ROBOTIC EXPLORATION FOR MAPPING: SYSTEMS AND ALGORITHMS

by

Hao Men

A DISSERTATION

Submitted to the Faculty of the Stevens Institute of Technology
in partial fulfillment of the requirement for the degree of

DOCTOR OF PHILOSOPHY

Hao Men, Candidate

ADVISORY COMMITTEE

Kishore Pochiraju, Chairman Date

David J. Cappelleri Date

George Kamberov Date

Souran Manoochehri Date

Yan Meng Date

STEVENS INSTITUTE OF TECHNOLOGY
Castle Point on Hudson
Hoboken, NJ 07030
2012

ROBOTIC EXPLORATION FOR MAPPING: SYSTEMS AND ALGORITHMS

ABSTRACT

Robotic exploration can enhance the quality, efficiency and completeness of mapping for surveying in unknown, distant, and hazardous environments. In most cases, multiple position sensors are installed on the mobile robots for precise localization of mapping tasks. Usually, laser ranging devices on the robots generate highly detailed and dimensionally accurate coordinate data in the form of point clouds. 3D Digital Terrain Model (DTM), Computer Aided Design (CAD) drawings and other surveying or visualization products can be synthesized from the point cloud data. Laser scanning while in motion is subject to position sensing errors, and point clouds obtained from a single static vantage position are generally incomplete because no data points exist in occluded areas. Position sensors that have high precision at the high sampling rate required for 3D mapping are usually complex and expensive. Therefore, the map registration algorithm is desired to automatically to produce a high precision global point cloud map and work independently with positioning sensors. Meanwhile, a path planning algorithm based on an uncompleted map is desired for mapping robot travel efficiently through unknown environments and complete mapping tasks.

The work in this thesis is focused on building a mobile scanning system and algorithms required for map registration, occluded area recognition, and vantage position determination as path planning. A mobile robotic system and a 3D LIDAR scanner with video cameras have been developed as the platform for experimentation and algorithm validation. This system produces color rendered point clouds as maps. Multiple positioning sensors including GPS, IMU, steering potentiometer, and odometers installed on the robot provide rough positioning estimates for

tracking the robot and validation. The color point clouds generated from the mobile mapping robot are used for verification and performance evaluation of the developed algorithms.

The contribution in map registration algorithms includes two parts: color assisted automatic point cloud registration based on point surface normal orientation histogram and occupancy grid correlation; Hue Assisted Iterative Closest Point (HICP) algorithm for fine registration. The first part does not require any pre-known positioning information. The registration process is completed by solving the transformation correlation in Fourier domain. However, registration accuracy relies on the resolution of orientation histogram and occupancy grid, and therefore can only produce coarse registration output. The second part requires rough-aligned point clouds before registration starts, which are exactly the coarse registration products. The HICP iteratively registers point clouds paired wisely together. The results of this come with high accuracy, but take more time than the first step. Above two algorithms work together to register color point clouds during exploration together into a global frame.

The contribution in path planning contains a vantage position generating algorithm based on occlusion detection and frontier based exploration. Methodologies for identifying and selecting candidate vantage positions for mapping are discussed. For complete and efficient map generation, the map completeness evaluation is based on grid occupancy in 2D space and point cloud density in 3D space. The path planning algorithm for frontier best view is contained by map overlap ratio for the registration. A trajectory evaluation is performed during next best view point selection to generate the next vantage position to complete the exploration process.

This research has produced efficient color assisted automatic and iterative algorithms for registration of 3D point cloud map segments, and a path planning algorithm to generate vantage positions for robotic carriers explore unknown environments to complete the autonomous exploration and mapping task.

Author: Hao Men

Advisor: Kishore Pochiraju

Date: March 12, 2012

Department: Mechanical Engineering

Degree: Doctor of Philosophy

Dedication

To my parents

Xiaojuan Ma & Junzhe Men

Acknowledgements

I want to specially thank my advisor, Dr. Kishore Pochiraju, for his valuable time, support and guidance during my dissertation work and my graduate study at Stevens Institute of Technology. Working with him has been a great learning experience and the best period of my life.

Special gratitude is extended to my dissertation committee members: Professor Souran Manoochehri, Professor Yan Meng, Professor David Cappelleri and Professor George Kamberov. Their valuable time, comments, and insights are faithfully appreciated.

I would like to dedicate my dissertation to my father Junzhe Men and my mother Xiaojuan Ma for their support during the past years. They wished to pursue a higher degree in their life, but never got a chance to have one. I would not have been able to join Stevens and pursue a PhD degree without their inspiration. I would like to thank my lovely sister Meng Men for her patience on listening and understanding. Specially thanks goes to my cousin Xinyu Song for his guidance and help during the past years. Thank you so much for your support!

I would like to thank all people in Design & Manufacturing Institute who helped me on my research and study. My gratitude especially goes to my dear friend Mr. Biruk Gebre for his time, friendship, and kind support on many experiment settings and data collections in the past years. I also want to thank other team members: Mr. Akin Tatoglu, Mr. Tejaswi Yerukalapudi, Mr. Jinkyu Han, Mr. Ahmet Uzun, Mr. Chirag Gardharia, and Mr. Wongyoung Kim from the robotics team at DMI. Special thanks to Dr. Daizong Li, Dr. Liwen Guo, and Dr. Yunn-Tzu Yu for their kind help during my study at DMI.

I would also like to thank all of my colleagues at DMI for their support and friendship during my graduate study. I am grateful for the support and friendship from my dear friends Xi Chen, Wei Xu and Weihe Xu. Special gratitude goes to Ms. Yuchen Qi for the happiness and support she

brought to me during the last year of my PhD work. My sincere acknowledgement goes to those who supported and inspired me during the course study and research work of my PhD time.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

This chapter introduces the concept and state of art of robotic exploration and related techniques for mapping. Development of mapping hardware platforms, algorithms about map registration and path planning have been included. The concept of complete mapping is introduced especially in regards to occlusion detection and complete coverage of a given area. Challenging problems are illustrated in different stages of robotic exploration for mapping and surveying. The last section in this chapter contains the organization and brief introduction of the remaining chapters in this thesis.

1.1 Surveying and Mapping

Maps are a static visual representation of an area to symbolically describe dimensions and relationships between elements of specific space such objects, regions, and themes[1, 2].This representation describes object geometric features based on different variants such as discrete range points, color, geometric shape and mathematical equation[3]. Range data and color property are usually acquired directly from range and imagery sensors [4, 5], such as laser ranger and stereo camera. Geometric features and mathematical equations are usually extracted from sensor data to describe geographical conditions and object surface features [1, 6, 7]. Maps are capable of providing quantitative descriptions about an environmental situation in many industrial and research fields such as civil engineering, geographical information systems, and autonomous navigation, etc.

The objective for surveying in an urban area is to provide a necessary description of physical measurements in map form about environmental objects. Traditionally, the survey process is finished by surveyors by using surveying equipment such as levels and theodolites to measure points on object surface [8-10]. Levels and theodolites have specially designed optical instruments for the purpose of relative angular and distance deviation measurement. This information is critical for civil engineers to evaluate design and build on certain areas. It is also saved into Geographical Information System (GIS) to effectively restore, manage and analyze surveying data [11, 12]. Especially in urban environment, the surveyor must lay out the routes of different facilities such as roads, buildings, pipelines, streets, highways, railways and other infrastructures for future application. The surveying can be briefly divided by land surveying and construction surveying. Land surveyors measure the terrain topography in a certain area, determine the boundary of a parcel, and create a map about the given land. The boundary of certain parcel is marked with legal description as the result of land survey [8, 10]. Construction surveying is more difficult and technical than land surveying. Construction surveying investigates existing conditions of a work environment. Terrain topography, building characteristics in three dimension space, and infrastructures have to be recorded in construction surveying. In some special cases, underground infrastructures should also be marked in a construction surveying map. Another function in construction surveying is verification, which includes verification of structures and locations for buildings under construction and at the end of construction. During the construction process, construction surveying helps to monitor and inspect progress. After the building process is completed, construction surveying performs as-built surveying to confirm that authorized work is completed and meets certain specifications.

Urban surveying aims to provide an as-is map in certain urban areas. The generated map knowledge should be processed and restored in GIS. Map information includes topographical

data, street, building and other infrastructure features, and in some cases internal building conditions should also be recorded. Masp can be represented as key point data sets, geometrical features, and topological relations. This map is mainly used to evaluate urban development, create a digital map about an urban area, and monitor building construction progress and historical remains repairs. Therefore, urban surveying is similar as construction surveying with some land surveying techniques.

In order to support various applications, the survey produces preliminary and reconnaissance information about urban environment. During the surveying period, the surveyor should determine distances, angles and areas for measurement. Every survey should be related to a reference point at both horizontal and vertical directions. Stakes, or lines, are set up as markers for measurement. Levels and theodolites are used to measure the markers' position relative to the ground. The layout of a building's structure and public infrastructure, such as culverts, should be measured using different techniques. Accuracy is the most critical requirement in urban surveys. Compared with other survey techniques, location surveys produce high precision maps about the environment. The degree of accuracy in surveying varies for different purposes. However, most urban survey maps have been used for civil engineering for construction and building evaluation. Therefore, the best survey data map should come with the highest accuracy and minimum of time cost.

A typical urban survey procedure can be divided as follows: Decide the survey area, establish a reference coordinate system, determine local survey locations, go to survey locations, follow certain sequence, set up markers for measurement, measure markers and record its coordinates until all local surveys are finished, shown as Figure 1.1.

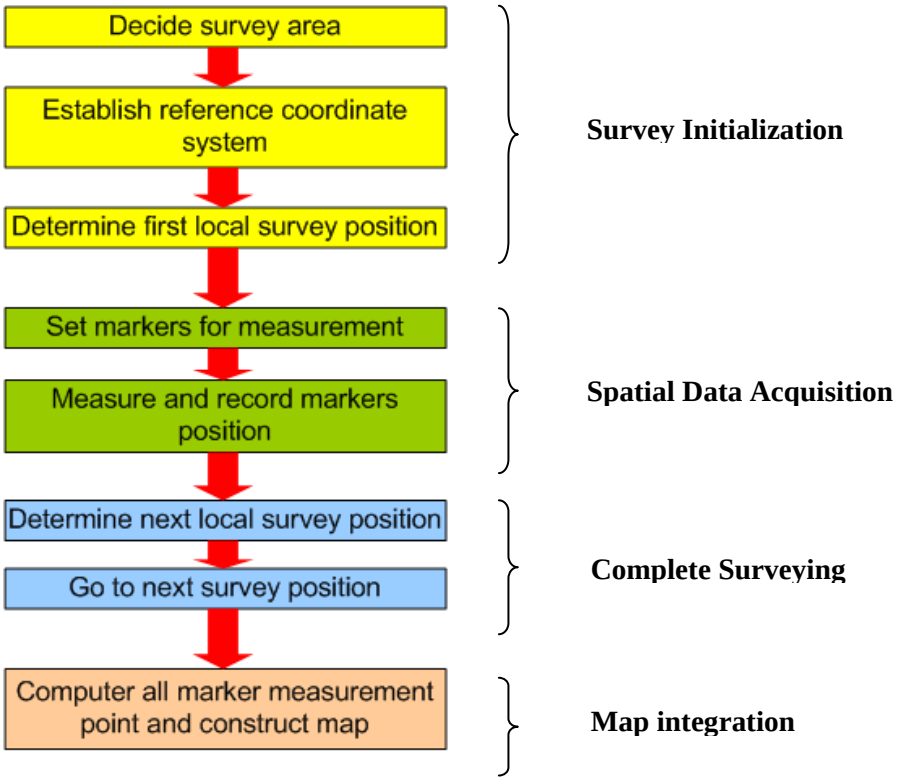


Figure 1.1 Surveying process

Traditional surveying process measures distance by using a level, tape and theodolite on a tripod with the triangulation method. By manually moving equipment from one location to another until all survey data is acquired, this process normally takes a long period of time, and a great deal of human resources to accomplish. The accuracy of a surveyed map in a large area cannot be guaranteed because of cumulative errors. Most recently developed surveying devices such as laser ranger, ultrasonic sensor and sonar have been applied to measure dimensions of object surfaces to construct a map under different environments [7, 13, 14]. The most commonly applied range sensor for urban surveying and mapping is Light Detection and Ranging (LIDAR) [15-17]. LIDAR measures object surface distance by using scattered laser pulses. Distance to the object is then calculated by measuring the time delay between the laser pulse's transmission and reflection.

LIDAR can be installed on different mobile platforms to finish surveying and mapping tasks under different environments [15, 18, 19]. For example, LIDAR is installed on airplanes to fly through certain areas. The map can then be extracted by calculating altitude deviation [10, 20]. Taking advantage of the portability, low power consumption, and robustness, LIDAR has become the most wildly applied mapping sensor in urban surveying [21, 22]. Because of the LIDAR range limit, orientation, and location of mapping vantage position, a series of maps taken from different observation vantage positions have to be combined together to construct the complete map. Acquired features and dimensional knowledge are then selected by the mapping system and transformed into a global map database. LIDAR scanner can be installed on a tripod and carried by surveyors to move from point to point to complete the mapping process. This usually costs time for operators to carry and adjust the device between different mapping positions to completely build a map. LIDAR also has been installed on a mobile platform to travel through the surveying area to construct a global map [23, 24]. The total time cost of the mapping process can be greatly decreased, as the mobile platform saves time for travelling.

1.2 Robotic Exploration for Mapping

Robotic exploration for mapping techniques can be utilized to complete surveying tasks in specific area, and does not necessarily require human interaction. Usually this exploration and mapping process is performed by a mobile platform traveling through the surveying area, and the LIDAR scanner and positioning sensor recording distance information in the reference coordinates system. Different mobile robotic carriers such as track ground vehicles and wheeled vehicles have been developed to explore in urban surveying. Both hardware and software on mobile robotic mapping platforms should ensure that the automated process travels through all

traversable areas and produces high precision maps within a given boundary. The concepts of robotic exploration and mapping with related methods have been introduced to surveying area as soon as fast distance measurement equipment and reliable high position sensors are developed [25, 26]. Mobile platforms provide the capability to travel in a mapping environment with precise location information.

Robotic exploration and mapping use ranging devices such as LIDAR, sonar, and radar to measure spatial data in an objective area within a very short time period. Those range sensors have been installed on mobile platforms as a 2D or 3D mapping system. For example, LIDAR was mounted on a ground vehicle to map road conditions. LIDAR was also installed on a helicopter to fly by urban areas to generate urban maps. Synthetic Aperture Radar (SAR) is specially designed to install on an airplane or satellite to produce very large scale 3D maps [20]. Compared with other measurement facilities, LIDAR owns small dimension and low power consumption. Most importantly, LIDAR is able to scan 2D and 3D at high frequency and the measurement precision is at millimeter level. Therefore, single or multiple LIDAR are installed on ground vehicles to perform urban surveying works.

A robotic carrier will locate position for itself based on rough estimation from its internal position sensor and precise distance measurement feedback correction [27, 28]. Local maps are then stitched together with computed accurate localization data and then registered into global coordinates as part of the complete map. This process is so called Simultaneous Localization And Mapping (SLAM) [28-30]. In order to autonomously explore and completely build the map in unknown areas, mobile robots need to be equipped with proper sensors to for autonomous operation. Calibrated single or multiple LIDAR devices with certain coverage range should be installed on the vehicle to generate highly detailed and dimensionally accurate coordinate data in the form of point clouds, odometer and steering encoder return speed and heading of the robot,

Initial Measurement Unit (IMU), which provides robot pitch/yaw/roll status, and Global Positioning System (GPS), which measures robot position in ground space coordinates. Depending on the position and movement information from sensor group, multiple point cloud maps obtained from different vantage position can be combined together into a complete map.

Point clouds with texture and laser reflection intensity generated from a LIDAR scanner have to be rigidly transformed and registered together depending on relative position and orientation at different vantage positions. Use of map registration algorithm should be efficient and economical if accuracy and speed of registration can be ensured so that large-scale global 3D maps can be constructed. Registration of map segments is trivial if precise position and orientation of the sensor area accurately known about the ground space reference frame. Positioning data from IMU and GPS are prone to errors and can be inaccurate under certain conditions. The quality of map registration can vary depending upon the sensor resolution and other factors related to the positioning sensor. While high definition surveying usually requires millimeter accuracy, robotic exploration or security applications may only require much coarser accuracy. Different techniques exist to apply common geometric shapes or exploit maximum likelihood between surface measurements to compute relative orientation and position between different vantage positions and stitch maps together. The most popular registration algorithm for point cloud registration is iterative closest point (ICP) algorithm. In ICP, the corresponding closest points in different point clouds are associated and the optimal rigid transformation required to minimize the mean-square error of separation between associated points of the two data sets is iteratively found. Color and optical intensity attributes can be integrated with point cloud to produce color point cloud. Color point cloud can enhance robotic mapping, navigation and exploration performance [31, 32]. In some mapping cases, if the environment surface is not arbitrary, two scanned point clouds from different positions cannot be registered together. The color and

intensity can be utilized in the ICP process to increase computational speed and provide higher registration accuracy. Prior work exists on the use of filtering the point set based on hue before conducting traditional ICP and on processing images to extract corresponding visual features.

There are several significant advantages of robotic mapping compared with traditional surveying methods. First, mapping scale can be expanded depend on mobility from mapping sensor carrier. Mapping platform can carry range sensor travel along large scale areas to collection location mappings. Compared with non-automated process, in which the surveyor has to carry the tripod and measurement equipment in surveying area, the size of the area has to be limited for certain surveyor groups. Second, the mapping period could be reduced depending on mobility of the carrier, intelligent local survey position selection, and new type of range sensors. Traveling time between different local mapping positions can be greatly compressed and the total surveying period can also be decreased. Total mapping time could also be compressed by efficiently picking up local mapping locations, which is also named as vantage positions. Optimized vantage positions ensure complete map generation with fewest local mapping and shortest travel time. Latest developed range sensors provide high mapping frequency which is able to finish a local mapping 3D environment in seconds. Together, all these efforts help to reduce the total surveying period. The last but the most important advantage is that mapping accuracy has been increased during robotic mapping process. Cumulative error has been reduced by probabilistic localization methods on mobile carrier. The accuracy for complete survey map has been improved by map construction techniques. Both probabilistic localization and map construction techniques ensure high accuracy maps can be produced after the mapping process.

Robotic exploration for mapping can be integrated into the urban survey process to enhance urban survey performance and reduce the surveying period. Compared with the urban surveying process in Figure 1.1, robotic exploration for mapping can take the place of rest progresses in urban

surveying except for the survey initialization phase. Urban surveying with an automated mapping technique does not require any human interruption before a complete measurement is finished. This procedure can be described in Figure 1.2.

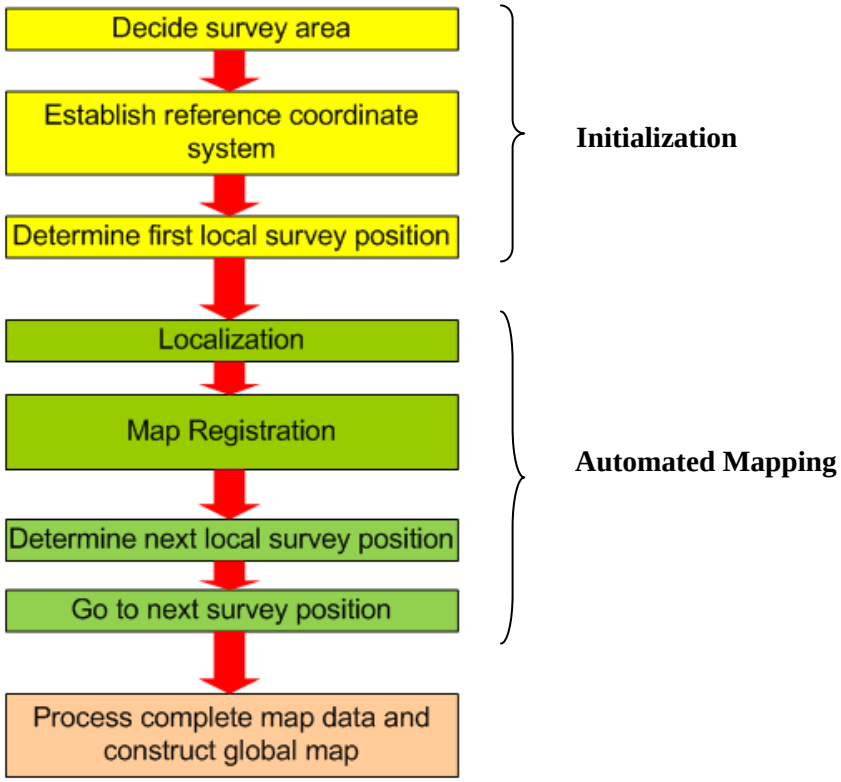


Figure 1.2 Robotic exploration for mapping process

As soon as initial parameters, such as survey area and reference coordinates system, have been input to automated mapping system, a mobile carrier will autonomously carry mapping sensors to completely survey the given area. After data analysis and noise canceling on raw survey data, processed information will be restored in GIS and output as 2D or 3D map format.

Exploration strategies are critical for autonomous robotic mapping besides SLAM. Due to partially known information about unknown environments, exploration strategies navigate the

mobile robot to travel and complete the map within optimal travel cost based on the map completeness evaluation of the current acquired map data. Also, exploration algorithms have to navigate the mapping robot to proper vantage position so that newly generated map scanning fills gaps left from occlusions in previous acquired maps. Selection of next mapping observation vantage position should consider current built map, incomplete and missed mapping space in the given area, and travel cost on different terrain conditions.

