Three-Dimensional Reconstruction and Modeling Using Low-Precision Vision Sensors for Automation and Robotics Applications in Construction

by

Yong Xiao

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Civil Engineering)
in the University of Michigan
2017

Doctoral Committee:

Professor Vineet R. Kamat, Chair
Assistant Professor Jia Deng
Associate Professor SangHyun Lee
Associate Professor Carol C. Menassa

Yong Xiao

yongxiao@umich.edu

ORCID iD: 0000-0002-2729-0795

DEDICATION

To My Lovely Parents,

Jinshui Xiao and Xiurong Xu.

To My Brother and Sister,

Yan and Lei.

ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to my advisor Professor Vineet R. Kamat for the continuous support of my Ph.D. study and research. He guided me with his patience, motivation, enthusiasm, and immense knowledge. When I struggled in conducting research or writing papers, I always received constructive guidance and constant support from Professor Kamat, which helped me to polish my skills in public presentations and scientific writings, and moreover, to complete my research projects and my dissertation. In addition, he has also been a great mentor in my life and has shared many helpful experiences and useful suggestions for my life and career choices which helped me to enjoy my Ph.D. life at the University of Michigan (UM).

I would like to gratefully and sincerely thank Professor Carol C. Menassa, who provided significant and valuable advice innumerable times for some of my research projects. Her insightful opinions allowed me to find solutions to tough problems. Moreover, she is so kind and patient to provide students including me a variety of significant comments and beneficial advice on scientific presentation and communication skills which helps me to better organize and present my research projects.

I would like to equally thank Professor SangHyun Lee who provided countless constructive and critical comments on some of my research projects. During the preliminary and qualification exams, his comments were highly helpful for me to better organize the research

background and motivation. He also contributed significant advice and instructions on the excavation slope stability monitoring project.

I would like to express my sincere thanks to Professor Jia Deng for the invaluable advice, insightful comments, and constructive suggestions for my research projects which assisted me to find the correct research directions and methods and thus accomplish my dissertation.

I am also indebted to Dr. Yuichi Taguchi from Mitsubishi Electric Research Laboratories (MERL) who collaborated with me for three years and supervised me for the internship during the summer of 2015. During my collaboration with him, I was deeply impressed by his enthusiasm, patience, and meticulousness for the research projects and the papers. When I was writing the two papers on dimensional analysis and point cloud completion, he patiently reviewed the manuscripts several times and gave many invaluable and critical comments. In addition, when I had practical challenges on the projects, he was always willing to scrutinize the technical details and was able to provide constructive suggestions to solve the problems.

I would also like to thank Professor Jerome Lynch and Professor Dimitrios Zekkos who provided a lot of constructive advice on the landslide detection project.

I am also deeply grateful to Dr. Suyang Dong and Dr. Chen Feng, who are my mentors both in academia and life. They helped me to adapt to the Ph.D. life at UM and gave significant advice about which courses were appropriate, how to conduct research projects, and how to collaborate with other researchers. Moreover, as good friends, they shared their Ph.D. and life experiences including doing research, keeping work-life balance, and writing scientific papers with me.

I would also like to thank my colleagues, Dr. Albert Thomas, Kurt M. Lundeen, Bharadwaj Mantha, Da Li, Zhiyuan Zuo and Lichao Xu. Their assistance and thoughtful

suggestions were essential for me to accomplish the research projects and complete this dissertation. Especially, I deeply appreciate Dr. Thomas' patience and kindness for providing me invaluable help and advice about dissertation writing and graduation preparation.

I would also like to express my gratitude to Meiyin Liu who gave me a lot of constructive advice on research and support in my personal life. I am also grateful to Professor Joon Oh Seo who gave me many useful suggestions when he was at UM. I am also honored to collaborate with William Greenwood and Mitsuhito Hirose on the landslide detection project and the UAV landslide monitoring project.

I deeply appreciate Dr. Dong and Kurt for their assistance in collecting the data and conducting the experiments for various projects. Their contribution enabled me to successfully complete the projects. I would also like to thank Mr. Christopher D. Klaft from the Walsh Construction Company for his assistance in obtaining the drone video imagery as well as ground-truth measurements for the excavation stability monitoring project. I would also like to thank Kevin Roback and Professor Marin K. Clark for providing the landslide satellite images and the labels for landslides.

Last but not least, I would express my deepest gratitude to my parents Jinshui Xiao and Xiurong Xu, my sister Yan Xiao and my brother Lei Xiao, without whom none of this would have happened. Their endless love and unconditional support enabled me to travel to another country, study without worries, and pursue my Ph.D. degree.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

ABSTRACT

Automation and robotics in construction (ARC) has the potential to assist in the performance of several mundane, repetitive, or dangerous construction tasks autonomously or under the supervision of human workers, and perform effective site and resource monitoring to stimulate productivity growth and facilitate safety management. When using ARC technologies, three-dimensional (3D) reconstruction is a primary requirement for perceiving and modeling the environment to generate 3D workplace models for various applications. Previous work in ARC has predominantly utilized 3D data captured from high-fidelity and expensive laser scanners for data collection and processing while paying little attention of 3D reconstruction and modeling using low-precision vision sensors, particularly for indoor ARC applications.

This dissertation explores 3D reconstruction and modeling for ARC applications using low-precision vision sensors for both outdoor and indoor applications. First, to handle occlusion for cluttered environments, a joint point cloud completion and surface relation inference framework using red-green-blue and depth (RGB-D) sensors (e.g., Microsoft® Kinect) is proposed to obtain complete 3D models and the surface relations. Then, to explore the integration of prior domain knowledge, a user-guided dimensional analysis method using RGB-D sensors is designed to interactively obtain dimensional information for indoor building environments. In order to allow deployed ARC systems to be aware of or monitor humans in the environment, a real-time human tracking method using a single RGB-D sensor is designed to track specific individuals under various illumination conditions in work environments. Finally,

this research also investigates the utilization of aerially collected video images for modeling ongoing excavations and automated geotechnical hazards detection and monitoring.

The efficacy of the researched methods has been evaluated and validated through several experiments. Specifically, the joint point cloud completion and surface relation inference method is demonstrated to be able to recover all surface connectivity relations, double the point cloud size by adding points of which more than 87% are correct, and thus create high-quality complete 3D models of the work environment. The user-guided dimensional analysis method can provide legitimate user guidance for obtaining dimensions of interest. The average relative errors for the example scenes are less than 7% while the absolute errors less than 36mm. The designed human worker tracking method can successfully track a specific individual in real-time with high detection accuracy. The excavation slope stability monitoring framework allows convenient data collection and efficient data processing for real-time job site monitoring. The designed geotechnical hazard detection and mapping methods enable automated identification of landslides using only aerial video images collected using drones.

Chapter 1

Introduction

1.1 Background

As the second largest construction market worldwide, the value of the United States (U.S.) construction industry was 4% of the Gross Domestic Product (GDP) in 2015, which also was the average value from 2007 to 2015 (Statista 2017). According to various research studies, it is found that there is a decline in construction productivity in the U.S. (Teicholz 2014). Even though Sveikauskas et al. (2016) claimed that in fact there exists a productivity growth by exploring more comprehensive measurements of labor productivity, the productivity growth rate is still low. For example, during 2002-2014 the average rate of the productivity growth in highway construction was 3.2% as reported in that research. The controversy over the construction productivity reflects the fact that the productivity growth remains stagnant or grows very slowly compared to non-farm industries whose productivity has increased by over 200 percent in the last 40 years (from the 1960s to 2000s) (Ennova 2014).

As it is widely agreed that integration of automation technologies contributed to the growth of productivity in the U.S. manufacturing industry (Brynjolfsson and Hitt 1996; Siegel 1997), it has been generally expected that automation technologies can similarly boost productivity in the construction industry (Zhai et al. 2009). Apart from the low productivity, the construction industry faces many safety problems during to co-existence of large machines and humans, harsh

work environment, heavy laboring tasks and so on. For example, one in five worker deaths in the calendar year 2015 were in construction (OSHA 2017). Employing automation techniques and robots in construction has the potential to make workers safer and reduce hazards, while also increasing productivity and benefitting the whole construction industry (Bernold 1987; Son et al. 2010).

As the development of research on computer vision, machine learning and robotics has occurred during the last two decades, the construction community has been gradually utilizing new automation technologies and robotic platforms. For example, computer vision techniques are widely employed to monitor conditions of infrastructure for existing civil infrastructure, e.g. bridge inspection (Adhikari et al. 2014; Zhu et al. 2010), tunnels inspection (Victores et al. 2011; Yu et al. 2007) and road defect detection (Jahanshahi et al. 2012; Koch et al. 2015), and to evaluate safety of workers so as to improve productivity and reduce potential hazards (Han and Lee 2013). Computer vision techniques can also be utilized for automated productivity analysis(Gong and Caldas 2010; Gong and Caldas 2011), automated performance monitoring (Golparvar-Fard et al. 2011; Yang et al. 2015), progress monitoring(Braun et al. 2015), materials and resources tracking (Park et al. 2012; Su and Liu 2012; Turkan and Bosch 2013), and so forth.

An important characteristic of the construction industry is the strong necessity for accurate construction site modeling or civil infrastructure surveying so as to obtain three-dimensional (3D) models, dimension measurements, and so forth. To meet this requirement, building information models (BIMs) (Azhar 2011) which contain rich 3D geometric models are employed in construction at different project phases, e.g. design, construction, and maintenance.

In the design phase, BIMs allow to integrate multiple disciplines including design and documentation and thus facilitate communications between different entities and better decision

(Yan and Damian 2008). During the construction phase, BIMs of the actual construction tasks as well as working sites can evaluate the project progress (Han and Golparvar-Fard 2015; Kim et al. 2013) as well as the safety of construction workers (Chi et al. 2012). Last but not least, researchers have also been exploring the applications of robotics in the construction industry to improve productivity and automation. For example, a bricklaying robot SAM100 (Construction Robotics 2017) has been developed and commercialized for onsite masonry construction. With the rapid development of robotics technologies, it is expected that robots integrated with various automation techniques will be widely used in the construction industry (Khemlani 2017).

In order to perform designated tasks, a robot needs to be able to capture the current environment, identify the present objects, and complete designed tasks (e.g., picking up a concrete slab, laying bricks, and so forth). In addition, for construction robotics, the robot should be capable of obtaining 3D models and dimension measurements of the construction site for many construction applications other than general robotics tasks (e.g., navigating in a cluttered environment). For example, for a mobile robot to lay bricks on a wall, it has to capture the 3D models of the current masonry work and place the next brick in the correct position and orientation. Otherwise, the walls might not meet the construction specifications. In addition, if a robot is designed to obtain 3D models for an excavation project for progress monitoring and safety analysis, it requires 3D perception to capture the uneven surface of the construction site and obtain sufficient data for creating the 3D models. Therefore, 3D data can significantly enhance the ability of construction robots to perform construction tasks automatically.

The characteristics of the construction industry challenge the use of robotics and also 3D data in construction projects. Firstly, the construction environment is usually cluttered with various construction materials, equipment, and human workers, which poses great challenges for

3D modeling, and general object detection and recognition. Secondly, the surrounding environment and the construction work vary according to the progress and the schedule of the project. The 3D data capturing and analysis should be performed frequently so as to be applied in real construction projects. Thirdly, precise 3D models or measurements of principal objects are crucial for many construction tasks. Therefore, for the 3D data and models, the accuracy should be carefully explored so as to meet the requirements of real construction projects.

Currently, the most common method to utilize 3D data in construction is to utilize a terrestrial laser scanner, which is able to obtain data with very high accuracy (e.g., the 3D position accuracy of Leica ScanStation P30/P40 is 3mm at 50m and 6mm at 100m (Leica Geosystem 2017)). However, such a laser scanner is very expensive, and the data collection as well as the data processing and modeling are also time-consuming. In addition, it is often impossible for a robot to carry such laser scanners to collect data. To meet the need for fast data acquisition and robotic platform, low-precision sensors, e.g. RGB-D sensors, stereo cameras, and normal RGB cameras, offer significant promise for construction applications. Even though low-precision sensors might not be able to capture data with a high accuracy comparable to a terrestrial laser scanner, it still can obtain data and models that meet the requirements of some automation and robotics applications in construction.

In addition, low-precision sensors allow the potential of being integrated into robotic platforms. Thus, with the robotic platform, they enable fast and comprehensive data acquisition which can to some extent mitigate problems caused by the cluttered environment. However, most previous research on automation in construction (including robotics) relies on accurate 3D data source from expensive equipment (terrestrial laser scanners) and rarely utilizes affordable sensors for 3D reconstruction and modeling. Even though there exist some methods utilizing

close range photogrammetry with RGB cameras to perform 3D reconstruction and modeling (Golparvar-Fard et al. 2011; Nassar et al. 2011), these studies focus more on 3D modeling for outdoor construction sites while neglecting indoor environments which need to be modeled for facility management as well as many robotics applications.

In summary, automation and robotics technologies are expected to improve productivity and safety for the construction industry, and 3D data and geometric modeling can play a significant role for construction automation and robotics. In order to enable and facilitate the usage of robotics in construction, 3D perception from low-precision sensors is crucial as it can capture the data fast while achieving acceptable accuracy for several construction tasks. In addition, there is a need to explore 3D perception and modeling from low-precision sensors for construction applications and projects in both indoor and outdoor environments, due to their potential benefit in the facility management phases of constructed facilities.

1.2 Literature Review

Due to the limitation of existing robotic platforms and unique characteristics of construction work (for example, the construction site is usually cluttered, unstructured, and has both static and moving human workers and equipment), robots specifically designed for the construction industry are not so prevalent although some construction robotic systems are developed (Construction Robotics 2017). However, as aforementioned, many researchers have been exploring to adopt newly developed automation techniques to improve automation for construction as well as promote utilization of robots in construction. Thus, this section will review the construction automation research which is also relevant to robotics as well as previous work on construction robotics. This section also investigates previous research on 3D modeling

and reconstruction in construction so as to discuss the challenges of utilizing 3D data from low-precision sensors in construction automation and robotics.

1.2.1 Construction Automation and Robotics

Starting from the 1970s, construction robots have been developed for prefabrication of modular homes, tasks in construction sites, maintenance and inspection (Bock 2007). Depending on the functions, Ruggiero et al. (2016) categorized construction robotics into several classes (e.g., demolition robots, bricklaying robots, exoskeletons, and forklift robots), and discussed their advantages and limitations. To promote efficient construction robotics in the industry, construction robots should not be constrained or designed for only one task or a few tasks. Otherwise, the high cost and low rate of return on investment might hinder usages of these robots.

Thus, this chapter utilizes the categories proposed by Son et al. (2010) to review related papers on construction automation and robotics technology. By generally following the key phases of construction projects, these categories are: (1) planning and design, (2) construction robotics (during the actual direct construction work), (3) intelligent job-site management, (4) operation and maintenance, and (5) others, which either combines some of the previous categories or belong to none of the four (Son et al. 2010). Instead of generally reviewing all the literature on these categories as done in previous work (Bock 2007; Son et al. 2010; Yamazaki 2004), this section will focus on reviewing and discussing previous work related to 3D data and reconstruction for the first four categories while ignoring the last one.

1.2.1.1 The Planning and Design Phase

The planning and design phase mainly involves the architects and engineers capturing the facility design in 2D blueprints and 3D models (BIMs), and interacting with contractors and other stakeholders to improve the designs and create plans for the construction phases. Therefore, the automation techniques for this phase aim at improving the design processes (e.g., parametric design) and facilitating communication between different entities, and thus concentrate on developing software products to improve productivity and communication. In this context, BIMs which can represent detailed 3D models and related properties allow designers to create appropriate models, present accurate and realistic visualization for contractors and users, and aid the contractors to create or update schedules quickly (Bryde et al. 2013; Jung and Joo 2011).

1.2.1.2 Construction Robotics

In the categories above, construction robotics refers to the robots that are designed to perform tasks directly related to construction projects (e.g., laying bricks, and performing excavation). Even though in the late 1990s several bricklaying robots were developed (Balaguer et al. 1996; Heintze et al. 1996; Pritschow et al. 1996), few of them were commercialized due to the high cost of the robotics system, sophistication of the robot control system, and necessity of special parts (e.g., bricks and blocks) (Balaguer 2000). With the development of computer science technology and also easy access to affordable and powerful sensors, research about automation and robotics in construction began to progress rapidly in the last two decades. New robotic platforms are able to incorporate various sensors and thus obtain useful data which improve the functionality, automation, and feasibility of the robotic systems.

A semi-automated mason robot, SAM100 (Construction Robotics 2017) has been developed and commercialized for onsite bricklaying tasks. The SAM100 places bricks

according to previously laid bricks and the laser is utilized to line up the bricks for precise placement and to ensure the quality of the walls. This robot can collaborate with the mason and increase masons' productivity by 3 to 5 times with consistent production and lower installation cost. However, this robot is not fully automated and needs to be monitored by humans, and the bricks have to be restocked by human workers. Feng et al. (2015) attached fiducial markers to bricks and commanded a robotic arm to pick up bricks automatically using cameras. The experimental results demonstrated the system can correctly identify the bricks if they are in the view of the camera and build user-designed shapes. They also explored the utilization of an RGB-D camera to capture 3D data and thus create 3D models of the construction site. However, this work mainly utilized the camera to capture images and localize bricks and did not fully integrate the 3D data in the bricklaying tasks.

In terms of excavation robots, similar to bricklaying robots, several robotic systems (Bernold 1993; Bradley and Seward 1998; Lever and Wang 1995; Salcudean et al. 1998) were developed in 1990s based on the conventional industrial robots equipped with a bucket and have less sensors integration and thus a low level of automation (Ha et al. 2002). Komatsu developed the world's first intelligent machine control excavator PC210LCi-10 (Komatsu 2015). The system is equipped with stroke sensing hydraulic cylinders, an IMU sensor, and GNSS antennas, and can semi-automatically perform the excavation tasks. However, the system does not have any perception sensors for capturing the environment. Apart from performing the basic excavation operation, a robotic excavator with a high level of automation should be able to perceive and model the current environment including the terrain, and then make decisions accordingly to complete the tasks. Therefore, 3D data and (real-time) modeling is necessary for modeling the work environment (Chae et al. 2011; Kim 2013) for robotic excavators.

Chae et al. (2012) presented a new method for real-time earth surveying using 3D laser scanners installed on a mobile platform. To automatically register different scans, several sphere targets were placed at arbitrary points on the site so as to calculate the transformation matrix between two scans. The system can obtain 3D models for an 80x80m earthwork site in about 130 minutes. Yoo and Kim (2016) developed a 3D local terrain modeling system using a 2D laser scanner to model the terrain for an intelligent excavator robot. They attempted to find the optimum location for installing the sensors to minimize blind spots and obtain front earthwork terrain models. Experimental results for an actual earthwork site show that the system can achieve excellent accuracy even with vibration from the excavator.

1.2.1.3 Intelligent Job-site Management

As aforementioned, safety and productivity are two primary issues in construction. Construction robotics and automation thus aim to reduce risks related to construction workers' life and health, and to improve productivity for construction activities. For construction safety management, various approaches have been developed and investigated by adopting mobile devices, 3D sensors, cameras or other sensors to capture data of construction workers' behavior patterns in work and data of construction work environment (Yang et al. 2015). These applications take advantage of perceptual sensors to acquire data at high frequency, and thus are able to generate information about safety issues quickly, which allows safety management to be made in time and efficiently. Depending on the construction characteristics, these methods aim to capture data on the construction site to detect and track construction entities (e.g., construction materials, working machines, and human workers) for safety management and progress monitoring. As this dissertation is related to 3D geometric modeling, previous work using 3D

point clouds is discussed. For the methods using 2D computer vision methods, the reader is referred to relevant work described in (Seo et al. 2015; Yang et al. 2015).

As noted earlier, terrestrial laser scanners can provide accurate 3D point clouds for a large area and thus are utilized to capture data for accurate modeling and tracking. Turkan et al. (2013) utilized terrestrial laser scanners to track secondary and temporary concrete construction objects (e.g., formwork, scaffolding and shoring). After obtaining multiple scans of the construction site, they registered the scans with the 3D building model using a robust iterative closest point (ICP) method using the point-to-plane framework (Rusinkiewicz and Levoy 2001). Then the secondary and temporary objects were recognized using a surface based recognition metric (Bosché 2010). Wang and Cho (2015) designed a smart scanning system to rapidly identify target objects and update the target's point clouds to aid the heavy equipment operation in rapidly perceiving 3D working environment at dynamic construction sites. A smart scanning method was developed to only capture data for a specified target object's point cloud data while the object model was reconstructed using the concave hull modeling. Han et al. (2015) presented a new appearance-based material classification method to monitor operation-level construction progress using 4D BIM and site photos. The images were utilized to reconstruct 3D models of the build, and object detection and recognition using a supervised classification framework. The method achieved detection accuracy of 92.4% on four real-world construction sites while obtained 3D BIMs at different time stages for progress monitoring.

Instead of expensive terrestrial laser scanners, 3D data obtained from low-precision vision sensors (e.g., RGB-D sensors, and stereo vision cameras) can also be utilized to detect, model and track construction entities on the construction site. Teizer et al. (2007) presented a real-time 3D modeling method to rapidly detect, model, and track static and moving obstacles by

utilizing a Flash LADAR. They utilized the 3D data to create and update an occupancy grid which is employed to detect and track objects. Park et al. (2012) employed stereo cameras to obtain 3D data for tracking construction resources using an on-site camera system. Each camera of the stereo camera system was utilized to perform 2D tracking using images while triangulation was conducted to obtain 3D coordinates of the entities. The method was proved to be able to effectively track a steel plate, a van, and a worker. Han and Lee (2013) utilized a motion capture and a 3D camcorder to extract 3D human skeleton in order to identify critical unsafe behaviors and actions for certain construction tasks.

1.2.1.4 Operation and Maintenance

Once the construction activities are completed, the condition of civil infrastructure and constructed facilities has to be monitored regularly to make sure that it is functioning well. Therefore, many researchers have focused on detecting and analyzing defects for civil infrastructure monitoring by collecting a series of images to detect and model defects.

Yu et al. (2007) utilized image processing techniques to detect cracks in a tunnel from a Charge-Coupled Device (CCD) camera installed on a mobile robot. Medina et al. (2010) presented an automated inspection system for road cracks. They detected and classified cracks by combining traditional image features and Gabor filters. Abdel-Qader et al. (2003) compared four image-based crack detection methods using concrete bridge images. These methods detected cracks by finding edges in images. (Barazzetti and Scaioni 2009) presented a method of processing a sequence of images to detect cracks and compute the width across the longitudinal profile in pixels.

Apart from detecting cracks, the dimensions of cracks are also of significance. To get the spatial dimensions of cracks, 3D sensing systems (e.g., stereo cameras, and laser scanners), are

utilized to get 3D point clouds of the scene. Adhikari et al. (2014a) used an image-based method to obtain crack length and width and change detection for bridge inspection. Tung et al. (2002) developed a mobile imaging system for bridge crack inspection. Two CCD cameras were installed on the platform and they compared the two images from the cameras to detect the crack and its position. Hampel and Maas (2009) utilized cascaded image analysis for dynamic crack detection from stereo images. By detecting the discontinuities in dense surface deformation vector fields, their method is able to identify cracks and obtain the dimension information.

Jahanshahi et al. (2013) developed another image-based crack detection and qualification method. Firstly, they collected a set of images with overlapping features and then performed Structure from Motion (SFM) analysis to get 3D point clouds of the scene. Finally, by utilizing segmentation, feature extraction, and classification, the cracks and their dimensions were detected from the point clouds. This method was tested on concrete cracks. Jahanshahi et al. (2012) used an RGB-D sensor to get 3D point clouds of pavements and automatically detect and measure the cracks. Torok et al. (2013) used a robotic platform to gather a set of images and then performed 3D reconstruction. They proposed a new automated method for detecting cracks from 3D meshes which are independent of the data sources (from images or laser scanner data).

1.2.2 3D Reconstruction and Geometric Modeling

Regarding 3D geometric modeling for construction automation and robotics, the primary research topics are related to reconstructing 3D models for existing buildings (i.e., as-built BIMs), buildings under reconstruction, construction sites, and so forth. Since as-built BIMs can be utilized for various applications and generating as-built BIMs requires 3D geometric modeling including estimating the model topological relations, this section will mainly discuss previous work related to as-built BIMs for buildings or infrastructure (especially, pipe models).

To obtain accurate as-built BIMs and extract dimensional information from built environments, high-end 3D laser scanners (time-of-flight, phase-shift) are widely utilized for their high accuracy and the capability of obtaining data for large scale scenes (Pătrăucean et al. 2015). Budroni and Boehm (2010) used a plane sweep algorithm and a priori knowledge to segment point clouds into floors, ceilings, and walls, and created a 3D interior model by intersecting these elements. Since this method utilized the Manhattan-world assumption to obtain rectangular primitives for objects, it failed to handle complicated geometric primitives or complicated structures. Nüchter and Hertzberg (2008) used semantic labeling to find coarse scene features (e.g., walls, floors) of indoor scenes from point clouds obtained by a 3D laser scanner. They employed common-sense knowledge about buildings to label planar surfaces as wall, floor, ceiling, and door. Díaz-Vilariño et al. (2015) combined laser scan data and high-resolution images to detect interior doors and walls and automatically obtained optimized 3D interior models.

Instead of primarily utilizing planes from point clouds, Dimitrov and Golparvar-Fard (2014) presented a new method to segment point clouds into non-uniform B-spline surfaces for as-built modeling. Brilakis et al. (2010) explored a framework for automated generation of parametric building information models (BIMs) of constructed infrastructure from hybrid video and laser scanning data. They developed several automated processes for generating BIMs from point clouds, for example, automated generation of colored point clouds from video and laser scanner data, and automated identification of most frequently occurring objects. A drawback of these approaches that use high-end 3D laser scanners is that they need professional setup and operation (e.g., attaching markers in the environment for registering point clouds). Moreover, the post-processing methods used to extract 3D models from point clouds are time-consuming and

labor intensive since such sensors typically obtain millions of points to represent surfaces as point clouds.

Since 3D facility models are significant for maintenance, operation, and construction management, pipeline extraction from point clouds is also a major research topic as piping may comprise 50% of the value of the facility (Ahmed et al. 2014). Rabbani and Van Den Heuvel (2005) employed a sequential low dimensional Hough transform instead of the 5D Hough transform for automatic detection of cylinders in point clouds. They first estimated the orientation of a cylinder using a 2D Hough transform and computed cylinder position and radius by a 3D Hough transform. Avoiding the Hough transform in the 5D space, this method can reduce the space and time complexity. Ahmed et al. (2014) presented practical and cost-effective approach using the Hough transform and domain constraints to automatically identify and model 3D pipes from laser-scan-acquired point clouds. They also performed detailed error-modeling to filter out the systematic errors in order to localize the pipe cross sections. By degrading the 5D Hough space to a systematic repetitive 2D Hough space, this method greatly reduces computation complexity. Son et al. (2015) proposed a fully automated as-built 3D pipeline extract method from laser-scanned data using curvature. They utilized a normal based region growing method to find candidate segments and extracted curvature features to decide whether the segments are pipelines. The method can successfully separate pipelines from other objects while achieving 100% precision and recall over a data set captured from a chemical plant.

1.2.3 Research Gaps for 3D Reconstruction and Modeling in ARC

According to the review of previous literature on construction automation and robotics, and 3D reconstruction and geometric modeling, in order to improve automation and promote the use of robotics in civil engineering with affordable sensors, the current state of knowledge and

research needs to be further explored and investigated in the following directions: (1) utilization of low-precision sensors to collect 3D point clouds to perform 3D reconstruction and modeling for as-built BIM modeling and facility management in indoor environments, (2) occlusion handling to obtain complete 3D models, (3) integration of domain knowledge and sensor properties for specific civil engineering applications, (4) low-precision sensors for terrain modeling.

1.2.3.1 3D Reconstruction and Modeling from Low-precision Sensors in Indoor Environments

Due to the application accuracy requirements, most of the 3D as-built BIM modeling in the previous literature in civil engineering adopts the data from laser scanner (specifically terrestrial laser scanner) which can provide accurate measurements. First, the sensor is expensive and must be operated by experts with professional training. In order to collect data using terrestrial laser scanner, the scanner has to be moved to multiple locations to capture the whole scene while attaching salient markers in the environment for further registration of multiple scans. In addition, the scanner will acquire billions of points, which lead to high space and time complexity. Moreover, although terrestrial laser scanner can work in indoor environments, due to the high occlusion, the scanner has to be moved to a large number of positions which increases the cost and time for data acquisition as well as the number of point measurements.

Point clouds obtained from a set of images have been proven to be efficient and sufficiently accurate for some construction applications. Based on a set of unorganized or sequential images, the structure from motion (SFM) method is usually adopted to reconstruct 3D points by estimating the camera poses and the 3D coordinates of features extracted from images. Thus, this method requires that there exists overlapping distinguishable features points among

images. It will fail to reconstruct the 3D points when there are a lot of repetitive patterns or featureless objects (e.g., walls and floors).

In summary, 3D point clouds obtained from laser scanner and camera are mainly utilized to create exterior models for as-built buildings. There lacks an investigation of 3D point cloud acquisition as well as 3D as-built modeling using low-precision vision sensors in indoor environments for construction automation and robotics, as well as for facility management.

1.2.3.2 Occlusion Handling to Obtain Complete 3D Models

When using time of flight or optical sensors to capture data, many objects will block the view of sensors or the objects, and thus, sensors usually cannot obtain all the points of objects. For example, when an RGB camera is utilized to capture images for a typical classroom at a high position, the chairs will create obstruct the view on some areas on the floor and thus lead to incomplete data. For many perceptual sensors (optical, thermal, time of flight), occlusion is inevitable especially in indoor environments or construction sites which are abundant with various objects. The incomplete point clouds present many challenges for 3D modeling and further analyses (e.g., object detection and scene understanding).

Regarding occlusion handling, many previous projects (Díaz-Vilariño et al. 2015; Quintana et al. 2016; Xiong et al. 2013) on reconstructing as-built BIMs for indoor environments aim to detect openings (doors or windows) on the walls. Xiong et al. (2013) utilized the ray-tracking method to detect free space and occluded points by projecting the plane points into a 2D space. Then the edges estimated from the depth images are utilized to detect openings (e.g., doors and windows) so as to obtain as-built models for the interior of the buildings. Díaz-Vilariño et al. (2015) utilized color images to perform the ray-casting to find visible image sources so as to generate orthoimages. The openings (closed doors) are extracted by detecting

rectangles in the 2D space by the generalized Hough Transform. Since these methods are designed to create building models, the openings in the walls, especially doors are the main focus for occlusion handling. The methods for handling the other type of occlusion as well as occlusion for non-planar objects are not discussed.

1.2.3.3 Integration of Domain Knowledge for Construction Applications

To promote automation techniques and robots in construction, the domain knowledge about the characteristics of construction should be taken into consideration so as to efficiently complete tasks. First, one of the key features for construction applications is the unstructured, cluttered and dynamic environment. For example, a construction site usually has various materials, static or moving construction machines, moving construction workers, and so on. As-built indoor environments are abundant with various furniture, interior decorations, occupancies, and so forth. These features require any automation techniques to handle occlusion caused by the clutter, and collect data frequently and cost-effectively to cope with the dynamic change. Moreover, before completely automated construction robots, it is unavoidable that construction robots will co-exist with human workers in the environment, interact with humans to ask for instructions or decision-making, or collaborate with human workers to complete certain tasks. Therefore, the robots should be able to be aware of and even detect and track human beings in the cluttered and unstructured environment.

Another key feature in construction is the need for obtaining dimension information for task execution, maintenance, and safety issues. In construction, there often exists a discrepancy between designed and built models due to the uncertainty in real construction tasks or neglected issues in the design phase. For example, when a robot is installing windows within window frames, it has to capture the dimensions of the frames in order to place the windows correctly

instead of exclusively following the design. Therefore, the dimensions of certain objects are crucial for many construction applications, which calls for automation techniques or robotic systems that are capable of obtaining these dimensions efficiently and quickly.

1.2.3.4 Low-Precision Sensors for Terrain Modeling

3D terrain models are important for many construction and civil applications. For example, to monitor the excavation progress and safety, it is necessary to obtain 3D terrain models of the excavation project to provide quantitate evaluation of the progress. Traditionally, the terrestrial laser scanners or high precision RGB cameras are utilized to obtain precise point clouds for 3D reconstruction and modeling. However, the cost of data collection and processing for the collected data is usually high. To allow fast data acquisition and processing, low-precision sensors (normal RGB cameras in this application specifically) can be utilized to obtain point cloud with a certain level of accuracy (e.g., ~1cm) which can meet requirements for civil applications that require less accuracy. In addition, the data collection and processing for using the terrestrial laser scanners or high precision RGB cameras requires experts with professional training. It is beneficial to explore and design frameworks that are easy-to-use to reduce the cost of data collection and processing.

1.3 Research Objectives

The overall research objective of this dissertation is to explore and investigate utilization of low-precision vision sensors in 3D reconstruction and modeling for construction automation and robotics. In this dissertation, low-precision vision sensors denote computer vision sensors that can be utilized to obtain 3D point cloud of the environments. Specifically, for applications in indoor environments (especially facility management and maintenance), this dissertation utilizes

RGB-D sensors while normal RGB cameras for outdoor environments. The specific objectives of this research are as follows:

- Design point cloud completion methods to address occlusion problems for cluttered indoor environments so as to create complete as-built BIMs.
- Investigate the integration of a priori knowledge of construction scenes and sensor properties to enhance data acquisition or dimension retrieval of building components.
- Explore efficient algorithms to detect and track humans using low-precision vision sensors in cluttered environments with various illuminations to enable robotic systems to recognize surrounding people.
- Investigate the utilization of drone-mounted sensors for geometric modeling of outdoor construction environments or terrains.

1.4 Research Methodology

The steps and results below outline the research methodology of this dissertation.

- Design point cloud completion methods by incorporating the surface geometric properties and sensor characteristics to overcome the occlusion problems for indoor environments using RGB-D sensors, and to obtain topological relations of surfaces so as to obtain 3D complete as-built BIMs. This method will process the surfaces according to their planarity and size in order to gradually complete point clouds and estimate surface relations with high confidence. By utilizing octree representation, it is able to compute visibility information of voxels and process data for a large scale.

- Develop a user-guided dimensional analysis system using RGB-D sensors in indoor environments by utilizing a priori knowledge and the sensor properties to allow for real-time, efficient, and interactive dimension estimation. This system utilizes three indoor scenarios (i.e., hallways, doors, stairs) as examples to investigate the integration of prior knowledge of the scenes and the sensor characteristics to allow for real-time dimension estimation.
- Design and implement algorithms to efficiently track a specific human using a single RGB-D sensor in cluttered indoor environment with various illuminations. Based on RGB-D data, this method investigates the utilization of color and 3D features in order to develop real-time human tracking methods using a single RGB-D sensor.
- Design and implement a system to utilize drones to collect videos for 3D geometric modeling and interactive analysis to evaluate the slope stability for safety evaluation.
- Investigate the detection of landslides from RGB images for automatic landslide detection and mapping using drones. This method explores the utilization of a supervised classification framework to extract landslide from RGB images in order to provide potential landslide areas for automatic landslide data collection and mapping using drones.

1.5 Dissertation Outline

This dissertation is the result of compiling manuscripts that are related to 3D reconstruction and modeling using low-precision vision sensors for construction automation and

robotics. Since each chapter from Chapter 2 to 6 is written as a self-contained paper, there exists some overlapping in the background introduction and literature review in multiple chapters for the sake of completeness. There are two major parts in this dissertation: Chapters 2, 3 and 4, present the 3D reconstruction and modeling methods in indoor environments for facility management and maintenance; Chapters 5 and 6 describe the usage of drones and 3D reconstruction and modeling for terrain surface modeling and mapping.

Chapter 2 presents a join point cloud completion and surface connectivity relation estimation method for as-built BIM modeling in indoor environments. The surface geometric properties and visibility information of the 3D space which is computed using sensor characteristics and observations are utilized to add missing point and infer surface relations so as to reconstruct complete 3D models as well as the surface relations. Chapter 3 designs a user-guided dimensional analysis system using RGB-D sensors in indoor environments for indoor facility management and robotics applications. The sensor properties as well as prior knowledge about the scenes (i.e., objects whose dimensions are of interest) are combined to generate user guidance of how to move the sensor to obtain the dimension information.

As the robots will be gradually employed in construction or other industries, it is inevitable that robots and humans will coexist in the environment and thus the robots should have the capability of being aware of humans, share the space or materials with humans, and collaborate with human coworkers. Chapter 4 describes the human tracking method using RGBD sensors using an online learning strategy. The tracking method integrates the 3D features, RGB features, and an online learning method into the Kalman filter so as to track a specific person using an RGB-D sensor.

Chapter 5 develops a readily-deployable slope stability monitoring framework using drones for excavation projects. The framework allows easy operation of the drones and simple data collection as well as data processing for generating the 3D terrain model and the slope map quickly with little supervision. Chapter 6 explores the detection of landslides from RGB satellite images in order to provide input for an autonomous landslide monitoring system using drones. Chapter 7 summarizes the major findings and contributions of from this research, and discusses further work directions.

