation metrics, showcasing average values exceeding 93% for specificity, 97% for precision, 97% for recall, and 98% for the F_1 score. Additionally, it demonstrates significantly faster computational speed than IR-RANSAC.

Furthermore, PiGPDS [3], a perspective-independent ground plane detection and segmentation method, was introduced for complex 3D indoor scenes, where the position and orientation of the sensor are unrestricted and unknown. PiGPDS demonstrated excellent performance in terms of four evaluation metrics: specificity, precision, recall, and F_1 score, with average experimental results of 98.28%, 95.48%, 96.64%, and 96.01%, respectively. The evaluation results demonstrate that PiGPDS is robust for perspective-independent ground plane detection, accurately segmenting ground surfaces of complex 3D indoor scenes acquired from different locations with varying pitches and yaws.

Finally, PiPCS [4], a Perspective-Independent Point Cloud Simplifier, was introduced as a significant advancement, building upon the foundational research laid out in earlier studies [1], [2], and [3]. PiPCS expands the conventional concept of background subtraction to encompass the identification, segmentation, and removal of 3D background bounding surfaces, such as walls, windows, curtains, ceilings, and floors, from indoor point clouds. Unlike conventional background subtraction techniques, PiPCS contextually segments and eliminates 3D background components, yielding precisely segmented 3D foreground objects without relying on colour or historical data. PiPCS demonstrated remarkable results across all metrics, with specificity, precision, recall, and F_1 score yielding average experimental results of 97.08%, 91.73%, 91.53%, and 91.27%, respectively. Moreover, PiPCS showcased significant size reductions, achieving an average reduction of 74.11% across all dataset point clouds. PiPCS optimizes 3D computer vision systems by streamlining their workflows, enhancing indoor scene perception, reducing point cloud size, and enabling precise labeling within complex indoor environments.

In future work, we will extend our dataset to include a larger number of 3D point clouds from diverse indoor environments. This expanded dataset will be meticulously labeled using our PiPCS and PiGPDS algorithms to annotate both background elements and foreground objects within each point cloud. Ultimately, our aim is to publicly release this annotated dataset, not only facilitating the development and evaluation of 3D machine learning-based computer vision applications and contributing to the academic community but also advancing our own research initiatives, including 3D object recognition and human activity recognition within complex indoor environments.

6.1 Resulting Publications

[1] A. Ebrahimi and S. Czarnuch, “Automatic super-surface removal in complex 3D indoor environments using iterative region-based RANSAC,” Sensors, vol. 21, no. 11, p. 3724, 2021.

[2] A. Ebrahimi and S. Czarnuch, “Bounding Surface Segmentation and Removal in Complex 3D Indoor Environments Using Orientation-based MSAC,” in the 30th Annual Newfoundland Electrical and Computer Engineering Conference (NECEC), 2021.

[3] © 2023 IEEE. Reprinted, with permission, from A. Ebrahimi and S. Czarnuch, “PiGPDS: Perspective Independent Ground Plane Detection and Segmentation for Complex 3D Indoor Scenes,” in 2023 International Conference on Digital Image Computing: Techniques and Applications (DICTA), IEEE, 2023, pp. 105–112.

[4] A. Ebrahimi and S. Czarnuch, “PiPCS: Perspective Independent Point Cloud Simplifier for Complex 3D Indoor Scenes,” in IEEE Access, doi: 10.1109/ACCESS.2024.3452633, 2024.