There are several specific requirements for mobile robots exploring and constructing maps in urban surveying. The biggest problem for urban surveying is cumulative errors. Cumulative errors normally come as the result from misplacement of measurement locations and inaccurately calibrated measurement equipment. As soon as a series of location measurement is complete, accumulated measurements referring to the previous location together may affect the precision of survey results. The second requirement is the completeness of mapping, which asks for the complete coverage of all objects in the travelled area. The contour of a building could be different, and concave shapes might not be reached because other objects blocked between range sensors. These concave shapes need to be filled by map measurement from another vantage position, which is determined based on map evaluation after every vantage position mapping process is finished. Another requirement is path planning on urban terrain. The costs for mapping process such as length of travelled trajectory, and time needs to be minimized from respect to surveyor's consideration. The trajectory needs to be located on the terrain where the mapping robot is able to walk through, thus robot dynamics must be considered. In some search and rescue cases, the robot is asked to search and find certain objects and construct a 3D map about this object. Mapping robots need to perform quick mapping and compare the map with a given model or pattern to exploit objects in minimum time. Sometimes mapping robots should incrementally learn different object models to label them and fit CAD model on labeled points as map product.

Since robotic exploration and mapping has been an active field for both research and industry in the past years [3], efforts and research went to construct a global map with accurate spatial information by using single or multiple robots. This subject includes complete map construction, optimal mapping path planning, and accurate map information acquisition [8, 33, 34]. Robotic exploration and mapping for urban surveying is one of the most challenging topics in robotic mapping research. Limited by some specific situations in urban environment above, a mapping robot has to generate the best route points to complete the map within the shortest time cost [33-35]. The mission for robotic exploration for mapping in urban environments is that the robot motivates itself to travel inside a given area, determine its movement to explore an unknown field, and complete mapping the whole area with the lowest cost and satisfied mapping precision [36].

1.3 State of Art

From the later 1980s and early 1990s, mapping became an important topic in robotics research area [3, 21, 36, 37]. Earlier works were focused on constructing reliable robotic platforms and applying different sensors to acquire accurate map data in experimental environments [38-41]. Research during this time was focused on map descriptions started by metric and topological approaches. The metric approach was developed to describe object contour layout in occupancy grid mapping [42]. Larger scale environments with obvious landmarks or objects was usually described by topological methods by arcs connections [43, 44]. Dimensionally accurate 3D map generation attracts more research interests in the later stage of robotic mapping. 3D mapping was created based on stereo vision devices by [45-47]. Ranging devices were applied and became the most popular equipment for robotic mapping because of its accuracy and reliability. Multiple 2D laser rangers were installed on mobile ground vehicles to accomplish 3D mapping missions by

fusing 2D range images into a 3D map [48, 49]. The single 3D laser ranger was developed and installed on top of unmanned ground vehicles to be deployed in outdoor terrain to finish exploration and mapping missions. Multiple 3D maps generated from different locations were combined together based on its position and orientation in global frame. Common geometric features and landmarks also were applied to stitch 3D maps together.

Registering the map segments is trivial if the precise position and orientation of the sensor is known about a global reference frame. However, sensing the position and orientation of the robot accurately is challenging and the use of registration algorithms based on identifying common features and geometries in the two map segments may be accurate, efficient, and economical. A rigid transformation with translation and rotation is obtained from the map registration process determining the map sensor position in 3D space and its orientation [22, 28, 29]. The map registration quality can vary depending upon the accuracy required by the application. While high definition surveying may need sub-centimeter or millimeter accuracy, robotic exploration may only require much coarser registration.

Generation of point clouds is the most common 3D map format in mobile robotic mapping. Discrete range points received from range sensors describe spatial information about the environment. Different techniques exist for merging 3D maps together by exploiting geometric features and measuring surfaces. The most popular registration algorithm for point cloud map registration is iterative closest point (ICP) algorithm [45, 49, 50], in which the corresponding closest points in different point clouds are associated and optimal rigid transformation to minimize a mean-square error of separation between associated points of the two data sets [51] is iteratively found. Upon convergence, ICP algorithm terminates at a minimum. Several algorithms are in existence for calculating the minimum average distance between two point clouds. The Singular Value Decomposition (SVD) method [49], eigen-system methods that exploit the

orthonormal properties of the rotation matrices, and unit and dual quaternion techniques were adopted in ICP techniques. Quaternion based algorithms have been used in ICP for map registration in . SVD based algorithms are widely used in ICP and 6DOF SLAM [49, 52] as they are robust enough [66] to reach local minimum and are easy to implement.

Different variants of ICP have been investigated [53]. Corresponding points sampling, matching, weighting, and rejecting are some methods used to accelerate the ICP algorithm. In the ICP algorithm, associating corresponding points in two point cloud data sets is the most critical step. Nearest neighbor search in 2D or 3D space is commonly used for associating the corresponding points. Parallel ICP algorithms have been developed [54] to accelerate computation speed.

As vision sensors are now integrated into laser ranging systems [46, 47, 55], 3D point clouds also contain the color properties in the scene. In this effort, the color attributes of the range point are utilized in the ICP process to increase computational speed and provide higher accuracy. The color attribute of the scanned point from Red-Green-Blue (RGB) space is translated into Hue-Saturation-Lightness (HSL) space, and the hue value is used along with the coordinate data during the corresponding point association step. Prior work on hue association is based on filtering the point set on hue before ICP [56]. Some preliminary work on processing images to extract corresponding visual features for registration has also been reported [57].

Different mobile robotic platforms were integrated with ranging sensors according to different mapping and exploration requirements. The ground wheeled robot is the most commonly applied platform for urban surveying. The ground vehicle platform can be modified from automobile [58] with mapping sensor mounted on top to construct outdoor 3D maps in certain area. Smaller size vehicles such as iRobot [23] have been developed to build maps especially in an indoor environment. This specially designed mapping robot can be utilized to map both indoor and outdoor environments. The size of these robots enables them to be able to travel through hallways

inside a building, and they also posess the mobility to travel outdoors to construct building outside surface dimension maps. These were proved to be reliable, and were then deployed to explore dangerous or hazardous conditions and complete surveying [35].

Due to position sensor noise, measurement error and drifting, local maps cannot be precisely registered together and inaccurate global maps were constructed after the robot finished its exploration[2, 59]. Probabilistic techniques have been introduced to build a statistical framework for simultaneously solving the mapping and localization problem relative to its growing map after 1990s. The Gaussian noise model was applied to estimate noise distribution on positioning sensor data. Kalman filters are employed to estimate map and robot location [2, 29, 59] by assuming noise distribution follows Gaussian distribution. Particle filters do not make any assumptions for the noise distribution, but re-sample the sensor data based on measurement acquisition. This method is more reliable to estimate noise distribution under different conditions[58, 60]. Objects in an environment can be extracted and used to identify relative locations, such as ceilings, walls, furniture, or doors (open/closed) [61-64]. Robot movement has to be simultaneously monitored during the mapping procedure. Location and orientation information provided from robot internal sensors are not enough for mapping and localization[22, 65]. The location feedback also comes from the map registration process. When the find registration is finished, a relative position and orientation in global space can be used to correct localization data from internal sensor. Simultaneous accurate localization has to be contained while robot is moving to escape any accidents. Trust worth localization information has to be provided to robot when it is travelling in an urban environment. However, location sensor and controller normally introduce noise in robot movement which makes localization inaccurate. Probabilistic methods have been introduced to provide most likely position data to the robot. The Simultaneous Localization And Mapping (SLAM) technique is the most wildly used localization algorithm in robot navigation

and control[66, 67]. Different filters are used in the SLAM process for location data updates. Extended Kalman Filter (EKF), Particle Filter (PF), and Graph Filter (GF) are the commonly used filter in SLAM process [2, 68].

Some map evaluation techniques have been introduced to navigate robot finish mapping process in an unknown environment. The occupancy grid mapping algorithm was introduced to help navigate the robot to complete mapping in a given area [13, 69]. The find-grained grid has been used to distinguish free and mapped regions. This method is applied in many robotic systems. Polyhedral [43, 44] sets are introduced to describe the geometric environments. Topological maps have an advantage in its size and computation efficiency in environment representation, which is also used for map description.

Vantage positions are computed based on evaluation of registered maps. Unexplored area can be extracted by computing the grid occupancy and point cloud map continuity. Cost function helps to determine the best next vantage position based on the acquired map and known environment situation. Next mapping position also has been decided based on the latest registered map. Vantage position keeps being modified based on the latest registered map and cost function until all accessible areas have been explored and global map has been accomplished [70]. To completely explore an unknown area with minimum cost is the objective to determine next vantage positions selection. Global coverage in 2D maps can be computed by a grid occupancy algorithm so that global mapped area is known[67, 71]. A rectangular/triangular cell occupancy algorithm was applied to increase the evaluation process and determine the best route for rescue missions. The lawnmower algorithm for robotic exploration was developed in 1993, and the seed spread and random walk algorithms were applied for complete exploration and mapping. Frontier based exploration and wall following for indoor exploration were utilized to help mapping robots exploring in an unknown field and return to starting position after the task has been accomplished.

A visibility graph is applied to navigate robot travel through an obstacle environment[32, 57], Voronoi diagram is applied in exploration by defining points set as site, free area and obstacle area can be divided by Vironoi region.

The Next Best View problem was proposed to solve mapping vantage position selection for autonomous robotic exploration. This algorithm was developed to efficiently travel through the exploration area.[24-26, 72]. The best view was selected based on the Frontier Exploration algorithm [70]. Multi-bots mapping has been developed in [25, 30, 70], different bots are deployed on mapping frontiers to expand mapping coverage area [70]. Motion model of mapping robot is included in exploration process, energy-efficient exploration is proposed by [73].

1.4 Challenging Problems

The most challenging problem in autonomous exploration for mapping and surveying is constructing a large scale dimensional accurate map from many small map segments. This problem can be divided into several sub-problems such as accurate registration of local map into global coordinates, acceleration of registration process, and robust and automated global map construction. Current pair-wised fine iterative registration procedures are relied on pre-alignment, which transforms robot centric local maps into global frames based on internal position and orientation sensor perception. Also the relative position between two 3D point cloud maps before registration affects the convergence of registration procedure, the map surface geometry has to be arbitrary so that correct convergence can be reached. The distribution of points in mapping a robot centric 3D point cloud map is not evenly distributed, density of points decreases as soon as the radius from objects surface goes far from the range sensor. In order to generate complete and accurate maps in large urban areas, the point density should maintain above a certain level so that

the CAD model can be created from range points measurement. Noise data due to light condition, object surface scattering and refraction need to be distinguished and removed from normal measurement.

Another challenge is path planning for exploration and mapping. Since local maps can be registered into global space depending on map registration techniques, determination of next vantage observation position and orientation is critical to solving this problem. Current exploration and path planning algorithms were focused on navigating travel through an unknown scene and finishing a search and rescue mission. The objectives of complete mapping require all objects surfaces in traversable environment to be measured by range sensor. Grid occupancy maps only help to decide the coverage of the mapped area but not coverage on object surface. Exploration occupancy grid and object surface contour continuity should both be considered for robot path planning. The map completeness evaluation criteria should be finished for complete surveying objective and be implemented after every local mapping is accomplished. Meanwhile, robot path should be determined to accomplish the whole mapping process within next best view and ensure the next view generated point cloud has enough overlap with the previous one, which is desired by the automated registration process.

1.5 Thesis Organization

This dissertation is aiming to solve the challenging problems during autonomous exploration for mapping processes. 3D color map registration is the first challenge in order to construct global maps from many map segments acquired on different vantage positions. The exploration and path planning problem needs to be solved for complete mapping missions. Map completeness is the most important objective to achieve in robotic exploration for mapping.

This thesis is organized as follows: this chapter introduces fundamental concepts and work flows about robotic mapping and robotic exploration for mapping. Development of robotic exploration and mapping algorithms with related hardware requirements for mapping robot are introduced. The state of art of robotic exploration and mapping with related systems and algorithms during past decades have also been reviewed in this chapter.

Chapter 2 illustrates the hardware platform of mapping robots (Remotely Operated Autonomously Mapping System). Different modules such as drive by wire, 3D color mapping station, and teleportation module have been introduced. The mobile vehicle dynamic model and software structure have been included so that autonomous exploration for mapping and movement control can be completed based on this platform. Few sets of 3D color point cloud map produced by the mapping platform are provided in this chapter.

Chapter 3 introduces point cloud processing methodologies. The Voxel grid based point cloud de-sampling algorithm has been discussed for color point cloud. The point cloud surface normal vector computation method is introduced. The 3D occlusion detection method is introduced to identify occluded spaces in point cloud map. The occlusion detection algorithm is accomplished by checking depth measurement continuity on range points and its neighbors. Points have been organized into a grid based data structure in order to accomplish efficient neighbor search and depth measurement comparison. Also, an exploration occupancy grid has been introduced for exploration and identifies occluded areas for path planning. Every acquired point cloud at latest vantage position is processed and elevation distribution is computed on each grid. The distribution pattern decides the exploration areas to determine occlusion and possible areas for exploration.

Chapter 4 introduces automated registration algorithms for coarse registration. The Extended Gaussian Image (EGI) based correlation algorithm is introduced as a fully automated registration

method which does not require any pre-alignment information for different local map segments. EGI registration can be applied as a rough registration process to produce pre-alignment for ICP based fine registration. A color assisted automated registration algorithm has been introduces with performance evaluation.

Chapter 5 discusses the point cloud find registration algorithm. The Iterative Closest Point (ICP) algorithm for 3D point cloud map registration is introduced. Color point cloud map registration for the map is generated by a ROAMS mapping robot. Hue data is extracted from color properties in the color point cloud map because of its invariance under different color conditions. The Hue assisted Iterative Closest Point (HICP) algorithm has been proposed to accelerate the 3D color map registration procedure. Noise effects on hue have been evaluated under different noise levels and situations. HICP is compared with typical 3D ICP algorithms on registering 3D point cloud maps from ROAMS exploration.

Chapter 6 describes the path planning algorithm for exploration and mapping. The exploration occupancy grid initialization and update algorithm are introduced for the path planning purpose. Vantage observation candidate positions are generated based on frontier best view and constrained by overlap with previous point clouds. The next position is determined by the shortest traversable distance between current position and candidate positions.

Chapter 7 concludes contributions and summarizes accomplished research, future work is also proposed in this chapter.

Chapter 2 Hardware Platform for Exploration and Mapping

This chapter describes the mobile 3D mapping robot prototype named ROAMS (Remotely Operated and Autonomous Mappings Systems) which is applied to generate high resolution 3D maps of indoor/outdoor urban environments. The robotic system generates 3D maps using a video registered LIDAR scanner integrated with a multiple degree of freedom actuator. This vehicle is also used as a test platform for conducting studies and real-time experiments on autonomous exploration operations. Environmental awareness sensors in combination with a long range wireless communications system are used to enable remote operation and monitoring of ROAMS.

2.1 Introduction to Remotely Operated and Autonomous Mapping System (ROAMS)

ROAMS was developed with an integrated 3D laser ranging device as a mobile facility for urban area mapping (Figure 2.1). ROAMS is able to construct a 3D map within 80 meters range at a single observation position. Multiple sensor groups have been installed on ROAMS to measure range data for spatial map construction, perceiving environmental situation to navigate itself, and monitoring internal pose for localization. Two different operating modes can be switched over to finish mapping in a given area. The remote operation mode allows the operator to control the robot's movement and disable/enable mapping progress, based on the acquired global map. The operator navigates the robot to complete a map of the given area. The second mode is autonomous mode, which fully takes control of robot's movement and navigation, pose mapping sensor to acquire map and construct the global map until it has completely mapped the given area. The second mode has been applied for robotic mapping in this dissertation.

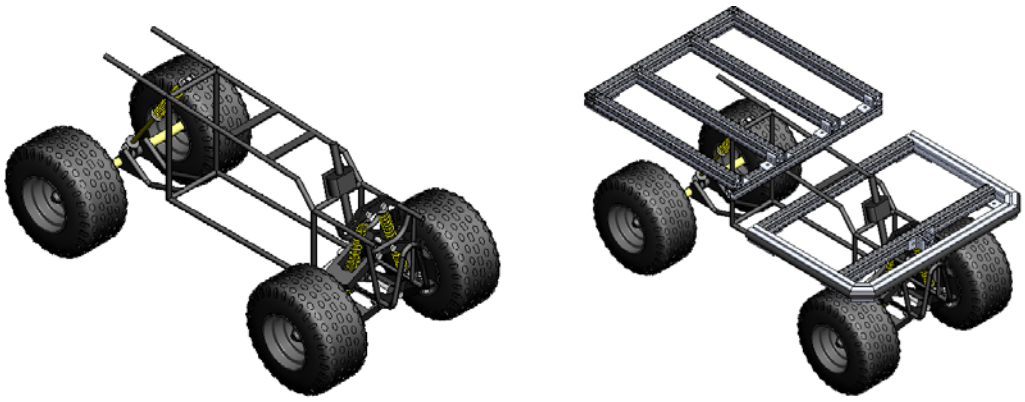
Figure 2.1 ROAMS for urban environment mapping

The ROAMS platform is designed as a four-wheel driven ground vehicle to build maps in an urban environment. The dimension and weight allow the robot to be easily carried and deployed in different areas. Moreover, ROAMS is able to accomplish internal building mapping and travel in hallways, and be lifted by elevator. Color cameras provide ROAMS with the capability to add color attributes on spatial point, and provide detailed information around the environment rather than pure range data. A Light Detection and Ranging (LIDAR) sensor is installed on the rotation base which is capable to rotate at pitch, yaw and roll direction. The LIDAR provides 2D range data by calculating inferred laser pulse reflection travelled time. A position sensor is installed on the rotation mechanism with LIDAR. Therefore, a 3D point cloud map can be generated for ROAMS by rotating LIDAR. ROAMS is constructed by several important modules: vehicle body, drive by wire system, onboard computer, sensor package, battery power management, communication, Mapping actuator and map generation, path planning, and navigation. These

modules work together to enable ROAMS to map specific urban areas under tele-operation or autonomous mode.

Vehicle Body

ROAMS utilizes the mechanical structure of a miniature ATV as the frame for the vehicle. This frame is constructed from high tensile strength hollow steel tubes and is connected to the all terrain wheels through the use of hinges and a front and rear suspension system (Figure 2.2). A secondary frame constructed from T-slot 80-20 Aluminum extrusion is mounted on top of the base frame to allow for easy mounting of hardware components. The 80-20 extrusions have a unique profile that makes them rigid and provide great flexibility for mounting devices. Hardware can be conveniently mounted anywhere along the frame without the need for extra machining.



(a) Base frame and suspension (b) secondary 80-20 Mounting Frame

Figure 2.2 3D model of ROAMS

Drive by wire system

The vehicle is driven using a 750W electric motor connected to the rear drive shaft using a chain and sprocket. Front wheel steering is finished by a high torque worm gear motor. This motor is connected to the steering shaft using a spur gear and an integrated slip clutch mechanism to disengage the motor from the steering shaft if excessive torque is applied. A potentiometer connected to the steering shaft provides steering angle position feedback, enabling the steering motor to be used as a high torque servo motor. Both the driving and the steering motors are controlled by a Dual channel DC motor controller. The motors and controller hardware used are inexpensive and commercially available items.

Onboard Computer

The onboard computer has a Quad-Core CPU and runs programs on Windows XP operating system. All data acquisition, processing, and communication are accomplished by this computer. The computer also translates control scripts into mobile robot movement and also writes integrated sensor reading into scripts. Latest acquired point cloud map is processed by computer and registered into the global map. The computer is learning and understanding the completeness of simultaneously acquired maps and computes its position and orientation based on mapped data and position sensor reading. It evaluates map completeness of given field and navigates the ROAMS vehicle to travel and map unmapped space until complete map is acquired. Communication data compressing/depressing is also finished by onboard computer.

Sensor Package, Communication and Power Management

During remote operations of ROAMS, onboard Sensors and wireless communications are employed to provide the remote operator with 360° situational awareness. These include three wide angle video cameras (one forward looking, one rear facing and one pan tilt birds-eye view), Eight Infrared (IR) proximity sensors (4 forward looking, 4 rear facing), a GPS sensor, a 3-axis gyroscope, an acoustic microphone, and the actuated LIDAR scanning system. While the vehicle is in motion the LIDAR is oriented to look forward and provide 2D ranging data for obstacle detection, or to run localization algorithms such as SLAM. In its current state, ROAMS is used to acquiring 3D maps only while stationary. Data is communicated to the Operator Control Unit (OCU) through the use of an 802.11 b/g Cisco Aironet 1300 wireless communication system. The onboard computer is connected to the Aironet 1300 using an Ethernet cable and a wireless antenna is used for communicating with other wireless enabled devices, such as the OCU. This system can be configured as an access point to enable communication with multiple wirelesses enabled devices, or is paired up with another Cisco Aironet 1300 for increased operational range and security, shown as Figure. 2.3.