Chapter 2

Point Cloud Completion and Surface Connectivity Relation Inference

2.1 Introduction

Building information models (BIMs) contain rich geometric properties and spatial relations of various building entities (Tang et al. 2010) and can play an important role in different project stages, including design, construction, and maintenance phases (Azhar 2011; Hardin and McCool 2015). Once buildings or infrastructure are constructed, as-built BIMs that represents the current state of buildings are necessary due to the discrepancy between designed models and real constructed facilities or a lack of as-designed BIMs (Pătrăucean et al. 2015). Depending on the existence of designed or previous BIMs, as-built BIMs can be generated by updating the as-planned (Golparvar-Fard et al. 2011) or creating new BIMs from scratch (Volk et al. 2014). Three-dimensional (3D) point clouds collected by various sensors, e.g., laser scanners (Giel and Issa 2012; Hajian and Becerik-Gerber 2010), cameras (Bhatla et al. 2012; Klein et al. 2012), and depth cameras(Arnaud et al. 2016; Zhu and Donia 2013), are widely used to detect geometric shapes and their spatial relations between building elements in order to create as-built BIMs.

When using the sensors to obtain 3D point clouds, due to object occlusions or sensor limitations, observed point clouds usually cover only some parts of scenes and miss some other

parts. These point clouds will be referred to as incomplete point clouds in this chapter. For example, when using the 3D sensors (e.g., laser scanners, and depth sensors) to obtain a point cloud of a typical classroom, any fixed furniture such as anchored tables or chairs may block each other or the building elements (e.g., walls and floors) such that the resulting point clouds do not contain the complete geometry of the building elements. Based on the incomplete point clouds, it is challenging to recover complete 3D object models or identify object labels.

In order to mitigate this problem, this chapter presents a framework to jointly recover missing points and infer connectivity relations between surfaces for creating complete 3D models in indoor environments. Our framework exploits the fact that indoor environments are dominated by planar surfaces and that intersections of the planar surfaces provide a strong cue for completing missing data: if two planar surfaces are physically intersecting and connected, there is likely no gap between them and missing points between them can be filled. Thus the main process of our framework consists of extracting planar surfaces from the incomplete point cloud, estimating such connectivity relations between intersecting planar surfaces, and filling the missing points between the planar surfaces if the connectivity relations are found. For estimating the connectivity relations and filling the missing points, the framework uses the visibility information of points in 3D space, which is obtained by generating a truncated signed distance function (TSDF) octree (Steinbruecker et al. 2014) from the incomplete point cloud, such that we do not connect planar surfaces and fill missing points when there are free space measurements between them. To obtain more comprehensive connectivity relations and fill more missing points, our framework also includes additional processes such as (1) estimating connectivity relations between parallel planar surfaces located close to each other; (2) extracting nonplanar

surfaces and connecting each of them to a planar surface that supports it; and (3) filling missing points within individual planar surfaces.

The rest of the chapter is organized as follows. Section 2.2 *Previous Work* reviews related work on completing point clouds and estimating spatial characteristics for 3D reconstruction and as-build BIM modeling. Section 2.3 *Methodology* introduces the proposed method in detail. The experimental results and discussion on real-world and synthetic datasets are presented in Section 2.4 *Experimental Results and Discussion*. Section 2.5 *Conclusions and Future Work* draws conclusions of the chapter as well as discusses its limitations and future work.

2.2 Previous Work

Our work involves point cloud completion and spatial relation inference, each of which is discussed separately in this section.

2.2.1 Point Cloud Completion

To complete point clouds of surfaces, a common approach is to apply interpolation using geometric properties (e.g., symmetry and smoothness) of the surfaces. Janaszewski et al. (2010) filled holes by extracting the Euclidean skeleton and closing holes in the skeleton using a modified hole closing algorithm and thickness of objects. Kroemer (2012) utilized planar reflection symmetries to detect extruded shapes and then employed the parametric representation of the extruded shapes to complete the point cloud. Wang and Oliveira (2007) used moving least squares to interpolate both geometry and shading information to fill holes. Sharf et al. (2004) estimated the characteristics of the surfaces and filled holes by copying the best matching patches from valid regions. Carr et al. (2001) utilized radial basis functions to reconstruct smooth surfaces and complete holes by interpolation. When reconstructing 3D models with known

parametric representations, e.g., cylindrical objects (Ahmed et al. 2014; Son et al. 2015), identifying the exact parameters also helps complete missing point clouds. The parameters are used to infer unobserved points on its surface and thus obtain 3D complete models. Li et al. (2011) proposed a method to simultaneously fit primitives and recover their global mutual relations from noisy and incomplete point sets. By estimating the global relations and shape alignments, complete models are constructed. Chauve et al. (2010) presented a piecewise-planar 3D reconstruction and completion method from point clouds with noise and outliers. They added ghost primitives composed of planar primitives to ensure the continuation of detected primitives and the prevalence of vertical structures. Xiong et al. (2013) utilized a ray-tracing method to detect occluded regions of the walls and filled them using a 3D inpainting algorithm.

Another type of methods for completing point clouds is to reconstruct the models from partial point clouds by referring to existing 3D object model libraries. Kim et al. (2012) first acquired 3D models of common objects and their variability models and then recognized these objects from a single scan. Sung et al. (2015) collected examples of 3D shapes to build structural part-based priors and learned the distribution of positions and orientations of each part of the shapes. When processing incomplete point clouds, they estimated the parts and symmetries of the data and fused data source, symmetry, and database to reconstruct 3D complete models. Nan et al. (2012) trained a classifier on a set of shape features and performed the segmentation and classification simultaneously. The 3D completion models are obtained by a template deform-to-fit reconstruction method. Song and Xiao (2014) created a collection of 3D Computer-Aided Design (CAD) models, rendered each model from different viewpoints to get synthetic depth maps, and then trained a support vector machine (SVM) classifier for each depth rendering. A 3D detection window was employed to detect objects and reconstruct the 3D complete models.

Shao et al. (2012) developed an interactive approach to generate better segmentation results and then replaced the segments with objects from a 3D model database to obtain semantic models of indoor scenes.

2.2.2 Spatial Relations Inference

While 3D models only delineate geometric property of objects, BIMs also need spatial relations (or topological relations) of building components to facilitate complicated analysis and decision making, e.g., building object classification (Brilakis et al. 2010). Apart from merely representing simple spatial information (e.g., connection, adjacency, and intersection), spatial relations can also depict or be used to infer physical relations, which helps object detection and scene understanding. Existing BIMs can be directly employed to estimate spatial relations between 3D objects for spatial queries or analysis. Nguyen et al. (2005) proposed algorithms to automatically estimate topological information of building components from 3D CAD models. Based on the boundary representation of 3D objects, the following topological relations were computed: adjacency, separation, containment, intersection, and connectivity. Nepal et al. (2008) analyzed topological relations to derive construction features from a BIM model using the Industry Foundation Classes. Borrmann and Rank (2009) extracted directional relations (e.g., above, below, and north of) between 3D spatial objects for BIMs. Daum and Borrmann (2014) estimated topological relations for spatial queries based on a novel boundary representation of 3D models for BIMs. These methods rely on an existing BIMs as well as specific representation of models to efficiently compute topological relations of building entities.

Instead of using existing BIMs, spatial relations are also estimated when creating as-built BIMs. Silberman et al. (2012) presented a supervised framework to segment visible regions and infer their support relations by utilizing physical constraints and statistical priors on support. This

method processed a single RGB-D image individually instead of processing registered point clouds. Shao et al. (2014) extrapolated the cuboids around objects to recover the geometric attributes and their spatial relations by making the cuboids physically stable. This method aimed to recover the support relations and thus provided cues for retrieving models from 3D model libraries. Zheng et al. (2013) estimated the geometric primitives by segmenting the point clouds and completing the volumetric space. The completion mainly utilized the occlusion information and the Manhattan assumption. After the completion and segmentation, they used Swendsen-Wang Cut (Barbu and Zhu 2005) to optimize the stability of surfaces.

Different from the aforementioned previous work, this chapter explores to complete the missing points and recover the spatial relations (especially connections between surfaces) simultaneously from a registered point cloud. Considering the noisy data, the proposed method couples the surface segmentation process with the point cloud completion and connectivity relation inference, so that the connectivity relations and surface completion are performed robustly. Different from the methods in (Shao et al. 2014; Silberman et al. 2012; Zheng et al. 2013), the proposed method can handle larger scale indoor scenes. In addition, based on the assumption that planar surfaces are dominant structures in indoor environments, the modeling process in our method starts from major planar surfaces and then handles iteratively using small planar surfaces and nonplanar surfaces, which allows creating reliable and complete models.

2.3 Methodology

Figure 2-1 shows how the proposed method simultaneously completes a point cloud and recovers the connectivity relations of surfaces from multiple RGB-D frames. The input to the method is a series of organized point clouds (depth maps) registered with each other by a

simultaneous localization and mapping (SLAM) system. A truncated signed distance function (TSDF) octree is created to label the visibility of each octree voxel using the observed point clouds and sensor poses. From the octree point cloud, a normal-based region growing algorithm is first employed to extracted major planar surfaces. The system tries to create connections between the planar surfaces by filling the gap between surfaces if necessary. A connectivity graph $\boldsymbol{G}$ is created using the planar surfaces where each node denotes a surface while an edge represents the connection between two surfaces. Then, the normal-based region growing algorithm is utilized to extract small planar surfaces and nonplanar surfaces from the remaining point cloud. These new detected surfaces are utilized to update $\boldsymbol{G}$ by estimating the connections between them and the surfaces in $\boldsymbol{G}$. Therefore, $\boldsymbol{G}$ is updated by three different types of surface sets, i.e., major planar surfaces, small planar surfaces, and nonplanar surfaces, and the surface completion and connectivity inference methods depend on the surface type. The surface detection is iteratively performed until no surface is detected. Finally, the 3D complete models and the connectivity relations of surfaces are reconstructed.

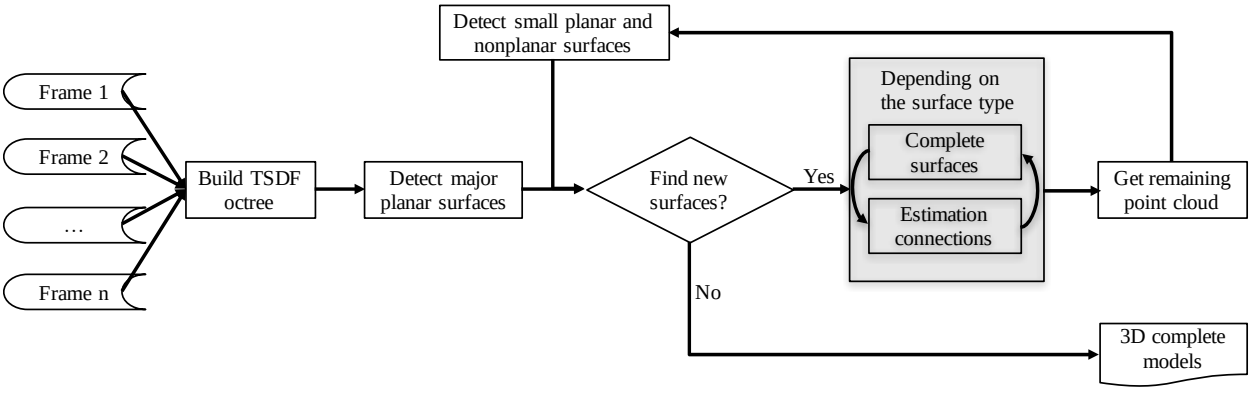


Figure 2-1: Point cloud completion and surface relation inference overview.

2.3.1 TSDF Octree Construction

The registered organized point clouds are fused by using a truncated signed distance function (TSDF) octree (Curless and Levoy 1996; Newcombe et al. 2011; Steinbruecker et al. 2014) to reduce the measurement noise and to obtain a single fused point cloud with fewer points compared to the raw point cloud. The TSDF octree representation can efficiently handle large-scale scenes while incorporating uncertainty of observed points. The TSDF octree generation is performed using the depth frames and the sensor poses (positions and orientations) computed by a SLAM system which registers all frames to the same coordinate system.

For each point in a frame of the organized point cloud, a ray from the sensor position to the point is cast to the TSDF volume. Then the TSDF values of certain octree voxels on the ray according to the depth measurement of the point are calculated for the first time or updated if they are computed using other points. The octree is incrementally expanded to cover all the measured points when the depth measurement falls into an uninitialized region. The octree generation iterates for all the points in all the frames.

After processing all the frames, the TSDF value of an octree voxel reflects the distance between the voxel and its nearest surface point. The TSDF value is close to zero for a measured point, while the TSDF value is positive and negative for a point in front of and behind a measured point, respectively. The visibility of an octree voxel is then determined according to its TSDF value. The voxels with zero TSDF values are categorized as occupied voxels, which form the single fused point cloud $\boldsymbol{P_{oct}}$. The voxels with positive and negative distance values respectively correspond to free space voxels (free space between the sensor and occupied voxels) and invisible voxels (occluded behind occupied voxels). In addition to these visibility labels, the voxels will be assigned a surface identification during the completion process.

2.3.2 Point Cloud Segmentation

Instead of segmenting the point cloud into separate objects in advance for later processes as previous methods (Zheng et al. 2013), in this chapter, the segmentation is coupled with estimating the connectivity relations and completing the point cloud. Since the point clouds contain noises due to the measurement noises or registration errors of multiple frames, it is challenging to attain the optimal and general threshold for both planar and nonplanar segmentation (e.g., the distance for a point belonging to a surface). For example, when detecting planes for the noisy point clouds, the segmentation method tends to find multiple planar clusters for the plane containing large noises. Therefore, in this chapter, the point cloud segmentation contains two separate steps, (1) major planar surface segmentation, and (2) small planar surface and nonplanar surface detection. Since most indoor objects contain planes, the proposed method processes the major planar surfaces before handling small planar and nonplanar surfaces. After the connection estimation and completion for the major planar surfaces, the small planar surface and nonplanar surface detection is iteratively performed on the remaining point cloud and the detected surfaces are processed for point cloud completion and surface relation inference.

A normal-based region growing algorithm (Xiao et al. 2014)is utilized to detect major planar surfaces. The normal vectors and curvatures of the points are estimated by performing principal component analysis of neighboring points. In order to find a planar cluster, the point with the smallest curvature is selected as the initial seed point from the points that are not classified to any cluster. Starting with this seed point $\boldsymbol{p_s}$ whose normal vector is $\boldsymbol{n_{p_s}}$, for each point in its neighborhood, $\boldsymbol{p} \in \mathbb{N}_{\boldsymbol{p_s}}$, if the difference between its normal vector and $\boldsymbol{n_{p_s}}$, is smaller than a threshold, $\boldsymbol{p}$ is assigned to the cluster $\boldsymbol{C_{p_s}}$ containing $\boldsymbol{p_s}$ and used as a new seed point. This process is iteratively performed until no point is added to $\boldsymbol{C_{p_s}}$ and all seed points are

explored. Then another qualified seed point, i.e., it has the smallest curvature among the remaining unassigned points while the curvature is smaller than the curvature threshold, is selected and the iterative growing process is performed again to find another cluster. The method stops until no qualified seed point is available or no cluster meeting the requirements (in this chapter, a cluster has to contain a minimum number of points) is found using the growing strategy. A small curvature threshold and a small normal vector discrepancy threshold are utilized to extract planes with high confidence. Based on these major planar surfaces, the connectivity graph $\boldsymbol{G}$ is constructed.

After the connectivity graph is created and updated by the detected major planar surfaces, for the remaining point cloud, the normal-based region growing algorithm is adopted to identify nonplanar surfaces and small planar surfaces by relaxing the thresholds for the curvature and the normal vector difference. Due to noise and irregular objects, some spurious clusters whose points scatter widely in 3D space may be obtained. To eliminate them, the point density of the cluster, i.e., the ratio of the number of points to the volume of its bounding box, is checked. The valid nonplanar surfaces and small planar surfaces are utilized to update the connectivity graph by finding connections between them and the planar surfaces in $\boldsymbol{G}$.

Since the algorithm of updating $\boldsymbol{G}$ by a small planar surface is different from that using a nonplanar surface, this work utilizes the cluster point distribution to distinguish a nonplanar surface from a small planar surface. The principal component analysis (PCA) is performed on the cluster points and the eigenvalues $\lambda_0, \lambda_1, \lambda_2$ ($\lambda_0 \leq \lambda_1 \leq \lambda_2$) and eigenvectors of the covariance matrix of the points are computed. The value $p \leftarrow 1 - \lambda_0/\lambda_1$ can reflect whether these points are from a plane. For a perfect plane, p is 1 because the points have zero variance along the normal vector (which is the same as the eigenvector corresponding to λ_0) of the plane,

and thus λ_0 is 0. For the points on a sphere, the three eigenvalues are identical and p is 0. If p is greater than a threshold (in this work, 0.9), the surface is viewed as a planar surface and used to update $\boldsymbol{G}$ using the corresponding method. Otherwise, it is processed as a nonplanar surface to update $\boldsymbol{G}$.

2.3.3 Connectivity Graph Construction

To represent connectivity relations between surfaces, an undirected graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ is constructed where the set of vertices $\boldsymbol{V}$ denotes surfaces segmented from the point cloud, and the set of edges $\boldsymbol{E}$ represents connections between vertices. If there exists a connection between two surfaces $\boldsymbol{v_i}$ and $\boldsymbol{v_j}$, an edge $\boldsymbol{e_{ij}}$ is added to $\boldsymbol{G}$. Since the planar surfaces are the major components of objects in indoor environment, the chapter first utilizes the major planar surfaces to construct $\boldsymbol{G}$ by recovering the connections among them. Then, $\boldsymbol{G}$ is updated using small planar surfaces and nonplanar surfaces by finding connections between them and the planar surfaces in $\boldsymbol{G}$.

2.3.3.1 Connection Inference and Point Completion for Planar Surfaces

Algorithm 1 describes the method of updating $\boldsymbol{G}$ based on the major planar surface set $\boldsymbol{S}$. In indoor environments, large planar surfaces (measured by the number of points in the observed point cloud) usually dominate the main structures of a scene and play an important role in surface connections within the scene. Thus, Algorithm 2-1 handles planar surface according to their sizes so as to recover the connections between larger planar surfaces before processing small planar surfaces. The algorithm contains two sub-processes where the first one finds connections between $\boldsymbol{S}$ and $\boldsymbol{G}$ while the other seeks connections within $\boldsymbol{S}$ and then adds them to $\boldsymbol{G}$. When building a connection between two surfaces, some points are added to the two surfaces and the completion can lead to changes of distances between surfaces, which improves the

possibility of building more connections. Since $\boldsymbol{G}$ already contains surface connections of major planar surfaces and partially filled surfaces during the building connection process, connections between $\boldsymbol{S}$ and $\boldsymbol{G}$ have higher confidence than those within $\boldsymbol{S}$. Therefore, Algorithm 2-1 first searches connections between $\boldsymbol{S}$ and $\boldsymbol{G}$ and then estimates connections within $\boldsymbol{S}$.

Algorithm 2-1: Update Connectivity Graph by the Major Planar Surface Set

```
1   function UpdateGraphByMajorPlaneSet(G, S_P)
2     S_P ← SortPlanesBySize(S_P)
3     do
4       for s ∈ S_P do
5         S_cs ← FindCandidateConnectedPlanes(s, V)
6         for s_cs ∈ S_cs do
7           if ThereIsAConnection(s, s_cs) = true then
8             S_P ← S_P\{s}
9             V ← V ∪ {s}
10            E ← E ∪ {e(s, s_CS)}
11          end if
12        end for
13      end for
14      for s_1 ∈ S_P, s_2 ∈ S_P, s_1 ≠ s_2 do
15        if ThereIsAConnection(s_1, s_2) = true then
16          S_P ← S_P\{s_1, s_2}
17          V ← {s_1, s_2} ∪ V
18          E ← E ∪ {e(s_1, s_2)}
19        end if
20      end for
21    while IsChanged(G) = true
22    return G
23  end function
```

As shown in Algorithm 2-1 Line 5, first, for each surface $\boldsymbol{s} \in \boldsymbol{S}$, its candidate connected surfaces $\boldsymbol{S_{cs}}$ are searched from the vertices in $\boldsymbol{V}$ by checking the spatial relations of surfaces. If the distance between two surfaces $\boldsymbol{s}$ and $\boldsymbol{s'} \in \boldsymbol{V}$ is smaller than a distance threshold ($15v_s$ where v_s is the octree voxel size) and they intersect with each other, there might exist a connection between the two surfaces. Thus $\boldsymbol{s'}$ is a candidate connected surface for $\boldsymbol{s}$, i.e. $\boldsymbol{S_{cs}} \leftarrow \boldsymbol{S_{cs}} \cup \{\boldsymbol{s'}\}$. In this chapter, the distance between two different surfaces, $\boldsymbol{s_1}$ and $\boldsymbol{s_2}$ is computed as the distance between the closest pair of points $(\boldsymbol{p_1}, \boldsymbol{p_2})$ while the two points are from different surfaces, i.e., $\boldsymbol{p_1} \in \boldsymbol{s_1}, \boldsymbol{p_2} \in \boldsymbol{s_2}$.

Once candidate connected planar surfaces $\boldsymbol{S_{cs}}$ are found, for each candidate surface $\boldsymbol{s_{cs}} \in \boldsymbol{S_{cs}}$ the algorithm will check whether there is a valid connection between $\boldsymbol{s_{cs}}$ and $\boldsymbol{s}$ (Algorithm 2-1 Line 7). The validity of a connection is related to the type of connections being estimated. This chapter estimates two connections for planar surfaces, i.e., the connection between two intersecting planar surfaces and the connection between parallel but not coplanar planar surfaces. The validity of the connections will be discussed later. If a valid connection is detected, the edge is constructed between the two surfaces and added to $\boldsymbol{G}$ (Algorithm 2-1 Line 10). Meanwhile, the surface $\boldsymbol{s}$ is moved from $\boldsymbol{S}$ to $\boldsymbol{G}$ (Algorithm 2-1 Lines 8-9). After trying to connect $\boldsymbol{S}$ to $\boldsymbol{G}$, Algorithm 2-1 detects connections within $\boldsymbol{S}$ as shown in Lines 14-20. If a connection between two planar surfaces, $\boldsymbol{s_1}$ and $\boldsymbol{s_2}$, is found, $\boldsymbol{s_1}$ and $\boldsymbol{s_2}$ are added to $\boldsymbol{V}$ while the edge between them is added to $\boldsymbol{E}$.

Algorithm 2-1 iteratively performs the two sub-processes until $\boldsymbol{G}$ is not updated, i.e., no surface is added to $\boldsymbol{G}$ and no more connection is detected.

Algorithm 2-1 is designed for the major planar surface set detected at the first stage the segmentation method. For the remaining planar surfaces (usually having a small number of

points), this chapter only tries to build a connection between them and the surfaces in $\boldsymbol{G}$, which is the same as Algorithm 2-1 Lines 4-13.

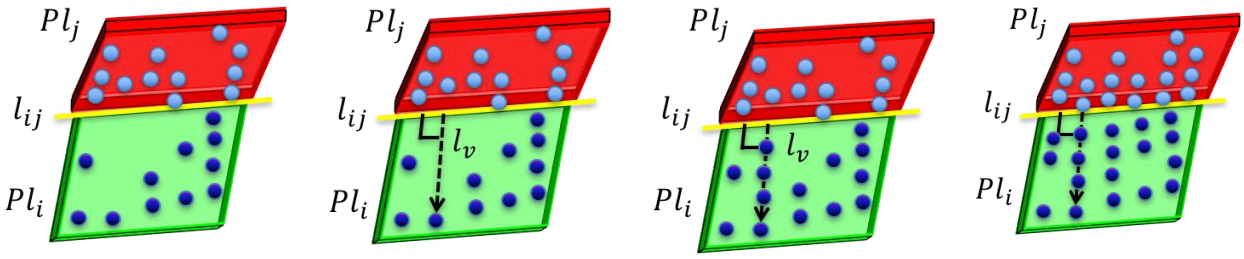
2.3.3.1.1 Connection Inference for Intersecting Planar Surfaces

The completion between two intersecting planar surfaces is performed by growing the two planar surfaces toward the intersection line as shown by Figure 2-2. The intersection line $\boldsymbol{l_{ij}}$ (Figure 2-2 (a)) of two planar surfaces $\boldsymbol{Pl_i}$ and $\boldsymbol{Pl_j}$ is estimated using the plane equations. Then as shown in Figure 2-2 (b), for each plane, e.g., $\boldsymbol{Pl_i}$ starting from a voxel $\boldsymbol{v}$ on $\boldsymbol{l_{ij}}$, a segment $\boldsymbol{l_v}$ orthogonal with $\boldsymbol{l_{ij}}$ is drawn toward the centroid of the surface until it hits a point $\boldsymbol{v_{Pl_i}}$ on that surface.

If all voxels on the segment $\boldsymbol{l_v}$ are unassigned to any surfaces or invisible, these voxels will be filled with points and added to the plane $\boldsymbol{Pl_i}$ (Figure 2-2 (c)). Otherwise, no voxels on $\boldsymbol{l_v}$ will be added to the octree. Thus, if $\boldsymbol{l_v}$ contains at least one free space voxel, none of these voxels are added since they have a high probability of being free space too. The process is iterated at all voxels on $\boldsymbol{l_{ij}}$. This completion process will fill the invisible or unassigned voxels around the intersection line of the two planar surfaces. This completion is temporarily performed first and finalized when the connection is valid. When the connection is valid, all the added points are maintained in the TSDF octree permanently.

Whether the connection between two intersecting planar surfaces is valid depends on the quality of the intersection segment between them. The intersection segment $\boldsymbol{seg_{ij}}$ is a segment on the intersection line $\boldsymbol{l_{ij}}$ between two surfaces and contains at least a certain number of points from both planar surfaces. If one of the planar surfaces have few points on $\boldsymbol{l_{ij}}$, there exists no intersection segment between the two planar surfaces. In order to compute the intersection

segment, a segment $\boldsymbol{seg_i}$ is estimated from the points that are in $\boldsymbol{Pl_i}$ and also on $\boldsymbol{l_{ij}}$. Similarly, $\boldsymbol{seg_j}$ is computed. Finally, the intersection segment $\boldsymbol{seg_{ij}}$ is estimated as the intersection of $\boldsymbol{seg_i}$ and $\boldsymbol{seg_j}$, i.e., $\boldsymbol{seg_{ij}} \leftarrow \boldsymbol{seg_i} \cap \boldsymbol{seg_j}$.



(a) Compute the intersection line $\boldsymbol{l_{ij}}$ (b) Draw a segment $\boldsymbol{l_v}$ perpendicular to $\boldsymbol{l_{ij}}$ (c) Fill the voxels on $\boldsymbol{l_v}$ (d) Iterate all the points on $\boldsymbol{l_{ij}}$

Figure 2-2: Completion between two intersecting planar surfaces.

If there exists an intersection segment $\boldsymbol{seg_{ij}}$ between two planar surfaces and $\boldsymbol{seg_{ij}}$ is sufficiently long and contains a certain number of points, the connection between them is valid. In this chapter, if the segment length is greater than $5v_s$ and the ratio of the number of points it contains to its length is greater than $0.9/v_s$, the intersection segment is good and the connection becomes valid. If a valid connection is detected, the connection is constructed between the two surfaces and the added points will be permanently assigned to the surfaces as well as the octree.

2.3.3.1.2 Connection inference for parallel planes

Other than intersecting connections, there exist connectivity relations between two parallel planar surfaces in the real world, e.g. a book lying on the table, and a television hanging on the wall. To estimate this connectivity relation, the two planar surfaces and their parallel relation (In this chapter, we assume that for plane relations, coplanar planes have the same

parameters and parallel planes only share the same normal vector.) should be correctly identified. Instead of simply utilizing a distance threshold between two parallel planes, this chapter also integrates the uncertainty of the planar surface to determine the parallel relation. If two planar surfaces are identified as parallel and can be connected, a connection between them is created and added to the graph $\boldsymbol{G}$.

In this chapter, to decide whether two parallel planar surfaces, $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ are connected to each other, their normal vectors should be parallel, i.e., the minimum angle between them is smaller than an angle threshold (10 degrees), and the distance between the planar surfaces is smaller than a threshold ($2.5v_s$). Then, all the points of the two planar surfaces are projected to a plane parallel to the planar surfaces and two corresponding 2D convex hulls, $\boldsymbol{ch_1}$ and $\boldsymbol{ch_2}$ for the projected points are computed. The overlapping value is computed as the ratio of the points of the smaller planar surface (for example, $\boldsymbol{P_1}$) falling within $\boldsymbol{ch_2}$. If the ratio is smaller than a threshold (20% in this chapter), there is no connection between $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$, and they are just parallel to each other. Otherwise, the distance $d(\boldsymbol{c_1}, \boldsymbol{P_2})$ from the centroid of the smaller surface $\boldsymbol{P_1}$ to $\boldsymbol{P_2}$ is computed. If the distance is greater than $\max(0.25(thick_{CBD_1} + thick_{CBD_2}), d_{thresh_p})$(in this chapter, $d_{thresh_p} = 3v_s$) where $thick_{CBD_1}$ and $thick_{CBD_2}$ represent the thickness of the cuboids of $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ (which will be explained in Section *Completion within Individual Planar Surfaces*), there is a connection between $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$. If the overlapping value is greater than the threshold and $d(\boldsymbol{c_1}, \boldsymbol{P_2})$ is smaller than the threshold, $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ are viewed as coplanar planes and will be merged into one planar surface.

2.3.3.2 Connection Inference and Point Completion for Nonplanar Surfaces

After the major planar surfaces are processed, nonplanar surfaces are utilized to update the connectivity graph $\boldsymbol{G}$. By assuming that a nonplanar surface is supported by a planar surface, this chapter aims to connect a nonplanar surface to at most one planar surface and build a connection between them. The candidate connected planar surfaces for a nonplanar surface are determined according to the distances between surfaces.

To find the best candidate, the method computes the weights for the candidate planar surfaces based on the gravity direction, the surface size, and the distances between surfaces. The gravity direction is employed to obtain physically reasonable connections. This chapter assumes that when capturing the first frame, the sensor is held almost horizontally and all the other frames are registered to the first frame. Therefore, the gravity direction is set using this prior knowledge according to the sensor coordinate system. Meanwhile, the surface size is utilized to ensure that the connection creation prefers large planar surfaces than small planar surfaces. The distance between surfaces is also considered to favor closer surfaces. Let $\omega_{\boldsymbol{NP_i}}(\boldsymbol{P_j})$ denote the weight of a planar surface $\boldsymbol{P_j}$ with respect to a nonplanar surface $\boldsymbol{NP_i}$. Then,

$$\omega_{\boldsymbol{NP}_i}(\boldsymbol{P_j}) = a_1 \cdot f(\boldsymbol{g}, \boldsymbol{n}_{\boldsymbol{P}_i}) + a_2 \cdot h(|\boldsymbol{P_j}|) + a_3 \cdot g\big(d(\boldsymbol{NP_i}, \boldsymbol{P_j}), dT_0, dT_1\big) \qquad (2.1)$$

where a_1, a_2, a_3 are weight coefficients. The first term $f(\boldsymbol{g}, \boldsymbol{n}_{\boldsymbol{P}_i}) = 1 - \angle(\boldsymbol{g}, \boldsymbol{n}_{\boldsymbol{P}_i})/\pi$ is a function of the gravity $\boldsymbol{g}$ and the normal vector of $\boldsymbol{P_j}$, $\boldsymbol{n}_{\boldsymbol{P}_i}$ where $\angle(\boldsymbol{g}, \boldsymbol{n}_{\boldsymbol{P}_i})$ is the minimum angle between $\boldsymbol{g}$ and $\boldsymbol{n}_{\boldsymbol{P}_i}$. The second term $h(|\boldsymbol{P_j}|)$ is a function of the planar surface size and favors large surfaces than small surfaces. In this chapter, if $|\boldsymbol{P_j}| > T_s$, $h(|\boldsymbol{P_j}|) = 0.7$. Otherwise, $h(|\boldsymbol{P_j}|) = 0.3$. As the octree is utilized, T_s can represent the number of voxels of an object model. In this chapter, T_s is set as the number of voxels of a square planar surface with a side

length of $50v_s$, i.e. $T_s \leftarrow 2{,}500$. The third term $g\left(d\left(\boldsymbol{NP_i}, \boldsymbol{P_j}\right), dT_0, dT_1\right)$ is defined as

$$g\left(d\left(\boldsymbol{NP_i}, \boldsymbol{P_j}\right), dT_0, dT_1\right) = \begin{cases} 1 - 0.5\frac{d\left(\boldsymbol{NP_i}, \boldsymbol{P_j}\right)}{dT_0} & d\left(\boldsymbol{NP_i}, \boldsymbol{P_j}\right) \leq dT_0 \\ 0.5 - 0.5\frac{d\left(\boldsymbol{NP_i}, \boldsymbol{P_j}\right) - dT_0}{dT_1 - dT_0} & dT_0 \leq d\left(\boldsymbol{NP_i}, \boldsymbol{P_j}\right) \leq dT_1 \\ -\infty & d\left(\boldsymbol{NP_i}, \boldsymbol{P_j}\right) > dT_1 \end{cases} \quad (2.2)$$

It is a function of the distance between $\boldsymbol{P_j}$ and $\boldsymbol{NP_i}$, and dT_0, dT_1 are utilized to categorize the distance into three different ranges (in this chapter, $dT_0 \leftarrow 15v_s$, $dT_1 \leftarrow 20v_s$). When the distance between the surfaces are too large (greater than dT_1), the weight is negative infinite and a connection between them will not be created.

The candidate planar surfaces are sorted by the weights. Starting from the planar surface with the largest weight, this chapter tries to construct a connection between the planar surface and the nonplanar surface by filling voxels that are not free space. For each point **p** on the nonplanar surface, its projection point on the planar surface $\mathbf{p_{proj}}$ is computed. If none of the voxels between **p** and $\mathbf{p_{proj}}$ are free space, these voxels are temporarily labeled as the nonplanar surface. If new points can be added between the two surfaces, there exists a valid connection between the two surfaces and the new points will be permanently added to the octree as well as the nonplanar surface.

2.3.4 Completion within Individual Planar Surfaces

To generate compact models, the planar surfaces are also completed using their parameters and the visibility information apart from completion when creating the connections between planar surfaces. Since the nonplanar surfaces in this chapter are represented by a cluster of points without any parametric representation, it is difficult to define complete models for them and thus they are only filled when finding the connections.

To facilitate the completion of individual planar surfaces, this chapter estimates a cuboid for each planar surface while considering the measurement error. Algorithm 2-2 shows the pseudocode of estimating the cuboid from the point cloud $\boldsymbol{P}$ assigned to the planar surface. Firstly, in Line 2 all the points on that planar surface are rotated to a plane parallel to XZ according to the normal vector of the planar surface. After this process, points of the rotated point cloud $\boldsymbol{P'}$ have nearly the same Y values. Then in Line 4, the minimum enclosing rectangle $rect$ of the 2D point sets containing X and Z of $\boldsymbol{P'}$ is estimated by ignoring Y values of these points. In Line 5, the mean μ_y and standard deviation σ_y of Y of $\boldsymbol{P'}$ are calculated. In Lines 6-11, the eight corners of the cuboid are computed based on $rect$, μ_y, and σ_y while the thickness of the estimated cuboid is $3\sigma_y$. Line 12 rotates the cuboid corners back as Line 2 transforms all points to a plane parallel to XZ.

Algorithm 2-2: Estimate a cuboid for a planar surface

```
1   function EstimateCuboid (P)
2       P' ← RotateToXZ(P)
3     P_XZ ← {P'.x, P'.z}
4     rect ← FindMinimumEnclosingRectangle(P_XZ)
5     (μ_y, σ_y) ← ComputeMeanAndSTD(P'.y)
6     for i ← [1,4] do
7       cuboid[i] ←Point(rect[i].x, μ_y − 1.5σ_y, rect[i].z)
8     end for
9     for i ← [1,4] do
10      cuboid[i + 4] ←Point(rect[i].x, μ_y + 1.5σ_y, rect[i].z)
11    end for
12    cuboid ←RotateBack(cuboid)
13    return cuboid
14  end function
```

The thickness of the cuboid of a planar surface reflects the uncertainty of this plane. For a planar surface close to the sensor, the thickness of its cuboid is usually smaller compared to the plane far away from the sensor. By taking into consideration the thickness of cuboids, the cuboid representation helps to distinguish coplanar and parallel relations between planar surfaces, and thus benefits the inference of the connectivity relations between parallel planes.

Another advantage of the cuboid representation is that combined with the visibility of voxels, the cuboid can be used to complete planar surfaces to get a complete planar surface model. After updating the connectivity graph and completing intersecting planar surfaces, the cuboid is utilized to add points to the planar surfaces.

The connected component analysis is performed to further complete a planar surface. Based on the voxels that are not labeled as free space within the cuboid of the planar surface, the Euclidean clustering method is performed to detect connected clusters of voxels. If the distance between a cluster and the planar surface is smaller than a threshold (in this chapter $2.5v_s$), the voxels of this cluster will be added to the planar surface. In addition, to avoid noise in computing the visibility information, all the free space voxels within the cuboid are also clustered using the Euclidean clustering method. The small connected components are assigned to the current planar surface while the others are not filled so as to maintain large free space of planar surfaces.