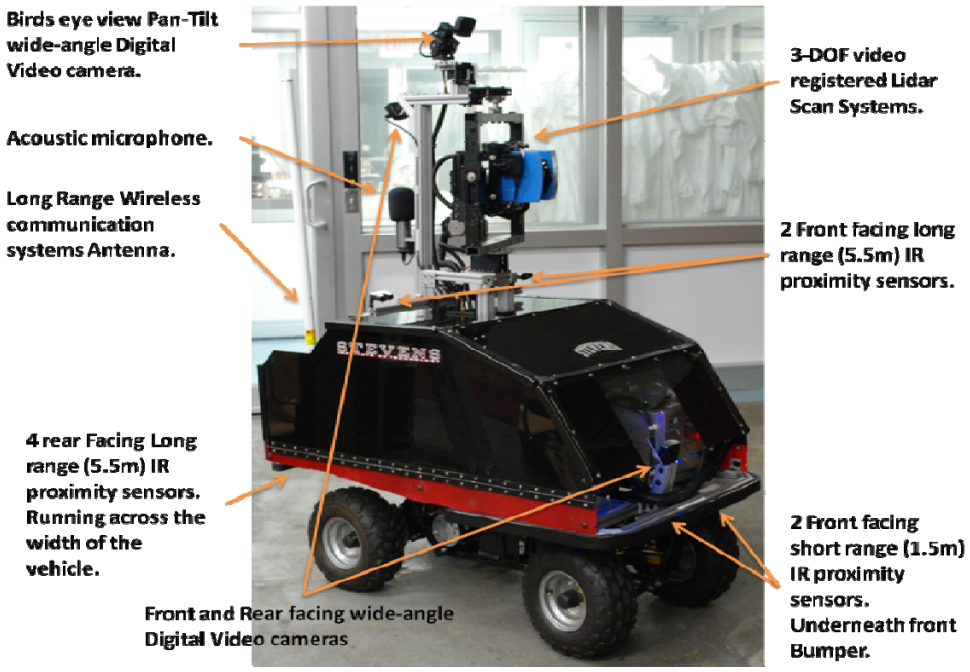


Figure 2.3 Sensor package on ROAMS robot

Twelve 12V Lead Acid batteries (connected as six 24V parallel battery banks) are used to power the computer and other onboard electronics. DC-DC converters are used for down stepping the 24V to lower voltages that are needed for low power electronics. A separate high power 37V lithium Ion battery pack is used for powering the driving and steering motors. The vehicle can be operated for approximately 10 – 14 hours on a single charge with this battery configuration.

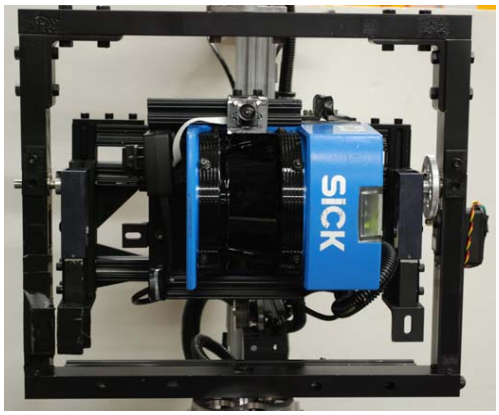
Mapping Actuator and Map generation

The LIDAR Scanner's actuator system utilizes three servo motors to enable orientation of the LIDAR with three degree of freedom (pan, tilt, and roll). This system allows ROAMS to use a single 2D LIDAR to perform tasks that usually require multiple LIDAR systems. 2D LIDAR can only scan a 2D space within certain angles. Each servo has a rotary potentiometer coupled to the rotating shaft to provide angular position feedback. The position feedbacks from these servos are

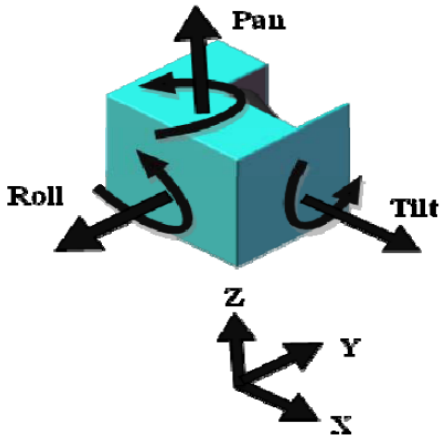
used to calculate the current orientation of the LIDAR and determine which part of the 3D scene is being scanned.

The system has two main configurations. The first is a horizontal configuration, where the roll servo is used to orient the LIDAR to scan horizontally. This configuration is used for performing 2D navigation and obstacle detection while the vehicle is in motion (Figure 2.4(a)). The Second is a vertical configuration, in which the LIDAR is oriented to scan vertically. This configuration is used for performing 3D mapping operations (Figure 2.4(b)). The roll servo is only used to switch the LIDAR between these two configurations. Using a high speed RS-422 serial port, data is received from the LIDAR at a maximum scan rate of 75 Hz for a 180° scan with 1° resolution (38 Hz for 0.5° resolution) While in the horizontal configuration the LIDAR is oriented to face forward using the pan servo, and the angle of elevation of the 2D LIDAR scan is adjusted using the tilt servo. Different angles of elevations can be used depending on the current use of the LIDAR (Figure 2.4(c-d)). For Example if the LIDAR is being used for obstacle Detection then the elevation angle can be lowered slightly, so that ground obstacles can be detected while the vehicle is in motion. After the point cloud map is acquired, all range points have to be analyzed before being transformed into a global map. Map generation filter out noise points, compute correct transformation parameters for the point cloud, and transform the point cloud into a global map. It also evaluates and provides most recent global map information to path planning module.

(a) SLAM Navigation mode



(b) Scanning Mode



(c) LIDAR on 3DOF Platform

(d) Rotation Degrees of Freedom of LIDAR

Figure 2.4 LIDAR for both map generation and navigation

The 3D color scanner used in this effort consists of a 2D LIDAR and two 1.3 megapixels, high frame rate video cameras installed on the LIDAR scanning plane. The LIDAR and cameras are rotated on an axis perpendicular to the scan plane generating 3D color point clouds. Figure 1 shows that both the 2D LIDAR and the cameras are driven on the Y axis (degree of freedom: θ) and on the Z axis (φ: degree of freedom). The rotations are controlled by servo motors installed on the axes. The cameras are calibrated to be on the LIDAR scan plane and a two- pixel wide

image stripe is extracted from the cameras and the color information is matched, in real-time, for the point ranged by the LIDAR. The relative position between cameras and LIDAR is pre-configured and images are aligned to reduce the interpolation burden to a single dimension. The 2D LIDAR generates scans at a frequency of 38 Hz and cameras can provide imagery at 60 frames per second. Time synchronization establishes the color of each ranged point. Two cameras are used to reduce occlusions due to the offset between the LIDAR mirror and the camera lens. All areas visible to the LIDAR are visible to one of the two cameras. The 2D range measurement along with the scanner rotation position (φ) is used to generate the coordinate in a spherical coordinate system which is transformed to Cartesian system as necessary. Figure 2.5 also shows a compact version of the system built for tripod mounted applications.

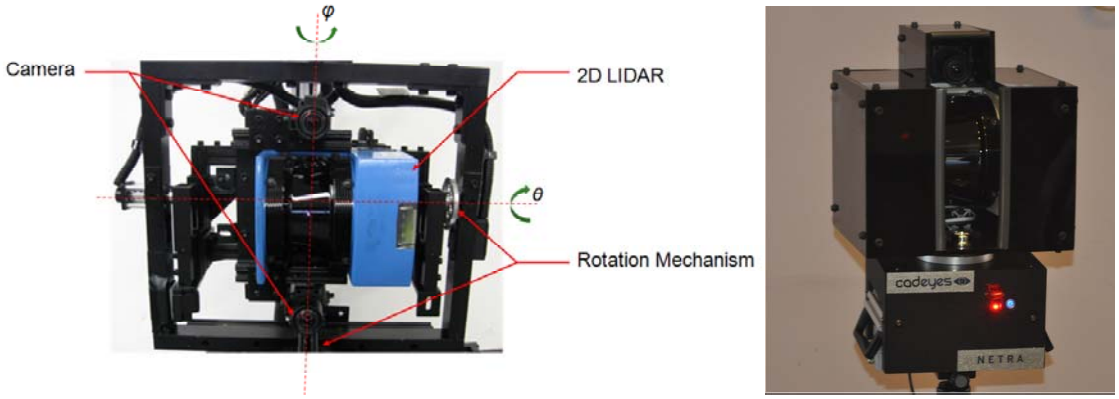


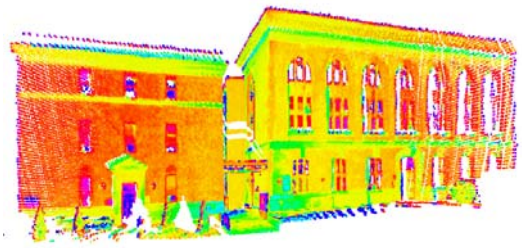
Figure 2.5 3D scanning devices built with 2D commercial scanners

The 3D color scanner is mounted on a mobile vehicle for mapping in large areas. This mobile mapping system can be remotely operated to explore and generate color point cloud data. Position and orientation sensors are installed on ROAMS to provide rough position and orientation estimates. Cameras and short range inferred sensors enable observation of terrain conditions, collision avoidance and allow a remote operator to drive the vehicle. Map data and video feeds

are transmitted using an on-board wireless communication system. This mobile mapping system performs scans only when stopped. The vehicle can localize itself from the map observations and moves directly from one vantage position to the next. This system can generate color point cloud maps with 0.5° angular resolution in the vertical scanning direction with a coverage angle of 100°. In the rotation (φ) direction, the resolution is at 0.1° with coverage angle 300°. The map segment from one vantage position covers a maximum radius of 80 meters.

The point cloud map produced by the LIDAR scanner is shown in Figure 2.6. Figure 2.6(a) shows the camera image taken from the vantage position depicting scenes visible to the scanner. The 3D color point cloud generated at that vantage position is shown in Figure 2.6(b). In this Figure, the coordinate (x,y,z) and the color (r,g,b) for all the pixels known. The point density (spatial resolution of the point cloud) varies on the left and right sides of the color scan scene depending upon the distance between the scanned point and the scanner. The closest area, in the middle of the image, has a shorter distance to the scanner and hence higher density of points. The scanner also records the optical reflection intensity of laser beam. The intensity information is combined with the range measurement data and shown in Figure 2.6(c). The object surface material, color and distance towards the scanner cause variations in intensity data. Similarities between intensity point cloud and color point cloud can be observed between Figure 2.6(b) and (c) on edges, doors, windows etc.

(a) Image of an urban building (b) Color point cloud map



(c) Laser reflection intensity map

Figure 2.6 High dimensional point cloud map segment from single vantage position

2.2 Dynamics and Control System

Rotational Platform

The 3D scanning system produces 3D point clouds with a 2D LIDAR and rotational platform around Z axis. The rotational mechanism is driven by a DC motor at the bottom. Rotational degree is measured by a potential meter installed on the rotation shaft. Rotation control is processed in a close-loop speed control system. This 3D scanning system operates at different speed level according to mission definition. A higher speed level is adopted when scanning time is limited, while lower speed can produce higher resolution 3D point clouds for mapping. Onboard computer transmit speed information to scanning controller, this information is translated and drives geared DC motor to rotate. Position data from sensors is simultaneously combined with 2D LIDAR measurements to construct 3D point cloud map, shown in Figure 2.7.

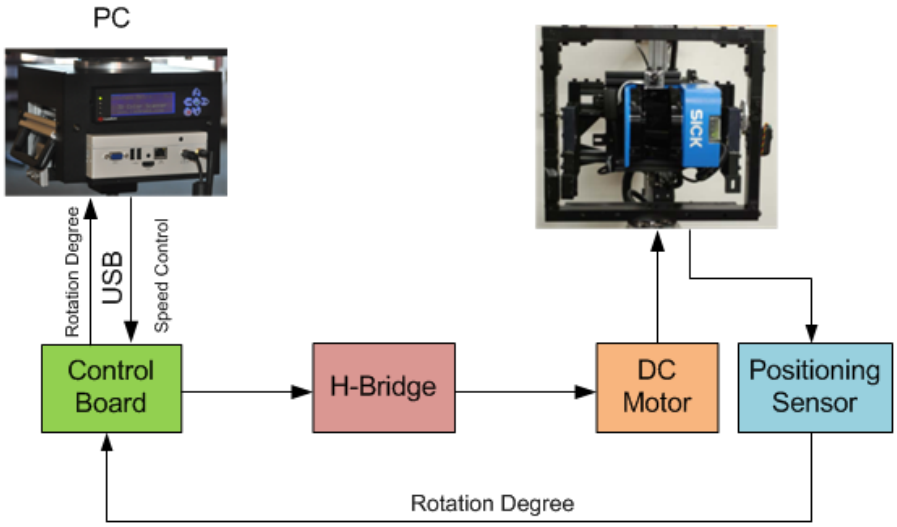


Figure 2.7 Rotation platform system diagram

The DC motor model can be modeled as:

$$J\ddot{\theta} + B\dot{\theta} = \mathrm{T} \tag{2.1}$$

In which, J is the moment of inertia, B is the system damping ratio, T is the torque provided by motor. The motor torque is related to the armature current ***I*** by a constant factor $\boldsymbol{K_t}$ in Eq.(2.2).

$$T = K_t \cdot i \tag{2.2}$$

Armature current ***I*** is defined as:

$$L \cdot \dot{i} + R \cdot i + e = V \tag{2.3}$$

In which L is the electric inductance, ***R*** is the electric resistance, ***e*** is the back emf related with rotational velocity by:

$$e = Ke \cdot \dot{\theta} \tag{2.4}$$

The DC motor rotational control can be modeled as:

$$\begin{cases} J\ddot{\theta} + B\dot{\theta} = K_t \cdot i \\ L \cdot i + \mathrm{R} \cdot i + K_e \cdot \theta = V \end{cases} \tag{2.5}$$

In SI units, $\boldsymbol{K_t}$ tie equal to $\boldsymbol{K_e}$ as $\boldsymbol{K}$. After Laplace transform, above equation is shown as:

$$\begin{aligned} Js^2\Theta(s) + Bs\Theta(s) &= K\ \mathrm{I}(s) \\ Ls\mathrm{I}(s) + R\mathrm{I}(s) &= V - Ks\Theta(s) \end{aligned} \tag{2.6}$$

Use input voltage as input, rotational speed as output. Open loop speed control transfer function is solved from Eq.(2.6) and defined as:

$$\frac{\dot{\theta}}{V} = \frac{K}{(Js+B)(Ls+R)+K^2} \tag{2.7}$$

The motor system is modeled in MATLAB Simulink in Figure 2.8. The physical parameters from the mechanical system is estimated by moment of rotor inertia $\boldsymbol{J}$=3.31 Kg.m^2/s^2, damping ratio of mechanical system $\boldsymbol{B}$=1.3Nms, electronicmotive force constant $\boldsymbol{K}$=2.5Nm/Amp, electric inductance $\boldsymbol{L}$=1.2 H, electric resistance $\boldsymbol{R}$=7.5 Ohm.

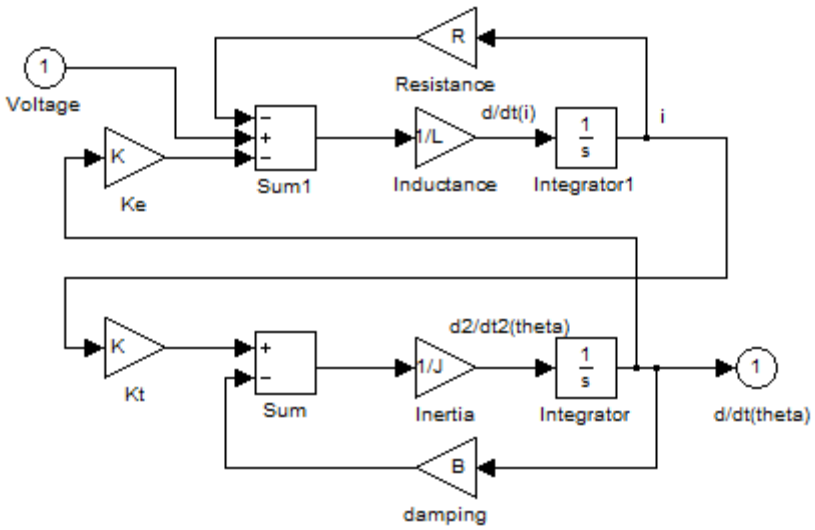


Figure 2.8 Simulink model of DC motor speed control

The DC motor plant model is described as above. The close-loop DC motor speed control system is shown in Figure 2.9. A PD controller is utilized to improve the performance of the system. Select Proportional variable as 10.0, Derivative variable as 2.0, the step response is shown as Figure 2.10.

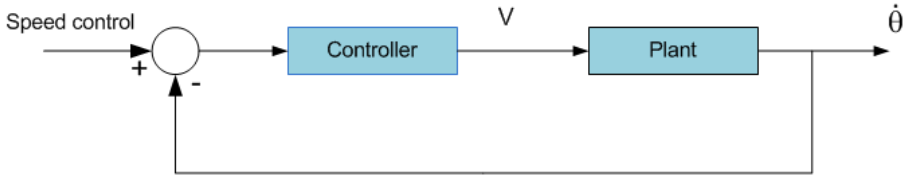


Figure 2.9 DC motor close-loop speed control system diagram

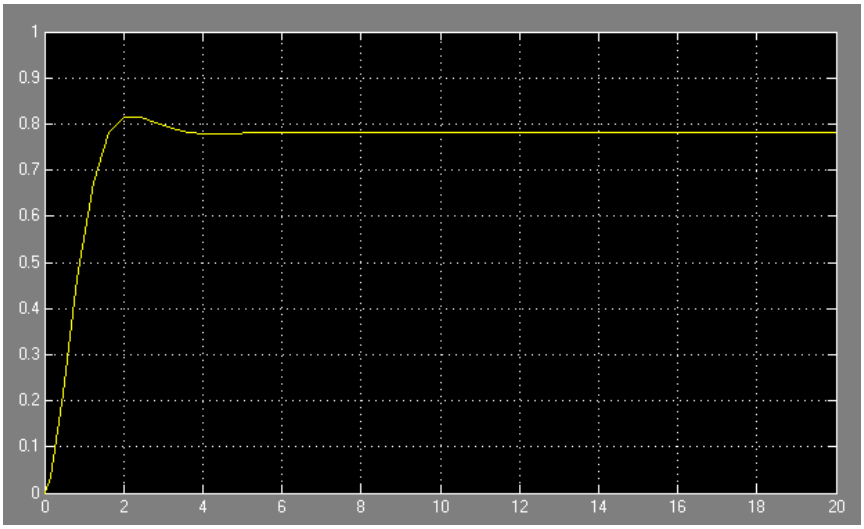


Figure 2.10 Step response of the PD close loop speed control model

ROAMS Dynamics

As mentioned in section 2.1, ROAMS is constructed on a 2 front wheel steering and 2 back wheel driving vehicle. This vehicle follows the Ackerman steering model. Fig. 2.11 presents the control model for the vehicle with thrust control and steering angle control as control inputs. The vehicle kinematics model determines current orientation, steering angle, and the instantaneous velocity.

The vehicle geometry is simplified as a rectangular box with its center of geometry. The steering wheel angle ( ) and driving wheel velocity (V_w) are shown, while the vehicle's orientation ( ) and vehicle velocity (V) are determined in (1) and (2).

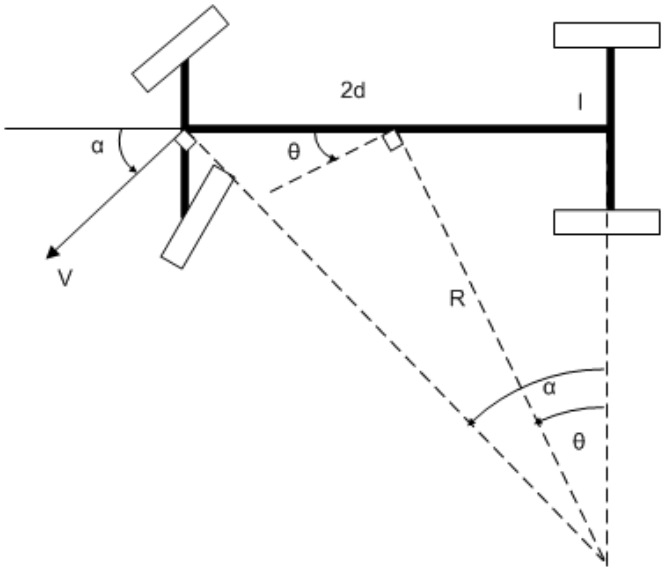


Figure 2.11 Kinematics for ROAMS vehicle

$$\begin{cases} \theta = \tan^{-1}(\frac{1}{2}\tan\alpha) \\ V = V_w \end{cases} \tag{2.1}$$

Turning radius, R, of the vehicle can be determined as

$$R = \frac{d}{\sin\theta} \tag{2.2}$$

Pose and location for ROAMS vehicle can be presented as a vector {*x, y,* }, respectively its movement can be presented as {*Vx, Vy, ω*} for velocity in 2D space. However, this robot is traveling in an urban environment. There are six Degree of Freedoms (6DOF) on the robot. The robot's position and orientation can be described as $\{x, y, z, \omega, \varphi, \theta\}^T$ representing its position at x, y, z direction and orientation angle around x, y, z axis in global coordinate system. Only steering

angle and driving velocity can be controlled as input and other variables can be acquired from sensor reading. Position sensor GPS provides displacement *x, y, z* data, IMU provides orientation information ω, φ, θ. Front wheel steering potential meter gives front where steering angle α which can also be transferred to vehicle steering angle θ. And rear odometer output vehicle speed at *y* direction. In this case, the max turning angle is 50 degree and max climbing angle is 30 degree.

Position sensors provide position and orientation variants of the robot, control command go through steering angle and driving velocity variance when mobile robot is moving. The control input will effect robot location in the next step. The accurate localization is accomplished by the navigation module in ROAMS based on control input, sensor feedback, and simultaneous localization and mapping algorithms in 6DOF.

Control System

The ROAMS robot is able to switch its working mode between teleoperation mode and autonomous working mode. This system structure is shown as Fig. 2.12. As shown in the Figure below, the ROAMS vehicle has 4 layers of data acquiring and processing, and it is also able to perform autonomous tasks like exploration in unknown space. Teleoperation has a higher priority so that if remote operator wants to take over control of the ROAMS vehicle, control permission will be transferred to the teleoperated control unit. Teleoperation transmits and receives response with movement controller via wireless connection, and sensory data is also transmitted to a remote control unit for manipulation, which allows full remote monitoring and control capabilities to the operator of the system. The operator reads sensor feedback data on Operating Control Unit (OCU) (shown in Fig. 2.13), and watches front/rear road situation and controls the robot's movements in unknown environment. The operator controls the mapping process and

brings ROAMS back to base. Autonomous robotic exploration and mapping tasks could also be finished on ROAMS platform. ROAMS has the capability to perceive an environmental situation, self-localize and determine future movement. The sensor layer acquires raw condition information from the environment. The perception layer accomplishes mapping process based on sensor data input, and finds roads and determine movement with optimal cost. The control layer drive ROMOS follows the upper layer's instruction. The UI layer provides the operator with a change to monitor robot's work simultaneously. All these layers cooperate together to finish the autonomous mapping task.

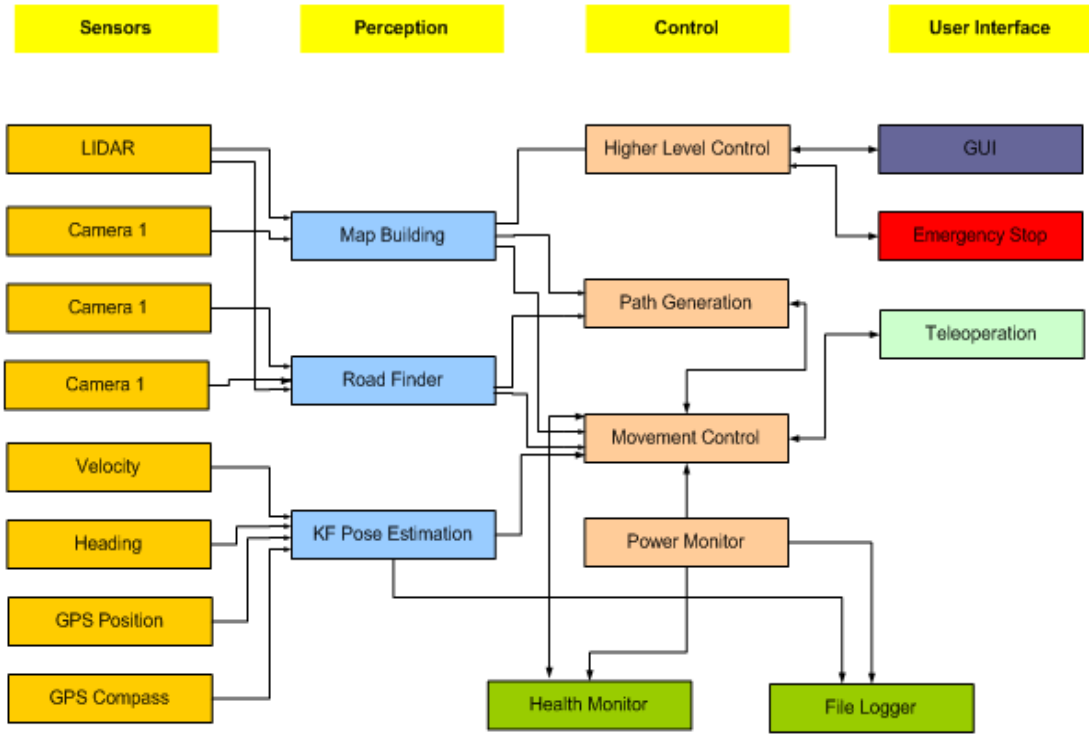


Figure 2.12 ROAMS system overview

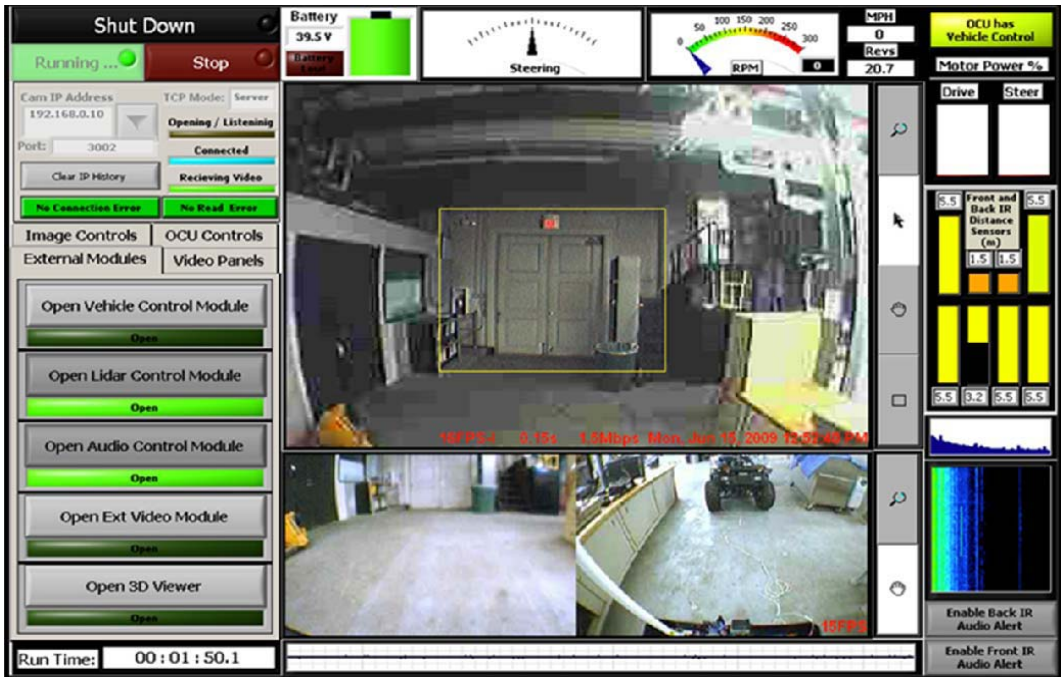
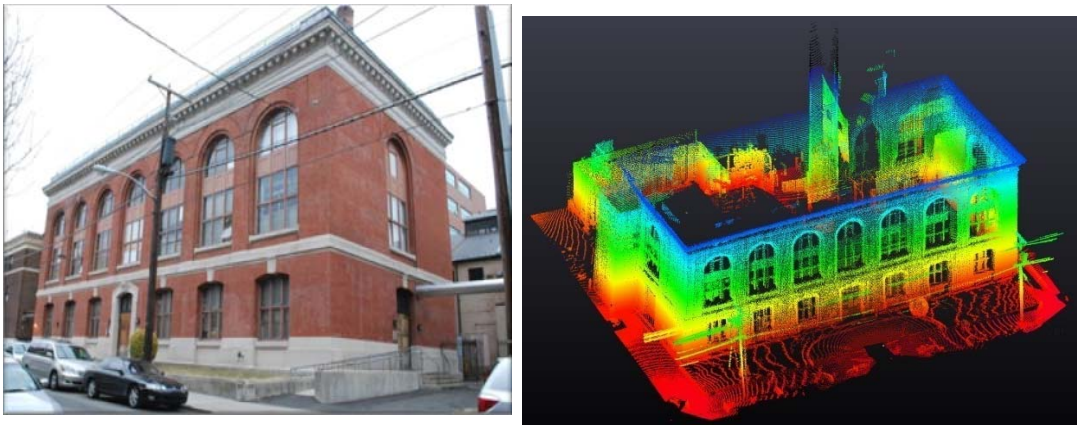


Figure 2.13 Remote control unit user interface

2.3 Application in Surveying and Mapping

Point cloud maps are generated from output data from ROAMS. A typical map that ROAMS can generate is shown in Fig. 2.14, Fig. 2.14 (a) shows a picture of Carnegie building in Stevens Institute of Technology, after a series of surrounding scans, digital surface map can then be constructed to provide high precision digital geometric measurement of the building (Fig.2.14(b)).



(a) Building for complete mapping (b) Constructed 3D map using LIDAR

Figure 2.14 ROAMS scan about urban building

A series of color point cloud maps about this building are generated on ROAMS. Figure 2.15 illustrates four color point cloud acquired from 4 different positions around Carnegie building, these point clouds are used as in the following chapter to validate map registration algorithm and comparison between different algorithms.

Figure 2.15 Color point cloud acquired around urban building

Figure 2.16 shows observation vantage positions about a hallway in Carnegie building in Stevens Institute of Technology. In total, eight local maps have been acquired at respective positions. These maps include both outdoor and indoor scenes of an urban environment and a few of them are shown in Figure 2.17. These color point clouds include accurate measurements on both dimensional building surface and color property. This set of maps is applied to evaluate map registration algorithms. Registered large-scale lower campus maps can be utilized to determine the completeness of the map and miss explored space. Scan positions in Figure 2.17 are applied to compare with autonomous path planning algorithm for efficiency.

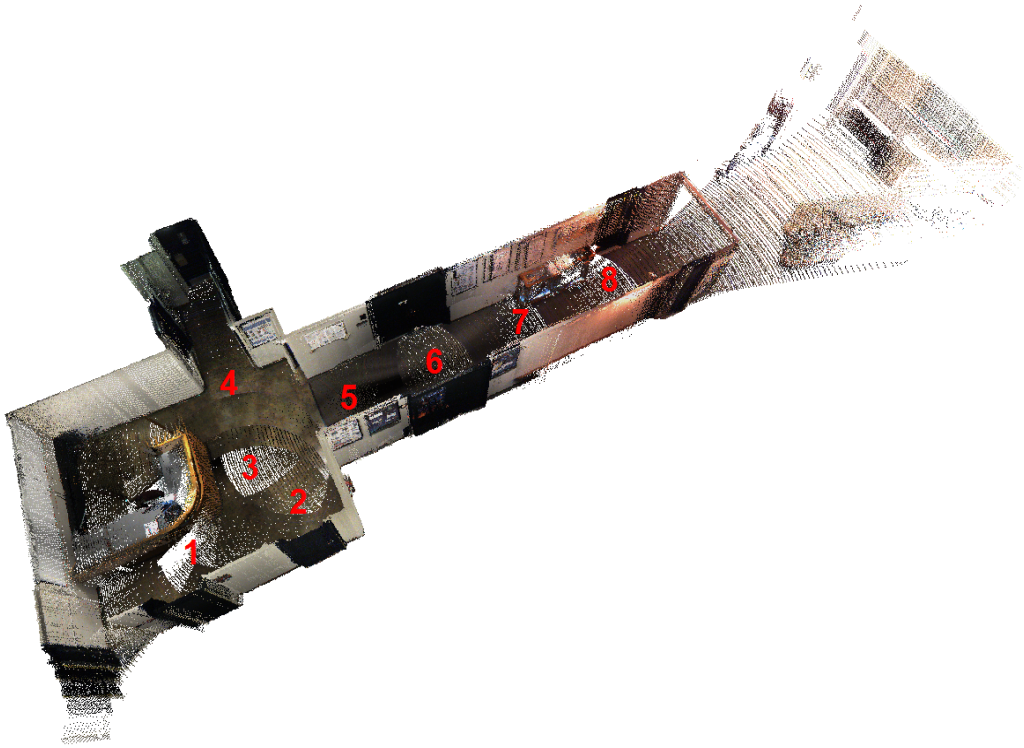


Figure 2.16 Nine mapping position on hallway in Carnegie Building

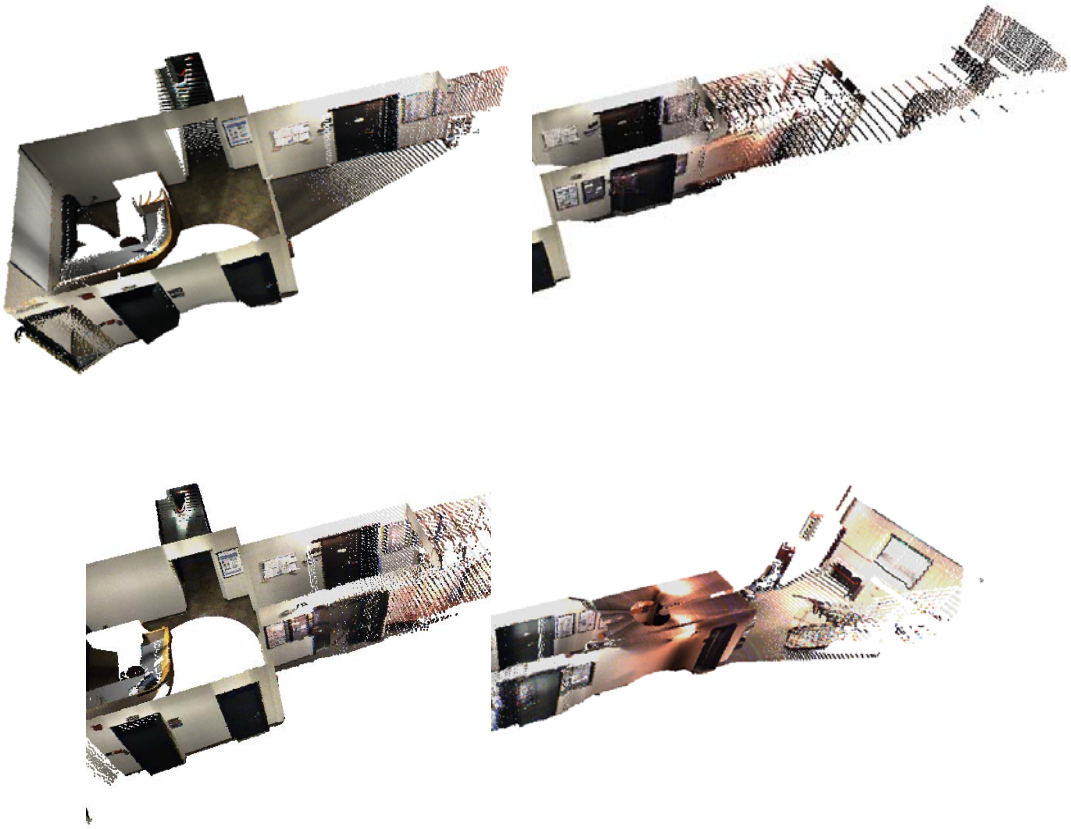


Figure 2.17 Indoor 3D color point cloud maps

The ROAMS mapping robot can be placed in both indoor and outdoor urban situations because of its proper dimension and drivability. Major components of this platform such as mechanical system, sensor package, driving by wire system, control system, mapping system, and software structure have been illustrated. The specs of 3D color mapping station and the depth adaptive mapping method are introduced. Groups of point cloud maps have been produced by ROAMS in this chapter to verify algorithms in map registration, completeness evaluation, and autonomous exploration strategies.

Chapter 3 Algorithms for Point Cloud Analysis

This chapter focuses on methods for analyzing point clouds, identifying occluded areas, and evaluating map completeness. The status of map completion in terms of area coverage, filling in occluded areas, and recognizing unreachable areas must be assessed. We introduce discretization of an exploration area into cells, a sampling density based occupancy assessment and a cell classification, in this chapter. We developed a height pattern assessment for determining if an exploration robot can position itself in a grid or navigate through a cell based on the scanned terrain and robot characteristics. This chapter presents various algorithms and descriptive numerical results illustrating their performance. These algorithms are used in the registration and exploration schemes presented in the subsequent chapters.

3.1 State of Art

The map completeness problem is generally addressed with several methodologies including grid occupancy, obstacle recognition, and object view completion detection. Grid occupancy techniques [71, 74] entail projecting the acquired point cloud onto a grid and calculating the density of sampling. Techniques investigated include grid based occupancy map, network/graph, cell based map [44] and template based completeness evaluation[20].

The occupancy grid map representation is one of the most commonly used methods to determine map completeness. The mapping area is turned into a 2D grid, and maps acquired from different vantage positions are projected onto the grid. Grid cells are marked as occupied when data exists in a cell of the grid. A notion for the level of occupancy can also be used based on the density of

point samples present in the cell. The contiguity of mapped cells can act as indicators for map completeness. For example, a map can be assumed to be complete when all the cells are classified into mapped or unreachable spaces.

A major challenge is determining whether it is possible to map a certain area due to the physical limitations of a robot. For example, spaces behind the walls of a hallway may not be accessible due to closed doors or other obstacles. Spaces marked as unreachable by the robot may require human intervention to enhance the completeness of the map. Analysis of both the local and global relationships between mapped, unmapped, and unreachable areas based on the available map information is critical to complete the map exploration. A criterion based on contours extracted from the scan data, and whether or not mapped areas form closed contours provide assessment methods for map completeness evaluation. Ascertaining that the gaps in map contours are indeed traversable paths requires recognizing pathways such as gateways, doors and roads, as well as knowledge of the capabilities of the robot. Therefore, notions for ground cells and navigability of the ground cells are desired.

As the range data is obtained from laser based devices, shadowed or occluded areas remain unmapped. Occluded areas in the scan data for the scanner described in the previous chapter are due to three reasons: i) LIDAR scanner range limitations (cannot see far enough), ii) scanner angle limitations (rotating mirror has constraints) leading to blind zones, and iii) occlusions from objects in the scene. Both range limitations and blind zones are known from the design of the scanner. Detection of occluded areas due to objects in the scene is more complicated.

3.2 LIDAR Scan Pattern

The 3D mapping system used in this effort was built from a 2D LIDAR and a rotating platform. The 2D LIDAR acquires range measurement data on its scanning plane and 3D maps are created

by acquiring multiple 2D scans and registering the scans. The raw data generated in polar coordinates is transformed into Cartesian coordinates for visualization and map construction of a reference coordinate system. As the mirror and the scanner head are actuated in a controlled manner, the scanner on the scene generates a known pattern of points. We exploited the distortion in this pattern to recognize the occlusions and areas of low accuracy in measurements.

Figure 3.1 shows the coordinate frames of the scanner. The angle represents the mirror angle reflecting the laser in the 2D LIDAR scanning plane. The angle represents the head rotation or the movement of the scanning plane about the z-axis. The 2D LIDAR frame samples the space with a resolution, $\Delta\theta$ of 0.25˚. Due to mirror rotation setting limitations, the angle θ spans from 40˚ to 140˚, and at maximum resolution generates a total of 401 discrete range points in one scan frame. The scanner motor controller determines the head rotation speed, $\dot{\phi}$, which is kept at a constant value. A position sensor is used to detect the head position, for every scan.