2.4 Experimental Results and Discussion

2.4.1 Experimental Setup

To evaluate the proposed method on real-world scenarios, we collected datasets of three different indoor scenes: (i) a cubic office desk, (Iii et al.) a typical officer corner with printers, and (Iii et al.) a table. The datasets were collected using an ASUS Xtion PRO while the point-

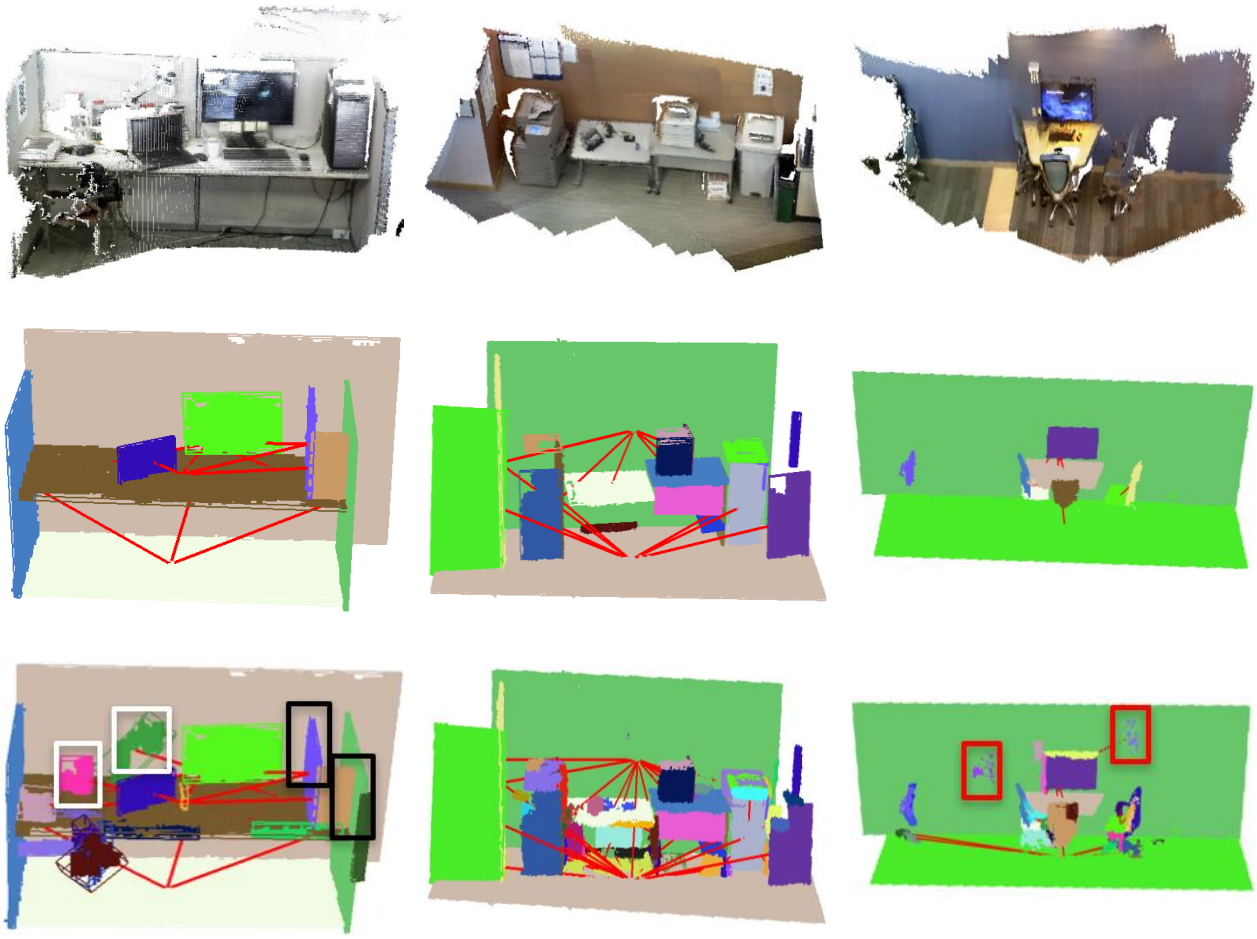
plane SLAM algorithm (Taguchi et al. 2013) was employed to obtain the registered point clouds and the sensor poses. The octree voxel size v_s was set as 0.02m.

In addition, to quantitatively evaluate the completion correctness and the model quality, the ICL-NUIM living room dataset (Handa et al. 2014) is utilized as it has ground truth mesh models. The ICL-NUIM dataset is a synthetic RGB-D dataset designed for evaluation of visual odometry and SLAM methods and contains the ground truth poses of the sensors. These sensor poses are used to register all the frames and build the TSDF octree. The octree voxel size v_s was set as 0.01m.

Regarding the thresholds in the point cloud segmentation, when processing the real-world datasets, to detect the major planar surfaces, the neighboring search radius is $5v_s$, the angle threshold for the normal difference is 6^o, the curvature threshold is 2, and the minimum cluster size is 300. For the ICL-NUIM datasets, the thresholds for detecting the major planar surfaces are $5v_s$, 3^o, 1 and 300, respectively. When detecting small planar and nonplanar surfaces, the thresholds are $3v_s$, 10^o, 10 and 150, respectively for all the datasets.

2.4.2 Results on Real-World Datasets

The accuracy of the connectivity relations between detected surfaces is evaluated using the real-world datasets. The ground truth connectivity relations of detected surfaces are manually identified and compared to those estimated by the proposed method. As the connectivity relations rely on the surface detection results, the connections related to undetected surfaces (small or irregular surfaces) are not considered.



(a) Scene **I** (b) Scene **II** (c) Scene **III**

Figure 2-3: Results on the real-world datasets.

Table 2-1: Evaluation of detected connections of the real-world datasets.

	Scene **I**		Scene **II**		Scene **III**	
	Plane	Final	Plane	Final	Plane	Final
$\|\boldsymbol{V}\|$	9	20	21	63	10	36
$\|\boldsymbol{E}\|$	15	21	24	65	6	27
$\|\boldsymbol{E_{err}}\|$	2	3	3	7	1	6
$\|\boldsymbol{E_{err}}\|/\|\boldsymbol{E}\|$	13%	14%	13%	11%	17%	22%
$\|\boldsymbol{E_{miss}}\|$	0	0	0	0	0	0

Figure 2-3 displays the results of processing the real-world datasets, including the original point cloud of the TSDF octree, the results after processing major planar surfaces, and the final results after processing small planar and nonplanar surfaces. The first row of Figure 2-3 shows the original point cloud in the octree $\boldsymbol{P_{oct}}$. The second row displays the results after processing the major planar surfaces $\boldsymbol{M_P}$, where each surface is rendered using a random color and the red segments represent the connections. The last row shows the results after processing the small planar and nonplanar surfaces $\boldsymbol{M_F}$. The black rectangles in Figure 2-3 (a) show the false connections due to over-filling of intersecting planar surfaces while the red rectangles in Figure 2-3 (c) include the false connections from wrong detected surfaces. The white rectangles in Figure 2-3 (a) cover example areas where the nonplanar surfaces and the connections are correctly identified.

Table 2-1 shows the connectivity evaluation results of the three real-world datasets. $|\boldsymbol{V}|$ is the number of surfaces in $\boldsymbol{V}$ while $|\boldsymbol{E}|$ represents the number of the detected connections among surfaces. $|\boldsymbol{E_{miss}}|$ denotes the number of undetected connections while $|\boldsymbol{E_{err}}|$ is the number of false detected connections. For each scene, the results after processing the major planar surfaces (the *Plane* column in Table 2-1), $\boldsymbol{M_P}$ are also evaluated as well as the final results $\boldsymbol{M_F}$ which are generated by processing small planar and nonplanar surfaces based on $\boldsymbol{M_P}$**.** As shown in Table 2-1, the number of false detected connections $|\boldsymbol{E_{err}}|$ for $\boldsymbol{M_F}$ is low, equal or less than seven for all the three scenes. The ratios of false detected connections, $|\boldsymbol{E_{err}}|/|\boldsymbol{E}|$ are equal or less than 22%, which demonstrates that at least 78% of the detected surface connections are correct.

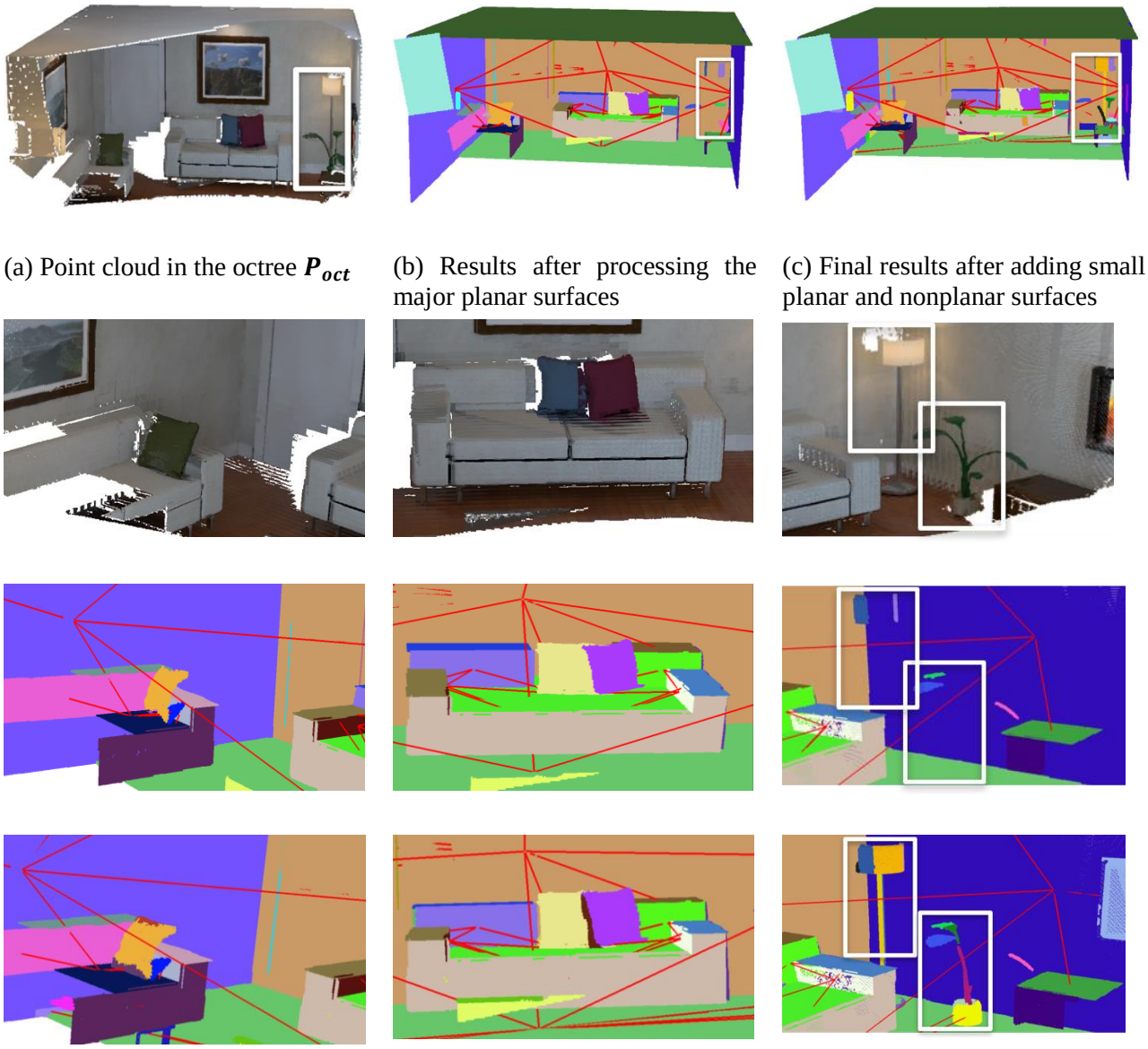
The false connections of $\boldsymbol{M_F}$ are caused by overfilling of intersecting planar surfaces and from wrong detected surfaces. The overfilling of intersecting planar surfaces denotes the case that when two intersecting planar surfaces are not connected in the real-world, the proposed

method creates a connection between them by adding points between them. It occurs when a valid connection can be created due to a lack of visibility information of the voxels between them. For example, the desktop box in Scene **I** is not connected to any of the walls. However, the proposed method fills the gap between the desktop box and the walls and thus create connections between them as shown in the black rectangles in the bottom image of Figure 2-3 (a).

False detected planar surfaces are mainly caused by registration errors of multiple depth frames and the uncertainty of sensor measurements. This is the reason why $|\boldsymbol{E}_{err}|$ increases after processing small planar and nonplanar surfaces, i.e., as shown in Table 1, the *Final* columns have larger $|\boldsymbol{E}_{err}|$ compared to the *Plane* columns. For example, for the wall in Scene **III** in Figure 2-3 (c), the nonplanar surface segmentation method detects multiple clusters around the wall as shown in the red rectangles of the bottom image of Figure 2-3 (c) and these clusters create false connections in the final results. Even though processing small planar and nonplanar surfaces leads to the increasing of errors in estimating connections, it can still identify some correct small planar or nonplanar surfaces and find the correct connections. As shown in Figure 2-3 (a), within the white rectangles, a lamp (in green) and a cup (in magenta) are correctly added to the model with correct connectivity relations.

The number of undetected connections $|\boldsymbol{E}_{miss}|$ is zero, which demonstrates that the proposed method is able to recover all the connections of surfaces in $\boldsymbol{V}$. According to our method, there exist three types of connections, (1) the connection between intersecting planar surfaces, (2) the connection between two parallel planar surfaces, and (3) the connection between a nonplanar surface and a planar surface. Based on the criteria for connecting surfaces, missing connections might occur when (1) many points between two actually connected surfaces (two planar surfaces, or a nonplanar surface and a planar surface) are not observed and their distance is too larger to be

considered for creating connections, and (2) a nonplanar surface is connected to more than one planar surface while our method only detects a single connection to a planar surface.



(a) Point cloud in the octree $\boldsymbol{P_{oct}}$ (b) Results after processing the major planar surfaces (c) Final results after adding small planar and nonplanar surfaces

(d) Close-up views of some areas of the three point clouds from left to right.

Figure 2-4: Results of kt0.

2.4.3 Results on ICL-NUIM Datasets

We utilize the ICL-NUIM living room dataset which contains four scenes, kt0, kt1, kt2, and kt3, to evaluate (1) the overall accuracy of the final models by comparing the final model point cloud with the ground truth point cloud, and (2) the completion results by counting the number of correctly filled points. As the four scenes have some overlapping areas, Figure 2-4 shows the modeling and connectivity inference results of kt0 which contains the major part of the living room dataset. Figure 2-4 (a) displays the original point cloud of the octree P_{oct} while (b) and (c) respectively show the results after processing the major planar surfaces and adding nonplanar surfaces. In Figure 2-4 (b) and (c), each surface is rendered by a random color while the red segment denotes that there is a connection between two surfaces. Figure 2-4 (d) shows the close-up view of some areas of the three point clouds in the above row. The white rectangles show example areas that nonplanar surfaces from a lamp and a plant pot are added to the final model after processing small planar and nonplanar surfaces.

By comparing Figure 2-4 (b) and (c) and using the close-up views in (d), i.e. the second and third rows of (d), it can be found that the results after processing small planar and nonplanar surfaces successfully add many small planar or nonplanar surfaces to the results, e.g., the plant pot and lamp in the white rectangles.

The comparison between the final model point cloud $\boldsymbol{P_m}$ and the ground truth model $\boldsymbol{P_{gt}}$ is performed by estimating the distance between a point pair that one is from $\boldsymbol{P_m}$ and the other in $\boldsymbol{P_{gt}}$. For each point $\boldsymbol{p_m} \in \boldsymbol{P_m}$, its nearest point in $\boldsymbol{P_{gt}}$ is searched and denoted as $\boldsymbol{p_{gm}}$. The Euclidean distance between $\boldsymbol{p_m}$ and $\boldsymbol{p_{gm}}$, $\|\boldsymbol{p_m} - \boldsymbol{p_{gm}}\|_2$ is calculated and referred to as the error of $\boldsymbol{p_m}$. Then the mean, median, standard deviation, and maximum values of the distances for all the four scenes, i.e., kt0, kt1, kt2, and kt3, are computed and displayed in Table 2-2. In addition,

using the same method, the original point cloud of the octree $\boldsymbol{P_{oct}}$ is evaluated and the results are presented in Table 2-2. To mitigate the distance differences due to the voxelization in creating the octree, both $\boldsymbol{P_{oct}}$ and $\boldsymbol{P_m}$ are aligned to $\boldsymbol{P_{gt}}$ using the Iterative Closest Point (ICP) (Besl and McKay 1992) algorithm.

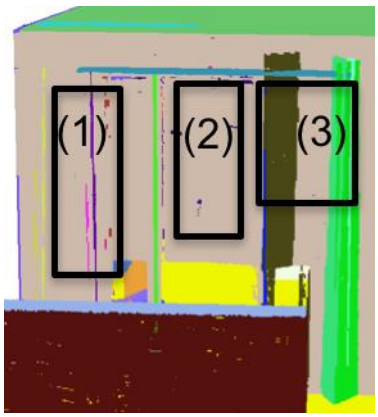
Table 2-2: Evaluation the quality of $\boldsymbol{P_{oct}}$ and $\boldsymbol{P_m}$ with respect to the $\boldsymbol{P_{gt}}$.

Point Cloud	Error (m)	kt0	kt1	kt2	kt3
$\boldsymbol{P_{oct}}$	Mean	0.006	0.006	0.006	0.007
	Median	0.005	0.006	0.007	0.006
	Std.	0.003	0.003	0.003	0.003
	Max.	0.019	0.021	0.021	0.023
$\boldsymbol{P_m}$	Mean	0.008	0.016	0.012	0.011
	Median	0.007	0.011	0.008	0.007
	Std.	0.007	0.024	0.022	0.02
	Max.	0.188	0.351	0.352	0.351

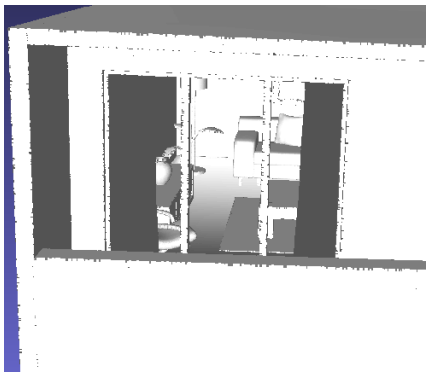
The lower mean values (less than or equal to 0.16m) in Table 2-2 demonstrate that the proposed method is able to reconstruct high-quality models. The fact that all the median values of $\boldsymbol{P_m}$ are lower than the mean values indicates half of the point errors are lower than the mean values. As shown in Table 2-2, the maximum errors of $\boldsymbol{P_m}$ (the row *Max.*) are larger than those of $\boldsymbol{P_{oct}}$, especially for the last three scenes. The main discrepancy for the last three scenes mainly occurs around a French window area. As shown in Figure 2-5, many points (for example the points within the middle rectangle, Rectangle (2) in Figure 2-5 (b)) are filled within the window frame. This is mainly because those filled voxels are not labeled as free space since there are no points behind the window glasses. However, Figure 2-5 (d) also indicates that the errors of points on the walls are small (blue indicates small point errors while red denotes large errors.), which

demonstrates that the proposed method is capable of recovering reliable points if the visibility information is correctly estimated.

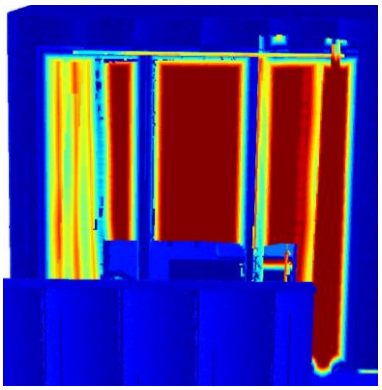
(a) Original point cloud of the window area.



(b) Final models of the window area.



(c) Ground truth for that area.



(d) Errors of that area.

Figure 2-5: Errors around the French window area in kt2.

The final model point errors can be categorized into three types, (1) errors of using planar surfaces to approximate nonplanar surfaces, (2) errors of over-filling of planar surfaces, and (3) errors of over-growing of planar surfaces, which correspond to the areas covered by the three rectangles in Figure 2-5 (b). Figure 2-5 (a) shows the original point cloud of the French window

are in the octree and (b) displays the models of that area where each surface is rendered by a random color. Figure 2-5 (c) displays the ground truth points while (d) depicts the errors of the models in (b) by rendering the errors from cool colors to warm colors, i.e. the dark red denotes large errors while the blue represents small errors. The three rectangles in Figure 2-5 (b) from left to right cover the areas of errors due to (1) processing nonplanar surfaces as planar surfaces, (2) over-filling of planar surfaces, and (3) over-growing of intersecting planar surfaces.

The errors of using planar surfaces to approximate nonplanar surfaces occur when the segmentation method detects planar surfaces from non-planar objects. For example, the segmentation method detects several planar surfaces, in particular for the area in the left rectangle, Rectangle (1) in Figure 2-5 (b), for the accordion folding doors before the French windows. During the processing of the planar surfaces, many wrong points are added in finding the connections and filling of the planar surfaces and thus cause large point errors.

The over-filling error is due to the missing of free space information for the voxels within a planar surface. For the French window as shown in Figure 2-5, only a small number of the points (mainly the lower parts of the window) within the window area are identified as free space by the points on the ground behind the window. The visibility information of other points (e.g., the points within the middle rectangle, Rectangle (2) in Figure 2-5 (b)) is unknown due to the visibility information computation strategy. Therefore, as shown in Figure 2-5 (b), those points are added to the point cloud during the planar surface filling process and lead to large (actually the largest) point errors.

The over-growing of planar surfaces occurs when the voxels between two intersecting planar surfaces are not labeled as free space due to a lack of information. When the distance between the intersecting planar surfaces is smaller than a distance threshold ($15v_s$), the proposed

method will add points between them and create a connection between them. In this dataset, there exist many intersecting planar surfaces due to the accordion folding doors before the French window. The segmentation method detects some planar surfaces and connects them by adding points behind the accordion folding doors. The large point errors within the right rectangle, Rectangle (3) in Figure 2-5 (d) are mainly caused by over-growing of planar surfaces.

Table 2-3: Evaluation of completion results.

	kt0	kt1	kt2	kt3
$\|\boldsymbol{P_m}\|/\|\boldsymbol{P_{oct}}\|$	2.74	3.20	2.60	2.39
n_{corr}/n_{add}	98%	87%	90%	93%

The number of correct points in $\boldsymbol{P_m}$ with respect to $\boldsymbol{P_{gt}}$ is also estimated using the point error. A point $\boldsymbol{p_m} \in \boldsymbol{P_m}$ is correct if $\left\|\boldsymbol{p_m} - \boldsymbol{p_{gm}}\right\|_2 \leq 2.5v_s$. By excluding $\boldsymbol{P_{oct}}$ from $\boldsymbol{P_m}$, we obtain the number of added points n_{add} and the number of correctly added points n_{corr} in $\boldsymbol{P_m}$. That is, $n_{add} = |\boldsymbol{P_m}| - |\boldsymbol{P_{oct}}|$ and n_{corr} is the difference between the size of correct added points in $\boldsymbol{P_m}$ and the size of $\boldsymbol{P_{oct}}$. The ratio of n_{corr} to n_{add} is also shown in Table 2-3. The ratios of correctly added points show that more than 87% of the added points from this proposed method are correct. In addition, Table 2-3 displays the ratio of the final model point cloud size $|\boldsymbol{P_m}|$ to the original point cloud size $|\boldsymbol{P_{oct}}|$. The results demonstrate that although the proposed method does not perform completion for isolated planar surfaces, it is able to at least double the point cloud size.

2.4.4 Computational Analysis

Regarding the computational time on the experimental datasets, the processing time of the proposed method ranges from several minutes to about half an hour on a standard desktop

personal computer (PCL) as shown in Table 2-4 where $|\boldsymbol{P_{oct}}|$ represents the number of points of the octree, $|\boldsymbol{V}|$ is the number of surfaces in the connectivity graph $\boldsymbol{G}$, and $|\boldsymbol{E}|$ denotes the number of surfaces in $\boldsymbol{G}$. Table 2-4 shows that the computational time is positively related to the number of frames. The larger the number of frames is, the more computational time the system takes. The computational time is also related to the scale of the scene, which can be reflected by the size of the original point cloud $|\boldsymbol{P_{oct}}|$. A large-scale scene usually contains more points and surfaces and thus requires more computational expense in filling the surfaces and estimating the connections.

Table 2-4: Computational time of all the datasets.

	# of frames	$\|\boldsymbol{P_{oct}}\|$	$\|\boldsymbol{V}\|/\|\boldsymbol{E}\|$	Time (minutes)
Scene **I**	60	54,567	20/21	0.78
Scene **II**	44	169,631	63/65	0.84
Scene **III**	70	188,967	36/27	1.42
kt0	1,509	660,577	68/64	34.73
kt1	966	832,596	117/158	29.23
kt2	881	1,046,417	127/176	31.66
kt3	1,241	742,691	93/104	32.77

In the proposed method, there are two main processes that affect the computational time: (i) creating the TSDF octree, and (Iii et al.) updating the connectivity graph using the major planar surfaces. When constructing the TSDF octree, the system loads each depth frame and utilizes each observed point of the frame to update the TSDF tree. Therefore, the number of frames greatly affects the computational time. Moreover, the graph updating using the major planar surfaces involves surface parameter updating (e.g., computing plane equations and estimating the cuboid) and traversing a large number of voxels in order to decide their labels.

Thus a large number of planar surfaces in the scene generally leads to more computational time compared to a scene containing a small number of planar surfaces.

2.5 Conclusions and Future Work

This chapter presented a framework that integrates point cloud completion and surface connectivity relation inference into a joint process to obtain complete 3D models and surface connections. The framework utilizes geometric properties of surfaces and the visibility of octree voxels to estimate the connections of surfaces and recover missing points between the surfaces. The method first processes the major planar surfaces to estimate their connectivity relations and fill the missing points. Then small planar surfaces and nonplanar surfaces are utilized to find more connections between them and the major planar surfaces by adding points if necessary. Furthermore, individual planar surfaces are further filled using the connected component analysis within the surface cuboid to obtain complete surface models. Experimental results demonstrated that the proposed method is able to recover all connectivity relations between surfaces, double the point cloud size by adding points of which more than 87% are correct, and obtain high-quality 3D models.

The proposed method handles nonplanar surfaces using a basic strategy, i.e. growing their points toward planar surfaces if possible. It does not incorporate the geometric properties of the nonplanar surfaces. In addition, the connectivity relations between nonplanar surfaces are not estimated in this chapter. Future work will explore to segment more primitives other than planes (e.g., cylinder, sphere, etc.) using non-uniform B-Spline surface fitting methods (Dimitrov et al. 2016) and infer connectivity relations between nonplanar surfaces as well.

Chapter 3

User-Guided Dimensional Analysis of Indoor Building Environments

3.1 Introduction

Three-dimensional (3D) geometry and, in particular, dimensional information about the built environment is required in a wide range of civil infrastructure applications (Bosch 2010). During the construction phase, dimensional information must be monitored on site so that the work can meet the requirements of the design and specifications. During the maintenance phase, dimensional information is necessary to check whether the built environment remains consistent with existing building codes and to quantify any developed flaws (e.g. deformations). In addition, in the context of construction automation, dimensional information is useful for any robot performing tasks in the construction or built environment. For example, a door installing robot must consider the actual size of the door frame on a construction site instead of the designed size due to potential tolerance discrepancies. Given such dimensional information, the robot is able to install a door correctly and ensure that it can fit the panel in a frame accurately. In addition, the dimensions of any openings are significant for an autonomous robot while moving in indoor environments. For example, when passing through a door, a robot has to detect the dimension of the opening space so that it can make an informed choice about whether to directly go through this door or to find another way.

Traditionally, dimensional information in the built environment is manually obtained by tape measurements, which is labor intensive and has limited accuracy. With the rapid development of sensors for capturing 3D point clouds, geometric models of the civil infrastructure can be obtained rapidly and accurately, thereby making the automatic retrieval of infrastructure dimensions a possibility. In order to obtain accurate dimensions of civil infrastructure, laser scanners are widely used to capture high-accuracy 3D point clouds to build 3D models that contain detailed dimensional information (Bennett 2009; Huber et al. 2010; Xiong et al. 2013). However, Tang et al. (2010) pointed out that this process is usually time-consuming and not fully automated.

Instead of using 3D laser scanners, RGB cameras (Bae et al. 2014; Brilakis et al. 2011; Golparvar-Fard et al. 2011) can be used to capture a series of images that are then processed using structure from motion (SFM) to generate 3D point clouds. This method is able to obtain point clouds for large-scale scenes and has a shorter data acquisition time compared to methods that use laser scanners. Another commonly used method to obtain colored 3D point clouds is to employ stereo cameras that are composed of two RGB cameras (Fathi and Brilakis 2011) or to utilize RGB-D cameras consisting of an RGB camera and a depth camera (Chen et al. 2015; Zhu and Donia 2013). One of the benefits of utilizing stereo or RGB-D cameras is that these cameras enable obtaining point clouds from single frames and thus performing data analysis in real time. Moreover, colored 3D point clouds provide the opportunity to extract semantic information compared to point clouds generated from laser scanners (Golparvar-Fard et al. 2011). Therefore, stereo or RGB-D cameras are well suited for geometry and dimension interpretation from 3D point clouds in contexts where human users or robots need to interact with the built environment in real time.

In this chapter, we propose a user-guided dimensional analysis approach that is able to compute dimensions in indoor built environments using a color and depth (RGB-D) sensor. The method performs dimensional analysis on a single frame obtained from an RGB-D sensor to achieve high computational efficiency and to avoid error accumulations in multi-frame registration. Due to the limited field of view and measurement range of the sensor, a single frame cannot guarantee that all dimensional information of interest can be computed. Therefore, a knowledge-based user guidance system is developed to guide a user (or a robot) to move the sensor to a better position so that complete data suitable for dimensional analysis is collected. After a complete frame data is collected, the geometric analysis is performed to obtain the necessary dimensional information.

The remainder of the chapter is organized as follows. Section 3.2 *Previous Work* reviews related work and outlines its limitations. Section 3.3 *The dimensional Analysis System* describes the designed method in detail. Section 3.4 *User Guidance* describes the conducted experiments and the obtained results. Finally, Section 3.5 *Conclusions and Future Work* draws conclusions and discusses future work.

3.2 Previous Work

In the context of getting dimensional information from built environments, several research studies have focused on creating 3D models by using high-end 3D laser scanners (2D rotational laser scanners or terrestrial laser scanners), which can provide accurate and rich 3D point clouds of a large environment. Budroni and Boehm (2010) used a plane sweep algorithm and a priori knowledge to segment point clouds into floors, ceilings, and walls, and created a 3D interior model by intersecting these elements. Since this method utilized the Manhattan-world assumption to obtain rectangular primitives for objects, it failed to handle complicated geometric

primitives or complicated structures. Nüchter and Hertzberg (2008) used semantic labeling to find coarse scene features (e.g., walls, floors) of indoor scenes from point clouds obtained by a 3D laser scanner. They employed common-sense knowledge about buildings to label planar surfaces as wall, floor, ceiling, and door. Díaz-Vilariño et al. (2015) combined laser scan data and high-resolution images to detect interior doors and walls and automatically obtained optimized 3D interior models. Instead of primarily utilizing planes from point clouds, Dimitrov and Golparvar-Fard (2014) presented a new method to segment point clouds into non-uniform B-spline surfaces for as-built modeling.

In addition, several researchers have also used high-accuracy laser scanners to obtain 3D models of dynamic construction environments and equipment. Wang and Cho (2015) designed a smart scanning system to rapidly identify target objects and update the target's point clouds. They then used concave hull surface modeling algorithms to get a 3D surface model. Cho and Gai (2014) used laser scanners to obtain 3D point clouds of the environment and identified 3D target models by comparing them to a model database. The field results of these two chapters demonstrated that the method could improve productivity and safety in heavy construction equipment operations. Brilakis et al. (2010) explored a framework for automated generation of parametric building information models (BIMs) of constructed infrastructure from hybrid video and laser scanning data. They developed several automated processes for generating BIMs from point clouds, for example, automated generation of colored point clouds from video and laser scanner data, and automated identification of most frequently occurring objects.

A drawback of these approaches that use high-end 3D laser scanners is that they need professional setup and operation (e.g., attaching markers in the environment for registering point clouds). Moreover, the post-processing methods used to extract 3D models from point clouds are

time-consuming and labor intensive since such sensors typically obtain millions of points to represent surfaces as point clouds.

Instead of using high-accuracy laser scanners, simultaneous localization and mapping (SLAM) techniques have been widely used for registering multiple 3D frames and obtaining 3D models of large-scale environments with affordable sensors (e.g. low-cost RGB-D sensors, cameras). Newcombe et al. (2011) presented KinectFusion, which employed an iterative closest point (ICP) algorithm to register a current depth map to a global model reconstructed by fusing all previous depth maps. Taguchi et al. (2013) proposed the point-plane SLAM system that uses both points and planes as primitives to achieve faster correspondence search and registration of data frames, and to generate 3D models composed of planar surfaces. Cabral and Furukawa (2014) proposed a method for reconstructing a piecewise planar and compact floor plan from multiple 2D images, which provides an improved visualization experience albeit with fewer geometric details. Although the 3D models generated by these methods enable dimensional analysis in large-scale environments, the accuracy is limited due to drift error accumulations in multi-frame registration.

Unlike previous work, the method described in this chapter aims to obtain dimensional information of indoor scenes from a single frame of an affordable RGB-D sensor. The proposed single-frame approach avoids the error accumulation problems inherent in multi-frame registration. In order to overcome the limitations of a single frame, such as the limited field of view and measurement range, this chapter describes a user guidance system that provides directional feedback for the user to obtain complete data suitable for dimensional analysis.

The most relevant prior work to our method is Kim et al. (2012) which presented a hand-held system for real-time interactive acquisition of residential floor plans. The system described

in that chapter integrates an RGB-D sensor, a micro-projector, and a button interface to help the user capture important architectural elements in indoor environments. Instead of obtaining the floor plan of a building using a SLAM technique as in Kim et al. (2012), the method in this chapter focuses on obtaining dimensional information of specific objects in indoor environments from a single frame. Moreover, the designed user guidance system guides the user in observing essential components for specified scenes.

The proposed user guidance system was inspired by Richardson et al. (2013) and Bae et al. (2010). Richardson et al. (2013) presented a user-assisted camera calibration method that suggests the position of calibration targets in the captured images to obtain reliable, stable, and accurate camera calibration results. Bae et al. (2010) proposed the computational rephotography system that, given a reference image, guides the user to capture an image from the same viewpoint. In order to obtain accurate dimensional information from a single frame of an RGB-D sensor, the proposed user guidance system evaluates the completeness of the current frame and then instructs the user to move the sensor to get improved results for the application. Using basic guidance, the proposed system can lead a non-expert user through the steps necessary to obtain complete data and thus accurate dimensional measurements.

3.3 The Dimensional Analysis System

In this chapter, the focus of the dimensional analysis is on civil infrastructure with planar surfaces in indoor environments using an RGB-D sensor. The proposed framework is shown in Figure 3-1. Firstly, one frame of 3D point clouds (for example Figure 3-1 (a)) is acquired by an RGB-D sensor. Then, the preprocessing is conducted on the point clouds to extract planar surfaces and compute topological relationships of these planes (Figure 3-1 (b)). Based on the planes and their topological relations, the geometric analysis is performed to compute the initial

dimensions of the scene (Figure 3-1 (c)). Combining the scene type and the initial dimensional measurements, the user guidance system evaluates the completeness of the current frame and dimensional measurements. If the data frame does not contain all components for computing the dimensions, the user guidance system provides instructions for moving the sensor to get a complete frame and thus accurate dimension measurements. Therefore, a new frame data (Figure 3-1 (d)) is captured by the sensor. The same processes, i.e. preprocessing (Figure 3-1 (e)) and geometric analysis (Figure 3-1 (f)), are performed to acquire new dimensions, which have a higher quality and are used as the final dimension estimation results.

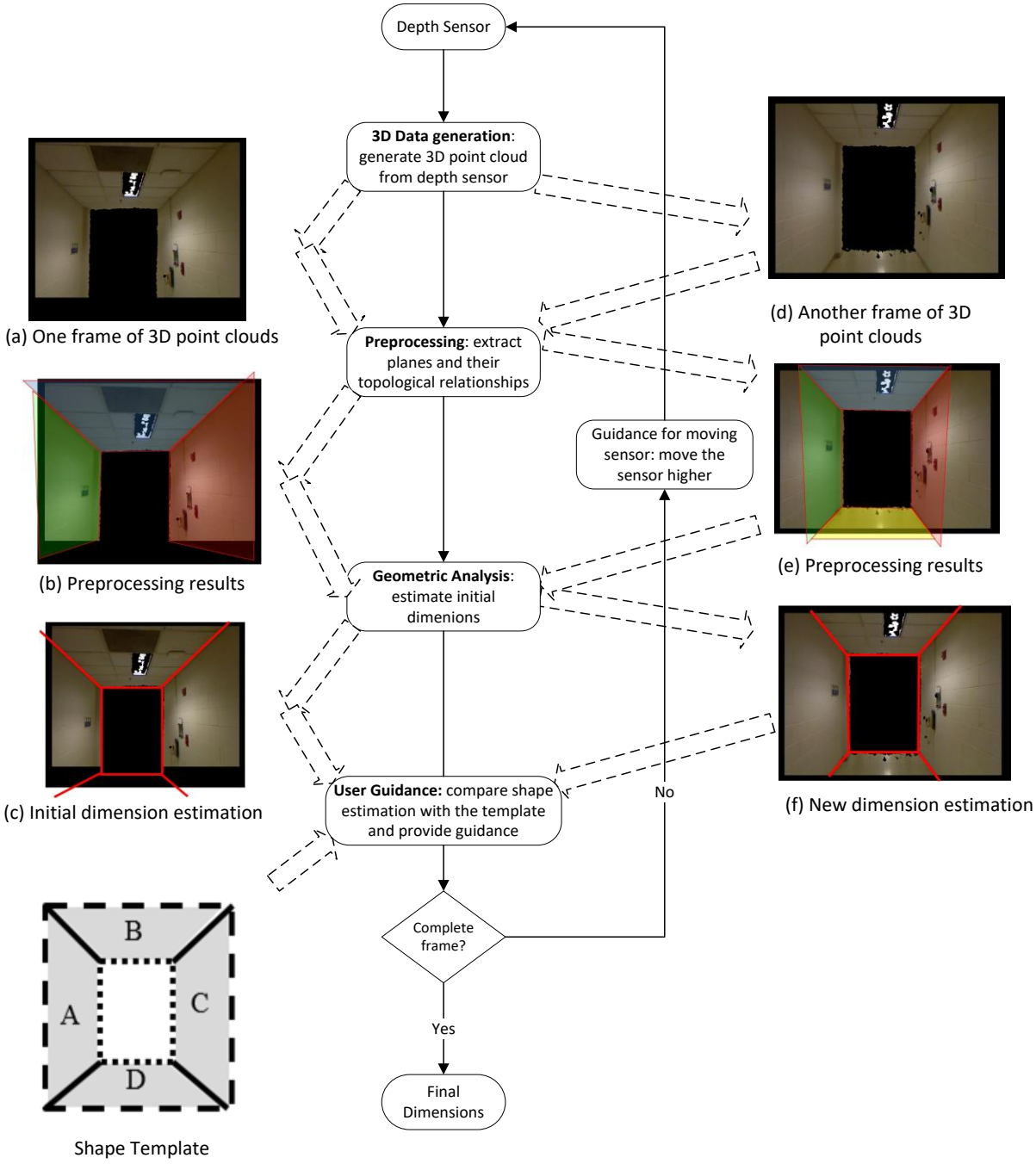


Figure 3-1: Overview of the user-guided dimensional analysis system.