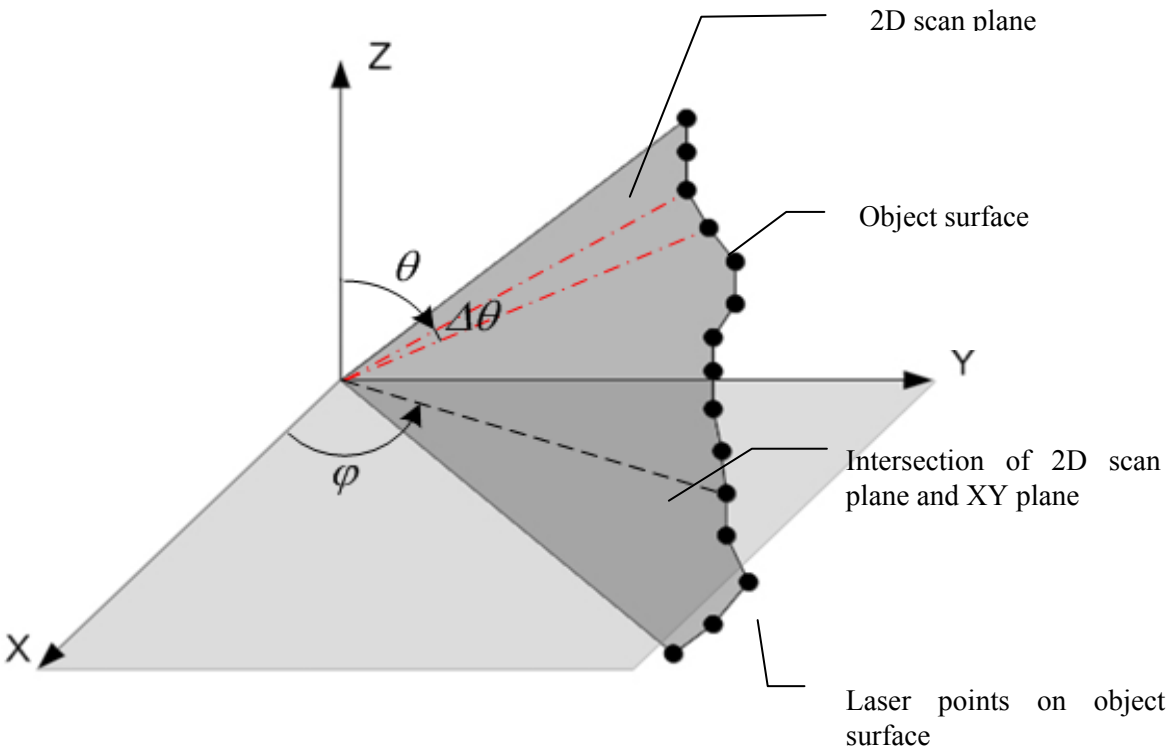


Figure 3.1 LIDAR scan coordinate reference

Figure 3.1 also illustrates the measurement pattern of the 3D LIDAR. The 2D scanning plane is identified by grey color in Cartesian coordinates. The 2D scanning plane is perpendicular to the xy plane and rotates around the z axis. The xz plane determines the reference for the measurement of the rotational angle φ. The mirror angle is denoted by θ and radial distance r denotes the range to the object surface. The coordinates in the Cartesian frame (x,y,z) can be calculated by:

$$\begin{cases} x = r\sin\theta\sin\varphi \\ y = r\sin\theta\cos\varphi \\ z = r\cos\theta \end{cases} \tag{5.1}$$

A single scan of point cloud data typically contains about 500,000-1,000,000 points for the scanner used. However, due to the limitations on the mirror rotation of the 2D LIDAR used, a 40˚

coverage area on both top and bottom cannot be scanned by the LIDAR and circular areas of self-occlusion – referred to as the blind areas will be present in all scans. Blind areas are located below/above the mapping position as a circular area with radius of r_{b1} *and* r_{b2}, when the scanner is placed at a height with distance h_1 from the ground and a ceiling is present at a height h_2, as shown in Figure 3.2. The distance from the LIDAR to the ground is h_1 and ground blind zone area is related as:

$$S_{b1} = \pi(h_1 \tan(40^o))^2 \tag{5.2}$$

Occlusions appear when one object surface closer to the LIDAR blocks the laser ray from other objects and surfaces behind it. A schematic of the situation is shown in Figure 3.3 (a). Two sides of the object facing toward the LIDAR are mapped; the other sides and objects behind this object are occluded as shown in Figure 3.3(b). This situation appears as though the object surface is discontinuous on scanning depth direction in Figure 5.3(b). A typical color point cloud generated indoors with this scanner has a blind zone with a radius of about 2.6m. The blind zone is depicted in Figure 3.4 (a). Occlusion generated by object surfaces is shown in Figure 3.4(b). A close-up view shows two major occlusion spaces in Figure 3.4(b).

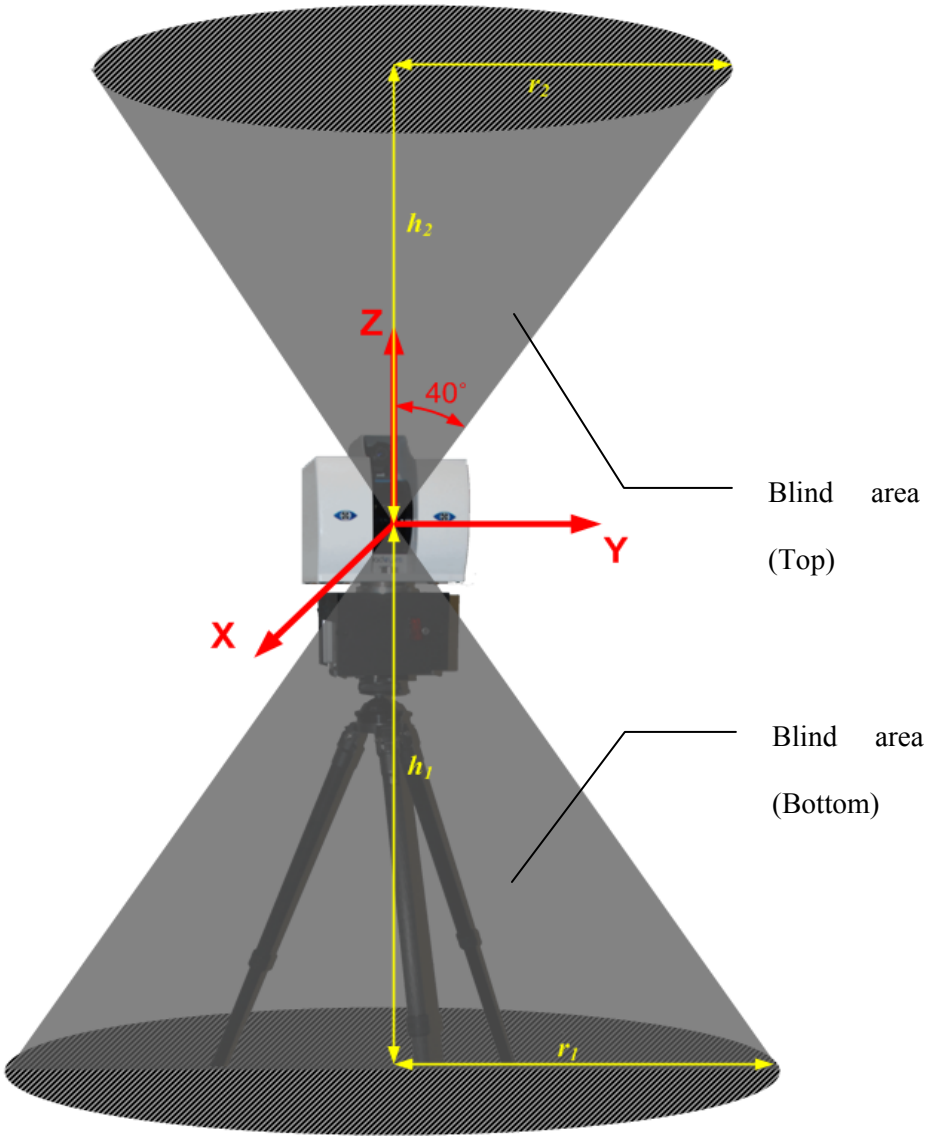
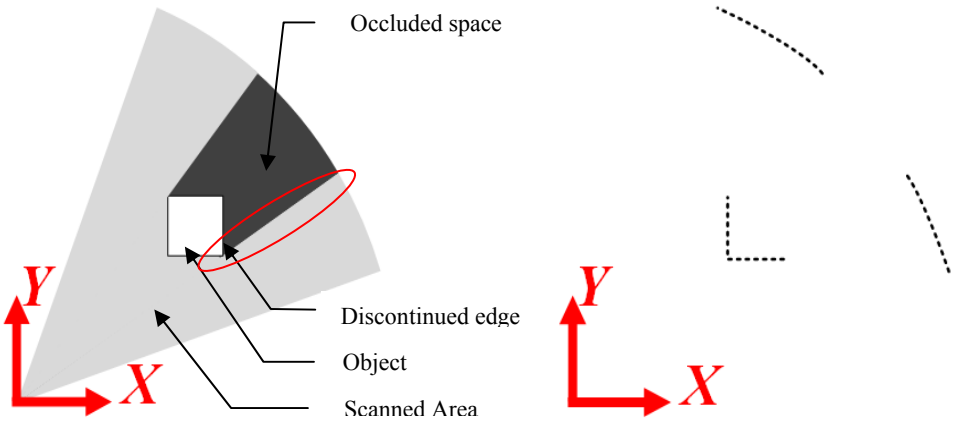


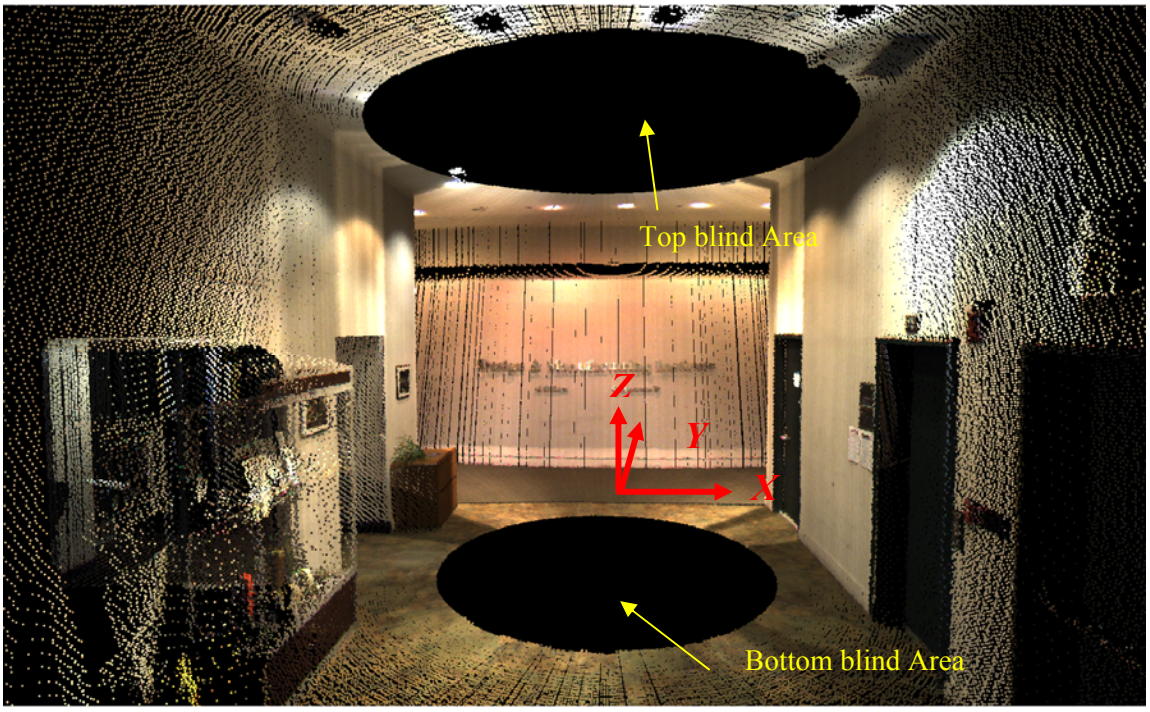
Figure 3.2 Blind areas in 3D LIDAR generated point cloud map

The finished map should not include blind areas, unmapped object surfaces, or shadows due to occlusions. A map from a single observation vantage position normally includes both occlusions and blind zones. The selection of the next observation vantage position must consider the location of the occluded spaces so that a global map can be incrementally completed.



(a) Occlusion created by surface closer to LIDAR (b) point cloud map with occlusion

Figure 3.3 Occlusion created by object surface



(a) Indoor scan color point cloud map including top & bottom blind area

(b) Close up view of occlusion in the scan

Figure 3.4 Indoor point cloud with occlusions

3.3 Algorithms for Occlusion Detection

The detection of occlusions in a point cloud generated from one vantage point can be accomplished with a pattern analysis. We use a range variance metric to measure the distortion of the scan pattern and identity areas of occlusions. The mapping system saves data sequentially by depth distance measurement, tilt, and pan angles. The raw measurement data format is shown in Table 3.1. The measurement data is organized for each scan into an N x M array. The row number represents tilt angle θ while the column number represents pan angle φ. The neighboring points are denoted as the North, South, West, and East points, as shown in Figure 3.5. The distances to the nearest neighbors can be determined without much of a computational burden if the data is organized as shown in Table 3.1.

$\grave{e}_1$	$r_{1,1}$	$r_{1,2}$	…	$r_{1,j}$	$r_{1,j+1}$	…	$r_{1,M}$
$\grave{e}_2$	$r_{2,1}$	$r_{2,2}$	…	$r_{2,j}$	$r_{2,j+1}$	…	$r_{1,M}$
…	…	…	…	…	…	…	…
$\grave{e}_i$	$r_{i,1}$	$r_{i,2}$	…	$r_{i,j}$	$r_{i,j+1}$	…	$r_{i,M}$
$\grave{e}_{i+1}$	$r_{i+1,1}$	$r_{i+1,2}$	…	$r_{i+1,j}$	$r_{i+1,j+1}$	…	$r_{i+1,M}$
…	…	…	…	…	…	…	…
$\grave{e}_N$	$r_{N,1}$	$r_{N,2}$	…	$r_{N,j}$	$r_{N,j+1}$	…	$r_{N,M}$
	$\ddot{o}_1$	$\ddot{o}_2$	…	$\ddot{o}_j$	$\ddot{o}_{j+1}$	…	$\ddot{o}_M$

Table 3.1 Organized 3D measurement data for efficient search

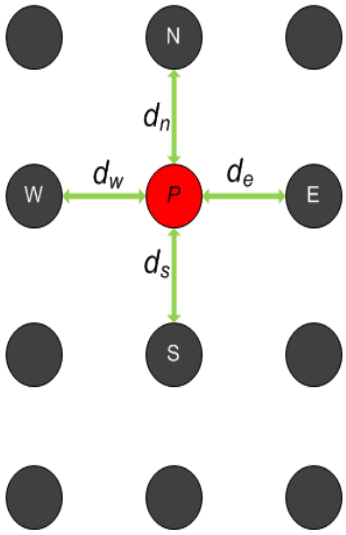


Figure 3.5 Depth measurement comparison with 4 neighbor points

In a 3D scene, the neighborhood points for given point $\boldsymbol{p_{i,j}}\{r_{i,j,}\ \theta_i\ ,\ \varphi_j\}$ are the four points of $p_{i+1,j}$; $p_{i-1,j}$; $p_{i,j+1}$ and $p_{i,j-1}$, denoted as the North, South, East and West neighbor points respectively. A maximum range variance metric is defined as the largest difference in range from the scanner location between two points: $v_{i,j}$, which is the maximum difference between ranges, $r_{i,j,}$ and its neighbor's range from the scanner position. For point $\boldsymbol{p}$ in Figure 3.5, the range variation for the four neighbors are calculated by:

$$d_n=|r_{i,j}-r_{i-1,j}|,$$

$$d_s=|r_{i,j}-r_{i+1,j}|,$$

$$d_e=|r_{i,j}-r_{i,j+1}|,$$

$$d_w=|r_{i,j}-r_{i,j+1}|.$$

A range variance metric is defined as $v_{ij}=\max\{d_n, d_s, d_e, d_w\}$.

An algorithm has been developed to detect points that are on the boundary of an occlusion by comparing the range variance metric to an expectation threshold value for the scene. The occlusion detection algorithm can be summarized as:

For i=1 to length (θ)

For j=1 to length (φ)

For point $\boldsymbol{p_{i,j}}\{r_{i,j,}\ \theta_i\ ,\ \varphi_j\}$

$d_n=|r_{i,j}-r_{i-1,j}|$,

$d_s=|r_{i,j}-r_{i+1,j}|$,

$d_e=|r_{i,j}-r_{i,j+1}|$,

$d_w=|r_{i,j}-r_{i,j+1}|$,

$v_{i,j}=\max\{d_n,\ d_s,\ d_e,\ d_w\}$,

If $v_{i,j}>v_{thres}$

Mark $\boldsymbol{p_{i,j}}$ as occlusion boundary points.

End for

End for

The threshold v_{thres} is determined from analysis of the scene. Geometrically, v_{thres} can be related to the angle between the ray cast and the geometry sensed by the LIDAR. Expecting that there is continuity of the geometry in the scene, it is assumed that occlusions are present when a severe discontinuity is sensed. An angle variation threshold value is determined and used to filter points belonging to the interface between mapped and occluded areas. Selection of the threshold value is discussed further in the next section.

Determination of Threshold

This algorithm relies on the scanner point pattern and the expectation of the object surface's characteristics. The distance Δs_i between the i^{th} point and $i+1^{th}$ point can be utilized to identify if the i^{th} point is located on occlusion boundary on the LIDAR scanning plane. Figure 3.6 shows the schematic of a typical range difference between two points. Points on occlusion boundaries have large Δs as the LIDAR senses a range discontinuity. However, the point spacing is a function of the range, i.e. Δs_i is a function of r_i and r_{i+1} as well as the scan position angles (and). Figure 3.7 shows the geometric relationship between Δs_i r_i and r_{i+1}. $\Delta\theta$ is a constant, e.g. 0.25°, and can be determined by the scanner resolution.

From Figure 3.7, the distance between neighboring points is determined by range difference as:

$$\Delta s_i = \frac{r_i \sin(\Delta\theta)}{\sin(\alpha - \Delta\theta)} \tag{5.3}$$

Since is determined by LIDAR resolution at 0.25° on the North-South direction, we expect a value for close to /2 for flat walls. When is small, it indicates a sharp corner in the wall and a high likelihood for an occlusion. Figure 5.8 illustrates the scenario in which a large distance separates two walls. Two contiguous range points sense two different walls and there is a sudden change in the range between the two points. This discontinuity in the range translates to low values of . We use as a measurement of the range discontinuities. Numerical experiments have shown that the occlusions can be detected robustly when the threshold is set to 5° for this scanner.

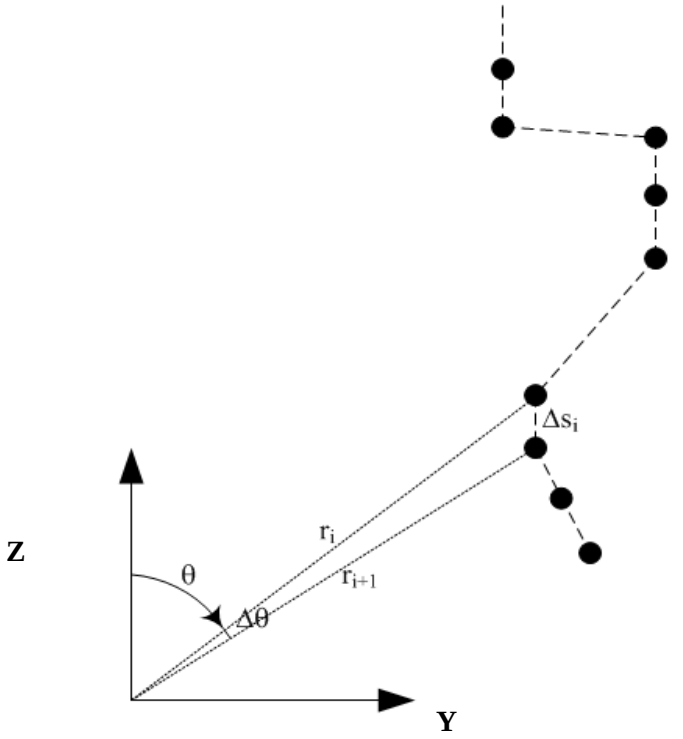


Figure 3.6 Points distribution on LIDAR scanning plane

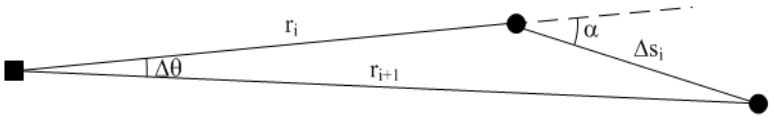


Figure 3.7 Neighboring points distance based on scanning plane range measurement

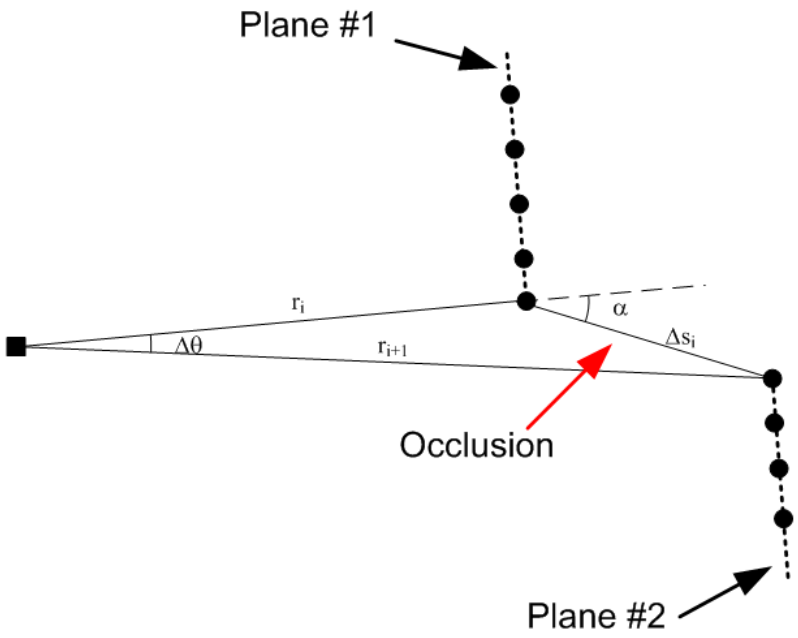


Figure 3.8 →0 near areas of occlusion

The point spacing s_i increases substantially when occlusions are present which translates to small values of Figure 3.8 shows the discontinuity and substantial distortion of the point spacing at occluded areas. The point spacing is normalized by range to reduce the range effect on the spacing. Eq. 3.4 shows the geometric relationship between various parameters.

$$\frac{\Delta s_i}{r_i} = \frac{\sin(\Delta\theta)}{\sin(\alpha - \Delta\theta)} \tag{5.4}$$

Large values of $\frac{\Delta s_i}{r_i}$ translate to low values of . In Figure 3.9, the relationship between the point spacing metric $\frac{\Delta s_i}{r_i}$ and the object surface angle ( ) is shown. When the angle is small, a large value of point spacing is seen. Although the absolute value depends upon the scanner resolution ( ), the point spacing can be easily determined when the object surface angle is known from this figure. We assume that when the object surface angle is detected as 5^o, the point

is on an occlusion boundary. In the event that the < 5 in the scene, the diameter of the laser spot is large and the reflection is obtained from two surfaces and the measured point may have a large error. In either case, we determine that the area in question must be rescanned.

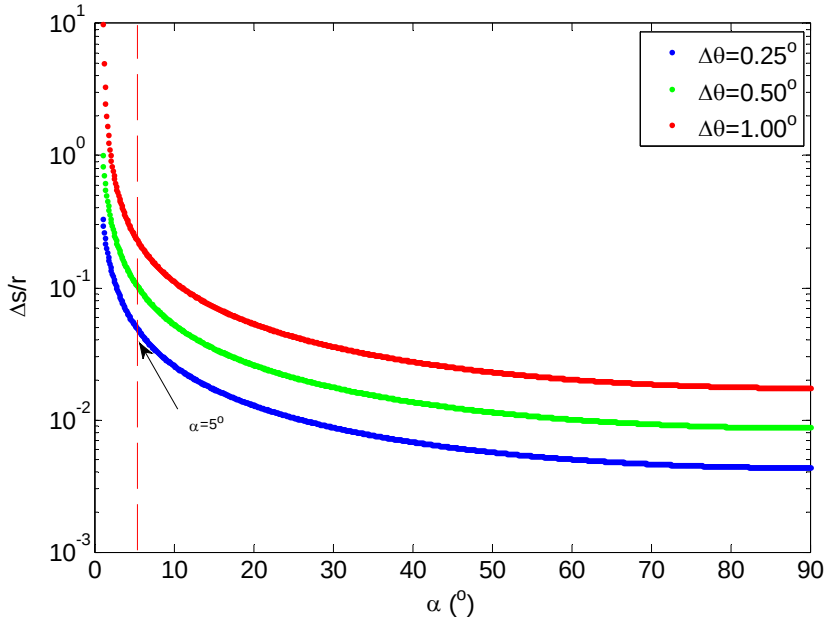


Figure 3.9 Relationship between the angle and the point spacing for various resolutions

From Eq. 3.4, we can establish that when $\frac{\Delta s_i}{r_i} \geq \frac{\sin(\Delta\theta)}{\sin(\alpha - \Delta\theta)}$ the point is on the occlusion boundary. Assuming that denotes the occlusion boundary, $\frac{\Delta s_i}{r_i}$ is determined and the point cloud, shown in Figure 3.4, is processed for occlusion boundaries. Figure 3.10 shows the

point spacing cut off and the points sorted as those in occluded boundaries ($\frac{\Delta s_i}{r_i} > 0.05$) and those within the surfaces.

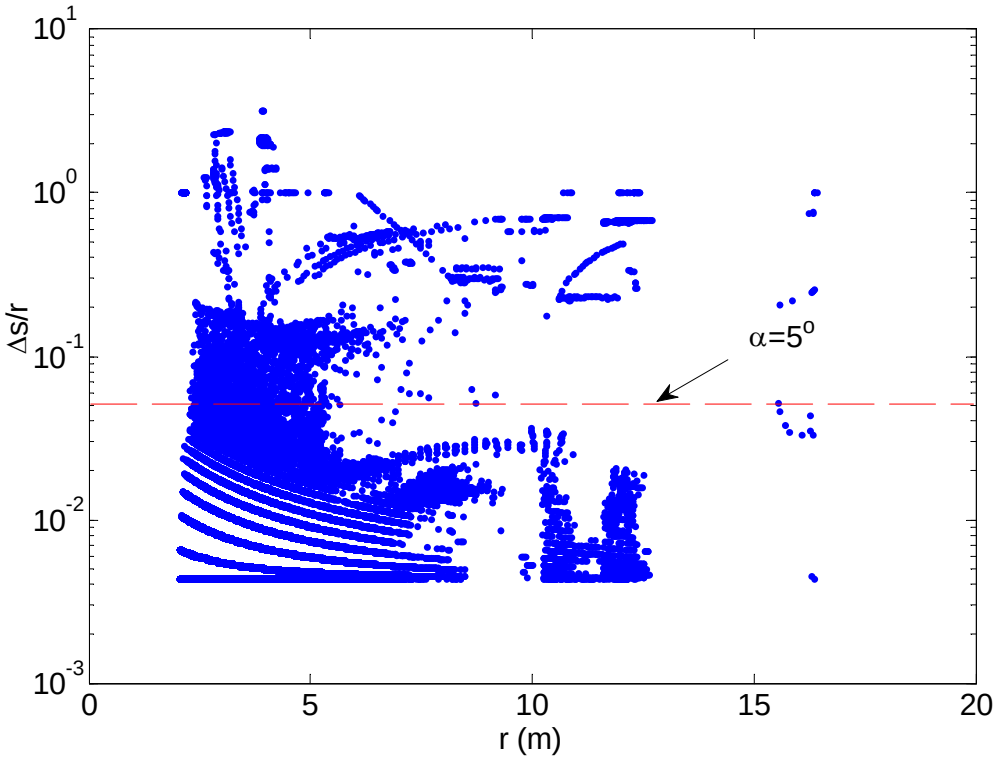


Figure 3.10 Range variations for tilt axis ( ) rotation

On the LIDAR rotation direction in Figure 3.11, the distance between neighboring points is

$$\Delta s_i = \frac{r_i \sin(\Delta\varphi)}{\sin(\beta - \Delta\varphi)} \tag{5.5}$$

In which is the angle between the object surface curvature and LIDAR scanning beam. Similar with the occlusion boundary detection criteria on the scan plane, the criteria on rotational direction can be described as:

$$\frac{\Delta s_i}{r_i} \leq \frac{\sin(\Delta\varphi)}{\sin(\beta - \Delta\varphi)}$$

In which is selected as 5°, $\Delta\varphi$ is 0.25°. This criterion is also applied to identify the occlusion on rotational direction. That scenario is shown in Figure 3.12. Combining the point classifications in both directions, the occlusion boundaries are extracted. Figure 3.13 shows the indoor scan being used as an example in this chapter with the detected occlusion boundaries. The figure also shows two close-up views of the occluded areas.

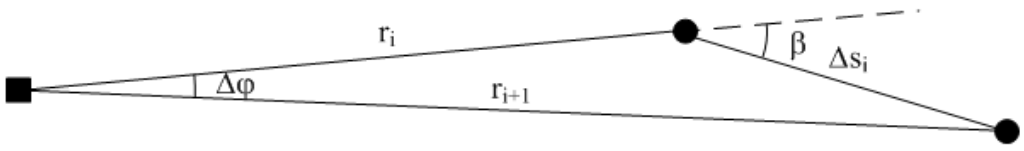


Figure 3.11 Range discontinuity definition for the pan ( ) axis rotation.

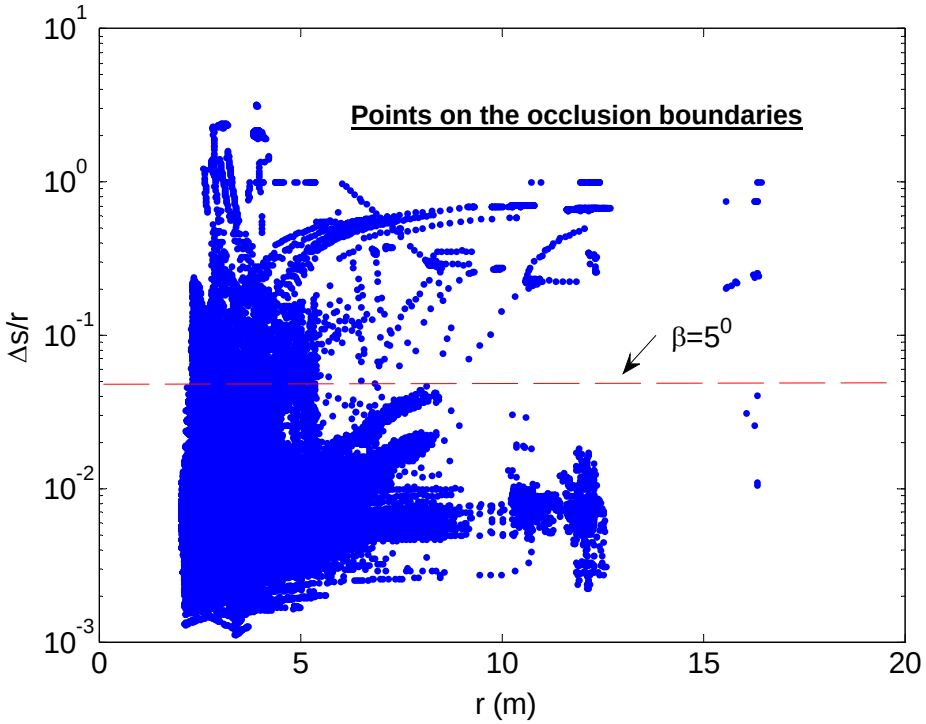
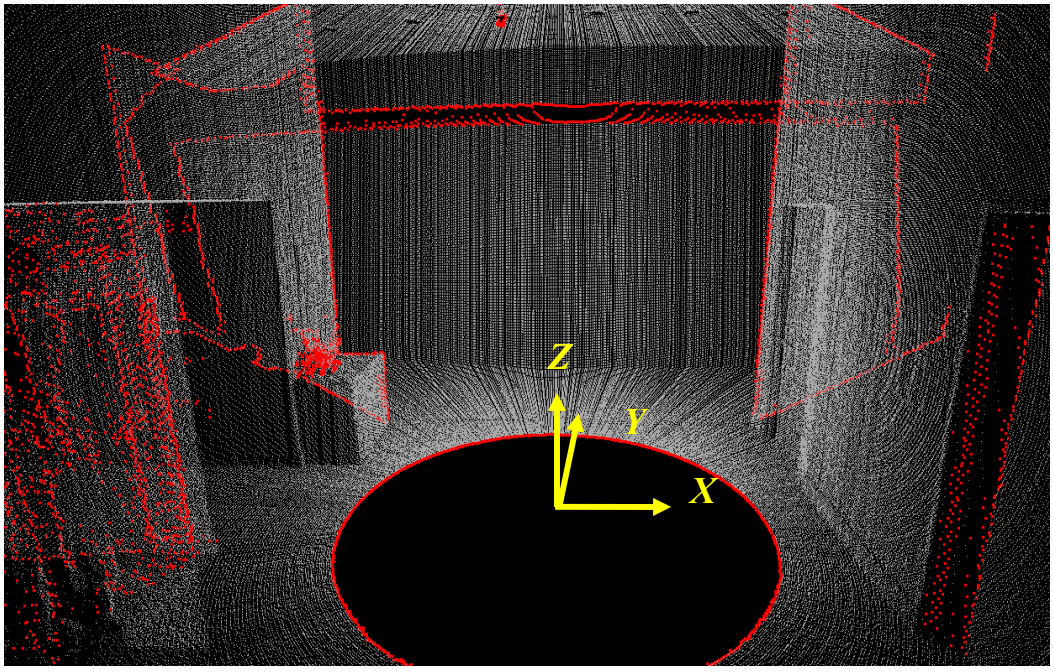
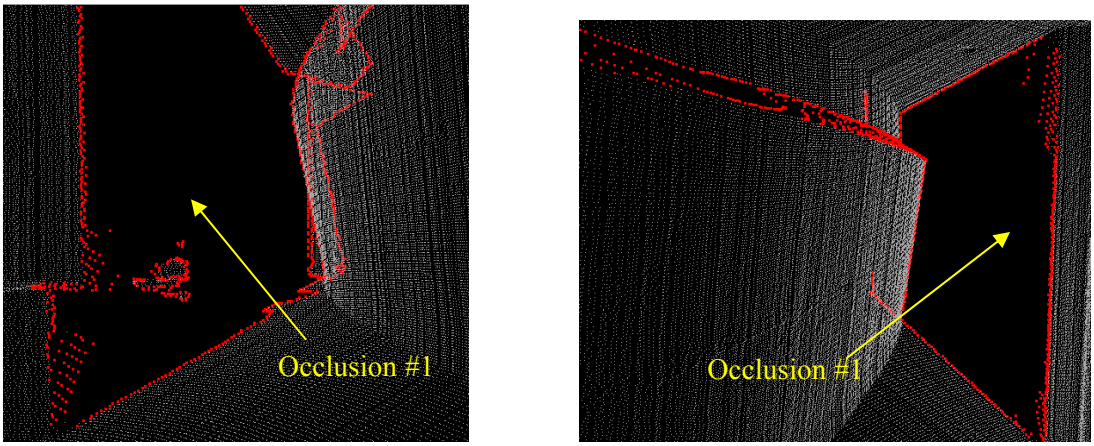


Figure 3.12 Range variations for the pan ( ) axis rotation



(a) Detected occlusion boundary (red points) in 3D point cloud map



(b) Close-up view of detected occlusion boundary (red points)

Figure 3.13 Detected occlusion boundary in 3D point cloud map

3.4 Voxel Point Cloud Sub-sampling

The point cloud must be pre-processed to remove heterogeneity in the point distribution that is inherent in most scanners. Typically, point clouds are registered together based on their distribution histograms of surface normal and space occupancy. Therefore, it is critical to compute accurate point surface normal vectors on every point to conduct reliable spherical correlation for rotational alignment. The occupancy grid in 3D space requires a sub-sampling algorithm for translational correlations. Points are not evenly distributed in every scene at every vantage position because the nature of LIDAR scanning. Figure 3.14 shows a typical scan for a hallway in which points closer to the scanner have considerably higher density than points at a distance. This heterogeneity in point distribution leads to substantial errors in normal computations and distance error computations.

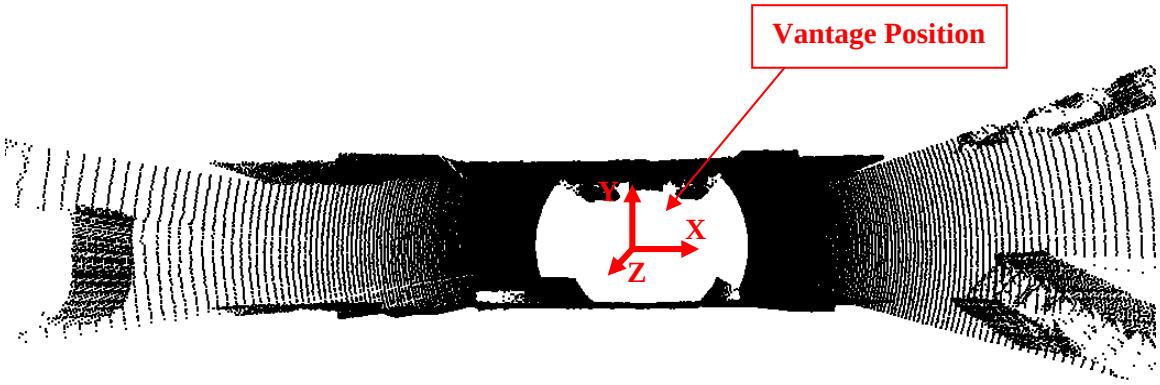
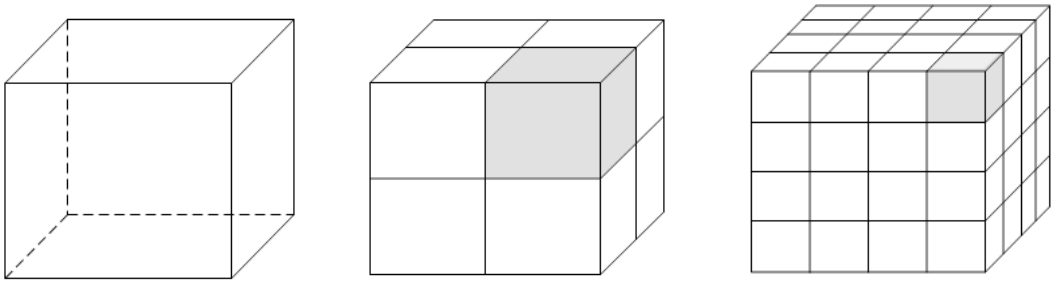


Figure 3.14 Point distribution for a point cloud. Point density closer to the scanning (vantage) point is substantially high than the density at a distance

The point distribution is determined by the LIDAR's position and distance from the object surface towards LIDAR. In Figure 3.14, points on ground surface far from vantage position have lower density than closer areas. Therefore, it is necessary to implement a voxel based sub-sampling algorithm before point surface normal computation and occupancy grid construction. A cubical voxel needs to be defined and only one point in this voxel is kept after sub-sampling. In this case the point distribution in 3D space will be less affected by LIDAR scanning pattern.

An octree based sub-sampling algorithm is implemented on point cloud data. The sub-sampling algorithm is performed on 3D space, and color and intensity attributes are retained or reassigned after the sub-sampling. For each point cloud, a bounding box is computed. An octree is constructed by dividing bounding box into 8 sub boxes in every step as shown in Figure 3.15. The bounding box of a point cloud is computed as $\boldsymbol{B}\{x_{max},x_{min},y_{max},y_{min},z_{max},z_{min}\}$ as shown in Figure 3.15(a). The first level of octree includes 8 sub-bounding box as shown in Figure 3.6(b) and the second level octree contains 64 child bounding boxes in Figure 3.6(c). The sub-division is recursively continued till a stop criterion is met.



(a) Bounding box (b) 1st level child boxes (c) 2nd level child boxes

Figure 3.15 Octree of point cloud bounding box

In a typical octree algorithm, the bounding box does not necessarily need to be a cube, but a general parallelepiped. However, the voxel grid based sub-sampling requires thst every voxel is cubic so that points can be evenly sampled in 3D space. The bounding box is appropriated and expanded to make it into a cube. The bounding box before octree sub-sampling is defined by $\boldsymbol{B_0}\{x_{min}, b_{xmax}, y_{min}, b_{ymax}, z_{min}, b_{zmax}\}$, in which $b_{xmax}=x_{min}+b_0$, $b_{ymax}=y_{min}+b_0$, $b_{zmax}=z_{min}+b_0$, $b_o=max\{ x_{max}-x_{min}, y_{max}-y_{min}, z_{max}-z_{min} \}$. The voxel size b_v is can be estimated by $b_v=b_o/N$, therefore for a given voxel size, the depth of octree can be calculated and an octree based voxel sub-sampling algorithm can be performed.

In order to illustrate this sub-sampling, a point cloud for an indoor scene before voxel sub-sampling is considered. The scene as scanned is shown in Figure 3.16. The points are not evenly distributed because of the distance from the scan position and object surface orientation. The result of octree sub-sampling for this point cloud is shown in Figure 3.17. The point distribution is homogenized because of this sub-sampling, as only one representative point is rendered for each voxel in the sub-sampled data.

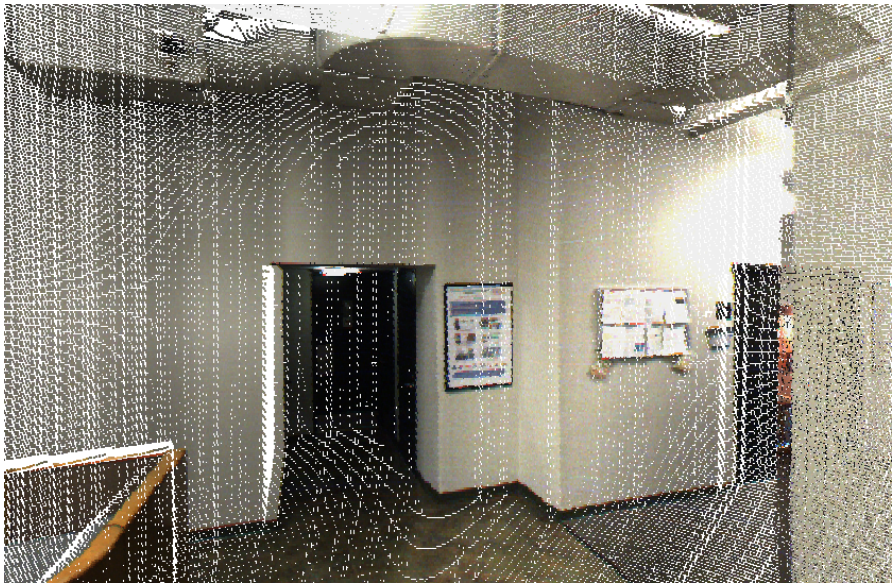


Figure 3.16 Color point cloud of an indoor scene at full resolution

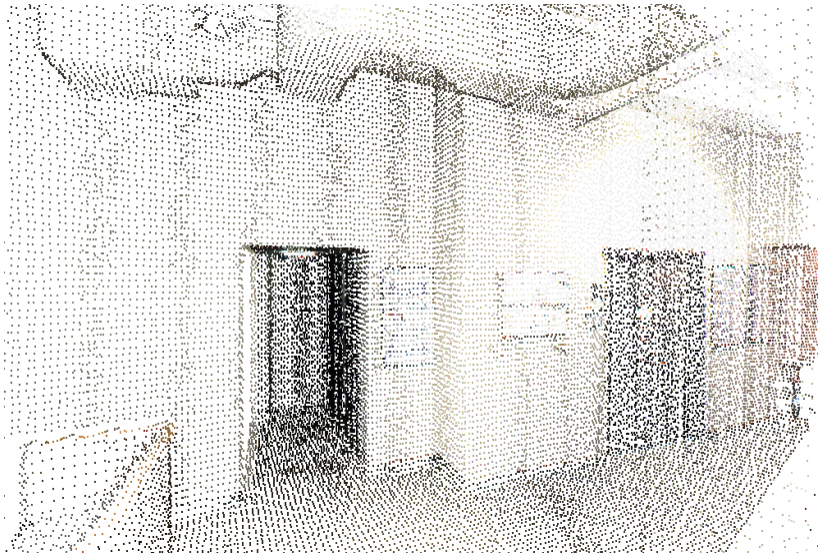


Figure 3.17 Color point cloud after voxel based sub-sampling. The point distribution is uniform after sub-sampling.