3.3.1 Data Preprocessing

In this chapter, it is assumed that the object of interest is composed of, supported by, or surrounded by planar surfaces. Since the proposed method is intended for dimensional analysis of indoor scenes, this assumption is reasonable as the common objects in indoor scenes have planar surfaces. Based on this assumption, the geometric analysis is performed to obtain dimensional information of specific infrastructure elements.

In order to extract planar surfaces efficiently, the fast plane extraction algorithm for organized point clouds proposed by Feng et al. (2014) is employed. This algorithm first segments the point clouds into groups and uses them as nodes to create a graph. Then, an agglomerative hierarchical clustering is performed on this graph to merge nodes on the same plane. Finally, the planes are refined by pixel-wise region growing.

This chapter focuses on estimating dimensions by utilizing plane topological relationships, which enables us to obtain robust and accurate measurements. Therefore, once all the planes are extracted from the point clouds, the topological relationships among these planes are estimated based on the plane parameters. Four types of plane topological relations of interest are defined as follows:

- Parallel: if the normal vectors of two planes are parallel to each other, the two planes are parallel planes.
- Coplanar: if two planes have the same geometric parameters, they are coplanar planes. Coplanar planes are also parallel planes.
- Intersecting: if two planes are not parallel to each other, they are intersecting planes.
- Perpendicular: if the normal vectors of two planes are perpendicular (orthogonal to each other), the two planes are perpendicular.

It should be noted that due to the uncertainty in sensor measurements, these relationships are approximately ascertained. For example, if the angle of the normal vectors between two planes is less than a specified α degrees, they are considered as parallel planes (α is empirically set as five to avoid classifying non-parallel planes as parallel due to large α or failure in detecting the parallel plane relationship).

3.3.2 Geometric Analysis

If all the measurements from the sensor were perfect, the dimensional information could be directly computed based on the geometric representations of the infrastructure. However, the sensor measurements have uncertainty inevitably and thus the geometric representations estimated from the point clouds are not perfect. In order to get robust and accurate dimensional information, least squares methods are utilized to mitigate measurements uncertainty. In this chapter, based on the scene types and experimental scenarios, the distance between two parallel planes and the distance between boundary points of coplanar planes are of interest. In addition, these two distances are also of interest in general for indoor environments which contain many regular planar surfaces. Methods for these two distance computations are proposed to obtain robust estimation.

3.3.2.1 Distance between Parallel Planes

After extracting the planes, the plane parameters are estimated from the points by least squares. Given the set of points $\boldsymbol{p}_i^k = \left[x_i^k, y_i^k, z_i^k\right], k = 1, \dots, K$ assigned to Plane i, whose parameters are represented by $\boldsymbol{P} = [a_i, b_i, c_i, d_i]^T$, the plane equation $a_i x_i^k + b_i y_i^k + c_i z_i^k + d_i = 0$ needs to be satisfied for all the K points. Thus, a homogeneous system can be constructed as follows

$$\boldsymbol{AP} = 0$$

where the matrix $\boldsymbol{A}$ can be constructed by stacking the row vectors $\left[x_i^k, y_i^k, z_i^k, 1\right]$. In order to get the least squares estimation, one possible solution is to perform singular value decomposition (SVD) (Mandel 1982) on the matrix $\boldsymbol{A}$ and then the plane parameters $\boldsymbol{P}$ can be extracted from the results of SVD. By the SVD theory a $m \times n$ real matrix $\boldsymbol{A}$ can be decomposed as $\boldsymbol{A} = \boldsymbol{U\Sigma V}^T$ where $\boldsymbol{U}$ is a $m \times m$ unitary matrix (i.e. $\boldsymbol{UU}^T = \boldsymbol{I}$), $\boldsymbol{\Sigma}$ is a $m \times n$ diagonal matrix with non-negative values, and $\boldsymbol{V}$ is a $n \times n$ unitary matrix. In order to find a least-squares solution, by imposing the constraints $\|\boldsymbol{P}\| = 1$, the solution aims to minimize $\|\boldsymbol{AP}\|$. As the rank of $\boldsymbol{A}$ is n ($m > n$ for our data), the solution of Equation (3.1) is the last column of $\boldsymbol{V}$.

Since it is assumed that there exist parallel plane sets, the plane parameter estimation results can be made more accurate by using this prior information. Suppose Plane i and Plane j are parallel to each other and the sets of points assigned to these planes are given as $\boldsymbol{p}_i^k, k = 1, \ldots, K$ and $\boldsymbol{p}_j^l, l = 1, \ldots, L$. To enforce the parallel constraint, Plane i and Plane j share the same normal vector and the equations are defined as

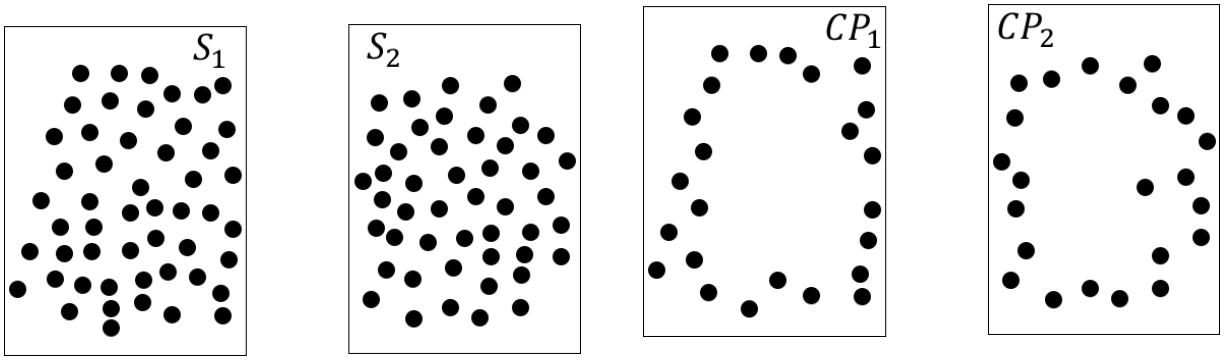
$$\begin{aligned} ax_i^k + by_i^k + bz_i^k + d_i &= 0 \\ ax_j^l + by_j^l + cz_j^l + d_j &= 0 \end{aligned} \tag{3.1}$$

Then a homogenous system similar to Equation (1) can be constructed with $\boldsymbol{P} = \left[a, b, c, d_i, d_j\right]^T$ and the matrix $\boldsymbol{A}$ constructed by stacking $\left[x_i^k, y_i^k, z_i^k, 1,0\right]$ and $\left[x_j^l, y_j^l, z_j^l, 0,1\right]$.

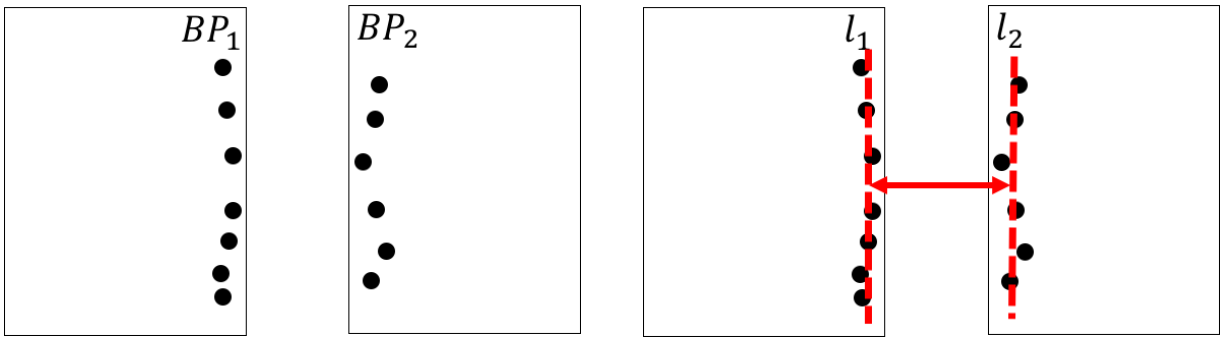
Therefore, by using SVD the plane parameters of parallel planes are computed using all the points on the planes.

Once the plane parameters are obtained, the distance d_{ij} between the parallel planes is calculated directly based on the plane parameters as

$$d_{ij} = \frac{|d_i - d_j|}{\sqrt{a^2 + b^2 + c^2}} \tag{3.2}$$



(a) Coplanar plane points (b) Boundary points of coplanar plane points



(c) Coplanar plane boundary points (d) Line fitting and distance computation

Figure 3-2: Estimation of the distance between two coplanar planes.

3.3.2.2 Distance between Boundary Points of Coplanar Planes

The coplanar planes boundary points refer to boundary points that are located between the two coplanar planes. For example, when measuring the width of the door while the door height is too high to be observed, the points on the wall near the door will be captured as two coplanar planes. To obtain the width of the door, the door frame points are extracted and used as the coplanar planes boundary points. In this context, the door width is the distance between boundary points of two coplanar planes as shown in Figure 3-2 (a).

In order to automatically find door frames, firstly the topological relationships between extracted 3D planar surfaces are estimated based on the plane fitting results. After detecting the coplanar planes, all the coplanar planes are rotated based on the plane parameters to make sure that the normal of the plane is parallel to the new Y axis and all Y values of the rotated points are almost the same. Then the boundary points (Figure 3-2 (b)) of the two planar surface, CP_1 and CP_2, are separately extracted by using the 2D alpha shape algorithm (Bernardini and Bajaj 1997). The 2D alpha shape algorithm moves a circle at a radius of α in the space while the circle must only contain points on its boundary and no points are allowed inside of the circle. Those points that allow the circles are the boundary points extracted by the 2D alpha shape algorithm. Based on the boundary points of each surface, the coplanar planes boundary points BP_1 and BP_2 (Figure 3-2 (c)), are obtained by utilizing a nearest points searching method. Finally, as shown in Figure 3-2 (d) a pair of parallel lines l_1 and l_2 are fitted from BP_1 and BP_2 using the similar method in the previous section. The two 2D lines are defined as

$$\begin{aligned} ax_1^i + bz_1^i + c_1 = 0 \\ ax_2^j + bz_2^j + c_2 = 0 \end{aligned} \tag{3.3}$$

where $[a, b, c_1]$ and $[a, b, c_2]$ are respectively the geometric parameters of the two 2D lines of BP_1 and BP_2, and $\left(x_1^i, z_1^i\right)$ is the i-th point of BP_1 while $\left(x_2^j, z_2^j\right)$ is the j-th point of BP_2. Therefore, a homogeneous system described by Equation 1 can be obtained where $\boldsymbol{P} = [a, b, c_1, c_2]$ and $\boldsymbol{A}$ is constructed by stacking $[x_1^i, z_1^i, 1,0]$ and $[x_2^j, z_2^j, 0,1]$. Based on the geometric parameters of the two 2D lines, the distance d_{12} between the two lines is computed as the following

$$d_{12} = \frac{|c_1 - c_2|}{\sqrt{a^2 + b^2}} \tag{3.4}$$

In this chapter d_{12} is viewed as the distance between the two coplanar boundary points.

In order to automatically extract the coplanar planes boundary points, a nearest point searching method as shown in Algorithm 3-1 is proposed. The boundary points of the two planes, CP_1 and CP_2, are separately extracted and used as input for that algorithm. For the first plane, for each point in CP_1, the nearest point in the second plane boundary points CP_2 is searched (Algorithm 3-1 Lines 7-11). After iterating all the points on the first plane, the points in CP_2 that have been searched as the nearest points, BP_2, belong to the coplanar boundary points from the second plane (Algorithm 3-1 Lines 12-16). By repeating the process for the second plane, the coplanar planes boundary points on the first plane, BP_1, can be also found.

This method utilizes the nearest neighbor search strategy to approximately find the coplanar planes boundary points. Since it employs the boundary points of each plane and the nearest neighbor search, it tends to prefer the point that is located closer to the other plane and thus to find a subset of the true coplanar planes boundary points. However, these points are sufficient for computing the distance between two coplanar planes as they are extracted from the

boundary points of the two planes. In addition, this method utilizes the boundary points of the two coplanar planes, which reduces the computation time.

Algorithm 3-1: Extract Coplanar Planes Boundary Point

```
1   function EXTRACTBOUNDARY(CP_1, CP_2)
2       BP_2 = EXTRACTEACHBOUNDARY (CP_1, CP_2)
3       BP_1 = EXTRACTEACHBOUNDARY (CP_2, CP_1)
4     return (BP_1, BP_2);

5   function EXTRACTEACHBOUNDARY(CP_1, CP_2)
6     is_searched[ 1:size(CP_2)] = false;
7     for each pt ∈ CP_1 do
8       // Search the nearest point to pt in CP_2
9       k = search_nearest_point(CP_2, pt);
10      is_searched[k] = true;
11    end for
12    for each i=1:size(CP_2) do
13      if is_searched[i] = true then
14        BP.add(CP_2 [i] );
15      end if
16    end for
17    return BP
```

3.4 User Guidance

The goal of the user guidance system is to generate instructions for moving the sensor to poses where the sensor can capture complete frames that contain all necessary elements of the scene and yield accurate and robust measurements. In this chapter, a complete frame denotes a single frame that includes all necessary components of the infrastructure features of interest. For example, a complete frame for a typical hallway contains the ground floor, the ceiling, and the two walls. The user guidance utilizes the prior knowledge of the scene, i.e. the scene type (box shape, opening structure, or parallel structure), the gravity direction, the shape template (which contains the topological relations between planar surfaces of the scene), etc., to identify whether a complete frame is captured by visualizing and checking the topological relations of planar surfaces.

Before using the sensor to collect data, it is assumed that the scene type is chosen by the user and there exists corresponding geometric and topological information of planar components. For a single frame, the system tries to identify the components of a scene and recover a hypothesis shape which is used for generating the user guidance for moving the sensor (Algorithm 3-2 Line 3). The system checks the completeness of the current frame by comparing the shape template and the hypothesis shape (Algorithm 3-2 Line 4). In order to generate quantitative guidance for the user, the user guidance system utilizes some of the sensor poses that are able to observe complete frames as baseline sensor poses. When an incomplete frame is obtained, by comparing the current sensor pose with the baseline sensor poses (Algorithm 3-2 Line 6), the user guidance proposes quantitative movement suggestions of the sensor to the user. The generated guidance describes the sensor movement suggestions in terms of translation and rotation of the sensors with respect to the default sensor coordinate system. In the text, for the

sake of illustration, simple cases of user guidance are used and the user guidance is described in words that are more user-friendly for human users. The user guidance generation stops if a complete frame is observed. The detailed user guidance system will be described for three general cases - box shape, opening structure, and parallel structure.

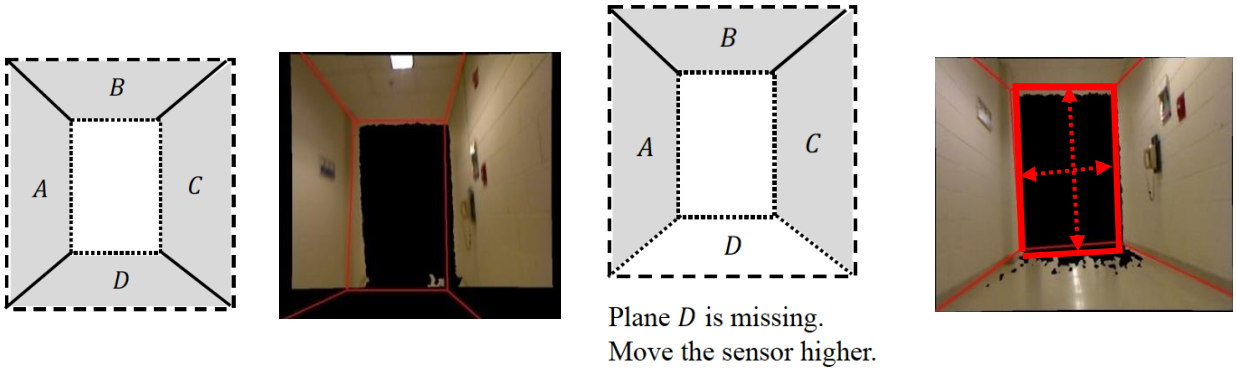
Algorithm 3-2: Generate user guidance for a single frame

```
1  function GENERATEUSERGUIDANCE(frame,gravity, baselineSensorPoses)
2     template =  GetSceneTemplate(sceneType)
3     shape = GeneateHypothesisShape (frame, template, gravity)
4     isComplete = Compare(template, shaple)
5     if isComplete == false do
6           userGuidance = ComputeGuidance(baselineSensorPoses)
7     end if
8  return userGuidance
```

3.4.1 Box Shape

A box shape is defined as the shape that contains two sets of two parallel planes while the two sets are perpendicular to each other. As shown in Figure 3-3 (a), Plane A and C are parallel to each other, so are Plane B and D. Moreover, Plane A is perpendicular to Plane D. The solid lines in Figure 3-3 (a) denote the intersection lines between two intersection planar surfaces. A typical example of a box shape is a hallway in indoor scenes and this chapter uses a hallway as an example to illustrate the method. To get the dimension of this structure (the width and height of the hallway), the points from all the four planes ($A, B, C,$ and D) should be observed by the sensor. Therefore, the sensor at the baseline poses should be in the center of the hallway and almost horizontal with its view direction parallel to Plane A. When the sensor acquires an incomplete frame which does not contain data from all the four planes, the user guidance will

identify the incompletion of the frame and provide user guidance for moving the sensor to capture sufficient points from all the four planar surfaces.



(a) The box shape template. (b) One frame data and the measurement. (c) The shape and the user guidance. (d) A better frame data and the measurement.

Figure 3-3: Box shape user guidance.

Since a typical hallway (composed of two walls, a ceiling, and a ground floor) is usually 2~3 meters high, an RGB-D sensor like Kinect is able to capture points from at least three planes of that hallway. When the sensor is too high away or too close to the ground floor, the ceiling or the ground floor cannot be observed by the sensor. If one planar surface is not obtained in the data, the geometric analysis is performed based on the partial data. Based on the prior information (i.e. the scene type, the related shape template and baseline sensor poses) and the captured data, the hypothesized shape is reconstructed to evaluate the completeness of this frame so as to guide the user.

Figure 3-3 shows an example of the user guidance for a box shape. Figure 3-3 (a) displays a priori knowledge about the box shape template, where gray shapes denote planar surfaces. This shape template also contains geometric and topological information of all the four

planes. As shown in Figure 3-3 (b), Plane D (i.e., the ground floor) is not detected in the data because it is too close to the sensor (closer than the minimum measurement distance of the sensor). Based on the observed planes and the shape template, the user guidance system generates a hypothesis box shape from that frame. Since the ceiling and the two walls are measured in the data, the intersection lines between the three planar surfaces can be derived, as denoted by the two solid lines (in fact horizontal) in Figure 3-3 (c). By vertically extending the end points (which are computed according to the line equation and the measured point clouds) of the two solid lines, the two vertical dotted lines are hypothesized and the other end points of the dotted lines are found based on their equations and the point clouds. The last two dotted lines are created by extending the end points while keeping it parallel to the two horizontal solid lines. Hence, the box shape (the red lines in Figure 3-3 (b)) is constructed for this frame and an abstract template (Figure 3-3 (c)) is also created. However, the height is not accurate since it is computed by hypothesizing the vertical dotted lines and their end points. By comparing the shape in Figure 3-3 (c) and the shape template in Figure 3-3 (a), the system identifies the fact that Plane D is not observed and then the user guidance system compares the current sensor pose with baseline sensor poses of the box shape and generates guidance for the user to move the sensor higher to obtain the accurate height.

Since the system detects that there are no points from Plane D, the system instructs the user to move the sensor higher in order to get points from Plane D, the floor. By following the guidance, the sensor is moved higher and then a new and better frame is obtained as shown in Figure 3-3 (d). In this frame, all the four planes can be extracted from the point clouds and a box shape similar to the template can be constructed without using any hypothesis. Thus, both the height and the width of the hallway can be computed by geometric analysis.

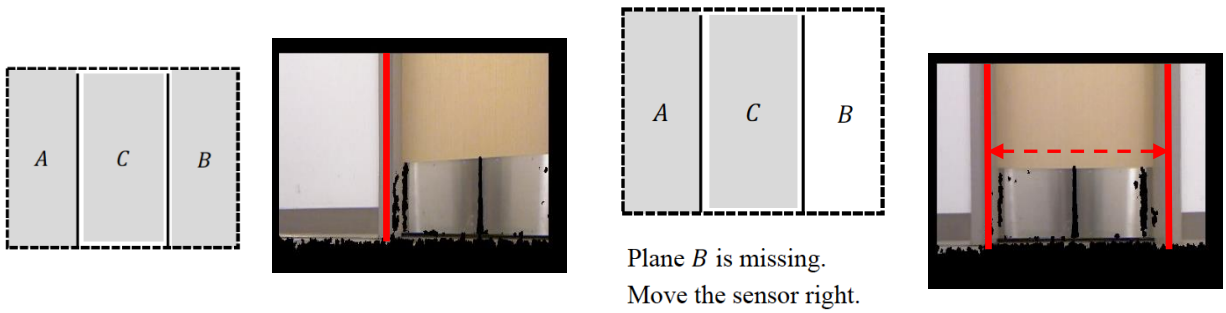
It should be noted that by assuming the sensor is held almost horizontally, even though only one plane is observed, the user guidance system is still able to generate user guidance for moving the sensor to find complete frames. For example, if Plane A is observed, based on the sensor pose assumption, the user guidance system will identify that at least a wall is captured and provide guidance for moving or rotating the sensor right or left to capture more data. Similarly, if two planes are captured by the sensor, the user guidance system works well too.

3.4.2 Opening Structure

An opening structure is defined as an opening in a planar surface, i.e., a rectangular hole within a planar surface. In this chapter, a door frame that is indented in a wall is used as an example of an opening structure. Since most doors in this chapter are located in the hallways, it is difficult to obtain both its width and height as the sensors cannot move as far from the door as possible when it is facing the door. Therefore, this chapter currently focuses on estimating the width of a door. As shown in Fig. 4 (a), Plane A and Plane B are vertical walls and they are on the same plane (their topological relation is coplanar). In order to get accurate width of the opening, the two planes A and B are necessary to provide constraints to reconstruct the shape of the opening. Thus, the user guidance is implemented to ensure that the two planes are observed by the sensor at an optimal pose, where the sensor at the baseline poses is almost horizontal and its view direction is orthogonal to Plane A and B, and moreover, it is close to the center of the opening.

Figure 3-4 displays an example of the user guidance for an opening shape. The opening shape template is shown in Figure 3-4 (a), where gray shapes denote planar surfaces and solid lines are components of the shape. For example, if Plane B is not captured in the data, a candidate wall is identified as follows: first the centroids of Plane A and Plane C are projected

onto a line that passes the sensor position and is perpendicular to both Plane A and C; since the projected point of the wall should be closer to the sensor compared to that of a door, Plane A is detected as a candidate wall. By assuming the door width, the system can still reconstruct an estimated shape as shown in Figure 3-4 (b). Here the vertical solid line is estimated by fitting a line using the boundary points between the two parallel surfaces, while the vertical dashed line is hypothesized from the door width assumption. By comparing Figure 3-4 (c) and (a), the user guidance system identifies that another wall, i.e., Plane B, is missing in the current frame. Therefore, using the baseline poses, the system instructs the user to move the sensor right so that the data of Plane B can be observed by the sensor. In this way, a new frame with better quality data that contains Plane A and B is obtained (Figure 3-4 (d)). Thus, the door width is computed using the method for estimating the distance between boundary points of coplanar planes.



(a) The opening shape template.

(b) One frame data and the measurement.

(c) The shape and the user guidance.

(d) A better frame data and the measurement.

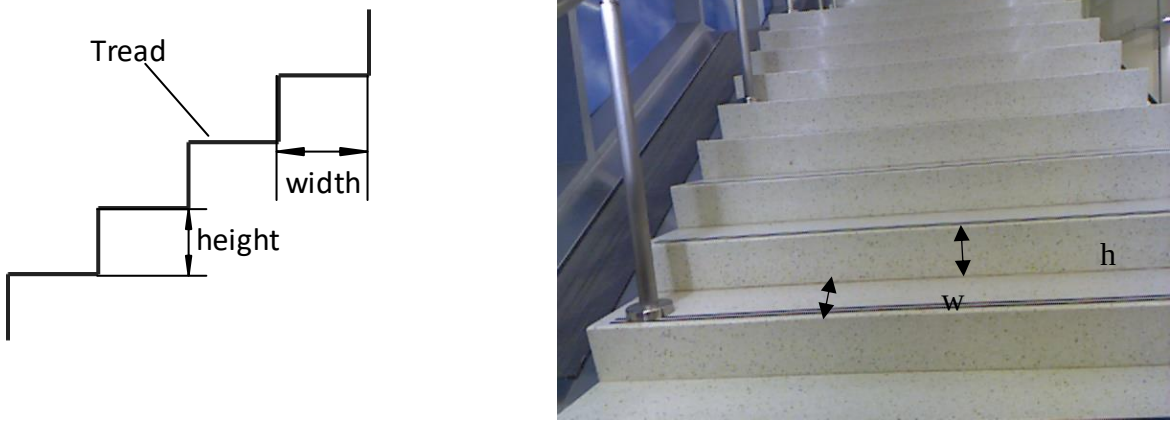
Figure 3-4: Opening shape user guidance.

It should be noted that for the simplicity of illustration, in Figure 3-4 only translation related user guidance is discussed. In fact, by comparing with the baseline poses, the user guidance also produces sensor movement guidance in terms of orientation. For example, using the incomplete frame in Figure 3-4 (b) as an example, the normal vector $\boldsymbol{n}_C$ of the candidate door

plane, i.e. Plane C , is on the right of the view direction $\boldsymbol{v}_{cam}$ if $\boldsymbol{n}_C$ and $\boldsymbol{v}_{cam}$ are both pointing to the door. To make and $\boldsymbol{n}_C$ and $\boldsymbol{v}_{cam}$ point to the same direction, the sensor should be rotated right in order to approach the baseline poses for observing complete frames.

3.4.3 Parallel Structure

A parallel structure is composed of multiple parallel planes. In this chapter, the stair is used for an example of parallel structures. The critical dimensions of a parallel structure are the distances between parallel planes. For stairs, interesting dimensions are defined as follows (Figure 3-5): the width is defined as the distance between two consecutive vertical planes and the height is defined as the distance between two consecutive horizontal planes.



(a) The definition of stair width and height in a side view of a stair.

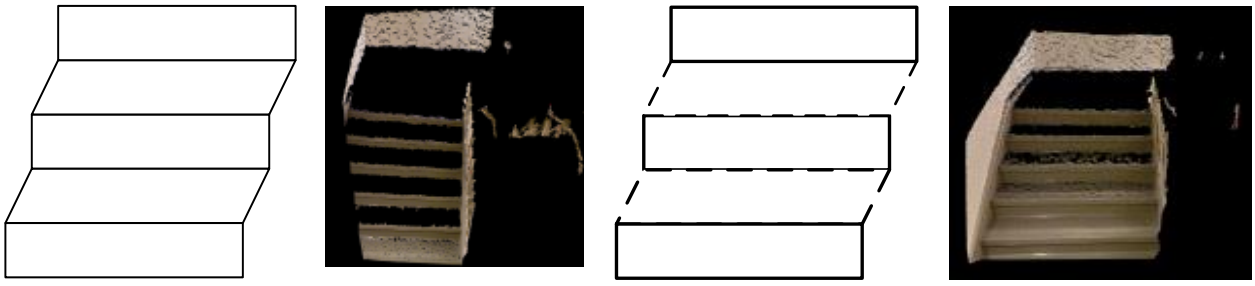
(b) The definition of stair width and height in a typical stair.

Figure 3-5: Stair dimensions.

There are two reasons for using parallel planes in the dimension definition. Firstly, for most applications in robotics and civil engineering, the dimensions defined in this way are sufficient even though stairs usually contain some protruding parts, for example, stair nosing (the

protruding part of a tread), and bump to avoid slipperiness on the tread. Secondly, from a practical perspective, it is complicated to fit a perfect rectangle for point clouds since the sensors usually fail to obtain all points of edges. Moreover, the methods utilizing least squares estimation to fit a rectangle to point clouds are inclined to obtain a smaller rectangle compared to the ground truth. Therefore, we use the distance between two parallel planes to define the dimensions of interest and this definition also provides hints for the subsequent user guidance.

In order to obtain the width and height of the stairs, two sets of parallel planar surfaces must be presented in the point clouds. Since the width and height of a stair are close to each other, the sensor is able to get sufficient points from both horizontal and vertical planes if its view direction is around 45 degrees with respect to both the horizontal and vertical planes of stairs. Based on this principle, the user guidance system estimates its orientation with respect to the stairs and then provides corresponding instructions for moving the sensor to get more vertical and horizontal planes.



(a) The stair template. (b) One frame data (c) The shape (d) A better frame data

Figure 3-6: Parallel structure user guidance.

Figure 3-6 shows an example of the user guidance system for the stair. The template of the stair is shown in Figure 3-6 (a). In Figure 3-6 (b) several vertical planar surfaces and a

horizontal planar surface are observed in that frame. In this case, only the width of the stair is able to be computed from the vertical surfaces by the geometric analysis. Based on the assumption of the shape template (Fig. 6 (a)) and the frame data, the height of the stair can be approximately computed by estimating the height of several vertical planar surfaces. Thus, as shown in Figure 3-6 (c), an instantiation of the shape template based on the data is derived from the frame data while the width has a good accuracy and the height has a poor accuracy. Based on Figure 3-6 (c) and the relative position and orientation of the sensor with respect to the stairs, the user guidance system provides guidance for moving the sensor to a better position with a better orientation. In this case, the sensor should be rotated toward the ground in order to obtain more points from the horizontal planes (Figure 3-6 (d)).

3.5 Experiments and Results

3.5.1 Experimental Setup and Sensor Calibration

In the conducted experiments, a Kinect for Xbox 360 sensor is used as the RGB-D sensor to obtain 3D point clouds of indoor scenes. This sensor can capture images with a resolution of 640x480 and work at a frame rate of 30 fps. The suggested operation range of this sensor is 0.8 to 5.0 meters and the depth resolution decreases quadratically with increasing distance from the sensor (approximately 7cm at the range of 5m) (Khoshelham and Elberink 2012).

The RGB-D camera has an infrared (IR) camera and a color (RGB) camera. With the assistance of an IR laser emitter, the IR camera is able to get a depth image of the environment. Meanwhile, the RGB camera is able to capture a color image. By using the intrinsic parameters of the two cameras and the relative transformation between the two cameras, the colored 3D point clouds can be computed from the color image and the depth image. When the Kinect

sensor is factory-assembled, the IR sensor and the RGB camera are fixed relative to each other and thus there exist default parameters for the two cameras, including the intrinsic parameters and their transformation matrix. However, due to imperfections in the manufacturing process, these default parameters cannot be expected to be exact for all Kinect sensors. Therefore, it is necessary to calibrate the Kinect sensor if it is used for applications that require high and repeatable accuracy. The sensor calibration in this chapter aims to obtain intrinsic and extrinsic parameters of the Kinect sensor and thus obtain accurate 3D colored point clouds from the sensor. By viewing the Kinect as a stereo system, a stereo camera calibration method is utilized to calibrate the Kinect and obtain its intrinsic parameters, and the extrinsic parameters between its IR camera and RGB camera.

The sensor is calibrated before gathering data. During the calibration, the IR emitter is covered by an opaque object and thus the IR sensor can obtain intensity instead of depth. To enable the IR sensor to capture a bright image, a lamp is used to provide more illumination for the calibration markers. In addition, to enable higher marker detection results, a fiducial marker system based on AprilTags (Olson 2011) is used instead of traditional checkboard for calibration. Based on multiple pairs of images by the IR sensor and the RGB sensor, the calibration obtains the parameters of the stereo system.

To fully utilize the knowledge of the measured indoor environment, during the experiments the sensor should be held almost horizontally to the extent possible by the user, which ensures that the gravity direction is consistent with the assumption used in recognizing components of scenes. The sensor can be tilted a little bit as a tolerance ($\pm 15^{o}$ within the desired gravity direction) is added to check the gravity direction. Within this context, for a hallway, the floor is almost horizontal while the wall is almost vertical in the point clouds. This assumption is

reasonable in terms of the potential applications. For a robotic platform, it is easy to mount the sensor in this position. For a user holding the device, the sensor can be easily adjusted to meet this assumption.

Regarding the user guidance, the scene type, i.e. box shape, opening structure, or parallel structure, is selected by the user. The user guidance utilizes the shape template of the scene and geometric analysis to identify these planar components and the completeness of the frame. If the frame is incomplete, the system will generate user guidance and prompt it in the command window for the user. The correctness of the generated user guidance is highly dependent on the geometric analysis results, especially the components detection results for the specific scene type. For example, when observing the door, if one frame only contains the partial data from a cuboid recycle bin and the wall, the system will identify the wall as a candidate door while viewing the recycle bin as a candidate wall. In this context, the user guidance provided by the sensor will not be able to help find the correct door. In summary, for the current implementation, if the observed scene matches the designated shape template and the components identification is correct, the system can generate correct user guidance.

3.5.2 Average Geometric Measurement Accuracy

To evaluate the geometric measurement accuracy, multiple complete frames are acquired by moving the sensor to different positions in order to obtain data at different viewpoints. The average values over all the measurements from those complete frames are used to demonstrate the accuracy and performance of the sensor in estimating the dimensions. The ground truth of the dimensional information is obtained using a tape measure by a carpenter having ten years of construction experience. The error of this system is calculated by subtracting the average value from the ground truth.

In terms of a hallway structure, the method is tested on ten hallways in four different buildings. The overall accuracy of the widths and the heights of the hallways is shown in Table 3-1. The mean absolute error of the width measurement is 22mm while that of the height is 36mm. Considering the accuracy of the Kinect sensor, it can be concluded that this method is able to obtain accurate hallway width and height. The standard deviations of the absolute errors of the width and height measurements are 15mm and 24mm respectively. As shown in Table 3-1, the width measurement usually has a lower error and relative error compared to the height and moreover, the standard deviation of the width measurement is smaller than that of the height measurement. The reason is that the width of a hallway is usually less than its height and Kinect tends to obtain low-quality data from the ceiling or the floor because the uncertainty of the sensor goes up as the distance increases.

Table 3-1: Absolute errors and relative errors of hallway dimensions.

ID	Error (mm)		Relative Error	
	Width	Height	Width	Height
Hallway 1	32	15	1.79%	0.59%
Hallway 2	33	55	1.81%	2.26%
Hallway 3	23	77	0.94%	2.65%
Hallway 4	48	27	1.96%	1.10%
Hallway 5	24	41	0.99%	1.67%
Hallway 6	4	15	0.16%	0.57%
Hallway 7	1	3	0.05%	0.11%
Hallway 8	32	50	1.33%	1.82%
Hallway 9	17	68	1.12%	1.98%
Hallway 10	4	12	0.20%	0.49%
Avg.	22	36	1.04%	1.32%
Std.	15	24	0.67%	0.82%

Table 3-2: Absolute errors and relative errors of door width.

ID	Error (mm)	Relative Error
Door 1	9	0.98%
Door 2	39	4.25%
Door 3	4	0.38%
Door 4	5	0.55%
Door 5	19	2.08%
Door 6	11	1.20%
Door 7	41	4.50%
Door 8	20	2.19%
Door 9	5	0.55%
Door 10	2	0.22%
Avg.	16	1.69%
Std.	14	1.49%

Table 3-3: Absolute errors and relative errors of stair dimensions.