3.5 Normal Estimation for Point Clouds

There are several different algorithms such as Delaunay ball methods and Voronoi based methods [13] that can be applied to estimate point cloud surface normal. In this case, point surface normal vector on a single point has been computed by solving the nearest points fitted plane normal. A plane in 3D space can be defined by at least three arbitrary points. A point's surface normal could be defined as the surface normal of its belonging plane, which can be fitted by the point itself and its nearest *M* neighboring points. A plane ***P*** can be define fitting M+1 points in 3D space, the normal vector of the plane is defined as the surface normal of the point, as shown in Figure 3.18.

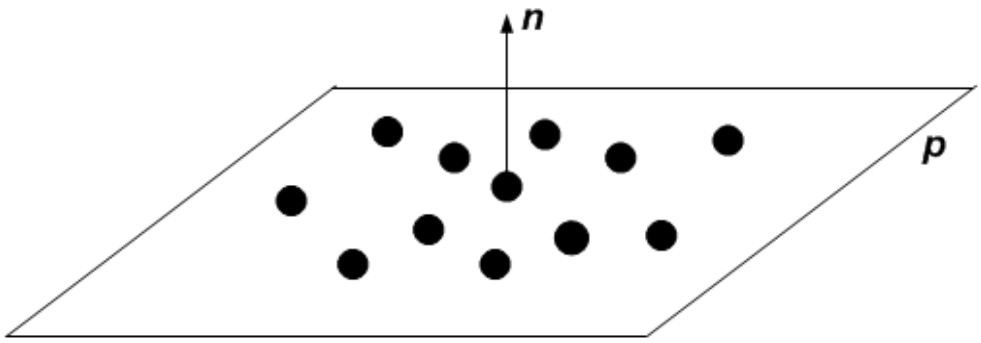


Figure 3.18 Point normal from nearest neighbor points. A bit fit plane is determined from nearest neighbors and plane normal is assigned to the point

3.6 Noise Filtering

The noise points occur while the laser beam from LIDAR hits the edge between different objects. These noise points are also named as outlier points. Due to the size and resolution of the laser

beam, noise points appear in the generated point cloud shown in Figure 3.19. The noise points are usually sparse between two different edges, and the spatial distance from itself to its nearest point neighbor is relatively large. Therefore, the noise filter algorithm is based on point spatial density. The nearest neighbor search in 3D space is implemented with a constraint of radius *r*. Apoint density higher than threshold within *r* defined sphere is recognized as regular object surface points. The noise filtered point cloud is shown in Figure 3.20. Outlier points between edges are successfully removed.

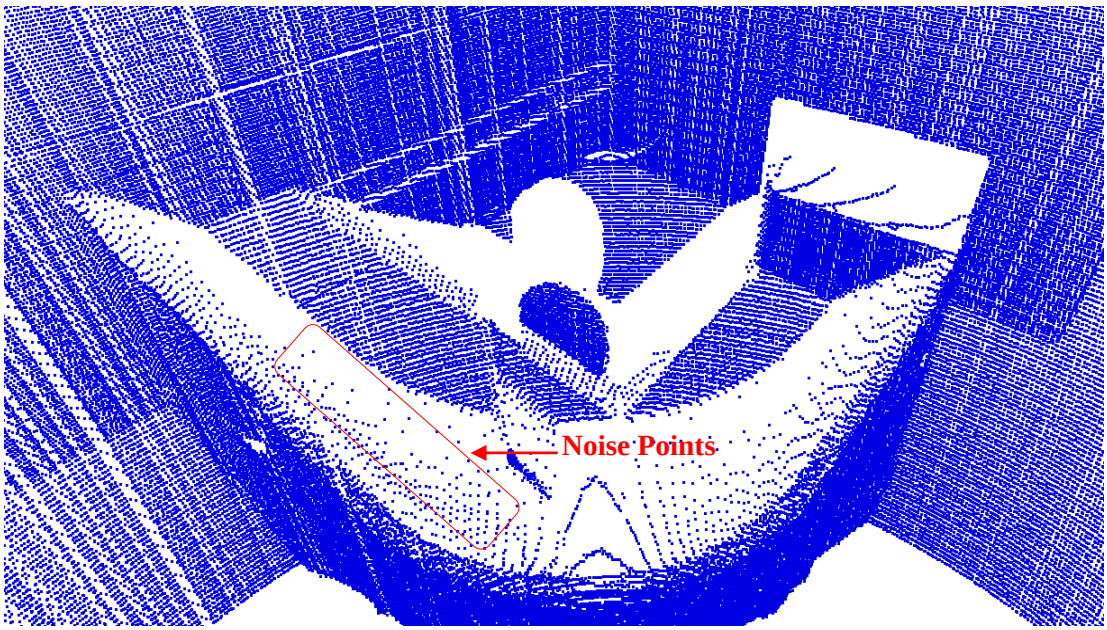


Figure 3.19 Point cloud with noise. Scanner produced considerable noise at corners and wall interfaces.

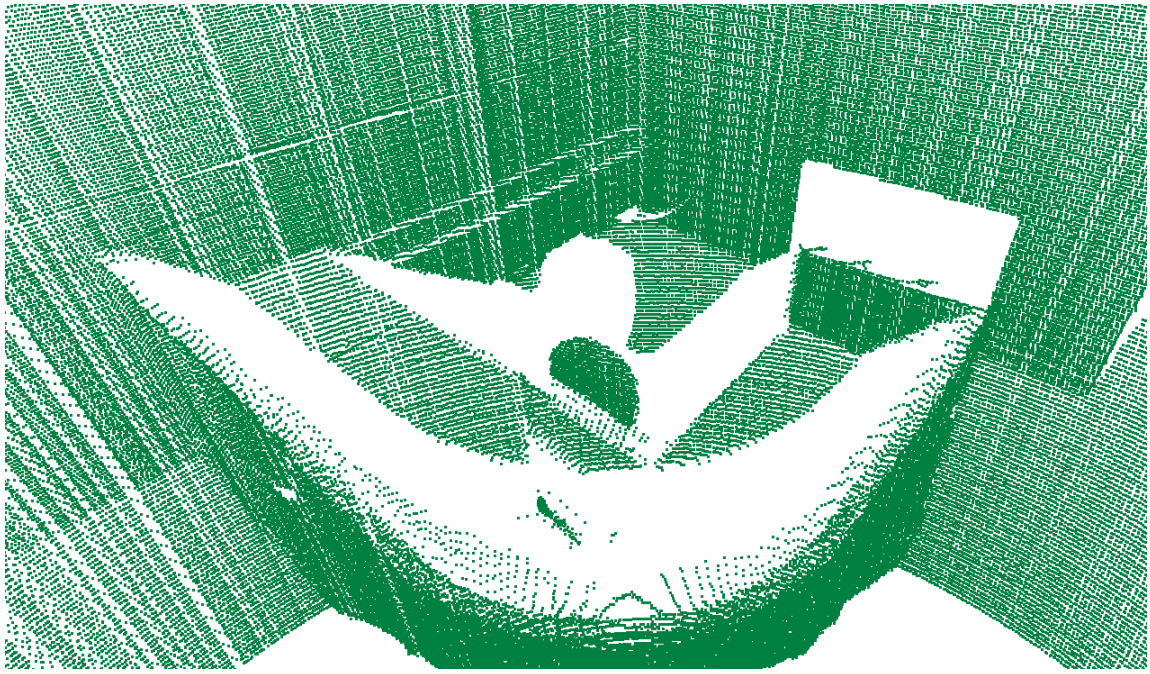


Figure 3.20 Point cloud after noise filtering

3.7 Grid Model and Cell Occupancy Analysis

Given a boundary of the area to be mapped, $\boldsymbol{B}\{x_{max},x_{min},y_{max},y_{min},z_{max},z_{min}\}$, we sub-divide the area into an exploration grid, ***Eg***. Figure 3.21 shows the terrain grid. The grid consists of a 2D cell array as shown in the figure and a 1D height histogram for each point in the cell. The resolution of the gird is defined by variable cell size $\boldsymbol{d_m}$ and can be related to both the size of the robot used for exploration and the general topographical features of the area. For an indoor scan with architectural features, a grid size of 100 cm (4 inches) may be appropriate as it is a typical thickness of a wall. For outdoors the cell size may be much larger. The exploration grid size has no correlation with the resolution of the architectural details that are being scanned, but with the navigation characteristics of the robot. In this analysis, the cell used is a parallelepiped with

anisotropic grid sizes. The grid size in the z-direction (elevation) is predefined with N_z levels. The exploration occupancy grid can be initialized in global space after destination area is given.

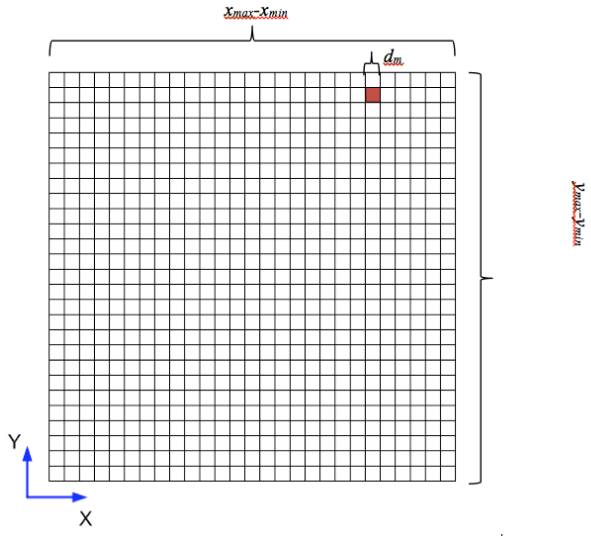


Figure 3.21 Two-dimensional Grid for identifying explored and unexplored areas

Once the exploration occupancy grid ***Eg*** is initialized, all the available point cloud segments are projected onto the grid. For each point cloud produced at one vantage, a voxel grid based sub-sampling is utilized to reduce the size and distribute points evenly in 3D space. Typically the point cloud size can been reduced by 90% using a voxel size of 50 cm. (from 737338 points to 70772 points). Figure 3.22 shows a point cloud taken in an interior space. The same point cloud is sub-sampled using the voxel grid and shown in Figure 3.23. The projection of the point cloud on the exploration grid ***Eg*** is shown in Figure 3.17.

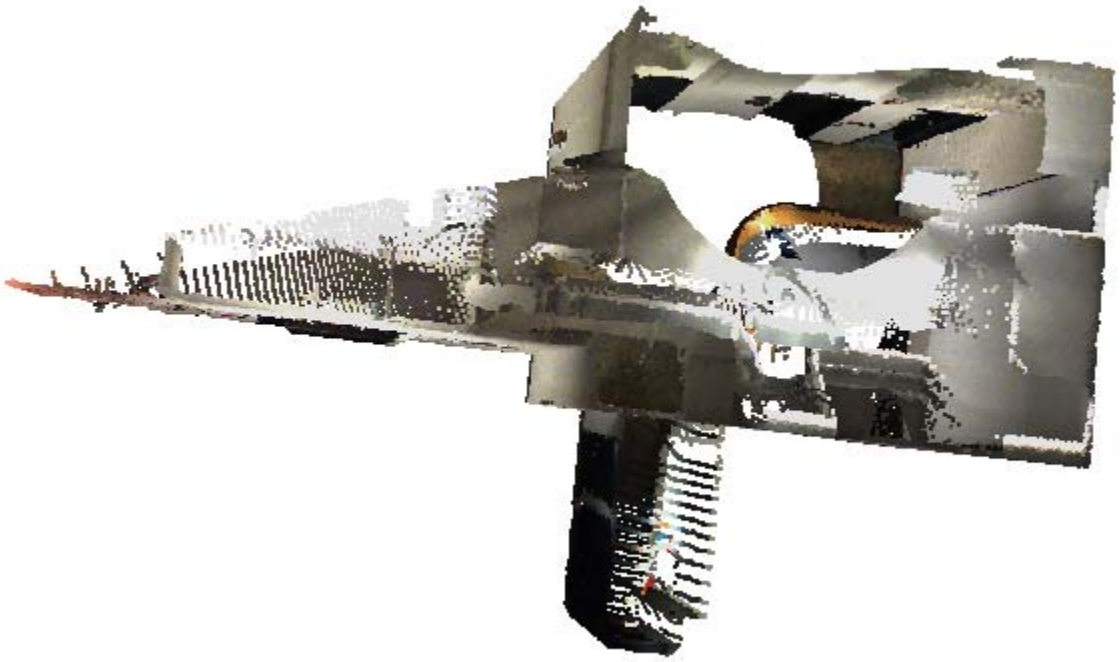
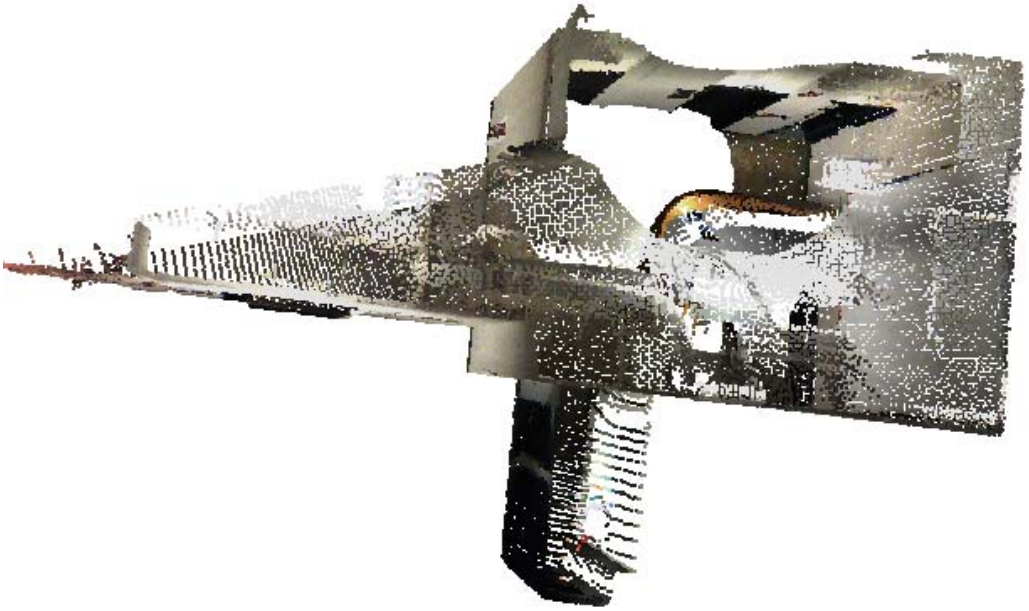
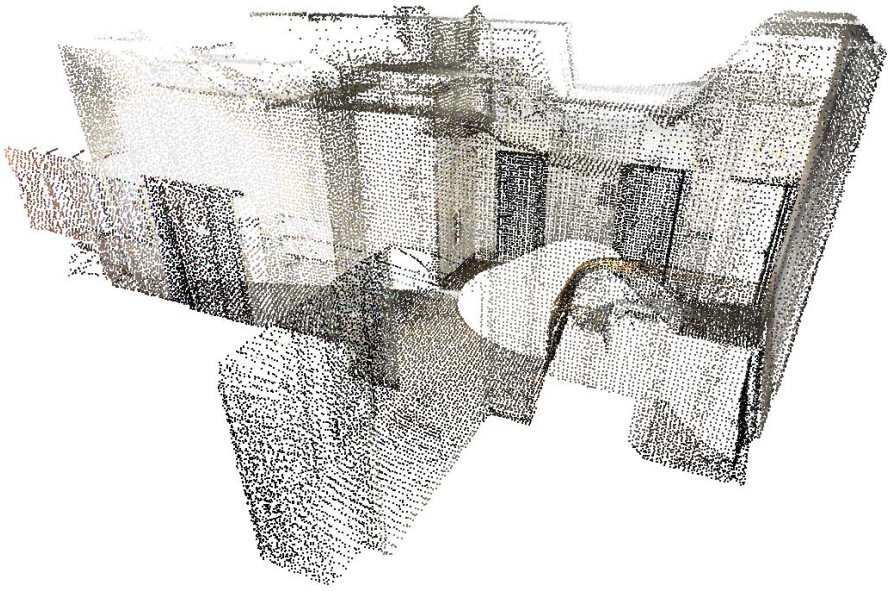


Figure 3.22 A typical point cloud generated for an interior space.



(a) Top view of voxel de-sampled color point cloud



(b) Side view of voxel de-samped color point cloud

Figure 3.23 Voxel grid de-sampled color point cloud at single vantage position

The exploration occupancy grid in Figure 5.17 is defined as $\boldsymbol{B}\{x_{max}=5.0,\ x_{min}=-30.0,\ y_{max}=10.0,\ y_{min}=-10.0\ ,\ z_{max}=3.0,\ z_{min}=-2.0\ \}$. The roof of the point cloud has been filtered out and the grid size on exploration occupancy grid is defined as 0.2m. The point cloud is then projected onto the exploration occupancy grid, as shown in Figure 3.24. The map shows areas where scan data is available (blue) and both unexplored and occulded areas. The map shows the blind spot from the scanner and occlusions due to shadows in the scan.

When the scan process generates more data in these areas, the scan segments are registered into the global refernce frame of the exploration grid and the grid occupany is updated.

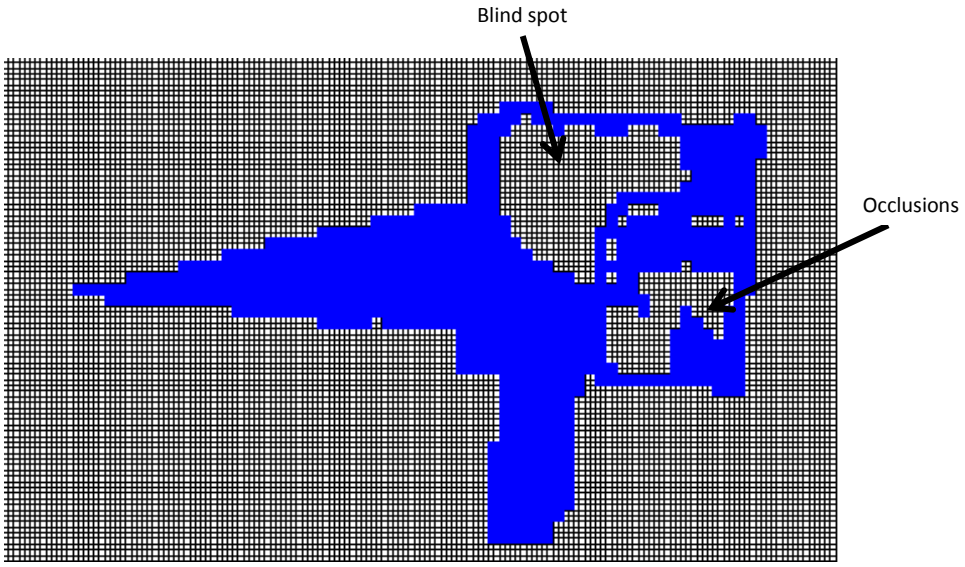


Figure 3.24 point cloud projected on exploration occupancy grid

3.8 Navigability Analysis for a Mapping Robot

The object surface point distribution at elevation direction on every grid is analyzed to identify objects. A typical point distribution on explorable area in this scene should include ground surface, walls, and other objects the robot can move across, and building roof, shown in Figure 3.25. This distribution of the z-heights in a cell is shown in the figure. The histogram in the figure indicates an equal probability of points at every height and indicates the presence of a wall in the cell. A pattern shown in Figure 3.26 corresponds to an area such as a hallway in which there is a floor and ceiling. The z-height histogram shows clean ground at the base of the robot (-1.5 m) and the ceiling at 1.75 m in the cell. Most common z-distributions are as shown in Figure 3.27. This distribution indicates the presence of obstacles on the ground and lower than expected ceiling. For most indoor scenes the ceiling expectation is at a minimum of 2.4 m (8 ft). If the physical dimensions, kinematic and dynamic conditions are satisfied, the geometry of the cell is conducive for robot navigaiton. These cells must be checked against robot capabilities to establish whether the robot can traverse the cell or if there is an obstracle in the cell.