ID	Error (mm)		Relative Error	
	Width	Height	Width	Height
Stair 1	4	11	1.43%	5.97%
Stair 2	6	24	2.12%	13.03%
Stair 3	2	10	0.68%	6.18%
Stair 4	15	28	5.91%	14.30%
Stair 5	0	5	0	2.72%
Stair 6	1	14	0.36%	7.41%
Stair 7	5	4	1.62%	2.25%
Stair 8	2	11	0.65%	6.42%
Stair 9	10	4	3.48%	2.22%
Stair 10	29	1	11.30%	0.55%
Avg.	7	11	2.76%	6.13%
Std.	8	8	1.62%	4.30%

For door frames, the method is tested on ten door frames in different buildings. The overall accuracy of the width of doors is shown in Table 3-2. The mean absolute error of the door width measurements is 16mm, which shows that the method measures door width with high accuracy. The standard deviation of the absolute errors is 14mm, which reflects the stability of this method in measuring door width.

For stairs, the method is tested on ten stairs in different buildings. The mean absolute errors of the width and the height of these ten stairs are 4mm and 15mm respectively while the standard deviations are 4mm and 9mm as shown in Table 3-3. Compared to the accuracy of Kinect, these errors demonstrate that using parallel planes to compute dimensions is able to get an accurate and stable estimation. In addition, compared to the dimension measurements of hallways and doors, the stair dimension measurements have a lower mean absolute error and standard deviation. This is partly due to the fact that the stair width and height estimated from a single frame are usually computed using multiple planes while the width and height of a hallway and the width of a door are estimated using two planes from a single frame.

Even though the mean absolute errors of the stair height and width are lower than those of hallway and door dimensions, both the relative errors of the stair width and height (2.67% ad 6.13%) are larger than those of the hallway and door dimension. This is mainly because the absolute values of the stair height (~180mm) and width (~300mm) are smaller compared to door width (~1,000mm), hallway width (~2,000mm), and hallway height (~2,500mm).

The developed methods are implemented in C++. The Point Cloud Library (PCL 2016) is utilized for capturing 3D point clouds from the Kinect sensor. The Computation Geometry Algorithms Library (CGAL 2016) is used for geometry computation. For the three cases, hallway, door, stairs, the average frame processing time are 0.03s, 0.8s, and 0.07s respectively.

The experiments were conducted on a desktop with Intel Core i7-4790K CPU of 4.00GHz and RAM of 16GB. The implementation does not employ any multi-threading or GPU techniques. The door frame takes longer time because many geometric operations (e.g. boundary extraction) are performed in data analysis. However, using multiple threading techniques, the processing time can be improved and thus the system will be feasible for real-time applications.

3.5.3 Relations between Sensor Poses and Dimension Measurements

To obtain complete frames of a scene, sensor poses (orientations and positions) have many options. This section will evaluate relations between sensor poses and the accuracy of dimension measurements for the three scenes. As aforementioned, the user guidance system generates instructions about moving the sensor's position and orientation. The hallway which has larger dimensions compared with the other two is used to evaluate the effect of sensor positions on the dimension measurement errors while the stairs to evaluate the effect of sensor orientations on the dimension measurement errors.

In terms of the hallway case, as the height of a hallway is usually larger than the width, we primarily evaluated the errors of height measurements corresponding to positions of the sensor by only varying the height of the sensor. We held the sensor horizontally, and vertically moved the sensor from a position close to the ground floor to a position close to the ceiling. Thus, the sensor poses in this way are assumed to have only variations in height. To obtain the relations between the sensor position and the error of the height measurement, the absolute distance difference d^* between the distance from the sensor to the ground floor and that from the sensor to the ceiling is computed. If the sensor is near the center of the hallway, the absolute difference d^* is near zero. On the contrary, if the sensor is close to the ground floor or the ceiling, d^* is larger and approaches the height of the hallway. For this hallway whose height is

2.91 meters, when the absolute difference d^* is larger than 2.6 meters, the sensor cannot observe any complete frames.

As shown in Figure 3-7, the average absolute error of height measurement increases as the absolute distance difference d^*. This is partly due to the fact that when the sensor is far away from the hallway center, it captures many lower quality points that are far away from the sensor. For example, when the sensor is close to the ground and far from the ceilings, the ceiling points will have larger uncertainty compared to the ground points. These points with large uncertainty might lead to large errors in dimension measurements. In addition, as shown in Figure 3-7 when the absolute distance difference d^* is less than 2 meters, the absolute error of height measurement has less variance compared to that within 2 and 2.5 meters. Thus, it is concluded that when the sensor is close to the center of a hallway it tends to provide robust and accurate hallway height estimation. However, it should be noted that if the sensor is located away from the center of a hallway, it does not necessarily indicate that higher accuracy dimensional measurements cannot be obtained. For example, when the absolute distance difference d^* is within 2 and 2.5 meters some of the absolute errors of height measurement are pretty accurate (less than 10mm). This is because the dimension measurements are computed using least squares estimation which mitigates large uncertainty of some points. Therefore, even though some points have higher uncertainty (which is still centimeter level), they do not dominate the least squares estimation results, i.e. dimension measurements in this chapter. This is also indicated in Figure 3-7 by the fact that the average height measurement errors of different ranges are within 20mm.

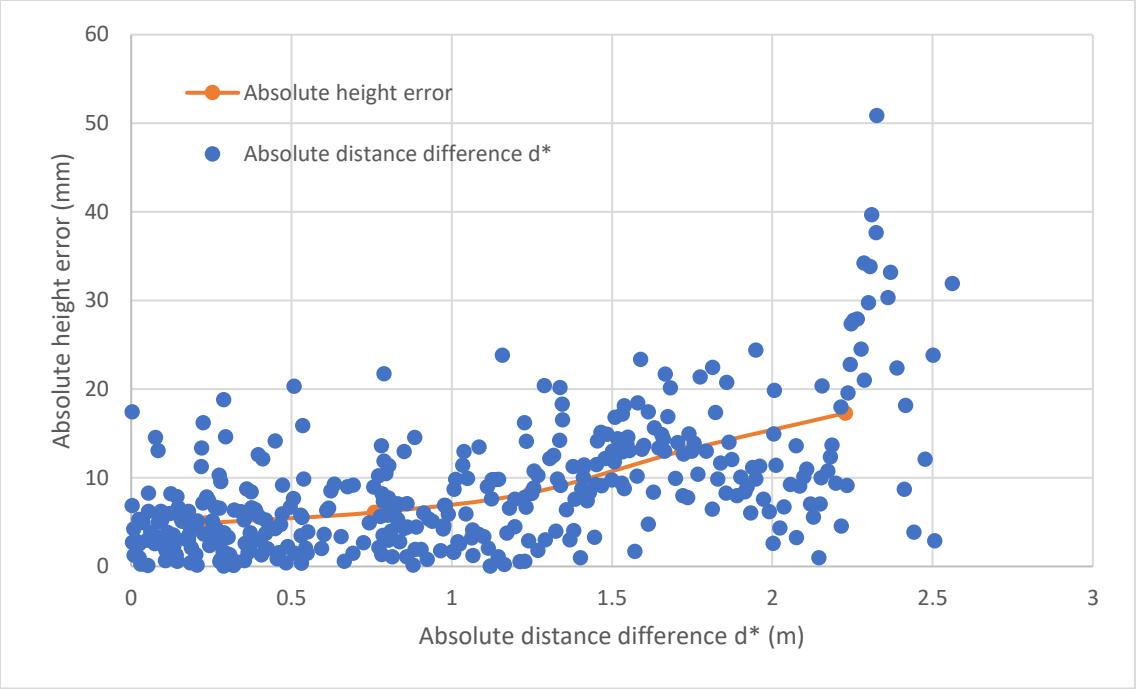


Figure 3-7: Absolute error of hallway height with respect to absolute distance difference d^*.

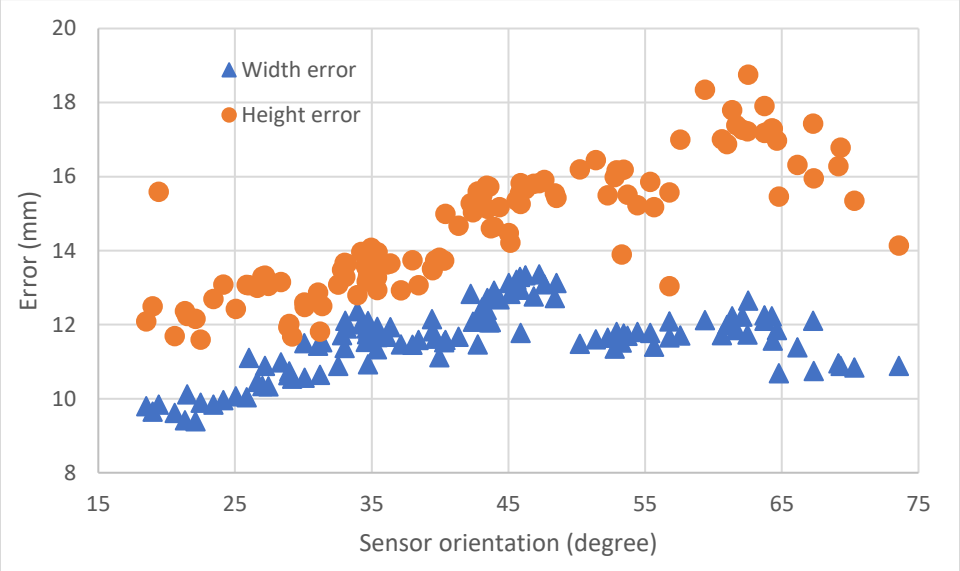


Figure 3-8: Error of dimensions with respect to sensor orientations for stairs.

For stairs, we collected data by only varying the view direction of the sensor to evaluate whether the sensor's orientation can improve the accuracy of dimensional measurements. We manually held the sensor horizontally and then rotated the sensor to change its view direction. From this dataset, the complete frames are extracted and their errors against the sensor orientation are shown in Figure 3-8. To obtain complete frames, the sensor's orientation with respect to the horizontal surfaces of the stairs should be greater than 18^{o} and less than 73^{o}. The results show that the errors are within 20mm and the dimension measurements have similar errors given that the error of Kinect point measurement is also on the order of centimeters. In addition, Figure 3-8 demonstrates that the sensor orientation does not significantly affect errors of dimension measurements. This is also due to the sensor uncertainty and the dimension estimation method.

3.6 Conclusions and Future Work

In this chapter, a user-guided dimensional analysis method for indoor building environments is introduced. The system uses a single frame from an RGB-D sensor to obtain the dimensions of an indoor scene by extracting planes and performing the geometric analysis. To overcome the disadvantage of the single frame data, a user guidance strategy is employed to provide guidance for better sensor poses in order to acquire complete data frames. Experimental results show that this method can obtain accurate dimensions of hallways, doors, and stairs with centimeters error. The user guidance system is able to provide useful guidance for moving the sensor to obtain complete frames. The experimental results also demonstrate that due to the uncertainty magnitude of the sensor and the dimension estimation method, when complete frames are captured the sensor poses have little effect on dimension measurements accuracy. Since the current user-guidance system only guides the user to obtain complete frames, future

work will explore how to systematically investigate the relations between various sensor poses and the dimension measurement accuracy in order to generate guidance for better frames in terms of high accuracy dimension measurements.

Due to the sensor, i.e. RGB-D cameras, used in the experiments, this research has two main limitations. Firstly, the RGB-D sensors do not function well in outdoor environments because the ambient IR affects the functionality of the IR sensor. Secondly, the RGB-D cameras have limited precision which is not sufficient for some construction tasks that require high precision especially during the construction phase (e.g., door installation). To overcome these two limitations, different sensors can be utilized to replace RGB-D cameras. For example, a stereo camera system can be used to get 3D point cloud in both indoor and outdoor environments. For those applications requiring high accuracy, more accurate sensors (e.g., laser scanners) can be adopted to acquire 3D point clouds. When different sensors are used, if the scene is the same, some minor changes need to be made according to the property of sensors. Future work will investigate using stereo camera systems for measuring dimensions of civil infrastructure elements in both indoor and outdoor environments. Another limitation of this chapter is that it can be only applied to infrastructure elements composed of planar surfaces. Future work will explore the design of corresponding geometric analysis and user guidance system for scenes containing non-planar surfaces.

Chapter 4

Human Detection and Tracking from a Single RGB-D Sensor

4.1 Introduction

Human detection and tracking allow a robot to be aware of a specific individual for performing various tasks, e.g., tour guiding, elderly care, surveillance and so forth (Bodor and Jackson 2003; Nez et al. 2016). The capability of identifying and tracking a certain human also allows a robot to interact or collaborate with that human in dynamic environments, and enables further applications (e.g., human following) (Chung et al. 2012; Dang and Suh 2011; Morioka et al. 2004). Various instances of previous research work have primarily explored the use of RGB or gray cameras to detect and track a certain person or multiple people by using visual appearances (Ghidary et al. 2000; Zarka et al. 2008; Zhou and Hoang 2005). However, the efficacy of these methods is affected by the illumination conditions, complicated background, and occlusion. Moreover, an RGB camera cannot obtain 3D information of the moving humans which can provide many significant features for human detection and tracking. Therefore, 3D sensors (e.g., stereo cameras, laser range finders, and RGB-D sensors) have been employed to obtain 3D data for human detection and tracking (Ali et al. 2013; Chung et al. 2012; Gritti et al. 2014).

Among the 3D sensors, the RGB-D sensors that can acquire organized color point clouds in indoor environments at frame rates of up to 30Hz have been proven to be useful for human

detection and tracking. As the RGB-D cameras have a limited field of view and sensing ranges, most of the human detection and tracking methods using RGB-D cameras integrate data captured from other sensors (e.g., laser range finders) (Susperregi et al. 2013), or utilize multiple RGB-D cameras to capture additional data (Nez et al. 2016). The methods using only a single RGB-D sensor highly rely on the existence of the ground plane which might not be realistic when a few number of points from the ground are observed due to occlusion or the sensor is too close or too far from the ground (Liu et al. 2015; Munaro and Menegatti 2014). In this chapter, a novel human tracking method is proposed to detect and track a specific individual from a single RGB-D sensor using online learning methods.

The rest of this chapter is organized as follows. Section 4.2 *Previous Work* reviews previous work on human detection and tracking using RGB-D sensors. Section 4.3 *Methodology* explains the proposed human tracking method in detail. Section 4.4 *Experimental Results and Discussion* shows the experimental results on the real-world datasets and discusses the performance of the proposed method. Section 4.5 *Conclusions and Future Work* draws the conclusion of this chapter and discusses the limitations and future work.

4.2 Previous Work

Human detection and tracking has been extensively investigated using various sensors, e.g., RGB cameras, thermal cameras, stereo cameras, RGB-D sensors, laser range finders, and so on. As this work mainly utilized RGB and 3D features from RGB-D images, this section first briefly discusses common methods for using images and 3D data to detect and track humans, and then reviews previous literature that utilized RGB-D sensors for human detection and tracking.

Regarding utilizing RGB or gray images in human detection and tracking, most of the methods extracted features from images and employed classifiers to detect or recognize parts of

the body, such as face (Suzuki et al. 2009), head (Xu et al. 2015), entire body (Dalal and Triggs 2005), and so forth. For a laser range finder (i.e., a 2D laser scanner), the common approach is to detect human legs from the point cloud (Chung et al. 2012). In terms of using the stereo cameras or RGB-D cameras which are able to obtain both RGB images and 3D point clouds, there exist three types of approaches: (1) 2D approach that mainly detects humans from RGB or depth images using image processing methods (Vo et al. 2014), (2) 3D approach that mainly detects humans from the 3D point clouds (Liu et al. 2015), (3) integration of the 2D and 3D approach (Munaro and Menegatti 2014). This section only reviews methods that utilized 3D data while ignoring the first category.

Munaro and Menegatti (2014) proposed a fast multi-people detection and tracking method using an RGB-D sensor. They downsampled the point cloud, extracted the ground plane, performed clustering in 3D space vertically, and detected multiple people by a Histogram of Gradients (HOG) people detector (Dalal and Triggs 2005) from the corresponding parts in the RGB image. The tracker utilized the online classifier based on Adaboost using the color histogram. The method can track multiple people with state-of-art accuracy and beyond state-of-the-art speed. However, this approach highly relies on the detection of the ground plane which might not be visible due to occlusion or the sensor positions.

Carraro et al. (2016) developed a cost-efficient human detection and tracking method using Kinect v2 with an embedded system, the NVidia Jetson TK1. Taking advantage of the embedded system, the method can generate a point cloud at 22 Hz and detect people at 14 Hz. Liu et al. (2015) utilized a novel point ensemble image (PEI) representation after the ground plane was detected. Based on this representation, a head crown detector was used to identify people candidates, and the histograms and the height statistics were utilized to detect people. The

method achieved high detection rates (more than 95%) while having a frame rate of 30~50 Hz. This approach faces the same problem of using the ground plane assumption.

Liu et al. (2016) proposed the new idea of spatial region of interest plan view maps for identifying human candidates after the ground plane was removed. A particle filter was adopted to track the motion models of multiple people. The method achieved high multiple object tracking (MOT) accuracies on two indoor datasets. However, it was not evaluated on public datasets, and the computational efficiency was not discussed. The methods that utilized HOG descriptors might fail when people squat down or are blocked by other objects (Liu et al. 2016).

Apart from a single RGB-D sensor, sensor fusions with other types of sensors (e.g., thermal cameras) or multiple RGB-D sensors can contribute data from different perspectives or containing different attributes for people detection and tracking. Susperregi et al. (2013) integrated an RGB-D sensor, a laser range finder and a thermal sensor on a mobile robotic platform to perform human detection and tracking. They proposed three independent detection methods based on the data captured by the three sensors and fused them into the particle filter system. A vest detection method was implemented from the RGB-D images; a leg detection method was designed using the laser range finder data; and the thermal detection identified possible human regions. This system can detect and track the target human with a safety vest robustly and accurately. However, this method requires the human to wear a safety vest in order to perform the vest detection in the RGB-D data.

Munaro et al. (2016) extended their previous work and developed an open source multi-camera calibration and people tracking method using RGB-D camera networks. The methods for human detection and tracking utilized the approach in (Munaro and Menegatti 2014) while allowing fusion of multiple RGB-D sensors. Luber et al. (2011) integrated two detectors, a novel

multi-cue person detector from RGB-D data and an on-line detector, into a multi-hypothesis tracker to perform people tracking without a ground plane assumption. The people detector was trained by extracting features (similar to HOG) from both the depth and RGB images while the tracker fused the online Boosting method into a Kalman filter based multi-hypothesis tracking framework. This system installed three RGB-D sensors to capture RGB-D data at 1.2m height. They tested it on a real-world dataset and obtained an improvement of tracking performance. However, the system was not compared with the other methods or evaluated on available public datasets. In addition, the facts that the frame rate was not reported and three RGB-D sensors were required make it inappropriate for robotics applications.

In this chapter, a novel human detection and tracking framework is designed to utilize 3D clustering for detection and feed 3D point cloud and 2D image features for updating an online classifier. The clustering is performed in 3D space using a normal-based region growing method while the classifier utilizes both 3D and 2D features. The online classifier in this work is the online support vector machines (SVM) which is based on the kernerlized structure output SVM (Hare et al. 2016) that supports usages of multiple kernerls.

4.3 Methodology

This work presents a human tracking framework from a single RGB-D sensor using online learning methods. Figure 4-1 shows the technical overview. The system starts with an RGB-D image and the bounding box of the object (the yellow box *Last result* in Figure 4-1) which is initialized by the user. In subsequent loops, the *Last result* will be updated by the previously detected results. To avoid confusion and clutter, the setting or update of the *Last result* is not shown in the figure. Then at one frame, the first 3D sampling method (referred to as the candidate sampling) is conducted to find the candidate human clusters in the point cloud. The

online learning method evaluates these candidates and tries to recognize the target human. If the target is found, another 3D sampling process (referred to as the universal sampling) is utilized to obtain various point clusters including both positive samples (candidate human clusters) and negative samples (non-human clusters, or clusters of other humans). Based on these samples, the online classifier is updated. With a new RGB-D frame, the same processes will be iterated.

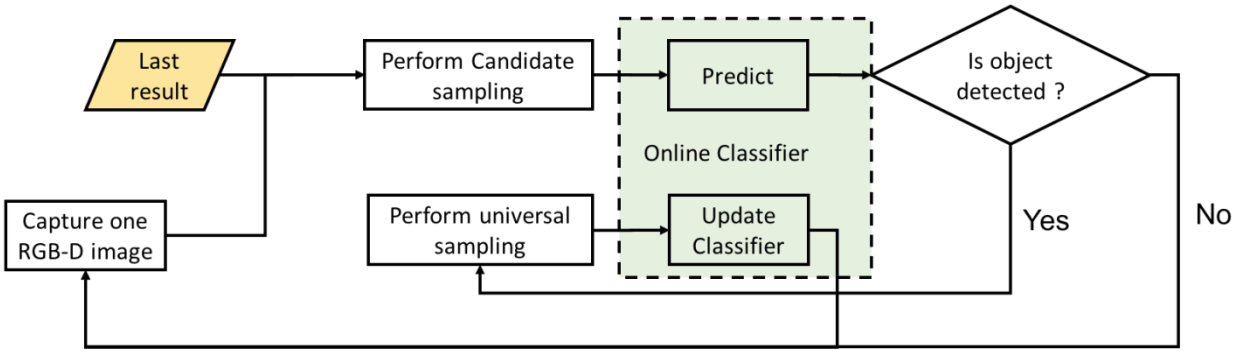


Figure 4-1: Technical overview of the human detection and tracking framework.

4.3.1 Term Explanations

The following terminology is utilized with specific meanings in this chapter:

- RGB-D sensor: An RGB-D sensor is composed of an RGB camera and a depth camera, and can obtain an RGB image and a depth image. Then, using the camera parameters, the RGB image and the depth image can generate an organized color point cloud that is also called an RGB-D image.
- 2D image: A 2D image refers to an image that contains only intensity information (color or gray) without any 3D information. When using the RGB-D sensors, a

2D RGB image is captured by the RGB camera while a 2D gray image can be obtained from the RGB image.

- RGB-D image: An RGB-D image $\boldsymbol{I_p}$ is an organized colored point cloud where every point can be found by a 2D index and contains color information (r, g, b) and 3D coordinates (x, y, z) with respect to the sensor. As $\boldsymbol{I_p}$ is created using a color image $\boldsymbol{I_c}$ and a depth image, $\boldsymbol{I_p}$ is associated with a specific $\boldsymbol{I_c}$.
- Sample: A sample is referred to as a point cluster which might be the target human cluster, and is represented by a 2D bounding box (rectangle) $\boldsymbol{R}$ and a 3D bounding box (cuboid) $\boldsymbol{C}$ where $\boldsymbol{R}$ denotes the 2D bounding box of the target human in $\boldsymbol{I_c}$ while $\boldsymbol{C}$ is the 3D bounding box of the target in $\boldsymbol{I_p}$. Using the transformation matrix between the RGB camera and the depth camera, $\boldsymbol{R}$ can be computed from $\boldsymbol{C}$. However**,** some of points within $\boldsymbol{R}$ might not belong to $\boldsymbol{C}$. Therefore, a sample can be uniquely represented by $\boldsymbol{C}$ or $\boldsymbol{(R, C)}$**.**
- Positive sample: A positive sample mainly contains points from the target human and is viewed as the target human by the classifier.
- Cuboid: A cuboid is the 3D bounding box of a point cluster whose edges are parallel to the corresponding three axes in the coordinate system. A cuboid is represented by a 6×1 vector, $[c_x, c_y, c_z, l_x, l_y, l_z]$ while $\boldsymbol{c_c} = (c_x, c_y, c_z)$ is the centroid of the cuboid and $\boldsymbol{c_d} = (l_x, l_y, l_z)$ are the dimensions of the bounding boxes along the three axes.

4.3.2 3D Sampling

For an input RGB-D image, given the previous object location, the sampling methods are performed to detect multiple point clusters one of which may contain the target object. Depending on the usages of the samples, the sampling method is categorized into candidate sampling and universal sampling where the former mainly aims to include the samples containing the object being tracked while the latter finds some negative samples to update the classifier. To utilize the 3D data, this work develops different approaches for the two sampling methods in order to effectively identify the object and update the classifiers.

4.3.2.1 Candidate Sampling

To effectively find the object candidates, the candidate sampling is composed of a two-stage cluster detection method. For many cases of human detection and tracking in RGB-D images, the target human is isolated in the environment and not connected to any other objects. Therefore, the first stage of the candidate sampling is to find isolated clusters based on the previous location of the target human. When the target is connected to other objects, the first stage fails while the second stage randomly obtain multiple samples using the previous location.

The first stage of candidate sampling utilizes the normal-based region growing method to cope with the variance of human sizes in the data. Due to the limited field of views of the sensor, occlusions of other objects and the poses of the person, the human object might not be fully captured by the sensor, and thus its dimensions vary in the RGB-D images. Therefore, instead of using the dimensional information from the previous tracking result, the candidate sampling method performs an efficient normal-based region growing algorithm around the center of the previous location. Based on the 2D rectangle of the previous location, the method dilates the rectangle in order to incorporate more points for consideration. Then, for all the points within the

2D rectangle, the normal-based region growing clustering is employed to detect connected components.

By taking advantages of the organized point cloud, the normal-based region growing clustering method searches neighbors using 2D indices instead of utilizing 3D neighbor finding methods (e.g., k-d tree (Bentley 1975)), which greatly saves time of constructing the searching structures and searching for neighbors. In addition, searching neighbors using 2D indices allows easy integration of downsampling which can greatly reduce the number of points being processed and thus improve time efficacy. For a point $\boldsymbol{p_0} = \{x_0, y_0, z_0\}$ whose 2D index is (r, c), its four neighbors in 2D space are the points corresponding to the indices $(r, c - s)$, $(r - s, c)$, $(r, c + s)$, $(r + s, c)$ while s is the down-sampling stride. If $s \leftarrow 1$, all the points are utilized and no downsampling is performed. If $s \leftarrow 2$, a quarter of the points are utilized since the points are downsampled in both dimensions using s. The neighbor points are also validated by comparing their distance to $p_0$to a distance threshold r (the neighboring searching radius). Even though this neighbor searching strategy cannot find all the neighboring points of p_0 within r, it is still able to effectively obtain the correct clusters for the region growing method.

When the tracked human is connected to other objects (e.g., other humans, furniture), the first stage detects clusters that contain both the target and the other objects, which affects the target detection. Therefore, to address this issue, the candidate sampler utilizes the previous target location to generate random samples. Based on the previous center of the cuboid, a certain number of cuboids (in this work, 50) are generated by fixing the cuboid size while randomly generating the cuboid center within a radius around the previous cuboid center. Since a randomly generated cuboid might not reflect the true bounding box of the contained points, for each cuboid sample, all the points inside this cuboid are utilized to update the cuboid parameters.

Once the samples are obtained, this approach removes some impossible samples by comparing the cuboids of the sample to the last cuboid of the target human. If the dimension changes of a sample with respect to the last target cuboid are larger than a threshold, it is unlikely that the sample contains the target human and thus is removed from the candidate samples. If the distance between a sample's centroid and the centroid of the last cuboid is larger than a distance threshold, the sample is eliminated too.

4.3.2.2 Universal Sampling

The universal sampling mainly serves to find negative samples around the current cuboid $\boldsymbol{C_{curr}}$ of the detected human while some of the positive samples might be found too. All the samples including the $\boldsymbol{C_{curr}}$ are utilized to update the classifiers. Compared to finding connected components in the candidate sampling process, the universal sampling maintains all the points within that area. To make the sampling method efficient, the sampling is performed based on the current rectangle $\boldsymbol{R_{curr}}$ to obtain multiple rectangles in the 2D space. By fixing the size, multiple rectangles around the center of $\boldsymbol{R_{curr}}$ are obtained**.** Then, the corresponding cuboids are computed using points with the rectangles. These samples are finally employed to update the classifiers.

4.3.3 Feature Representations

The histogram features (Hare et al. 2016) from the 2D image $\boldsymbol{I_c}$ are utilized in this work. To efficiently compute the color histogram, a spatial pyramid of four levels is constructed for the whole image. At each level L, the image is divided into $L \times L$ small patches each of which yields a histogram. The final histogram is a combination of the histograms of all the levels. When there

are multiple samples with overlapping areas, this implementation can reduce redundant computation.

3D features for a sample $\boldsymbol{s} = \{\boldsymbol{R}, \boldsymbol{C}\}$ are also investigated in this work. In order to achieve real-time performance, features that involve high computational cost are not considered. The statistic and geometric features of the point cluster are utilized in this work. The covariance matrix of the points within $\boldsymbol{C}$ is computed. The eigenvectors and eigenvalues are calculated and utilized in the feature representation. In addition, the point density of this cuboid is included in the feature representation.

4.4 Experiments and Discussion

4.4.1 Experimental Setup

To evaluate the tracking performance, six RGB-D videos which contain people moving in various illumination conditions are selected from the Princeton Tracking Benchmark datasets (Song and Xiao 2013) that are designed for evaluation of RGB-D tracking methods. To quantitatively evaluate our method, the six videos are manually labeled using the 2D images. The 2D object tracking method (Hare et al. 2016), Struck, and the human detection method using the ground plane assumption (Munaro and Menegatti 2014) which is referred to as PCL-Human, are tested on the videos for comparison.

The implementation in Point Cloud Library (PCL 2016) of the PCL-Human method is adopted by using the recommended parameters while the Struck implementation provided by the authors is utilized using default settings except that all the features proposed in that paper to allow better accuracy by sacrificing the time efficacy. In addition, to explore different normal computation methods which affect the computational efficiency, the proposed methods have two

different implementations, RGBD-N1 and RGBD-N2 by just varying the normal computation methods. The first one RGBD-N1 used a fixed number of points (in this work, 30) to compute normals while the second one utilized the fast integral normal computation for organized point clouds (PCL 2016) .

Regarding significant parameters in the proposed method, to reduce the size of the point cloud, the sampling stride is 3, which means that only $1/9$ of the points are utilized in the clustering process for the candidate sampling. The normal vector difference threshold is 60^o. All the parameters setting are shared for all the six videos.

In addition, the intrinsic camera parameters for the depth camera are provided to compute the 3D point cloud while the extrinsic parameters (i.e., the rotation and transformation relations) between the RGB camera and depth camera are not provided. As the extrinsic parameters are necessary to convert 3D cuboids to 2D bounding boxes (i.e., rectangles), the factory parameters for the extrinsic parameters are utilized.

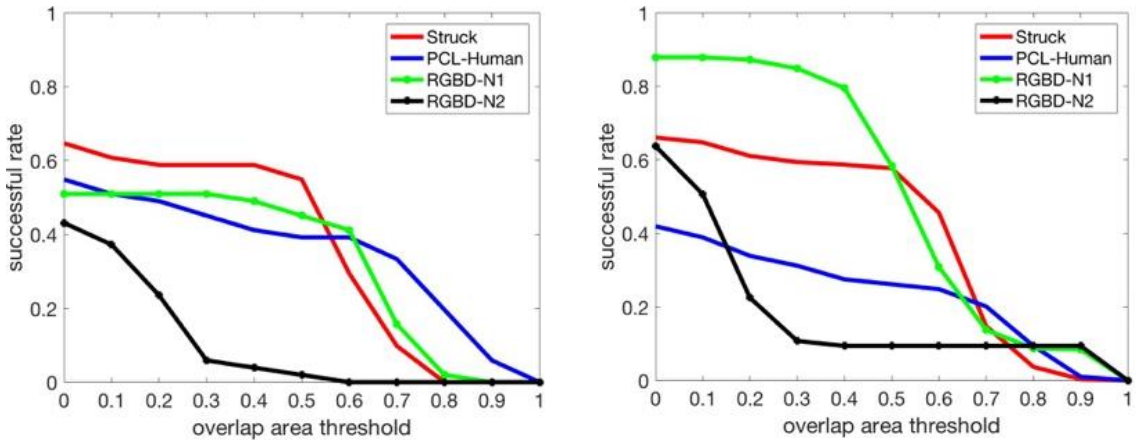
4.4.2 Detection and Tracking Performance Evaluation

As the proposed method relies on 3D clustering, the 2D bounding boxes are computed from the cuboids of the 3D clusters, which applies to the results of PCL-Human. Since 3D cuboid ground truth labels are difficult to be obtained, the evaluation is based on the 2D rectangles. The evaluation criterion utilized in (Everingham et al. 2010; Song and Xiao 2013) is employed. Given a ground truth rectangle $\boldsymbol{R_{g_i}}$ and a detected rectangle $\boldsymbol{R_{d_i}}$, the ratio of overlap r_i is computed as:

$$r_i = \begin{cases} \dfrac{area(\boldsymbol{R_{g_i}} \cap \boldsymbol{R_{d_i}})}{area(\boldsymbol{R_{g_i}} \cup \boldsymbol{R_{d_i}})} & \boldsymbol{R_{g_i}} \neq \emptyset, \boldsymbol{R_{d_i}} \neq \emptyset \\ 1 & \boldsymbol{R_{g_i}} = \emptyset, \boldsymbol{R_{d_i}} = \emptyset \\ -1 & otherwise \end{cases} \tag{4.1}$$

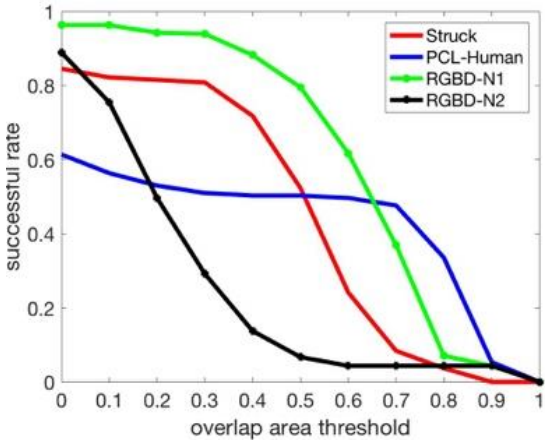
where $\boldsymbol{R}_{\boldsymbol{g}_i} = \emptyset$ denotes that the target is not visible in the image, $\boldsymbol{R}_{\boldsymbol{d}_i} = \emptyset$ means that no detection result is generated. If r_i is greater than a minimum overlapping area threshold r_t, the human in this frame is correctly identified. Thus the successful rate R of a tracker is computed as $R = \frac{1}{N}\sum_{i=1}^{N} u_i$ where $u_i = 1$ if $r_i > r\_t$, otherwise $u_i = 0$. The successful rates of the four methods, i.e., Struck, PCL-Human, RGBD-N1, and RGBD-N2, for all the six videos are shown in Figure 4-2. As PCL-Human tries to detect all humans and thus usually generates more than one rectangle, among detected rectangles by PCL-Human, the one that has the largest the ratio of overlap r_i is selected as the final result for computing the successful rate.

As shown in Figure 4-2, except for the fifth video, the RGBD-N1 method has higher or comparable successful rates compared to other methods. Several example images from the last three videos are shown in Figure 4-3 where the green rectangles are the ground truth and the red ones denote the detection results.

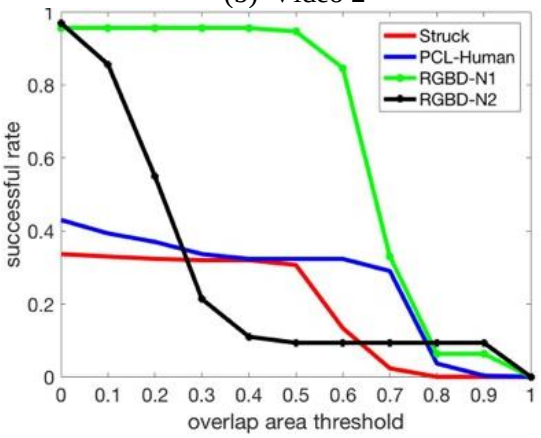


(a) Video 1

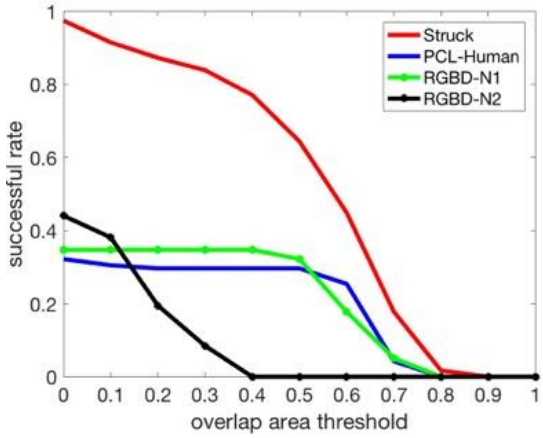
(b) Video 2



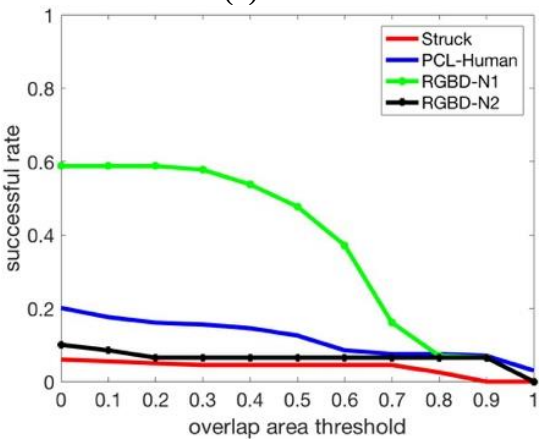
(c) Video 3



(d) Video 4

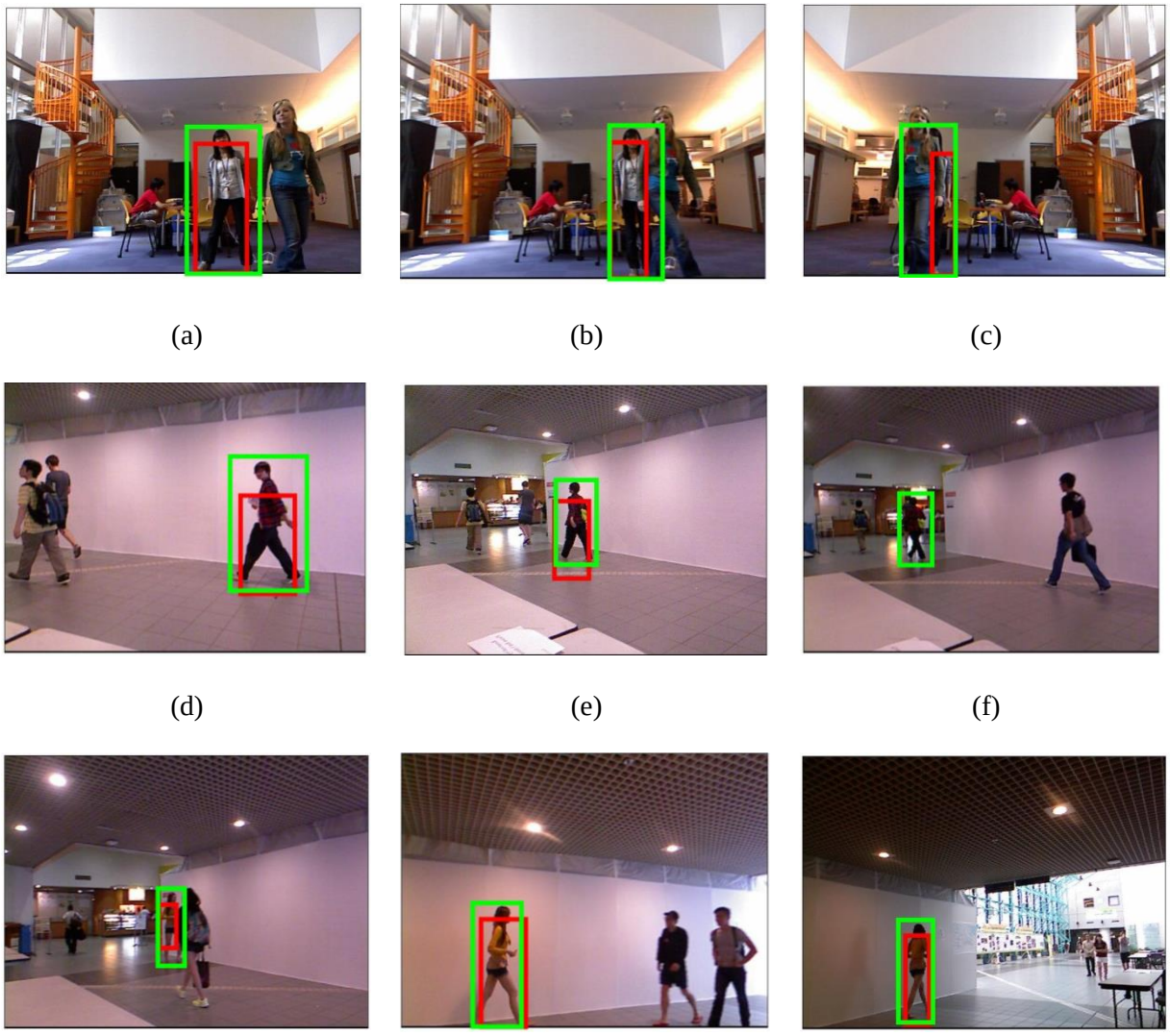


(e) Video 5



(f) Video 6

Figure 4-2: Successful rates vs. the overlap ratio threshold.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 4-3: Example results (in red) of RGBD-N1 on the last three videos.

First, by comparing the proposed methods, RGBD-N1 and RGBD-N2, it is found that RGBD-N1 (the green lines) outperforms than RGBD-N2 (the black lines) over all the videos as shown in Figure 4-2. This fact demonstrates that the normal computation method is important for the proposed method in order to predict the correct rectangles. In fact, RGBD-N2 usually

generates smaller rectangles around the target humans compared to RGBD-N1. The normals computed in RGBD-N2 create more differences of normals around the target and lead to several clusters after the region growing clustering, which makes the final rectangles smaller compared to the ground truth rectangle.

Second, comparing RGBD-N1 (green lines) against Struck (red lines) in Figure 4-2, it is found that other than the first and fifth videos, RGBD-N1 achieves higher successful rates than Struck. It should be noted that due to the error of the sensor extrinsic parameters and the fact the clustering ignores isolated points, the rectangles predicted by our method are usually smaller than the actual ones due to missing points from the header or the legs as shown in Figure 4-3 (a, b).

On the other hand, Struck predicts the rectangles with the same size even though the object is moving toward or away from the sensor. Therefore, when the human sizes change in the videos, our method is able to capture this variety compared to Struck. However, for the fifth video, the proposed method predicts rectangles that contain some points from the ground while ignoring some points from the head. In addition, in this video this person moves from one side to another and disappears, and our method fails to detect the human as shown in Figure 4-3 (f). As the distance between the sensor and the person increases, fewer points are observed due to the observation range limitation of the RGB-D sensor, and thus the clustering method fails to detect the target human cluster.

By comparing RGBD-N1 (green lines) against PCL-Human (blue lines) in Figure 4-2, it is observed that RGBD-N1 obtains higher successful rates. One of the reasons is that PCL-Human only performs detection without tracking and it does not utilize any previous results. Another reason is that it requires the ground plane to find the human candidate clusters. For the

last video, as the sensor poses vary and some frames have few points from the ground, the ground plane models are not updated correctly, which leads to many false alarms in the results.

4.4.3 Time Performance

As Struck performs 2D tracking, we only compare the time efficiency between the proposed method and PCL-Human. The average frame rates are 0.7, 3.8 and 8.3 fps, for PCL-Human, RGBD-N1, and RGB-D-N2, respectively overall for all the videos. The PCL-Human method is not coupled with a tracker and thus searches candidates over the 3D space in the RGB-D data, which costs more time to perform detection compared to tracking methods that utilize a local searching strategy. The normal computation method of RGB-D-N2 takes advantages of the organized point cloud and thus computes normals faster than the normal computation method in RGBD-N1, which contributes to the frame rates difference between them. This fact indicates the direction for improving the computational efficiency of the proposed method. Since this approach only utilizes a fraction of the points in the RGB-D data, the normals for the other points are not involved. Therefore, the following strategy can be used to reduce unnecessary normal computation: if the normal of a point is needed, it will be computed and saved for further usages.

In addition, the clustering process in the proposed method consumes some time for iterating all the points and comparing the distances and normals. By setting a larger sampling step, it will save time for clustering. However, since this sampling is performed in 2D space and thus causes uneven sampling in 3D space, a larger sampling step will affect the clustering results especially for clusters located far from the sensor. One solution to overcome this issue is to adjust the sampling step according to the distance of the cluster to the sensor. For clusters closer to the sensors, the sampling step s can be set as a larger number while s should be smaller when the cluster is far from the sensor.

4.5 Conclusions and Future Work

This chapter presents a real-time human detection and tracking method using a single RGB-D sensor. Two 3D sampling methods are utilized to detect candidate human clusters and obtain negative samples for training the online classifier. The online classifier detects the human being tracked from the candidate human clusters, and utilizes both the positive and negative samples to update its parameters. The method is tested on six RGB-D videos collected in real-world settings under different illuminations. The experimental results demonstrated that the proposed method achieves high success rates compared to a 2D tracker and a 3D human detection method. The proposed method can achieve an average frame rate of 3.8 fps, which is appropriate for real-time applications.

However, the current method does not estimate the moving models of the target person and sometimes fails to identify the correct person when the human appears after being blocked. Future work will explore the use of a Kalman filter to predict the human movements in order to provide hints for possible human locations that can also facilitate the candidate sampling. In addition, the current method can only track a single person. Future work will investigate multi-people detection and tracking.

Chapter 5

Excavation Slope Stability Monitoring Using 3D Reconstruction and Modeling from Aerial Images

5.1 Introduction

Due to its fast data acquisition and operational mobility, Unmanned Aerial Vehicles (UAVs), i.e., drones are seeing applicability in many civil infrastructure applications such as seismic risk assessment, surveying and mapping, and construction monitoring (Liu et al. 2014). Drones can be used to collect images frequently and fast for visual inspection and damage detection on existing civil structures by using computer vision methods (Morgenthal and Hallermann 2016). In addition, based on videos collected by drones and by adopting the structure from motion (SFM) system (Hartley and Zisserman 2003), three-dimensional (3D) point clouds of the environment can be reconstructed for metric applications of the data. The ability to facilitate rapid 3D modeling makes the application of drones very promising in construction processes such as excavation, which involve continuous and dramatic changes in the geometry of the work environment. Excavation involves the movement of large amounts of earth, resulting in continuously evolving changes to the ground surface geometry. Ongoing excavation operations present several safety issues for site personnel, and are primarily related to the stability of the excavated slopes and their cave-in potential.

Many research projects have explored the utilization of drones to obtain 3D models for earthwork (Nassar et al. 2011; Siebert and Teizer 2014). Due to their special applications, the proposed systems either rely on high-resolution cameras (Siebert and Teizer 2014) to acquire accurate 3D models, or multiple sensor fusion, and assume professional knowledge on surveying and mapping in data collection to obtain geo-referenced data. Different from previous methods, this chapter aims to present a readily deployable framework for monitoring excavation slopes using drones. The only data needed from the drone is video imagery captured by color cameras without any other sensor data. In addition, the drone can be controlled by a person without professional knowledge in surveying and mapping. In order to monitor the excavation slope safety, based on the collected videos, this chapter presents a comprehensive data processing scheme that contains constructing 3D point clouds from the video images, obtaining terrain models, and slope analysis.

The remaining sections of this chapter are organized as follows: Section 5.2 reviews related work on utilizing drones (especially when creating 3D models from drone collected videos) in civil engineering applications. Section 5.3 presents the technical approach including obtaining the comprehensive point cloud from videos, extracting ground points, generating terrain model, and performing slope analysis. Section 5.4 introduces the experimental details and results for a real excavation project. The last section draws conclusions about this research project, and identifies a future research agenda.

5.2 Related Work

Before drones were used for collecting data for 3D reconstruction, airborne or terrestrial laser scanners (Tang et al. 2010; Xiao et al. 2015) have been widely used for 3D building model generation. Laser scanner based data collection requires professional operators to obtain the data

and process the raw data for 3D reconstruction. In addition, the data collection is limited by the accessibility of view positions or orientations. In contrast, equipped with high-resolution cameras, drones allow faster data collection at various positions and orientations. In terms of the direct data used for domain related analysis, the applications of drones in civil engineering mainly can be categorized into two areas: (1) image based applications, (2) point cloud based applications. The first one directly utilizes images collected by drones to perform image analysis methods for specified projects (e.g. site monitoring (Zollmann et al. 2014), bridge inspection (Metni and Hamel 2007)). The second one analyzes the 3D point cloud data generated from the video images for projects involving model changing, e.g. earthwork planning (Siebert and Teizer 2014), as-built BIM generation (Wefelscheid et al. 2011). As this research belongs to the second category, this section focuses on related work for point cloud (generated from images) applications in civil engineering.

Xie et al. (2012) mounted four cameras on a drone and collected images in order to create 3D building models for urban areas. By performing self-calibration among the four cameras and utilizing triangulation for four images, the 3D building models are reconstructed from the triangulation results and images. This method is tested on a university campus and was shown to obtain high accuracy for large scale mapping. Wefelscheid et al. (2011) presented a processing chain of using images collected by a drone to created 3D as-built building models. By matching features from images and detecting the loop closure of drone trajectories, a 3D point cloud is obtained by performing dense reconstruction (Furukawa and Ponce 2010). The experimental performance and results on two benchmark datasets and a real-world dataset show that the proposed method is able to obtain high-quality models and the precision is comparable to that derived from Light Detection And Range (LiDAR) systems. To obtain accurate as-built BIM,

these systems usually require high-resolution cameras. Golparvar-Fard et al. (2011) proposed a system to monitor changes of 3D building elements from unordered photo collections. They first reconstructed as-built BIM using 3D point cloud from unordered photo collections. Then the as-build model is registered with as-planned BIM model. A machine learning algorithm is utilized to automatically detect and track changes of 3D building elements. The system is tested on three image collections for two building construction projects and demonstrates its feasibility for generating 4D as-built models and for detecting progress for building elements. This method needs the as-planned model to register against the as-build model. In addition, in order to adopt the machine learning algorithm, a large dataset is required to avoid overfitting. Moreover, in order to make the trained classifiers more general, more comprehensive datasets are necessary. Zollmann et al. (2014) utilized 3D reconstruction from images collected by drones to develop a mobile augmented reality (AR) system for construction management and documentation. The 3D point cloud is reconstructed and registered to the absolute coordinate system while incorporating 2.5-D as-planned data geo-referenced camera images. For visualizing data in AR, 3D models and video images are registered by a multi-sensor fusion system using Real-Time Kinematic (RTK) Global Positioning System (GPS), a vision-based panorama-tracker (Schall et al. 2009) and an inertial measurement unit (Batista et al.). This system allows for automated data collection by utilizing multiple sensors. To design the drone trajectory and utilize those sensors, it usually requires the operator to receive professional training.

Other than creating 3D models for existing or under-construction buildings, drones are also utilized to collect data and create 3D models for other construction projects, especially earthwork (e.g. landfill, excavation). Nassar et al. (2011) employ surface reconstruction techniques to model and quantify earthwork. The Autodesk software toolkits, Project Photofly

and Photo Scene Editor, are utilized to obtain 3D models from a set of high-resolution images (5 million to 10 million pixels). The software also allows topographic modeling and calculation of quantities. The authors tested these techniques on 23 excavation sites and found the appropriate ranges (e.g., pit excavations less than 2000 square meters and depths up to 5 meters) for using the techniques in earthwork applications. This method is based on very high-resolution camera and utilization of software kits. Siebert and Teizer (2014) developed a novel program for photogrammetric flight planning and its execution for obtaining 3D point clouds from images collected by drones. By performing the flight planning with geo-referenced coordinates, the drone is able to automatically collect videos with the GPS model onboard. Based on the geo-coordinated photos, a 3D point cloud, orthophoto, and a digital elevation model are obtained. Experimental results on a test bed environment and a landfill project demonstrate the successful applicability of drones and photogrammetric surveying for earthmoving projects. As this method aims to design flight planning and perform photogrammetric survey, the system needs GPS to have geo-referenced coordinates for the 3D models. Kim et al. (2015) utilized two types of drones to explore the generation of a 3D model of mesh image of excavation work. They compared several different models of UAV systems and selected two among them for getting data for the excavation work by setting the specified target performance. By utilizing the two drones on an excavation project, this method processed the data using the SFM system to generate 3D point clouds. As they focused on exploring different types of drones for excavation projects, no further data processing (e.g., 3D modeling) was conductedd based on the 3D point clouds reconstructed by SFM.

Different from previous work on utilizing drones for earth moving projects, this chapter explores a deployable framework for monitoring excavation slopes using drones. The data

collection only requires controlling the drones to collect videos without using any other sensors (GPS or IMU). In addition, the framework also includes 3D terrain model reconstruction from video images and interactive slope analysis.

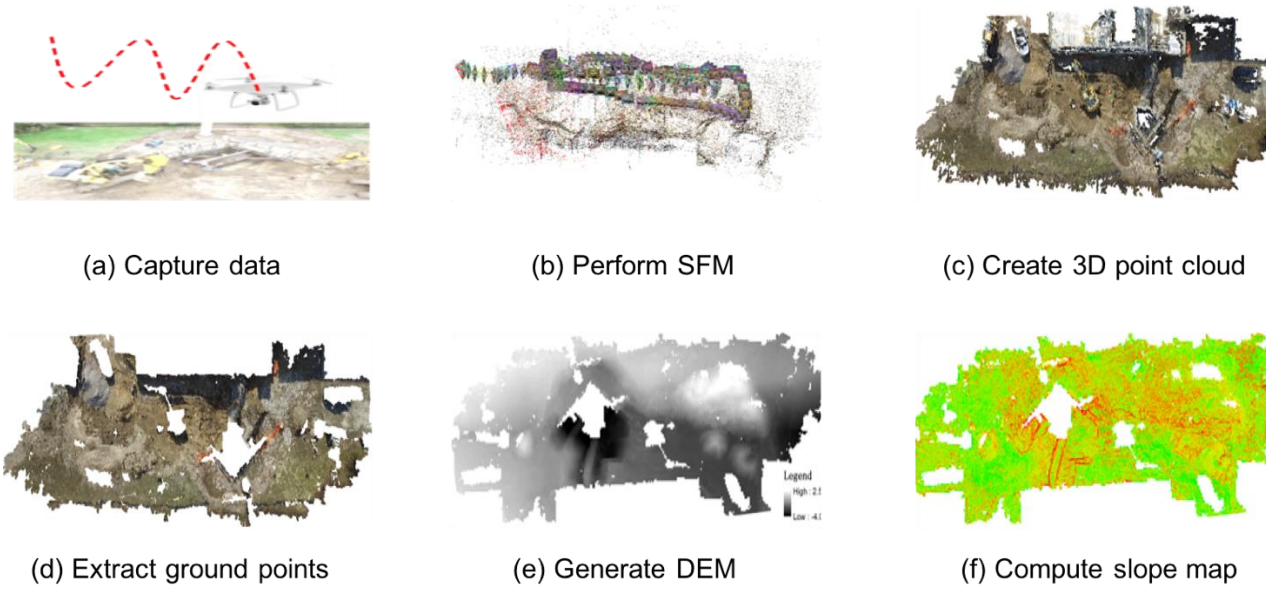


Figure 5-1: Overview of the excavation slope stability monitoring system.

5.3 Technical Approach

5.3.1 System Overview

As shown in Figure 5-1, the proposed excavation slope stability monitoring system workflow starts from collecting videos by flying a drone equipped with a camera. Using the video images, the SFM method is performed to generate a dense 3D point cloud of the excavation site. Then the ground points are separated from the non-ground points by computing

the elevation changes of points and clustering smooth surfaces. Based on the ground points, a digital surface elevation model (Demir et al. 2015) for the terrain is created for visualization of the terrain surface and supports further slope analysis (Figure 5-1 (f)).

5.3.2 Point Cloud Reconstruction

In order to obtain comprehensive and practical point clouds from videos taken by a drone, the Structure from Motion (SFM) method is firstly applied to create a raw point cloud from video images. Then by utilizing known measurements in the real world, the raw point cloud is scaled. Finally, the coordinate system of the point cloud is aligned to a generally used world coordinate system for better data interpretation and analysis.

5.3.2.1 Structure from Motion

The Structure from Motion (SFM) system is able to create 3D point clouds from a sequence of images. SFM aims to recover the 3D point positions (structure) and camera poses (motions) by optimization methods (e.g. bundle adjustment). SFM matches feature points, e.g. the scale-invariant feature transform feature (SIFT) (Lowe 2004), the speed up robust feature (SURF) (Bay and Tuytelaars 2006), in the images to build relations between sensor poses. When sufficient feature points matching and images are available, the camera poses including the camera parameters and 3D structure are iteratively estimated. For large scale scene reconstruction, the incremental SFM (Wu 2013) is applied to process a large number of images for a large area.

One of the key points in reconstruction of high-quality point clouds is to ensure that the captured environment contains sufficient distinct feature points and the images observe the objects of interest at different positions and orientations while having sufficient overlapping

areas. When capturing videos for excavation projects, the natural environment is able to provide sufficient feature points. In addition, a drone can fly above the environment and thus acquire images at different positions and view angles. Thus the videos captured by a drone are particularly suitable for reconstructing high-quality point clouds for excavation projects.

Due to the scale ambiguity, SFM is not able to obtain point clouds with absolute scale metrics. Thus, the unit of the point cloud constructed from SFM is unknown. To enable further analysis, the point cloud has to be scaled. A common way is to take some measurements in the real world, identify the corresponding measurements in the point cloud and then scale the point cloud to make it consistent with the measurements.

5.3.2.2 Point Cloud Alignment

Due to the initialization of the optimization method, SFM cannot estimate a unique coordinate frame of the point cloud from monocular visual images (Szeliski 2010). The coordinate system orientations, i.e., the directions of the three axes, in the reconstructed point clouds from SFM are usually not consistent with those of the real world. For example, a flat terrain is horizontal and usually lies in the XY plane. To align the point cloud to a commonly used coordinate system, one option is to manually identify the three axes in the point cloud and rotate the point cloud correspondingly. Given that the terrain is the main component of a point cloud for excavation data, this chapter utilizes the Manhattan world assumption (Coughlan and Yuille 2003) to automatically align the coordinate system orientations.

According to the Manhattan world assumption, there exist three main axes in man-made environments. Based on this theory, the normal vectors of points in these environments can be clustered into three classes whose centers are the three main axes. For flat terrain, most of the normal vectors of terrain points should be parallel and point toward the sky (assuming that the

normal vectors are aligned.) This direction is usually used as the Z-axis for terrain modeling. The other two axes fall in the flat terrain. Even though the flat terrain assumption is not true for an arbitrary excavation project, considering the large scale of captured points and some man-made objects (e.g. buildings), the Manhattan world assumption can be used in this scenario.

This chapter utilizes the method proposed in (Yaguchi et al. 2013) to compute the three main axes for a point cloud. The normal vectors $\mathbb{N} = \{n_1, n_2, \dots, n_N\}$ (N is the number of points, the length of a normal vector n_i is one) of all the points are first calculated by fitting a plane using its neighboring points. Then by utilizing the geodesic dome model (a uniform density sphere distribution) $\mathbb{M}$, the histogram of the normal vectors $\mathbb{N}$ is created. The bin values in the histogram are the sampled unit vectors in the geodesic dome model. Each normal vector $v \in \mathbb{N}$ is assigned to the bin from searching the closest point in $\mathbb{M}$ by viewing all normal vectors as 3D points. Based on the histogram, the first main axis a_1 is determined by as the unit vector associated with the highest bin. The second one a_2 is searched by finding the largest bin from vectors within the plane perpendicular to a_1 to make sure that $a_1 \perp a_2$. Finally, the last main axis a_3 is computed as it is orthogonal to both a_1 and a_2.

Given the three main axes, the point cloud is then rotated to make the coordinate system orientations parallel to the three axes.

5.3.3 Terrain Modeling

Apart from terrain points, a point cloud also contains several non-terrain points, such as construction equipment (e.g., excavators), construction workers, and buildings. In order to prepare clean data for analysis, the terrain points should be extracted to create terrain models.

This research utilized the progressive morphological filter (Zhang et al. 2003) to separate terrain points from non-terrain points. This method firstly rasterizes the point clouds into a raster

grid, which greatly reduces the size of the point cloud and improves the computational time. Then for a certain window size, the slope for each point with respect to the lowest point is computed. If the slope is greater than a threshold (which is also updated by the slope and the window size), the point is classified as a non-terrain point. By increasing the window size, the method iteratively identifies the non-terrain points in the grid. This method is able to effectively extract terrain points in both urban and rural areas.

However, this approach is designed for extracting ground points for large-scale airborne LiDAR point clouds which rarely capture points from vertical objects (e.g., building walls). For a small-scale point cloud with large terrain slope variation, this filtering algorithm fails to completely remove all non-ground points even it is able to remove some non-ground points and almost all points from vertical walls. To remove the remaining non-ground points, a normal-based region growing algorithm is utilized to cluster the point clouds.

The normal-based region algorithm firstly computes the normal vectors and curvature of points using their neighborhood points. Firstly, a stack of seed points S is initialized by selecting the point with the largest curvature value. For each point $p \in S$, for every point q within its neighborhood N_p, if the difference between their normal vectors is smaller than a threshold, these two points are assigned to the same cluster. Then, q is used as a future seed point and pushed into the stack S. Once all the seed points in S are visited, a cluster is found and S is cleared as empty. Then a new seed point is selected from the remaining points and added to S. The same iteration is performed for S until all points of S are visited. The algorithm is iteratively performed until all the points are assigned to a cluster.

By utilizing the normal vector and Euclidean distance (searching points in the neighborhood) in the clustering, the normal-based region growing algorithm is able to detect

compact and smooth clusters. Since the ground surface is smooth and usually contains the majority of the points, the largest cluster of the clustering results from the algorithm is identified as the ground points cluster.

The terrain model is typically represented by the Digital Elevation Model (Demir et al.) which is a continuous surface depicting the elevation of terrain points. Based on the terrain points, the DEM is constructed by interpolation.

5.3.4 Slope Stability Analysis

After aligning the point cloud and extracting the terrain points, ArcGIS is utilized to create DEMs and their slope maps by performing the inverse distance weighting interpolation method. The slope functionality in ArcGIS computes the rate of elevation changes between a cell point and its neighbors. By depicting the slope information for all cells (points) in the grid, the slope map presents a visualization of slopes for the whole DEM.

In order to estimate the slope stability, the excavation pit and the spoil pile in the excavation project are of interest. Since the slope map computes the rate of elevation changes in a certain time window, it cannot be utilized for slope stability analysis for an excavation project in terms of recognizing possible slope failures. In this project, since the two main slope values are necessary, the corresponding surfaces (the trench surface, the ground surface, and the spoil pile surface) of interest are manually identified. The points for the three main surfaces are manually selected from the ground points and their plane parameters are computed using least square estimation. Based on the plane parameters, the slope values are calculated.

5.4 Experiments and Applications

5.4.1 Data Collection and Processing

The videos are collected by a DJI Inspire 1 drone equipped with a camera that can provide 4K resolution videos at up to 25 frames per second and capture 12 megapixel photos. The drone is manually controlled by a construction site personnel and each video lasts for less than 10 minutes. To scale the point cloud, the measurements are obtained by measuring the distances of several distinguishable objects (e.g., the dimensions of a small building structure adjacent to the excavation site) in the environment. After sampling images from the videos, the SFM system, VisualSFM (Wu 2007; Wu 2011) is carried out to obtain raw dense point clouds. VisualSFM calculates the SIFT feature points using Graphical Processing Units (GPU) and enables parallel computing when estimating the 3D positions and sensor poses. The basic VisualSFM system only reconstructs a sparse 3D point cloud. In order to reconstruct rich and dense point clouds, the dense reconstruction method proposed by Furukawa and Ponce (Furukawa and Ponce 2010) is performed. In the last step, the dense point cloud is scaled using tape measurements.

There are totally five videos collected for this excavation project including (1) before excavation, (2) phase 1: pre-excavation to determine the locations of buried utilities, (3) phase 2: in excavation, (4) phase 3: in excavation, (5) phase 4: the beginning of backfilling.

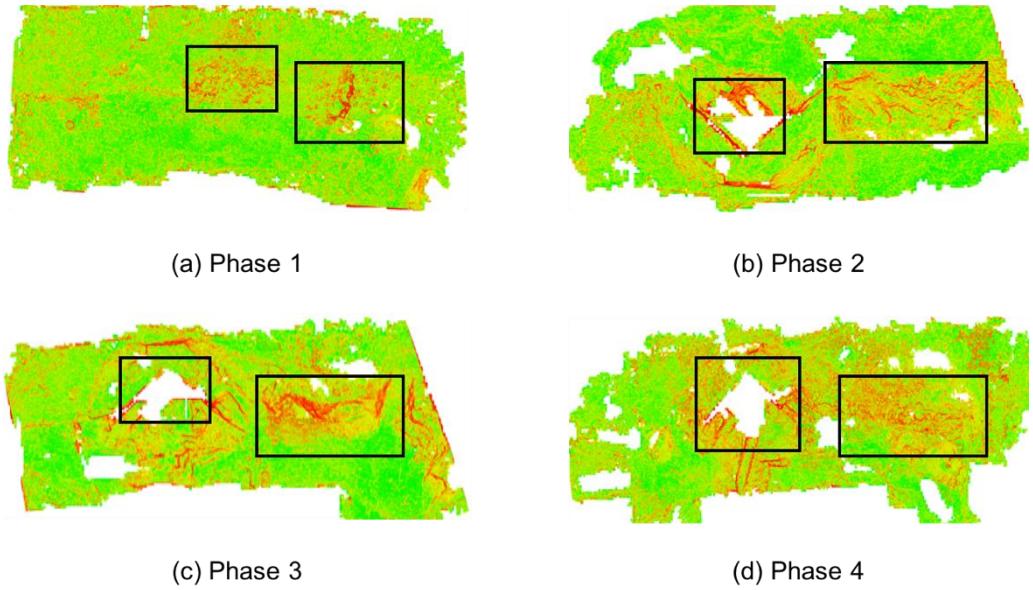


Figure 5-2: Slope Maps from Last Four Videos.

5.4.2 Slope Stability Analysis

Figure 5-2 displays the four slope maps of the excavation site starting from the pre-excavation phase while the left and right rectangles in each image cover the areas for the excavation pit and the spoil pile, respectively. These four images depict the slope changes in the excavation project, especially for the spoil pile (within the right rectangle of the images). In Figure 5-2 (a,b,c), increasing red color points appearing within an area indicate that the slopes in that area are increasing. By comparing Figure 5-2 (c) and (d), the red color points become less, and the slopes turn into smaller values during the backfilling.

In order to quantitatively evaluate the slope stability, the surfaces of interest, i.e., the trench and the spoil pile are identified manually in the ground points using CloudCompare (http://www.danielgm.net/cc/). For the trench, the surface containing the most points is selected along with the closest ground surface. Then a plane is fitted to each surface using least square estimation. The slope is computed as the angle between the two normal vectors of the surfaces.

The same selection rules apply to the spoil pile. Theoretically, the normal vector of the ground surface should be parallel to Z-axis, i.e. (0,0,1). However, the ground surface in the real world is not flat especially on an excavation construction site. Moreover, whether the slope value is safe is directly related to the ground surface close to the slope area.

Table 5-1: The slope values (in degrees) for the last three videos.

	Pit	Spoil Pile
Phase 2	19.5	31.0
Phase 3	30.2	34.4
Phase 4	3.9	26.5

Table 5-1 shows the slope values for the trench and the spoil pile for the last three videos (Figure 2(a) phase 1 does not involve much excavation). Based on the Michigan Occupational Safety and Health Administration (MIOSHA) standards (MIOSHA Regulatory Services Section 2017) and the type of soil, the maximum allowable slope of the excavation side is 35^{o}. As shown in Table 2, all the slope values computed in the evaluated datasets successfully meet the MIOSHA safety requirement. This is also consistent with the field construction workers as they paid spatial attention to maintain the slope values according to the requirements.

5.4.3 Data Processing Time

As noted previously, this research aims to simplify the data collection and processing steps for excavation slope stability monitoring and thus enables fast data acquisition and processing for rapid analysis and decision-making in construction safety and productivity monitoring. In terms of slope stability quantitative analysis, the system is composed of three main steps: (1) video collection by a drone, (2) data processing to obtain ground points, (3) quantitative slope computation. The data collection takes about 15-20 minutes including

operating the drone to collect videos and tape measurements. The data processing from videos to 3D terrain model consumes less than three hours while the slope computation only takes a few minutes. The most time-consuming processing is 3D dense point cloud reconstruction from the videos, which is almost 85% of all the data processing time. The data processing was conducted on a personal desktop with Intel® Core™ i7-4790K @ 4.00 GHz equipped with the NVIDIA GeForce GTX 970 graphic card. Therefore, the total system from collecting data to slope stability analysis takes less than three hours.

5.5 Conclusions and Future Work

This chapter presents a readily-deployable slope stability monitoring system using drones for excavation projects. The drones for collecting the data of the construction site can be operated by users without professional knowledge of surveying or mapping. The data processing pipeline is able to generate the 3D terrain model and the slope map in less than three hours with little supervision. In addition, the slope stability analysis for the excavation pit and the spoil pile can be obtained from the ground points interactively. Experimental results show that the proposed framework is able to collect data with drones quickly and obtain 3D model and slope stability analysis in time for monitoring the excavation project. The current data processing pipeline generates slope stability analysis with user interaction. Future work will try to automatically identify surfaces of interest and allow automated slope analysis and other safety related computation (e.g., whether the distance between the spoil pile and the trench is smaller than a threshold).

Chapter 6

Object-Based Landslide Detection from RGB Images Toward Automatic Landslide Detection and Mapping

The ultimate goal of the research program proposed in this chapter is to create a new autonomous data collection and decision support system for post-event reconnaissance of geotechnical engineering systems using unmanned autonomous aerial vehicles (UAAVs). Thus, an automated data collection system which allows the UAAVs to collect data with different density will be designed. For example, for geotechnical hazards, the UAAVs are supposed to collect more data at closer distances and different view directions compared to other areas. To enable this functionality, the system should be capable of detecting candidate geotechnical hazards in real time or before the actual flight. Given the easy accessibility of satellite images, we explored to detect geotechnical hazards merely from RGB images. As the RGB satellite images are similar to images collected by UAAVs, it is feasible that the proposed geotechnical hazard detection method using RGB satellite images can be applied to images collected by UAAVs. In addition, by using appropriate techniques (e.g., powerful workstation collected to drone by a wireless network) to enable real-time computation, the real-time collection can by implemented. Therefore, this chapter explored to utilize a supervised classification framework to train classifiers for detecting specified geotechnical hazards, specifically landslides, from RGB satellites images.

6.1 Introduction

Landslides are global geological hazards that can lead to large losses of industrial, agricultural, and forestry productivity, reduced real estate values, and loss of human and animal productivity (Kjekstad and Highland 2009). Specifically, in the United States (U.S.), landslides cost $3.5 billion per year in damage repair, cause between 25 and 50 deaths annually (USGS 2005). Localizing and Mapping the landslides in a timely and effective manner can benefit hazard assessment and management. Remote sensing techniques that can capture images using various sensors (e.g., optical sensors, and infrared sensors) non-invasively and frequently using different platforms (e.g., airplanes, and satellites), have been proven to be time and cost effective for landslide detection and mapping (Mantovani et al. 1996; Metternicht et al. 2005; Wasowski and Bovenga 2015).

When using satellite images to detect landslides, most of the previous research projects rely on integrating multiple data sources, e.g., multispectral images, digital elevation models (DEMs), Light Detection and Range (LIDAR) data, and interferometric synthetic aperture radar (InSAR) images, or performing change detection using data collected before and after hazards (Guzzetti et al. 2012; Lu et al. 2011; Nichol and Wong 2005; Roering et al. 2009; Zhao et al. 2012). For the first type of methods, multiple data sources can capture different characteristics of landslides and other objects (e.g., DEMs can reflect elevation variations for landslides), and thus facilitate to distinguish landslides from other objects in the data. However, these methods require that data from multiple sensors are available, which might lead to high costs to acquire data and more labor cost to interpret the data.

Based on comparing the data before and after landslides, the second type of methods is able to generate accurate landslide mapping by finding the changes in the data. Therefore, the

data before landslides should be present while the data after landslides should be captured in time using remote sensing techniques. In addition, more data sources lead to more computational time and cost for processing the data.

Compared to previous work, this research explores how to utilize object-based methods to efficiently detect landslides from only RGB satellite images after landslides occur without any other data sources or satellite images before landslides to identify potential landslide areas. One of the motivations of this research is to quickly identify potential landslide locations to allow other platforms, e.g., airborne photogrammetry, and unmanned aerial vehicles (UAVs), to collect more data for detailed landslide identification and mapping.

UAVs are able to capture data rapidly at closer ranges compared to satellite platforms, and have been utilized to collect landslide images using RGB cameras (Fernández et al. 2015; Lucieer et al. 2013; Niethammer et al. 2010; Turner et al. 2015). UAVs are able to capture data (mainly RGB images) at close range with various view directions and to fly to areas where humans cannot reach due to harsh environments, or where other platforms (airplanes or satellites) fail to obtain data because of occlusion or limitation view angles. However, due to the battery limitation, UAVs cannot capture data on a large scale compared to airborne and space-based platforms, and thus need landslide candidate areas so as to allow efficient and potentially automatic data capture. Moreover, using only RGB satellite images can reduce computational time for further analysis of other data sources by providing candidate landslide mapping.

The rest of the chapter is organized as follows. Section 6.2 reviews related work on landslide detection mainly utilizing remote sensing techniques. Section 6.3 explains the object-based landslide detection from RGB satellite images while Section 6.4 presents the experimental

results and related discussion on images after landslides. Section 6.5 draws conclusions about this chapter and discusses future work.

6.2 Related work

As this work is focused on landslide detection instead of landslide mapping, e.g., detailed landslide mapping using LIDAR data (Jaboyedoff et al. 2010; McKean and Roering 2004), this section reviews related literature on landslide detection from space-borne or airborne remote sensing images.

With the existence of satellite images available after landslides, the most reliable method is visual interpretation by experts (Brardinoni et al. 2003), which is very time and labor consuming. Thus, image processing techniques and supervised or unsupervised machine learning methods are explored to automatically detect landslides from images. As different sensors can capture different characteristics of landslides (for example, optical sensors are able to obtain color and texture information while LIDAR can capture three-dimensional points of the surfaces), the methods for landslide detection are highly related to the data sources. Thus, this section first briefly reviews landslide detection methods using synthetic aperture radar (SAR) imagery, and then discusses the methods mainly from optical images obtained by both space-borne and airborne platforms.

As surface movements can also reflect the slope stability, repeat-pass synthetic aperture radar interferometry (InSAR) on space-borne platforms enables effective landslide detection by providing surface displacement observations at various spatial and temporal scales (Rosen et al. 2000; Rott and Nagler 2006). Rott and Nagler (2006) explored how to utilize differential InSAR methods to detect and map landslide motion by means of single interferometric pairs. Using the SAR images, the DEMs and the surface motion maps are generated and combined to create maps

for landslide motion. Their case study proved that the satellite-based InSAR was able to provide accurate surface displacements for identifying very slow slope deformation and thus can be used for landslide mapping.

Similarly, Colesanti and Wasowski (2006) also utilized the innovative Permanent Scatterers (PS) technique for satellite InSAR to estimate very slow ground surface displacements in order to detect landslides. They also discussed the advantages (e.g., cost-effective for wide-area applications) and disadvantages (e.g., a limited range of detectable displacement velocities) of InSAR methods for landslide detection and mapping. Herrera et al. (2013) combined multi-sensor and multi-temporal SAR data (e.g., ALOS PALSAR, ERS & Envisat, and TerraSAR-X) to monitor the capacity of very slow landslides. They estimated the line of sight (LOS) displacement velocity (Vlos) from those SAR images using an advanced differential interferometric processing technique (Arnaud et al. 2003). Then, based on the Vlos data, a three-step procedure was carried out to generate the landslide damage map. Experimental results showed that the combination of these multi-sensor data allowed them to distinguish different landslide displacement directions, measure different velocity patterns, and separate the slower and faster landslides.

By obtaining color and texture information of landslides, optical images can also be employed to automatically detect landslides. The basic pipeline is to extract various features from the images for each pixel or a group of pixels, and then utilize supervised or unsupervised machine learning methods to separate landslides from other objects based on the features. Barlow et al. (2003) employed Landsat Enhanced Thematic Mapper Plus (ETM+) images and DEM data to perform object-based segmentation (segmenting the images into clusters of pixels which are referred to as objects in this context) and extracted features for the objects. Then, they

utilized a user-specified hierarchical classification structure to remove non-landslide objects so as to identify landslides. The accuracy of this method depends on the classification structure and might not work for different data sources or images collected under different illumination conditions.

Stumpf and Kerle (2011) employed object-based image segmentation on a variety of sample datasets (i.e., Quickbird, IKONOS, GEoeye-1, and so forth), feature selection, and object classification to identify landslides. By integrating the error balancing method, their proposed method achieved accuracies between 73% and 87% for the affected areas. Cheng et al. (2013) performed scene classicization based on the bag-of-visual-words (BoVW) representation with an unsupervised probabilistic latent semantic analysis model to differentiate landslides and non-landslides. Experimental results showed that this method was robust and obtained good performance. Rau et al. (2014) utilized satellite images, airborne digital images and DEMs to perform multi-resolution image segmentation, and then extracted features (e.g., slope gradients, and vegetation indices) for a hierarchical semantic classification network to detect landslides. The experimental results show that the method can achieve accuracy up to 90.3% while requiring less training samples.

In addition, when datasets before and after hazards exist, change detection methods can be used directly or indirectly (assisting the above methods) to extract landslides. Park and Chi (2010) proposed a supervised change detection analysis method by performing multi-temporal object-based segmentation on IKNONOS and QuickBird images to detect landslide-prone areas. After performing the object segmentation on the images, the thresholding was utilized to extract forest areas, and the change detection was performed to generate landslide maps. In order to perform change detection, the images before and after landslides should be available.

Different from the previous work that requires multiple sources of data (e.g., images captured by different sensors), this work focuses on object-based landslide detection using only RGB satellite images so as to rapidly obtain landslide candidate areas. The most relevant prior work to our research is (Stumpf and Kerle 2011). However, apart from using only RGB image data sources, our work also explores multi-scale image segmentation using a superpixel segmentation method instead of the multi-resolution segmentation method (Baatz 2000) in eCognition®. In addition, to deal with imbalanced data, the over-sampling method is utilized instead of under-sampling.

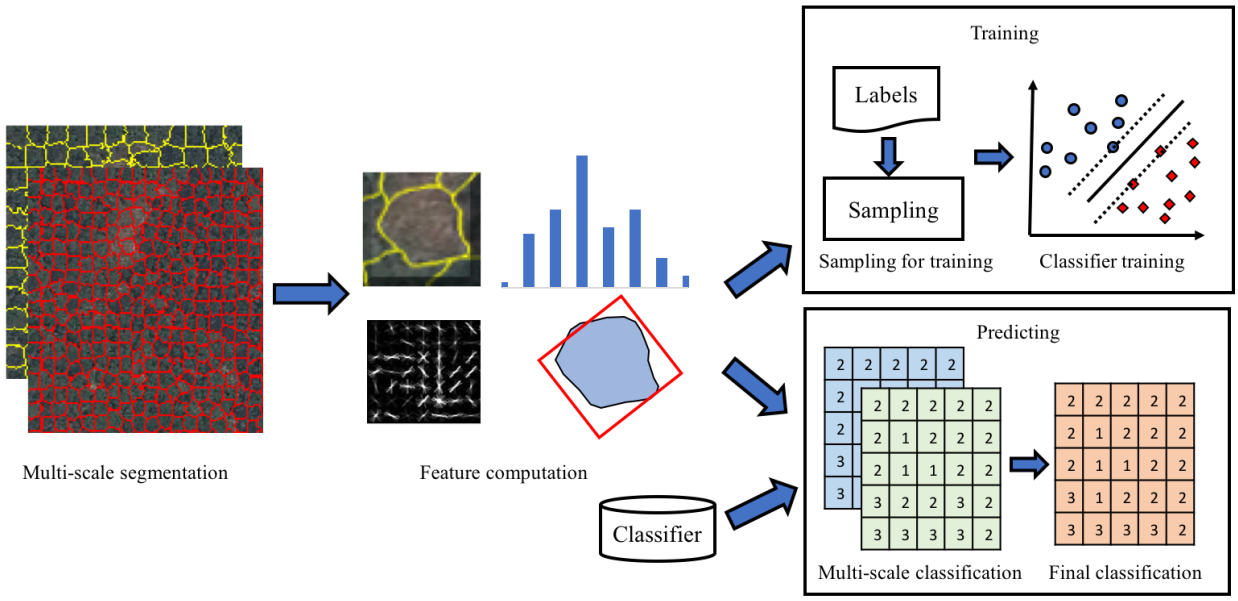


Figure 6-1: Technical overview of the landslide detection method.

6.3 Object-based landslide detection

6.3.1 Technical Overview

Figure 6-1 shows the overview of the object-based landslide detection method using RGB satellite images. First, the multiple scale image segmentation method is performed on the RGB

images to over segment the image into a large number of groups of pixels. Then various computer vision features (i.e., color, texture and shape features) are computed for each group of pixels. In the training stage, given the labeled data sets, sampling is performed to handle the imbalanced training dataset to make all classes have the same number of samples. Using the sampled datasets, the optimal classifiers for all the multi-scale segmentation results are learned. In the predicting stage, based on the multi-scale segmentation results and calculated superpixel features, the classifiers assign labels to each superpixel as well as each pixel for all the scales. The final labels for the image are obtained by performing majority voting over the classification results for all the scales.

6.3.2 Multi-scale Superpixel Segmentation

For the object based image analysis, the method of obtaining small image patches is significant as it should avoid clustering pixels from different classes into one object while aim to group pixels from the same class into one object. Therefore, an image patch usually contains pixels from one class and thus image patches from the same class have similar features so as to be categorized to the same class. Different from performing the region growing method to obtain image patches, the superpixel segmentation method is employed to extract superpixels (Ren and Malik 2003) which are a perceptually meaningful region of pixels (Veksler et al. 2010). The simple linear iterative clustering (SLIC) algorithm (Achanta et al. 2012) which is memory efficient and adheres to image boundaries is utilized to segment an image into small image patches, i.e., superpixels.

The SLIC method generates superpixels by performing clustering in a five dimensional (5D) space which contains L, a, b of the CIELAB color space and x, y of the pixel coordinates in the image. To compute the distance between two points in the 5D space, instead of using

Euclidean distances, a new distance computation method is utilized to integrate the Euclidean distances using the L, a, b components and the Euclidean distances using x, y while taking the compactness of a superpixel into consideration. Using this distance computation strategy, the improved k-means clustering is performed to group pixels into superpixels. To reduce the computational cost, the improved k-means only computes the distance between a pixel and some specific clusters that are at a certain range instead of all the clusters.

The SLIC method can enable the user to set the approximate number of labels in the output image, which allows multi-scale superpixel segmentation of the images. The multi-scale image segmentation method clusters pixels at different levels of detail and thus obtains different feature representations for objects. At different scales, the superpixels around a pixel usually have different sizes, which affects the corresponding features computation. For example, when the superpixel is small, a tree superpixel might be classified as landslide or ground as a small number of pixels within this superpixel cannot present distinctive features for trees. In this context, a large superpixel that groups more neighboring pixels can yield features that are more similar to tree features. Therefore, the multi-scale superpixel segmentation framework allows better classification of objects.

6.3.3 Feature Extraction

Considering the characteristics of landslides, this research computes the three categories of features from RGB images for detecting landslides: (1) color, (2) texture, (3) shape. The detailed feature representations and their computation for each category are discussed as follows.

6.3.3.1 Color Features

Color mean and covariance matrix: For a superpixel $\boldsymbol{sp}$, the color mean $\boldsymbol{m_c}$ and covariance matrix $\boldsymbol{C_c}$ are computed by treating each pixel as a vector containing R, G, and B values as follows:

$$\boldsymbol{m_c} = \frac{1}{|\boldsymbol{sp}|} \sum_{\boldsymbol{p} \in \boldsymbol{sp}} \boldsymbol{I}(\boldsymbol{p})$$

$$\boldsymbol{C_c} = \frac{1}{|\boldsymbol{sp}|} \sum_{\boldsymbol{p} \in \boldsymbol{sp}} (\boldsymbol{I}(\boldsymbol{p}) - \boldsymbol{m_c})(\boldsymbol{I}(\boldsymbol{p}) - \boldsymbol{m_c})^T$$

where $\boldsymbol{sp}$ is a superpixel, $|\boldsymbol{sp}|$ denotes the number of pixels in $\boldsymbol{sp}$, $\boldsymbol{p}$ is one of the pixels in $\boldsymbol{sp}$, $\boldsymbol{I}$ is an RGB image, $\boldsymbol{I}(\boldsymbol{p})$ is the a 3×1 vector containing the R, G, B channels of $\boldsymbol{I}$ at $\boldsymbol{p}$. The color covariance matrix of a superpixel is able to capture the relations between the three color channels of all the pixels with the superpixel. In addition, since the covariance matrix is obtained by subtracting the mean color, it is invariant to light conditions to some extent.

Color histogram: As the RGB satellite image contains three color channels, a histogram of the pixel intensity is computed for each channel for the pixels within a superpixel $\boldsymbol{sp}$. The color histogram of $\boldsymbol{sp}$ is obtained by concatenating the three histograms into a vector. The color histogram summarizes the color distribution within the superpixel, and can help to distinguish objects with different colors, especially to separate trees from grounds or landslides.

Color coherence vector: A color coherence vector (CCV) records the number of coherent and incoherent pixels at each discretized level where a pixel is coherent if it belongs to a contiguous region (Pass et al. 1996). For example, for a gray image, the color space is discretized into N (for example, $N = 2$) while the initial intensity space ranges from 1 to 256. Then, the discretized image only contains 1 if the initial color intensity is less than 127 and 2

otherwise. Based on the new image, the connected component analysis is performed to detect clusters with the same intensity. If a component contains more than τ pixels, all of its pixels are classified as coherent. Otherwise, all the pixels are labeled as incoherent.

For the discretized image, the number of coherent and incoherent pixels is computed, which yields a $N \times 2$ vector. In order to compute the CCV feature for an RGB image, for a gray image containing each channel of the RGB image, the CCV feature is calculated. The three CCV feature vectors are concatenated into one vector which corresponds to the CCV feature for the RGB image. To compute the CCV feature for a superpixel $\boldsymbol{sp}$, the bounding box of $\boldsymbol{sp}$ is utilized to extract a local patch from the RGB image, which is utilized to calculate the CCV feature. According to the computation strategy, the incoherent pixels represent isolated pixels within the image patch. Therefore, the CCV feature is able to provide finer distinction than the color histogram.

Green-red vegetation index: The Green-Red Vegetation Index (GRVI) (Motohka et al. 2010) is traditionally computed using near-infrared images to distinguish green vegetation from other objects (e.g., ground). In this project, based on RGB images, the adopted GRVI value (Rau et al. 2012) for each pixel is calculated as follows:

$$GRVI = \frac{Green - Red}{Green + Red}$$

For a superpixel, its GRVI value is computed as an average of all GRVI values of its pixels.

6.3.3.2 Texture Features

Grey-level co-occurrence matrix: The Grey-Level Co-occurrence Matrix (GLCM) (Haralick et al. 1973) is able to capture the probability of different combinations of pixel density in an image and allows to extract second order of statistical texture features (Albregtsen and

others 2008). This matrix and its derived features have been used in many classification and segmentation applications (Arzandeh and Wang 2002; Blaschke et al. 2014; Stumpf and Kerle 2011). An element $P_\delta(i,j)$ in GLCM represents the probability density of two neighboring pixels separated by distance $\delta = (dx, dy)$ and having gray level i and j. Based on the selection of distance δ computation, different co-occurrence matrices can be computed for an image. For example, when $\delta = (0,d),(-d,d),(d,0),(d,d)$ which correspond to the four directions $(0^o, 45^o, 90^o, 135^o)$ in a two dimensional (2D) grid, four co-occurrence matrices are computed for an image. Based on the GLCM, the following statistical parameters, energy, contrast, entropy, correlation and means and variances of all the parameters are calculated.

Histogram of oriented gradients: The histogram of oriented gradients (HOG) feature represents the edge and gradients within a local window by obtaining the histogram of its gradients orientation (Dalal and Triggs 2005). It divides the image window into regular small regions and computes the histogram of gradient directions or edge orientations for each small region. The combination of the histograms for all the regions is the HOG feature of the image window. Since it is computed on local cells, the HOG feature is invariant to geometric and photometric transformations.

6.3.3.3 Shape Features

To compute shape features for each superpixel $\boldsymbol{sp}$, the minimum enclosing rectangle $\boldsymbol{R}$ and the minimum enclosing circle $\boldsymbol{C}$ for all the pixels of the superpixel are computed. The first shape feature is computed as the ratio of the rectangle width to the rectangle height (assuming that the width is smaller than the height), i.e., $width(\boldsymbol{R})/height(\boldsymbol{R})$. This value reaches 1 if the rectangle is a square and if the rectangle is narrow and long, it approaches 0. The second shape feature is defined as the ratio of the rectangle area to the circle area, $area(\boldsymbol{R})/area(\boldsymbol{C})$. If the

pixels in the superpixel form a circle, this feature is 1. For a long and narrow rectangle, it will decrease. The third shape feature is the ratio of the rectangle width to the circle radius, $widht(\boldsymbol{R})/radius(\boldsymbol{C})$ while the fourth one is the density of $\boldsymbol{R}$, $|\boldsymbol{sp}|/area(\boldsymbol{R})$.

The last shape feature is computed using principal component analysis (PCA) results. For all the pixels within $\boldsymbol{sp}$, PCA computes the covariance matrix of all the pixel locations and estimates its eigenvalues and eigenvectors of the covariance matrix. Then the last feature is calculated as $\lambda_0/\lambda_1(\lambda_0 < \lambda_1)$.

6.3.4 Supervised Classification Methods

In this research, two popular classification methods, support vector machine (SVM) and random forest are explored to train a classifier for identifying landslides.

6.3.4.1 Support Vector Machine

A support vector machine (SVM) is a supervised discriminative classifier which separates different classes using hyperplanes. For a linearly separable two-class dataset, the SVM classifier finds the hyperplane to separate the two categories and utilizes the points that define the margins between two classes to represent the model. An SVM model contains significant point samples (i.e., support vectors) to represent the gaps between classes. By integrating the kernel trick, SVM allows non-linear classification by projecting the points into higher dimensional space.

6.3.4.2 Random Forest

A random forest classifier is a large collection of decision trees where each tree is trained using a random vector sample independently (Breiman 2001). By sampling at random with

replacement, a new training set is obtained from the original training set. Then a decision tree is trained on this new training set using random feature selection. At each node of the forest, a binary split is conducted for that node depending on the best split based on the selected input variables. When classifying, each decision generates a vote, and the classification results are obtained by using the majority vote. The random forest classifier can handle large databases and balance errors in class population for unbalanced datasets.

6.3.5 Sampling for Imbalanced Data

For a large satellite image, the landslide areas are limited in the image and have a small number of pixels compared to other objects (e.g., trees). This causes a serious imbalance in the training dataset which has lots of samples from non-landslide objects. Such imbalanced data can affect the classifier to generate more labels towards non-landslide objects while still achieving high classification accuracy. In order to deal with imbalanced data, sampling methods ranging from under-sampling to over-sampling are commonly utilized (Hoens and Chawla 2013).

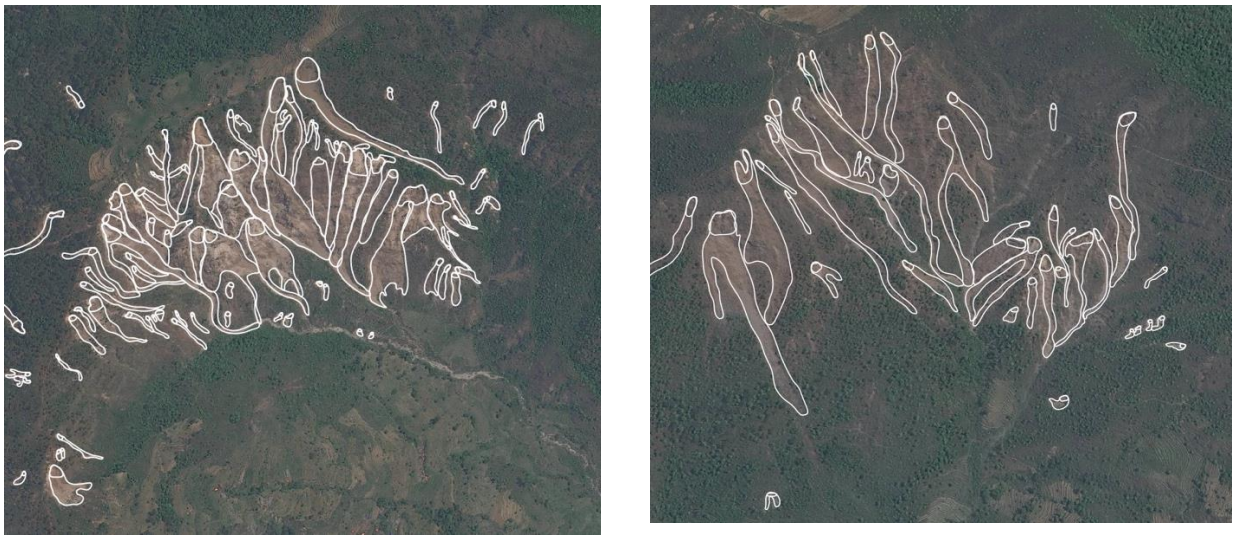
Some prior research suggests utilizing the under-sampling strategy to sample data from the majority class (Stumpf and Kerle 2011). However, under-sampling removes some of the training samples from classes with more samples, which might make the classifier fail to learn optimal parameters to identify data belonging to these categories. Considering the characteristics of the current data (i.e., the landslide class has fewer samples compared to other classes, and the appearance models of the landslide and ground are very similar.), the classifier needs to distinguish landslides from the ground. As under-sampling only utilizes some of the data from the ground class, the classifiers might not be optimized to separate landslides and ground. To estimate the optimal classifiers for this classification task, this work explores both over-sampling and under-sampling on the training dataset to decide the best sampling method.

The under-sampling method, NearMiss-3 (Zhang and Mani 2003) is employed to under-sample the majority classes (tree and ground in this work). This approach under-samples the data to make sure that the neighborhood of every position sample contains some negative samples, which can lead to high accuracy but low recall. The random-over sampling method is utilized to randomly select samples from the minority classes (mainly, the landslides in this work) with replacement. Thus, many landslide features are duplicated in the final training datasets. These two sampling methods generate different samples which are employed to train and test the classifiers so as to evaluate the influence of sampling on the classification performance.

6.4 Experiments

6.4.1 Study Area and Dataset

The experimental satellite images were collected by DigitalGlobe Worldview-2 on May 8 2015 in Barpak, Nepal after the April 25, 2015 M_w 7.8 Gorkha earthquake. The image resolution is 0.5m, and the landslide labels are obtained manually by visual interpretation. The areas of labeled landslides range from hundreds of pixels to millions of image pixels. Figure 6-2 shows two sample areas of satellite images and landslide labels within the white polygons. The area covered by the satellite images is a mountainous region mainly containing ground, trees and some rural residential areas, and lies within the longititudes from $84^o36'$E to $84^o51'$E, and the latitudes from $28^o1'$N to $28^o21'$N. The supervised classification task aims to recognize three types of objects from the images, i.e., landslides, trees, grounds. The latter two classes are also manually labeled from an area of the satellite images for training.



(a) Sample area 1. (b) Sample area 2.

Figure 6-2: Sample data and manually labeled landslides.

6.4.2 Experimental Setup

As the size of most of the original satellite images, (13,684x13,684 pixels) is too large for processing, each image is split into 16x16 sub-images with a size of 1,200x1,200 pixels. There exists overlapping between neighboring sub-images in order to generate accurate and continuous labels in the boundaries of the sub-images. For each sub-image, the SLIC superpixel segmentation in the scikit-image library (van der Walt et al. 2014) is performed at five scales by setting the approximate number of superpixels per image as follows: $400, 600, 800, 1{,}000, 1{,}200$. All the features are computed based on the multi-scale superpixel segmentation results. To generate the training data, from one satellite image, some trees and ground regions are manually labeled as well as the landslides, which create ground truth labels for some pixels. Then, the ground truth label of a superpixel is determined by the max voting strategy. To avoid overfitting,

the ground truth superpixels are randomly split into a training and testing set (in this work, 67% of the ground truth data are used for training, while the remaining for testing).

In order to find the optimal parameters for the classifiers and avoid overfitting, the parameter estimation using grid search with cross-validation (GridSearchCV) toolkit in scikit-learn (Pedregosa et al. 2011) is employed. For each parameter combination, the GridSearchCV toolkit performs a 3-fold cross-validation (the data is divided into three consecutive folds and each fold is used once as a validation set while the other two are for training.) to evaluate the classifier in terms of precision and recall on the dataset. For the SVM classifier, the linear and radial basis function (RB) kernels are both explored with various parameter combinations. After the cross-validation, the best parameters for the classifiers are determined and employed to train the classifier on the training datasets. The GridSearchCV method is conducted for both the classifiers for each scale to estimate the best parameter settings.

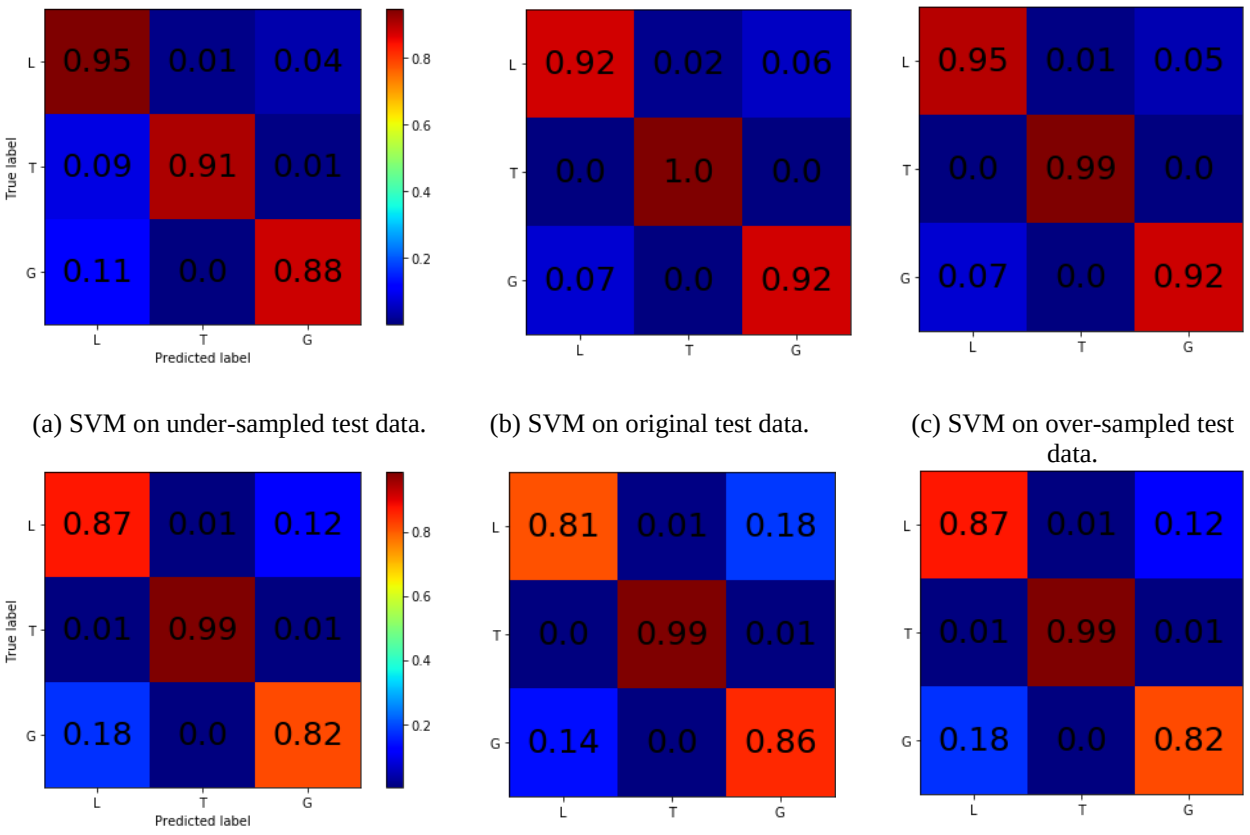
6.4.3 Performance on the Test Data

As shown in Figure 6-3, the two classifiers SVM and Random Forest (RF) trained using the under-sampled, the original without any sampling method, and the over-sampled training datasets, are evaluated by the testing dataset in terms of the confusion matrix, respectively. A confusion matrix shows the number or ratio of correct and incorrect predictions for each class, which reflects the performance of the classifier on the dataset. The imbalanced original test dataset have more samples for trees and grounds. The under-sampling method removes a certain amount of samples from the trees and grounds and makes all the classes have the same number of samples while the over-sampling method adds samples for the minority classes (landslide and tree). The classifiers are trained using the three different training datasets and evaluated using the same imbalanced test datasets.

By comparing the two rows in Figure 6-3, it can be found that the SVM classifier achieves better classification results on the test dataset than the Random Forest (RF) classifier for all the three training datasets. The RF classifier is unable to distinguish between landslides and grounds well, and thus achieves lower accuracy on them compared to SVM.

The difference between the three columns for the first row in Figure 6-3 demonstrates that the over-sampling method yields better classification results compared to the under-sampling method or just using the original training dataset without any sampling. As aforementioned, the over-sampling keeps all training samples for the majority classes (tree and ground) while randomly duplicating samples from the minority class (landslide), which enables the classifier to increase the capability to identify the majority classes better, specifically ground, and thus distinguish it better from landslides.

Table 6-1 shows the classifier performance in terms of F1 scores where $F1 = 2\frac{P*R}{P+R}$ while P is the precision and R is the recall. As shown in Table 6-1, the classifiers trained on the over-sampled training datasets outperform those trained on the original and under-sampled datasets, especially for the landslides. By adding more landslide samples, over-sampling the training dataset also allows the classifier to gain more information on the landslides. As shown in Table 6-1, the SVM classifier constructed on the over-sampled training dataset surpasses the others, which is consistent with Figure 6-3.



(a) SVM on under-sampled test data.

(b) SVM on original test data.

(c) SVM on over-sampled test data.

(d) RF on under-sampled test data.

(e) RF on original test data.

(f) RF on over-sampled test data.

Figure 6-3: Confusion matrices for the test dataset at Scale 5 for SVM and Random Forest.

Table 6-1: F1 scores of the classifiers on the test dataset.

	SVM			Random Forest (RF)		
	under	origin	over	under	origin	over
Landslide	0.82	0.92	0.93	0.82	0.81	0.92
Tree	0.95	0.99	1.00	0.99	0.99	0.99
Ground	0.92	0.93	0.94	0.84	0.84	0.85
Average	0.90	0.95	**0.96**	0.89	0.88	0.92

The confusion matrices in Figure 6-3 also reflect the classification performance for each class. As shown in Figure 6-3, the confusion matrix, the tree is at 99% correctly classified for all the six classifiers. Compared to the other two classes, landslide and ground, trees have distinct features in terms of color and texture. For example, within a tree superpixel, the green channel dominates over the other channels and thus the color features are quite different from those of superpixels belonging to either landslide or ground. Regarding the texture, a tree superpixel is more texturally isotropic compared to landslide or ground. Therefore, the classifiers can easily identify trees from the other two classes while making fewer mistakes of labeling the other two classes as trees.

Figure 6-3 also demonstrates that the classifier tends to confuse between ground and landslide. As the satellite image resolution is 0.5m per pixel, the landslide and the ground (e.g., roads) have a very similar visual appearance. Due to the characteristics of landslides (e.g., rough surface in a local area), the elevation distribution of the local area is an ideal feature if the elevation data (e.g. DEM) are available. If more accurate images are captured, detailed textural features of landslides can also enable the classifiers to learn more discriminative and useful features for the two classes. However, the confusion matrices on the test dataset demonstrate that the classifiers trained only using RGB images can still correctly identify at least 89% of the landslides.

In summary, the confusion matrices on the test dataset indicate that (1) trees are usually correctly recognized, and (2) the landslide and ground might need to exchange label in some cases. Moreover, based on these results, the methods of improving the landslide detection accuracy can utilize neighboring superpixel features or labels, and improve the identification of ground as there exists an abundance of samples from the ground class.

6.4.4 Classification Performance on the Whole Dataset

Since the primary objective of this chapter is to detect potential landslide areas, the accuracy of the landslide detection results is evaluated while the tree and ground accuracies are not discussed. As this research aims to detect correct landslide candidate areas, this work evaluates the landslide classification accuracy by computing how many landslides are correctly identified. The connected component analysis is utilized to calculate the number of landslides in the ground truth labels and the predicted labels. If a ground truth cluster contains at least one pixel that is predicted as the landslide by the proposed method, the cluster in the predicted labels containing that pixel is viewed as a correct landslide.

The recall accuracies evaluated in terms of how many landslides are correctly identified are 90.7% and 90.1% (from totally 2,081 landslides) by the SVM classifiers and the RF classifiers respectively. The high recall accuracy demonstrates that based on the RGB images, the proposed landslide detection method is able to identify most of the landslides. Moreover, the results also indicate that the RF classifiers achieve almost the same accuracy as the SVM classifier on the whole dataset even though its accuracy on the testing datasets is smaller than the SVM classifiers.

However, the proposed method generates many false alarms due to the fact that the method wrongly classifies many ground pixels into landslide. As indicated in Section 4.3, the classifiers tend to make mistakes about the ground and landslide labels. There are two main reasons for this phenomenon: (1) the visual appearance features of the landslide and ground are very similar sometimes; (2) the proposed method will fail to detect small, or long and narrow landslides.

For the study area, as the landslides usually happen near forest areas and cause movements of a mass of earth or rock, the landslide areas have similar visual appearance compared to some of the bare ground in mountaineous regions of the vicinity. Figure 6-4 shows an example of similar visual appearance of landslides and ground. The red rectangle in Figure 6-4 covers the area for a landslide while the green one includes an area of a section of regular ground. Regarding the color features utilized in this work, their color representations are very similar to each other as they have a similar color appearance by comparing the close-up look images in the middle column in Figure 6-4.

Even though the shape features are dependent on the multi-scale segmentation results, according to the shape feature computation (fitting the bounding rectangle and circle), it can be inferred that their shape features are similar too. By carefully scrutinizing the two patches, the contextures of the landslide seem to be different as the landslide area has more elevation changes which lead to more edges. However, due to the limited resolution and a lack of elevation data, these differences computed from RGB images are not sufficient to allow the classifiers to distinguish them. Therefore, the ground is labeled as landslide in the final results as shown in the right image in Figure 6-4.

Figure 6-4: Example of similar visual appearance of landslide and ground.

Since the multi-scale superpixel segmentation is performed to compute superpixels with different levels of details, a small landslide will have more neighboring pixels at a large scale where a superpixel usually contains more pixels than that on a small scale. As some of the neighboring pixels do not belong to landslide, they will affect the features computation in the superpixel and thus cause wrong labels in the predicting stage. Regarding the long and narrow landslide, since the SLIC algorithm tends to find regular superpixels, the method tends to detect several small superpixels for the long and narrow landslide, which will lead to problems for the small landslides as aforementioned above. Thus, some of the long and narrow landslides are neglected by the proposed method.

6.5 Conclusion

This chapter presents an object-based landslide detection method using only RGB satellite images using supervised classifiers. The multi-scale superpixel segmentation method is performed on the RGB images to find superpixels which are perceptually meaningful groups of pixels. The various visual features (i.e., color, contexture, and shape) are computed for each superpixel. In the training stage, the over-sampling method is utilized to handle the imbalanced data while the SVM classifiers and the RF classifiers are employed for comparison. The experimental results on the datasets demonstrate the proposed method can correctly identify 90% of the landslides and the two classifiers have similar performance on the datasets.

However, due to the limitation of features provided from RGB images, the proposed method finds many false alarms in regular ground regions. Moreover, the inherent features of the segmentation method lead to failures on the small, and the long and narrow landslides. Future work will explore the utilization of prior knowledge or other data sources to eliminate false alarms using this method. In addition, the proposed method will be tested on RGB images collected by airborne platforms or drones to allow for automatic landslide data collection.

Chapter 7

Conclusions

7.1 Summary of Research Methods

This dissertation primarily aimed to address issues and challenges in 3D reconstruction and modeling using low-precision vision sensors for both outdoor and indoor construction automation and robotics applications. Since occlusion, which is especially critical in indoor environments, can cause several problems for 3D reconstruction, modeling, and further analysis (e.g., object recognition, scene understanding), a joint point cloud completion and surface relation inference method is proposed to recover the missing points and infer the surface relations by integrating the visibility information and the surface geometric properties. This research also designed a user-guided dimensional analysis system to utilize prior knowledge of the scenes and the sensors in order to interactively obtain complete frames for estimating the dimensions of interest.

For facilitating worker-robot interaction, a human tracking framework from a single RGB-D sensor is proposed to combine an online learning method and various features to effectively detect and track a specific individual under various illumination conditions. This research also investigated the usage of drones for monitoring earthwork safety by proposing a comprehensive data processing scheme that involves constructing 3D point clouds from the video images, obtaining terrain models, and slope analysis. In addition, to allow a drone to

automatically map geotechnical hazards, this dissertation presented an efficient method of detecting landslides from only RGB images in order to identify potential hazard areas.

7.2 Research Contributions

This research contributes to construction automation and robotics literature by investigating 3D reconstruction and modeling using low-precision vision sensors. The researched methods can be readily integrated into robotic platforms for construction projects that need 3D as-built models, important dimensional information, frequent site monitoring and facility management. The research for detecting geotechnical hazards can facilitate further work on developing automated landslide detection and mapping by drones.

The specific research contributions and tangible outcomes of this dissertation that were described in the preceding chapters are summarized as follows:

- A general-purpose point cloud completion system that is able to correctly recover missing point clouds and generate 3D complete models for handling occlusion in indoor environments.
- A user-guided dimensional analysis system that can obtain complete frames to compute dimensions by providing correct guidance for facility management as well as investigation for integration of domain knowledge.
- A general-purpose human tracking framework that can detect and track a specific individual in real-time in various illumination conditions.
- An effective excavation slope stability monitoring system using 3D reconstruction and modeling from images collected by drones.

- An efficient object-based landslide detection method from RGB images which enables automatic landslide detection and mapping using drones.

7.3 Future Research Directions

This research focused on investigating 3D reconstruction and modeling using low-precision vision sensors in construction automation and robotics to provide 3D models. There exist certain limitations for the research methods that have been mentioned in the preceding chapters. These limitations provide the following directions for future research: (1) point cloud completion for complicated scenes, (2) user-guidance systems for complex scenes, (3) multiple human tracking using sensor fusion. In addition, for intelligent automation or robotic systems, it is significant to have the capability of scene understanding using 3D data.

7.3.1 Point Cloud Completion for Complicated Scenes

The point cloud completion method proposed in this dissertation utilizes the geometric properties of planar surfaces to find missing points between them and within individual planar surfaces, while the nonplanar surfaces are processed using a naive strategy. This method can be applied for various indoor applications (e.g., reconstructing as-built BIMs). However, for many civil engineering environments (e.g., utility tunnels with pipes, outdoor construction sites), there exist some nonplanar surfaces which are of significance for the applications. Therefore, it is necessary to design point cloud completion algorithms for certain nonplanar surfaces (e.g., cylinders and spheres) by utilizing the geometric characteristics of these nonplanar surfaces.

7.3.2 User-Guidance Systems for Complex Scenes

The user-guidance system in this dissertation was tested on three scenes that are composed of planar surfaces. As discussed in the previous subsection, for some environment applications involving nonplanar surfaces, the dimensions of objects that contain nonplanar surfaces are of interest as well. Therefore, there is a need to develop a specified user-guidance system for these dimensions by defining the dimensions of interest and designing the complete template for the scenes. In addition, the current user-guidance system utilizes low-cost RGB-D sensors which have a limited precision which is inappropriate for some civil engineering applications that require higher precision. More accurate sensors (e.g., laser scanners) can be integrated into the user-guidance framework by incorporating the sensor features into the design of the user-guidance generation.

7.3.3 Multiple Human Tracking Using Sensor Fusion

The current human tracking system utilizes only a single RGB-D sensor to detect and track one specific individual. However, there might be multiple people surrounding the robots and it is necessary to detect and track all the persons for some applications. In addition, a single RGB-D sensor has a limited field of view and observation range, which affects the applicability of this system. By integrating multiple sensors (e.g., laser scanners, RGB cameras), a robot can capture the surrounding environments with a large field of view and provide data for efficient multiple human detection and tracking.

7.3.4 Scene Understanding Using 3D Data

The results of this research are primarily 3D models and related information (i.e., surface connections) or direct uses of the 3D models (e.g., obtaining dimensions, computing slopes)

from low-precision vision sensors. However, it is also important for a robot to interpret and understand the environments intelligently (e.g., recognizing objects in the environment, and identifying their activities and interactions). The unstructured, cluttered, and dynamic environments in construction present challenges for developing appropriate scene understanding methods. By incorporating 3D data and models, automation and robotic systems can obtain comprehensive representations of the environment and effectively interpret the environment using multiple features computed from 3D data. Thus, future work will explore the design of scene understanding methods for ARC using 3D data.

APPENDICES

APPENDIX A Introduction to RGB-D Sensors

An RGB-D camera (e.g., Microsoft® Kinect XBox 360, AUSU Xtion PRO LIVE) is equipped with an infrared IR emitter, a color (RGB) camera and an IR depth sensor as shown in Figure 8-1. With the assistance of the IR emitter, the IR depth sensor is able to capture a depth image where each pixel contains the depth from the point to the sensor. Therefore, the IR depth sensor is also referred to as a depth camera. Meanwhile, the RGB camera can obtain a color image where each pixel contains a color represented by the red, green, and blue components. By using the intrinsic parameters of the depth camera, the 3D point cloud can be derived from the depth image. With the relative transformation between the depth and RGB cameras, each valid point of the 3D point cloud can be associated with a color from the RGB image.

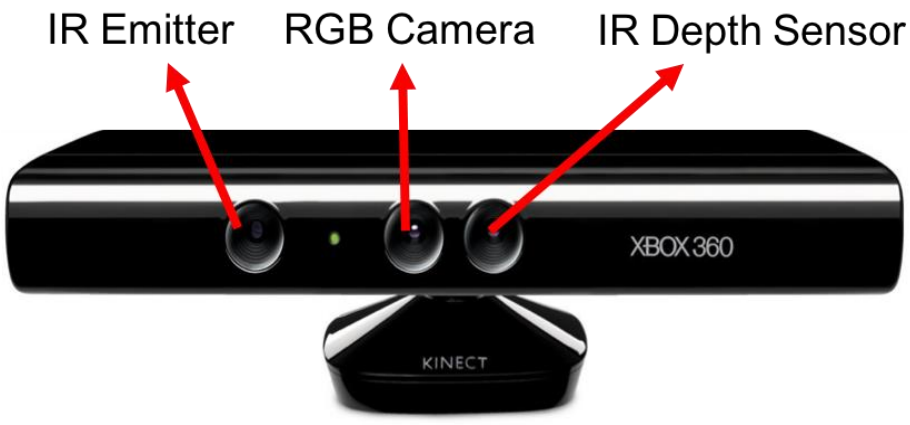


Figure 8-1: An example of RGB-D sensor

The 3D colored point cloud obtained by RGB-D sensors is also called an organized point cloud because each point is associated with a unique 2D index. It can be viewed as an image where each pixel contains a 6×1 vector $[x, y, z, R, G, B]$ where (x, y, z) is the coordinates of the point in the coordinate frame of the RGB-D sensor and (R, G, B) is the color of the point. Therefore, the 3D color point cloud captured by RGB-D sensors is also referred to as an RGB-D image.

If the intrinsic parameters, i.e., the camera matrix that defines the mapping of a pinhole camera from 3D points in the world to 2D points in an image, of the depth camera are known, the 3D point cloud can be reconstructed from a depth image. Assume that camera matrix of the depth camera is $\boldsymbol{K_d}$,

$$\boldsymbol{K_d} = \begin{bmatrix} f_{x_d} & 0 & c_{x_d} \\ 0 & f_{y_d} & c_{y_d} \\ 0 & 0 & 1 \end{bmatrix} \tag{8.1}$$

where the focal lengths are f_{x_d} and f_{y_d}, and the principal point coordinates are c_{x_d}, c_{y_d}. Using the pinhole camera geometry model, a point in the 3D world $\boldsymbol{p} = (x, y, z)$ is projected on to a point $\boldsymbol{p'} = (u, v)$ within the image where u, v are the image coordinates. Thus, $\boldsymbol{p}$ and $\boldsymbol{p'}$ satisfie $\boldsymbol{p} = \boldsymbol{K_d} \cdot \boldsymbol{p'}$, that is,

$$\begin{cases} u = \dfrac{f_{x_d} x}{z} + c_{x_d} \\ v = \dfrac{f_{y_d} y}{z} + c_{y_d} \end{cases} \tag{8.2}$$

Therefore, when the depth z is known, the 3D points can be retrieved from the depth image as follows,

$$\begin{cases} x = \dfrac{u - c_{x_d}}{f_{x_d}} z \\ y = \dfrac{v - c_{y_d}}{f_{y_d}} z \end{cases} \tag{8.3}$$

Note that z denotes the depth from the point to the sensor. Thus, (x, y, z) are the coordinates of $\boldsymbol{p}$ with respect to the coordinate system of the RGB-D sensor.

To get colored 3D point clouds, each 3D point should be assigned RGB values captured by the RGB camera. For this stereo camera system, this can be realized by utilizing the transformation (rotation and translation) between the RGB camera and the depth camera. Consider that the rotation and translation from the depth sensor to the RGB camera are $\boldsymbol{R}$ and $\boldsymbol{T}$, respectively, where $\boldsymbol{R}$ is a 3x3 rotation matrix and $\boldsymbol{T}$ is a 3x1 translation vector. Then, each 3D point can be projected on to the RGB image using the following procedure:

(1) Translate the point by T

$$\boldsymbol{q} = \boldsymbol{R}\boldsymbol{p} + \boldsymbol{T} \tag{8.4}$$

(2) Project the point on to the RGB image by using Equation (8.3) where f_{x_c} and f_{y_c} are the focal lengths of the RGB camera, and c_{x_c} and c_{y_c} are the principal points.

$$\begin{cases} u_c = \dfrac{f_{x_c} \boldsymbol{q}'_x}{\boldsymbol{q}'_z} + c_{x_c} \\ v_c = \dfrac{f_{y_c} \boldsymbol{q}'_y}{\boldsymbol{q}'_z} + c_{y_c} \end{cases} \tag{8.5}$$

Once the corresponding image coordinates of this point on the RGB image are obtained, RGB values can be associated with this point. Thus, the 3D colored point cloud is computed using a depth image and a color image.

When an RGB-D sensor is factory-assembled, the IR depth sensor and the RGB camera are fixed and thus there exist default parameters for the two cameras, including the intrinsic parameters of both cameras and their relative transformation relations. However, due to imperfections in the manufacturing process, these default parameters cannot be expected to be exact for all RGB-D sensors. Therefore, it is necessary to calibrate the RGB-D sensor if it is used for applications that require high and repeatable accuracy.

The sensor calibration aims to obtain the intrinsic and extrinsic (the transformation between the depth and RGB cameras) parameters of an RGB-D sensor in order to obtain accurate 3D colored point clouds from the sensor. By viewing the RGB-D sensor as a stereo camera system, the stereo camera calibration method can be utilized to calibrate the RGB-D sensor and obtain its intrinsic parameters and extrinsic parameters.

APPENDIX B 3D Reconstruction Using Structure from Motion

1. Introduction to Structure from Motion

As indicated by Equation (8.2), if only the image location of a point is known, (u, v), it is impossible to compute its 3D coordinates (x, y, z). Therefore, a single RGB camera is unable to obtain 3D information from the image. A stereo camera system can utilize the triangulation to compute 3D coordinates for a point if it is observed by both cameras, and the intrinsic and extrinsic parameters of the stereo camera system are known. However, if a single camera is utilized to capture a series of images for an object, it is possible to reconstruct the 3D structures of the object by estimating the camera motions. The method of obtaining 3D information from multiple view images is called structure from motion (SFM).

If some images $\boldsymbol{I_S}$ observe the same point $\boldsymbol{p} = (x, y, z)$ and its image coordinates in the images are known, $\boldsymbol{p_i'} = (u_i, v_i) \in \boldsymbol{I_i}, \boldsymbol{I_i} \in \boldsymbol{I_S}$, the following equations can be obtained:

$$\begin{cases} x_i = \dfrac{u_i - c_x}{f_x} z_i \\ y_i = \dfrac{v_i - c_y}{f_y} z_i \end{cases}, \quad \boldsymbol{I_i} \in \boldsymbol{I_S} \tag{8.6}$$

where f_x, f_y, c_x, c_y are the instrinsic parameters of the camera as in Equation (8.1), and $\boldsymbol{p_i} = (x_i, y_i, z_i)$ denotes the 3D coordinates of $\boldsymbol{p}$ with respect to the camera center. Meanwhile, the realtions between $\boldsymbol{p_i}, \boldsymbol{I_i} \in \boldsymbol{I_S}$ can also lead to equations related to the camera poses in the world. For example, for $\boldsymbol{I_i} \in \boldsymbol{I_S}, \boldsymbol{I_j} \in \boldsymbol{I_S}$, $\boldsymbol{p_i} = \boldsymbol{R_j^i p_j} + \boldsymbol{T_j^i}$ where $\boldsymbol{R_j^i}$ and $\boldsymbol{T_j^i}$ represent the roation and translation of the two camera coordiates from the j-th postion to the i-th postion, respectivly. Base on a sufficient number of the aforementioned equations, the point locations (3D structures) and the transformations between cameras (camera motions) can be estimated.

However, it is impossible to recover the absolute scale of point clouds reconstructed from SFM. If we scale the entire scene by k, and scale the camera matrices by the factor of $1/k$, all the equations remain the same. One common approach to obtain the absolute scale is to take some measurements in the real world and use them to scale the reconstructed 3D point clouds from SFM.

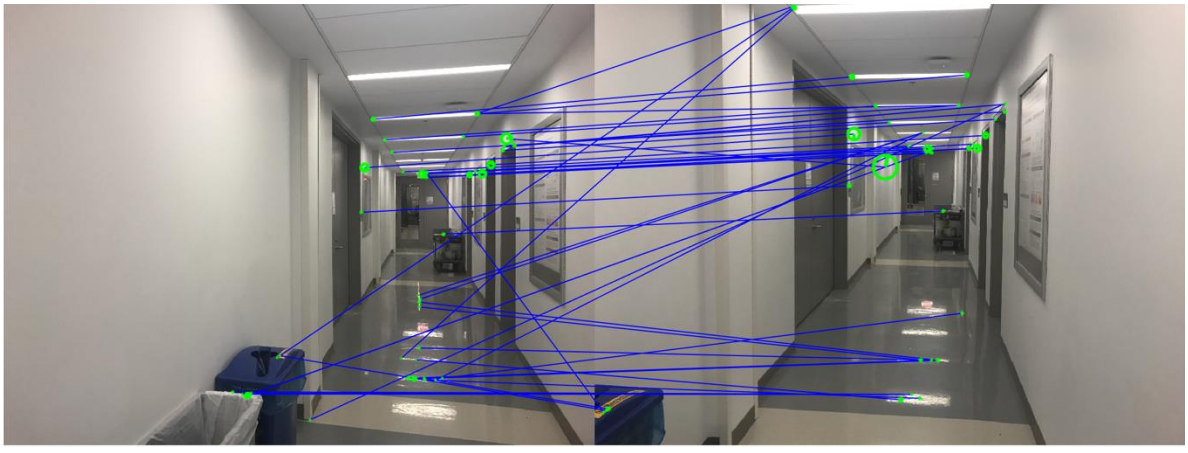
2. Challenges for SFM in Indoor Environments

As aforementioned, identifying the point pairs from multiple images is crucial for SFM to create correct and sufficient equations. A commonly used approach is to extract various distinctive points (e.g., corners points) using feature representations, e.g., Harr, SIFT, and SURF, and then search point pairs by comparing the feature representations. Among these feature representations, the SIFT feature descriptor is one of the most popular because it is invariant to uniform scaling, orientation, illumination changes, and partially to affine distortion (Lowe 1999). It has been proven to be efficient in 3D reconstruction and modeling for many outdoor applications as mentioned in Chapter 1 and Chapter 6.

However, as the indoor environment contains many featureless objects or repetitive patterns, it is challenging to obtain correct point pairs. By using the SIFT features as an example, Figure 8-2 (b) shows the some of SIFT feature points of a hallway in a campus building. To perform SFM, it is necessary to find correct point pair from images so as to estimate the camera motions for 3D reconstruction. Therefore, the feature points detected should be unique or distinctive. However, as shown in Figure 8-2 (b), some of the feature points are not unique or distinctive. For example, many features points are detected on the floors due to the reflections of the light. By comparing the left and right images in Figure 8-2 (b), it can be found that the reflections move on the floor and thus the corresponding features are not reliable.

(a) Original images.

(b) SIFT features points (the centers of the circles).



(c) Point matching by using the SIFT features.

Figure 8-2: An example of feature point matching issues in indoor environments.

Based on these feature points, it is difficult to find a sufficient number of correct point pairs to perform SFM and moreover, many wrong point pairs will generate unreliable 3D reconstruction results. As shown in Figure 8-2 (c), the features points on trash cans (at the left corner of the left image) should not be matched to any points as the corresponding points are not observed in the right image. In addition, as discussed above, the features points on the ground floor are not reliable and thus the related point pairs are not reliable for SFM. As shown in Figure 8-2 (c), many point pairs do not contain the same points in the world, and therefore SFM is unable to reconstruct the correct 3D point clouds based on these point pairs.

It should be noted that if we capture images for a room that contain different objects, correct point pairs are likely to be found for performing SFM to obtain 3D sparse point clouds for that room. For example, Furukawa et al. (2009) presented a fully automated system for architectural scene reconstruction and visualization for challenging textureless scenes (e.g. indoor scenes).

REFERENCES

Ahmed, M. F., Haas, C. T., and Haas, R. (2014). "Automatic detection of cylindrical objects in built facilities." *Journal of Computing in Civil Engineering*, 28(3), 04014009.

Ali, B., Iqbal, K. F., and Ayaz, Y. (2013). "Human detection and following by a mobile robot using 3d features." *\ldots and Automation (ICMA)*.

Arnaud, A., Christophe, J., Gouiffes, M., and Ammi, M. (2016). *3D reconstruction of indoor building environments with new generation of tablets*.

Azhar, S. (2011). "Building Information Modeling (BIM): trends, benefits, risks, and challenges for the AEC industry." *Leadership and Management in Engineering*, 11(3), 241--252.

Bae, H., Golparvar-Fard, M., and White, J. (2014). "Image-Based Localization and Content Authoring in Structure-from-Motion Point Cloud Models for Real-Time Field Reporting Applications." *Journal of Computing in Civil Engineering*, 637--644.

Barbu, A., and Zhu, S.-C. (2005). "Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8), 1239--1253.

Batista, G. E. A. P. A., Prati, R. C., Monard, M. C., Kegelmeyer, W. P., Ambrogi, F., Biganzoli, E., Gariboldi, M., Pierotti, M. A., Harris, A. L., Liu, E. T., Hamada, K., Nakayama, H., Ishitsuka, H., Miyamoto, T., Hirabayashi, A., Uchimura, S., Hamamoto, Y., Lander, E. S., Aster, J. C., and Tr, G. (2004). "A study of the behavior of several methods for balancing machine learning training data." *ACM SIGKDD Explorations Newsletter*, 6(1), 20.

Bay, H., and Tuytelaars, T. a. (2006). "SURF: Speeded Up Robust Features." 404--417.

Bennett, T. (2009). "BIM and laser scanning for as-built and adaptive reuse projects: the opportunity for surveyors." *The American Surveyor*, 6(6), 15.

Bentley, J. L. (1975). "Multidimensional Binary Search Trees Used for Associative Searching." *Commun. ACM*, 18(9), 509--517.

Bernardini, F., and Bajaj, C. L. (1997). "Sampling and Reconstructing Manifolds Using Alpha-Shapes." *Purdue e-Pubs, a service of the Purdue University Libraries*, 1--11.

Besl, P. J., and McKay, N. D. (1992). "A Method for Registration of 3-D Shapes." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), 239--256.

Bhatla, A., Choe, S. Y., Fierro, O., and Leite, F. (2012). "Evaluation of accuracy of as-built 3D modeling from photos taken by handheld digital cameras." *Automation in Construction*, 28, 116--127.

Bodor, R., and Jackson, B. "Vision-based human tracking and activity recognition." *Proc., Proceedings of the 11th Mediterranean Conference on Control and Automation*, 18-20.

Borrmann, A., and Rank, E. (2009). "Specification and implementation of directional operators in a 3D spatial query language for building information models." *Advanced Engineering Informatics*, 23(1), 32--44.

Bosch (2010). "Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction." *Advanced Engineering Informatics*, 24(1), 107--118.

Braun, A., Tuttas, S., and Borrmann, A. (2015). "Automated progress monitoring based on photogrammetric point clouds and precedence relationship graphs." *ISARC Proceedings of \ \ldots *.

Brilakis, I., Fathi, H., and Rashidi, A. (2011). "Progressive 3D reconstruction of infrastructure with videogrammetry." *Automation in Construction*, 20(7), 884--895.

Brilakis, I., Lourakis, M., Sacks, R., Savarese, S., Christodoulou, S., Teizer, J., and Makhmalbaf, A. (2010). "Toward automated generation of parametric BIMs based on hybrid video and laser scanning data." *Advanced Engineering Informatics*, 24(4), 456--465.

Budroni, A., and Boehm, J. (2010). "Automatic 3d Modelling Of Indoor Manhattan-world Scenes From Laser Data." *ISPRS Symp. Close Range Image Measurement Techniques*.

Cabral, R., and Furukawa, Y. "Piecewise Planar and Compact Floorplan Reconstruction from Images." *Proc., Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 628-635.

Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C., and Evans, T. R. (2001). "Reconstruction and representation of 3D objects with radial basis functions." *Proceedings of Annual Conference on Computer Graphics and Interactive Techniques*, 67--76.

Carraro, M., Munaro, M., and Menegatti, E. (2016). "Cost-efficient RGB-D smart camera for people detection and tracking." *Journal of Electronic Imaging*, 25(4), 041007--041007.

CGAL (2016). "Computational Geometry Algorithms Library." <http://www.cgal.org/>. (Feb 7, 2016).

Chauve, A. L., Labatut, P., and Pons, J. P. (2010). "Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data." *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1261--1268.

Chen, K., Lai, Y.-K., and Hu, S.-M. (2015). "3D indoor scene modeling from RGB-D data: a survey." *Computational Visual Media*, 1(4), 267-278.

Cho, Y. K., and Gai, M. (2014). "Projection-Recognition-Projection Method for Automatic Object Recognition and Registration for Dynamic Heavy Equipment Operations." *Journal of Computing in Civil Engineering*, 27(5), 511--521.

Chung, W., Kim, H., Yoo, Y., Moon, C.-B., and Park, J. (2012). "The Detection and Following of Human Legs Through Inductive Approaches for a Mobile Robot With a Single Laser Range Finder." *IEEE Transactions on Industrial Electronics*, 59(8), 3156--3166.

Coughlan, J. M., and Yuille, A. L. (2003). "Manhattan World: Orientation and Outlier Detection by Bayesian Inference." *Neural Computation*, 15(5), 1063--1088.

Curless, B., and Levoy, M. (1996). "A volumetric method for building complex models from range images." 303--312.

Dalal, N., and Triggs, B. (2005). "Histograms of Oriented Gradients for Human Detection." *Cvpr*, 1, 886--893.

Dang, Q. K., and Suh, Y. S. (2011). "Human-following robot using infrared camera." *Control*.

Daum, S., and Borrmann, A. (2014). "Processing of Topological BIM Queries using Boundary Representation Based Methods." *Advanced Engineering Informatics*, 28(4), 272--286.

Demir, I., Aliaga, D. G., and Benes, B. (2015). "Procedural Editing of 3D Building Point Clouds." *Iccv*, 2147--2155.

Díaz-Vilariño, L., Khoshelham, K., Martínez-Sánchez, J., and Arias, P. (2015). "3D Modeling of Building Indoor Spaces and Closed Doors from Imagery and Point Clouds." *Sensors*, 15(2), 3491-3512.

Dimitrov, A., and Golparvar-Fard, M. (2014). "Robust NURBS Surface Fitting from Unorganized 3D Point Clouds for Infrastructure As-Built Modeling." *Computing in Civil and Building Engineering*, 81-88.

Everingham, M., Van Gool, L. J., Williams, C. K. I., Winn, J. M., and Zisserman, A. (2010). "The Pascal Visual Object Classes (VOC) Challenge." *International Journal of Computer Vision*, 88(2), 303--338.

Fathi, H., and Brilakis, I. (2011). "Automated sparse 3D point cloud generation of infrastructure using its distinctive visual features." *Advanced Engineering Informatics*, 25(4), 760--770.

Feng, C., Taguchi, Y., and Kamat, V. R. (2014). "Fast plane extraction in organized point clouds using agglomerative hierarchical clustering." 6218--6225.

Furukawa, Y., Curless, B., Seitz, S. M., and Szeliski, R. (2009). "Reconstructing building interiors from images." 80--87.

Furukawa, Y., and Ponce, J. (2010). "Accurate, Dense, and Robust Multiview Stereopsis." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8), 1362--1376.

Ghidary, S. S., Nakata, Y., Takamori, T., and Hattori, M. (2000). "Human detection and localization at indoor environment by home robot." 1360--1365.

Giel, B., and Issa, R. R. A. (2012). "Using Laser Scanning to Access the Accuracy of As-Built BIM." 665--672.

Golparvar-Fard, M., Bohn, J., Teizer, J., Savarese, S., and Peña-Mora, F. (2011). "Evaluation of image-based modeling and laser scanning accuracy for emerging automated performance monitoring techniques." *Automation in Construction*, 20(8), 1143-1155.

Golparvar-Fard, M., Pena-Mora, F., and Savarese, S. (2011). "Monitoring changes of 3D building elements from unordered photo collections." 249--256.

Gong, J., and Caldas, C. H. (2010). "Computer Vision-Based Video Interpretation Model for Automated Productivity Analysis of Construction Operations." *Journal of Computing in Civil Engineering*, 24(3), 252--263.

Gong, J., and Caldas, C. H. (2011). "An object recognition, tracking, and contextual reasoning-based video interpretation method for rapid productivity analysis of construction operations." *Automation in Construction*, 20(8), 1211--1226.

Gritti, A. P., Tarabini, O., Guzzi, J., and Di Caro, G. A. "Kinect-based people detection and tracking from small-footprint ground robots." *Proc., Proceedings of 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4096-4103.

Hajian, H., and Becerik-Gerber, B. (2010). "Scan to BIM: factors affecting operational and computational errors and productivity loss." 265--272.

Handa, A., Whelan, T., McDonald, J. B., and Davison, A. J. (2014). "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM." 1524--1531.

Hardin, B., and McCool, D. (2015). *$\backslash$emph \ BIM and construction management: proven tools, methods, and workflows* \, John Wiley \ \& \ Sons Incorporated.

Hare, S., Golodetz, S., Saffari, A., Vineet, V., Cheng, M.-M., Hicks, S. L., and Torr, P. H. S. (2016). "Struck: Structured Output Tracking with Kernels." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10), 2096--2109.

Hartley, R., and Zisserman, A. (2003). *Multiple View Geometry in Computer Vision*, cambridge university press.

Huber, D., Akinci, B., Tang, P., Adan, A., Okorn, B., and Xiong, X. (2010). "Using laser scanners for modeling and analysis in architecture, engineering, and construction." 1--6.

Iii, J. G. R., Trevor, A. J. B., Nieto-granda, C., Cunningham, A., Paluri, M., Michael, N., Dellaert, F., Christensen, H. I., and Kumar, V. (2014). "Experimental Robotics." 79, 433--446.

Janaszewski, M., Couprie, M., and Babout, L. (2010). "Hole filling in 3D volumetric objects." *Pattern Recognition*, 43(10), 3548--3559.

Khoshelham, K., and Elberink, S. O. (2012). "Accuracy and resolution of kinect depth data for indoor mapping applications." *Sensors*, 12(2), 1437--1454.

Kim, D. H., Kwon, S. W., Jung, S. W., and Park, S. (2015). "A Study on Generation of 3D Model and Mesh Image of Excavation Work using UAV." \ *ldots* \ *Proceedings of the* \ *ldots* \.

Kim, Y. M., Dolson, J., Sokolsky, M., Koltun, V., and Thrun, S. (2012). "Interactive acquisition of residential floor plans." *Proceedings - IEEE International Conference on Robotics and Automation*, 3055--3062.

Kim, Y. M., Mitra, N. J., Yan, D.-M., and Guibas, L. (2012). "Acquiring 3D indoor environments with variability and repetition." *ACM Transactions on Graphics*, 31(6), 138 131--138 111.

Klein, L., Li, N., and Becerik-Gerber, B. (2012). "Imaged-based verification of as-built documentation of operational buildings." *Automation in Construction*, 21, 161--171.

Kroemer, O. a. (2012). "Point cloud completion using extrusions." *Proceedings of IEEE-RAS International Conference on Humanoid Robots*, 680--685.

Li, Y., Wu, X., Chrysathou, Y., Sharf, A., Cohen-Or, D., and Mitra, N. J. (2011). "GlobFit: consistently fitting primitives by discovering global relations." *ACM Transactions on Graphics*, 30(4), 1.

Liu, H., Luo, J., Wu, P., Xie, S., and Li, H. (2016). "People detection and tracking using RGB-D cameras for mobile robots." *International Journal of Advanced Robotic Systems*, 13(5).

Liu, J., Liu, Y., Zhang, G., Zhu, P., and Chen, Y. Q. (2015). "Detecting and tracking people in real time with RGB-D camera." *Pattern Recognition Letters*, 53, 16--23.

Liu, P., Chen, A. Y., Huang, Y. N., Han, J. Y., Lai, J. S., and Kang, S. C. (2014). "A Review of Rotorcraft Unmanned Aerial Vehicle (UAV) Developments and Applications in Civil Engineering. Smart Structures and Systems." *Smart Structures and Systems*, 13.

Lowe, D. G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints." *International Journal of Computer Vision*, 60(2), 91--110.

Luber, M., Spinello, L., and Arras, K. O. (2011). "People tracking in rgb-d data with on-line boosted target models." *Intelligent Robots and Systems (\ldots*.

Mandel, J. (1982). "Use of the Singular Value Decomposition in Regression Analysis." *The American Statistician*, 36(1), 15--24.

Metni, N., and Hamel, T. (2007). "A UAV for bridge inspection: Visual servoing control law with orientation limits." *Automation in Construction*, 17(1), 3--10.

MIOSHA Regulatory Services Section (2017). "MIOSHA: Excavation, Trenching & Shoring." <http://www.michigan.gov/documents/lara/lara_miosha_CS_9_3-18-2013_414603_7.pdf>. (2017-03-12).

Morgenthal, G., and Hallermann, N. (2016). "Quality Assessment of Unmanned Aerial Vehicle (UAV) Based Visual Inspection of Structures." *Advances in Structural Engineering*, 17(3), 289--302.

Morioka, K., Lee, J. H., and Hashimoto, H. (2004). "Human-Following Mobile Robot in a Distributed Intelligent Sensor Network." *IEEE Transactions on Industrial Electronics*, 51(1), 229--237.

Munaro, M., Basso, F., and Menegatti, E. (2016). "OpenPTrack - Open source multi-camera calibration and people tracking for RGB-D camera networks." *Robotics and Autonomous Systems*, 75, 525--538.

Munaro, M., and Menegatti, E. (2014). "Fast RGB-D people tracking for service robots." *Autonomous Robots*, 37(3), 227--242.

Nan, L., Xie, K., and Sharf, A. (2012). "A search-classify approach for cluttered indoor scene understanding." *ACM Trans. Graph.*, 31(6), 137 131----137 110.

Nassar, K., Aly, E. A., and Jung, Y. (2011). "Structure-from-motion for earthwork planning." *Proc 28th ISARC*.

Nepal, M. P., Staub-French, S., Zhang, J., Lawrence, M., and Pottinger, R. (2008). "Deriving construction features from an IFC model." *Proceedings of Annual Conference of the Canadian Society for Civil Engineering*, 426--436.

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohi, P., Shotton, J., Hodges, S., and Fitzgibbon, A. "KinectFusion: Real-time dense surface mapping and tracking." *Proc., Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, 127-136.

Newcombe, R. a., Molyneaux, D., Kim, D., Davison, A. J., Shotton, J., Hodges, S., and Fitzgibbon, A. "KinectFusion: Real-Time Dense Surface Mapping and Tracking." *Proc., 2011 10th IEEE international symposium on Mixed and augmented reality (ISMAR)*, 127-136.

Nez, J. C., Cabido, R., Montemayor, A. S., and Pantrigo, J. J. (2016). "Real-time human body tracking based on data fusion from multiple RGB-D sensors." *Multimedia Tools and Applications*, 76(3), 4249--4271.

Nguyen, T.-H., Oloufa, A. A., and Nassar, K. (2005). "Algorithms for automated deduction of topological information." *Automation in Construction*, 14(1), 59--70.

Nüchter, A., and Hertzberg, J. (2008). "Towards semantic maps for mobile robots." *Robotics and Autonomous Systems*, 56(11), 915-926.

Park, M.-W., Koch, C., and Brilakis, I. (2012). "Three-Dimensional Tracking of Construction Resources Using an On-Site Camera System." *Journal of Computing in Civil Engineering*, 26(4), 541--549.

Pătrăucean, V., Armeni, I., Nahangi, M., Yeung, J., Brilakis, I., and Haas, C. (2015). "State of research in automatic as-built modelling." *Advanced Engineering Informatics*, 29(2), 162--171.

PCL (2016). "Point Cloud Library." <http://pointclouds.org/>. (Feb 7, 2016).

Schall, G., Wagner, D., Reitmayr, G., Taichmann, E., Wieser, M., Schmalstieg, D., and Hofmann-Wellenhof, B. (2009). "Global pose estimation using multi-sensor fusion for outdoor Augmented Reality." *Proceedings of IEEE International Symposium on Mixed and Augmented Reality*, 153--162.

Shao, T., Monszpart, A., Zheng, Y., Koo, B., and Xu, W. (2014). "Imagining the Unseen: Stability-based Cuboid Arrangements for Scene Understanding." *SIGGRAPH Asia*.

Shao, T., Xu, W., Zhou, K., Wang, J., Li, D., and Guo, B. (2012). "An interactive approach to semantic modeling of indoor scenes with an RGBD camera." *ACM Trans. Graph.*, 31(6), 136 131----136 111.

Sharf, A., Alexa, M., and Cohen-Or, D. (2004). "Context-based surface completion." *ACM Transactions on Graphics*, 23(3), 878--887.

Siebert, S., and Teizer, J. (2014). "Mobile 3D mapping for surveying earthwork projects using an Unmanned Aerial Vehicle (UAV) system." *Automation in Construction*, 41, 1--14.

Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). "Indoor segmentation and support inference from RGBD images." *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7576 LNCS(PART 5), 746--760.

Son, H., Kim, C., and Kim, C. (2015). "Fully automated as-built 3D pipeline extraction method from laser-scanned data based on curvature computation." *Journal of Computing in Civil Engineering*, 29(4), B4014003.

Song, S., and Xiao, J. (2013). "Tracking Revisited Using RGBD Camera - Unified Benchmark and Baselines." *Iccv*, 233--240.

Song, S., and Xiao, J. (2014). "Sliding shapes for 3D object detection in depth images." 634--651.

Steinbruecker, F., Sturm, J., and Cremers, D. (2014). "Volumetric 3D Mapping in Real-Time on a CPU."

Su, Y. Y., and Liu, L. Y. (2012). "Real-Time Construction Operation Tracking from Resource Positions." 200--207.

Sung, M., Kim, V. G., Angst, R., and Guibas, L. (2015). "Data-driven structural priors for shape completion." *ACM Transactions on Graphics*, 34(6), 175 171--175 111.

Susperregi, L., Martnez-Otzeta, J. M., Ansuategui, A., Ibarguren, A., and Sierra, B. (2013). "RGB-D, Laser and Thermal Sensor Fusion for People following in a Mobile Robot." *International Journal of Advanced Robotic Systems*, 10(6), 271.

Suzuki, S., Mitsukura, Y., Takimoto, H., Tanabata, T., Kimura, N., and Moriya, T. (2009). "A human tracking mobile-robot with face detection." 4217--4222.

Szeliski, R. (2010). *Computer vision: algorithms and applications*, Springer Science \& Business Media.

Taguchi, Y., Jian, Y. D., Ramalingam, S., and Feng, C. (2013). "Point-plane SLAM for hand-held 3D sensors." *Proceedings - IEEE International Conference on Robotics and Automation*, 5182--5189.

Tang, P., Huber, D., Akinci, B., Lipman, R., and Lytle, A. (2010). "Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques." *Automation in Construction*, 19(7), 829--843.

Turkan, Y., and Bosch (2013). "Tracking Secondary and Temporary Concrete Construction Objects Using 3D Imaging Technologies." 749--756.

Vo, D. M., Jiang, L., and Zell, A. (2014). "Real time person detection and tracking by mobile robots using RGB-D images." 689--694.

Volk, R., Stengel, J., and Schultmann, F. (2014). "Building Information Modeling (BIM) for existing buildings: Literature review and future needs." *Automation in Construction*, 38, 109--127.

Wang, C., and Cho, Y. K. (2015). "Smart scanning and near real-time 3D surface modeling of dynamic construction equipment from a point cloud." *Automation in Construction*, 49(0), 239--249.

Wang, J., and Oliveira, M. M. (2007). "Filling holes on locally smooth surfaces reconstructed from point clouds." *Image and Vision Computing*, 25(1), 103--113.

Wefelscheid, C., Hansch, R., and Hellwich, O. (2011). "Three-dimensional building reconstruction using images obtained by unmanned aerial vehicles." *ISPRS -- Int Arch Photogramm Remote Sens Spatial Inform Sci*, XXXVIII-1/.

Wu, C. (2007). "SiftGPU: A GPU implementation of Scale Invaraint Feature Transform (SIFT)."

Wu, C. (2011). "VisualSFM: A Visual Structure from Motion System."

Wu, C. (2013). "Towards Linear-Time Incremental Structure from Motion." 127--134.

Xiao, Y., Wang, C., Li, J., Zhang, W., Xi, X., Wang, C., and Dong, P. (2015). "Building segmentation and modeling from airborne LiDAR data." *International Journal of Digital Earth*, 8(9), 694--709.

Xiao, Y., Wang, C., Xi, X. H., and Zhang, W. M. (2014). "A comprehensive framework of building model reconstruction from airborne LIDAR data." 17, 12178.

Xie, F., Lin, Z., Gui, D., and Lin, H. (2012). "Study on construction of 3D building based on UAV images." *ISPRS -- Int Arch Photogramm Remote Sens Spatial Inform Sci*, XXXIX-B1.

Xiong, X., Adan, A., Akinci, B., and Huber, D. (2013). "Automatic creation of semantically rich 3D building models from laser scanner data." *Automation in Construction*, 31, 325--337.

Xu, R., Guan, Y., and Huang, Y. (2015). "Multiple human detection and tracking based on head detection for real-time video surveillance." *Multimedia Tools Appl.*, 74(3), 729--742.

Yaguchi, H., Takaoka, Y., Yamamoto, T., and Inaba, M. (2013). "A method of 3D model generation of indoor environment with Manhattan world assumption using 3D camera." 759--765.

Yang, J., Park, M. W., Vela, P. A., and Golparvar-Fard, M. (2015). "Construction performance monitoring via still images, time-lapse photos, and video streams: Now, tomorrow, and the future." *Advanced Engineering Informatics*, 29.

Zarka, N., Alhalah, Z., and Deeb, R. (2008). "Real-Time Human Motion Detection and Tracking." 1--6.

Zhang, K., Chen, S., Whitman, D., Shyu, M.-L., Yan, J., and Zhang, C. (2003). "A progressive morphological filter for removing nonground measurements from airborne LIDAR data." *IEEE Transactions on Geoscience and Remote Sensing*, 41(4), 872--882.

Zheng, B., Zhao, Y., Yu, J. C., Ikeuchi, K., and Zhu, S. C. (2013). "Beyond point clouds: Scene understanding by reasoning geometry and physics." *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3127--3134.

Zhou, J., and Hoang, J. (2005). "Real Time Robust Human Detection and Tracking System." *CVPR Workshops*.

Zhu, Z., and Donia, S. (2013). "Potentials of RGB-D Cameras in As-Built Indoor Environment Modeling." 605--612.

Zhu, Z., and Donia, S. "Potentials of RGB-D cameras in as-built indoor environments modeling." *Proc., Los Angeles, CA: 2013 ASCE International Workshop on Computing in Civil Engineering*, 23-25.

Zollmann, S., Hoppe, C., Kluckner, S., Poglitsch, C., Bischof, H., and Reitmayr, G. (2014). "Augmented Reality for Construction Site Monitoring and Documentation." *Proceedings of IEEE, Special Issue on Application of Augmented Reality*, 102(2), 137--154.