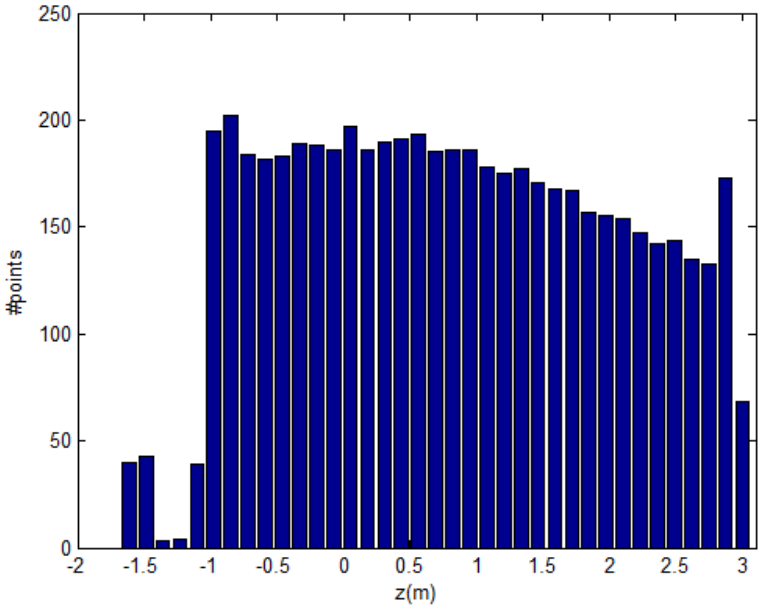


Figure 3.25 Z-Occupancy for a grid cell with wall

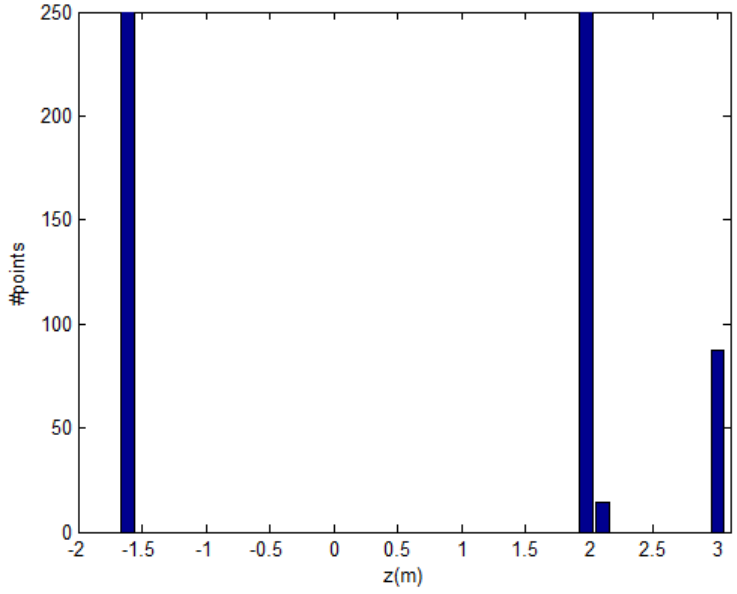


Figure 3.26 Z-Occupany for a grid cell with a floor and ceiling

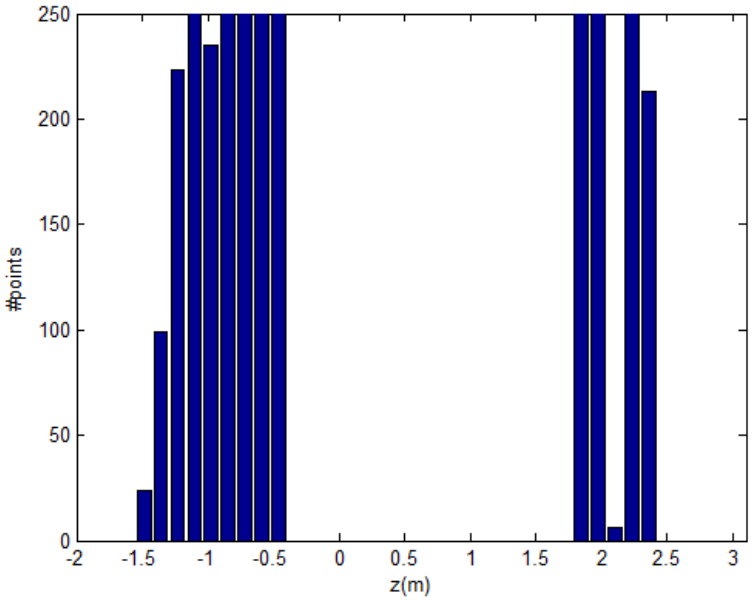


Figure 3.27 Point distribution pattern showing ground, roof and objects on the ground

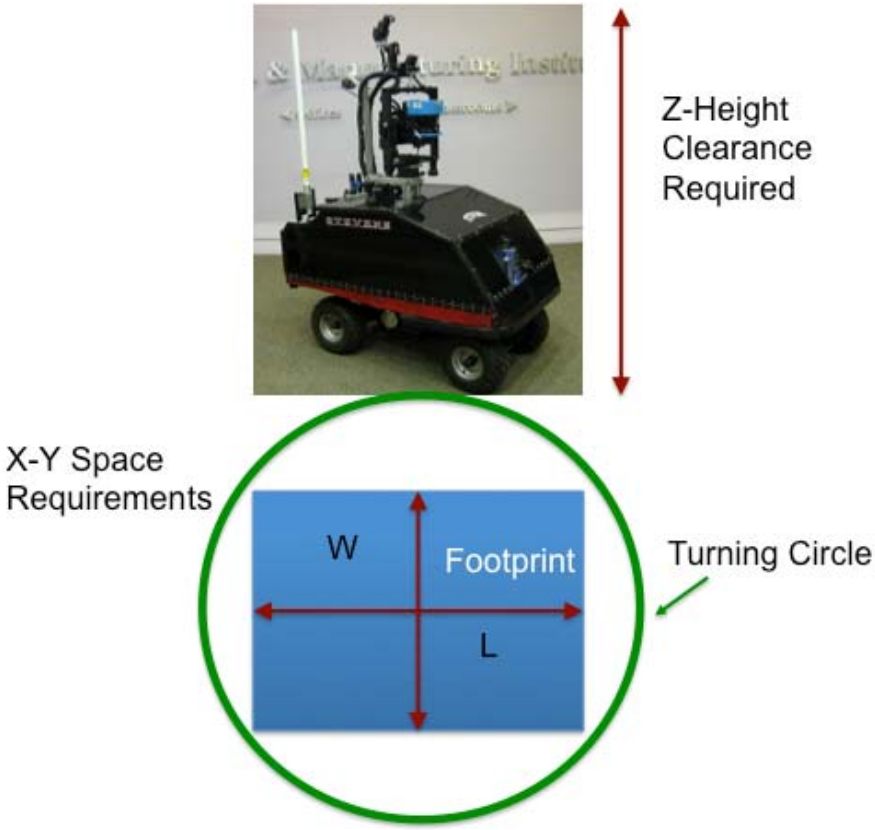


Figure 3.28 Space constraints of the mapping robot. Footprint and turning circle area determines the obstruction free cell size required for robot to traverse an area. The Z-height shows the overhead clearance required for traversal.

The z-distributions must be checked against the robot capabilities if the cell does not have a clean floor and has obstacles either on the floor or hanging from the ceiling. The mapping robot used for this analysis is shown in Figure 3.28. The robot requires a z-height clearance of 1.5 m and has known footprint dimensions and turning circle. The terrain in the exploration grid cell as determined by the z-height distribution is evaluated for navigability of the robot through the cell. The robot can be contained in the cell if the height distribution has a void near the center point (z=0) corresponding to 1.5 m. We further analze the characteristics of the terrain in the cell. The height distributions are classified into ground and ceiling features. For the ground features, the

maximum height and depth are determined. Also, cut plane are taken along the width and length axis to detect the presence of obstructions in the cell. When all the criteria are met and the cell is obstruction-free, the cell is marked as navigable.

An example exploration grid with navigability information is shown in Figure 3.29. The red squares indicate the cells with walls and the green areas are navigable areas. The yellow regions are where no infomrmation is present. Analysis of continguity of the green areas produces navigable regions and separates them from the walls.

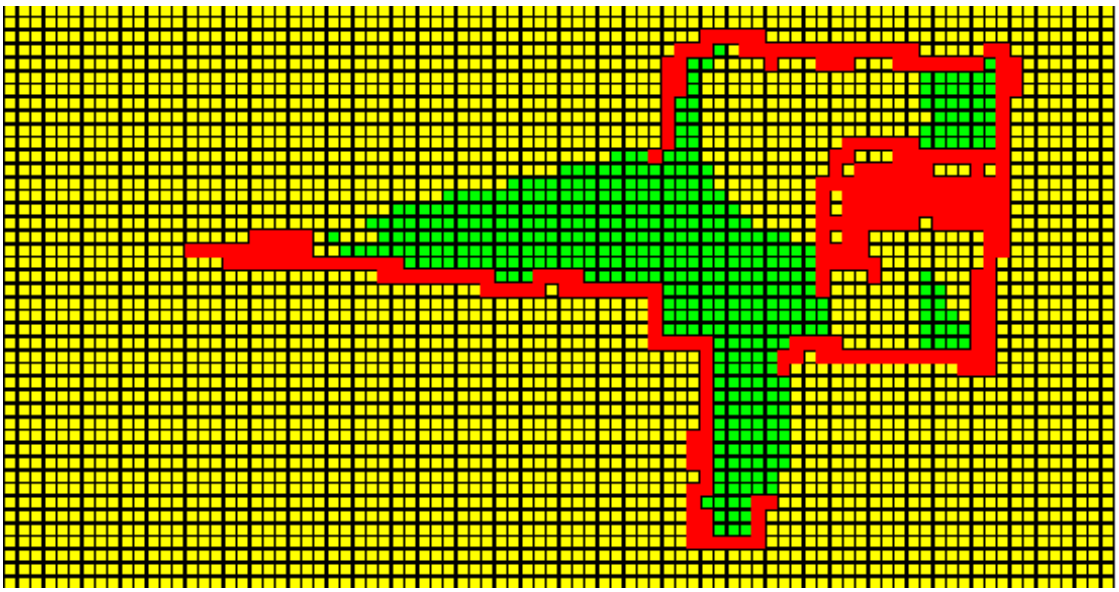


Figure 3.29 Navigability analysis on the exploration grid.

3.9 Concluding Remarks

In this chapter several methods for the analysis of a single point cloud are presented. A concept of exploration occupancy grid map has been introduced for exploration and path planning. The size of the exploration occupancy grid is initialized when the exploration and mapping task is

given. The scene generated at a single vantage position is analyzed based on elevation distribution. Different distribution patterns have been analyzed to identify the ground surface for exploration and movement and object surfaces. The output from point cloud processing results will be utilized to drive the mapping mission and path planning for the next vantage position.

Chapter 4 Coarse Registration of Color Point Cloud Segments

This chapter introduces a fully automatic, and comparatively fast, point cloud registration algorithm based on point cloud surface normal distribution and grid occupancy. Acceleration by using hue information obtained from 3D point clouds with color attributes is a major contribution in this chapter. Map registration without the need for any position information is a significant development for effective mobile mapping indoors. The algorithm enables automatic registration of overlapping point clouds without the need for any scan location information. The methodology can also detect any overlap in point clouds based on the hue distributions. The registration process cross-correlates the distribution of normal in two point clouds in Fourier domain. Rigid transformation solved using this algorithm can be utilized for rough alignment and in real-time on site. Map coverage assessments and, as an initial translation, estimates can be a good starting point for the fine registration process. Adding hue constrains to this registration process accelerates the performance for point cloud automatic registration. Experimental results from hue constrained registration demonstrate faster registration speeds and better registration accuracies

4.1 State of Art

Merging a pair or multiple point cloud segments from ranging device at different positions is important for constructing a 3D global map in a large-scale area [48, 55, 75]. The point cloud is registered into global space by performing a rigid transformation from local registration. The parameters of rigid transformation in 6 degree of freedom (6DOF) can also be applied for

simultaneous localization in 3D space [27, 52, 60], which is helpful on unmanned vehicle movement control. Localization and mapping capabilities help an unmanned robot efficiently explore unknown dangerous or hostile environments. Moreover, these capabilities can be applied in surveying industry and reverse engineering in manufacturing to improve the efficiency of measurement quality and enhance productivity.

Position and orientation information which is required for registration can be provided directly by mobile platform sensors such as GPS and IMU [58]. In most cases, position information acquired from the sensor is reasonably accurate. However the orientation attribute information is expensive and relatively imprecise because orientation sensor measurements can be effected by external disturbances like magnetic field variations and sensor integration drift with time. Position and orientation information can also be provided by indirect techniques based on both rough position sensor measurement and common geometric feature identification.

Compared to SLAM algorithms, map registration techniques focus on generating accurate map details rather than the location of the robot in a global coordinate system [60, 65, 76]. Discrete range points received from the mapping sensor contain detailed spatial information about the environment. Different techniques exist for merging such point clouds together by exploiting geometric features and measuring surfaces. Map registration techniques such as the Iterative Closest Point (ICP) [51] algorithm have been applied to stitch two neighbor 3D point cloud maps together into one map based on their common coverage area. Upon convergence, the ICP algorithm terminates at a minimum [51]. Several algorithms are in existence for calculating the minimum average distance between two point clouds. The Singular Value Decomposition (SVD) method [49], eigen-system methods that exploit the orthonormal properties of the rotation matrices, and unit and dual quaternion techniques were adopted in the ICP process [77]. Quaternion based algorithms have been used in ICP for map fusion in [51], SVD based

algorithms are widely used in ICP and 6DOF SLAM [49, 52] as they are robust [78] to reach local minimum and are easy to implement. Several variants of ICP are reported in the literature to increase speed and precision[53]. Corresponding points sampling, matching, weighting, and rejecting are some methods used to accelerate the ICP algorithm. In the ICP algorithm, associating corresponding points in two point cloud data sets is the most critical step. Nearest neighbor search in 2D or 3D space is commonly used for associating the corresponding points. Parallel ICP algorithms have been developed [54] to accelerate computation speed. The point to plane registration method [53] accelerates the ICP iteration and convergence.

Other techniques include the point signature method [79], which uses signature points to describe the curvature of a point cloud and matches corresponding signature points during the registration process. Spin image based methods compute 2D spin image to represent surface characterization and solve the registration problem by finding the best correspondence between two different scan spin images[80]. Other methods like principle component analysis[81] and algebraic surface model [82] are based on the point cloud surface geometrical features. The normal vector distribution can be translated into an orientation histogram in an Extended Gaussian Image (EGI). Rigid motion required to register two point clouds is solved from the cross covariance function of the two EGI images. Rigid motion could also be solved in the Fourier domain by computing Discrete Fourier Transform on Rotation Group on SO(3) (SOFT) .

4.2 Automatic Registration of Point Cloud Segments

The EGI registration algorithm can be applied for automatic 3D point cloud registration because it does not require any pre-alignment. EGI registration also can be combined with the ICP algorithm to accomplish the whole point cloud map registration process. Range point clouds

generated from a ROAMS mapping robot in Chapter 2 can be applied to take advantage of this algorithm to construct global map without any known position information.

The Extended Gaussian Image based registration algorithm provides a non-iterative solution for point cloud registration. This technique does not require any pre-alignment, however the accuracy after registration is not as good as ICP related techniques. It can be applied for fully automatic 3D map registration as pre-alignment for ICP fine registration. The EGI algorithm utilized point cloud normal distribution to construct surface normal histogram on an extended Gaussian sphere. The alignment between different point clouds is completed by maximizing the correlation function between their extended Gaussian image surface normal histograms.

An Extended Gaussian Image of a given point cloud can be constructed by computing surface point normal and project histogram on Gaussian sphere. Surface normal with each point can be extracted from a meshed or fitted plane. The Extended Gaussian Image normal histogram can be constructed by following algorithm.

Surface point normal histogram

Point cloud surface normal has been computed as the first step. Normal vector $\vec{n}\ \{n_x, n_y, n_z\}$ can be represented by its orientation angle $\{\theta,\ \varphi\}$ in Figure 4.1.

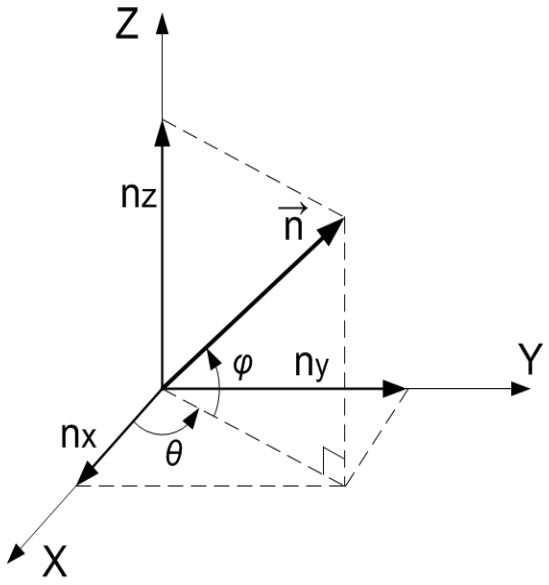


Figure 4.1 Surface normal orientation angle and vector

Surface normal orientation histogram can then be constructed after normal computation of every point. The orientation histogram ***H( )*** on plane is shown in Figure 4.2, where $\in SO(3)$. Extended Gaussian Image is a unit sphere that has been developed to represent surface normal vector information, the orientation histogram can be projected on EGI and shown as Figure 4.2.

EGI Construction

1. Initialize bandwidth for Gaussian Image, set longitude and latitude bandwidth as bw_1 & bw_2, a number of M grids are created on Gaussian sphere;

2. ***for*** i=1:*N*

for j=1:*M*

Compare $\boldsymbol{n}_i$ with j^{th} grid

if $\boldsymbol{n}_i \in j^{th}$ grid

h_j++;

end if

end for

end for

In which, N is the number of points in point cloud map, $\boldsymbol{n}_i$ is the surface normal on i^{th} point. h_j represents the total number of points normal fall in j^{th} grid.

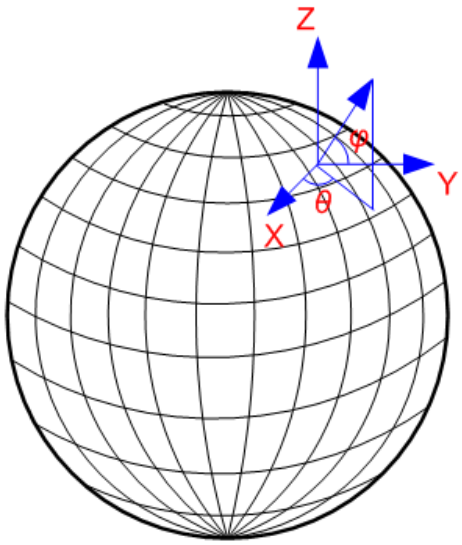


Figure 4.2 Surface point normal histogram on EGI

4.2.1 Rotation Estimation

As soon as two or multiple point clouds surface histograms have been generated in the form of Extended Gaussian Image, the global likelihood function about rotational alignment between EGI can be computed as:

$$G(R) = \int_{\omega \in S^2} H_1(\omega) H_2(R^T \omega) d\omega \tag{4.1}$$

In which, rotation matrix $R \in SO(3)$. $H_1(\omega)$ and $H_2(\omega)$ are the orientation histogram generated from two separate point clouds. Direct computation of $G(R)$ requires a complexity of $O(M^2)$, where M is the size of the histogram. The total computation complexity is on the order of $O(K^3M^2)$, where K is the number of samples in each dimension of $SO(3)$.

To reduce the complexity to solve rotation, the correlation integral can be expressed as a simple pointwise multiplication in the Fourier spectrum. The Fast Fourier Transform helps to reduce complexity dramatically.

Spherical correlation

The spherical harmonics (Y^l_m: S^2->C) from an eigenspace of harmonic homogeneous polynomials of dimensional $2l+1$. The $2l+1$ spherical harmonics for each $l \geq 0$ generate orthonormal basis for $f(\omega) \in L^2(S^2)$, where L^2 denotes square-integrability.

$$f(\omega) = \sum_{l \in M} \sum_{m=-l}^{l} \hat{f}^l_m Y^l_m(\omega) \tag{4.2}$$

And

$$\hat{f}^l_m = \int_{\omega \in S^2} f(\omega) \overline{Y^l_m(\omega)} d\omega \tag{4.3}$$

In which, $\hat{f}_m^l$ is the coefficient of Spherical Fourier Transform (SFT). The complexity to compute STF is $O(L^2 log^2 L)$, L is the bandwidth of spherical signal, in this case, bw_1 and bw_2 in EGI construction are equal to L.

Rotation $R \in SO(3)$ permits Fourier Transform because it has a basis of irreducible unitary representations as a compact Lie group. R can be represented as $R_z R_Y R_z$ ZYZ rotation group with corresponding Euler angle α, β and γ as the parameterization of $R \in SO(3)$.

$$f(R) = \sum_{l \in M} \sum_{m=-l}^{l} \sum_{p=-l}^{l} \hat{f}_{mp}^l U_{mp}^l(R) \tag{4.4}$$

Where,

$$\hat{f}_{mp}^l = \int_{R \in SO(3)} f(R) \overline{U_{mp}^l(R)} dR \tag{4.5}$$

$$U_{mp}^l(R(\alpha, \beta, \gamma)) = e^{-im\gamma} P_{mn}^l(\cos(\beta)) e^{-in\alpha} \tag{4.6}$$

In which, P^l_{mn} is the generalized associated Legendre polynomials. The $\hat{f}_{mp}^l$ is the coefficient of SO(3) Fourier Transform (SOFT), a fast discrete SOFT can be computed with complexity of $O(L^3 log^2 L)$.

For example, two point clouds scanned from different vantage points of a conference room are shown, in their local coordinate system, in Figure 4.3. The data point cloud is marked as blue and model is marked as black. The point normal orientation histograms on plane have been computed and shown in Figure 4.4. Orientation histograms projected on EGI are shown in Figure 4.5. After spherical correlation, the data point cloud has been rotated and aligned with the model point cloud, shown in Figure 4.6.

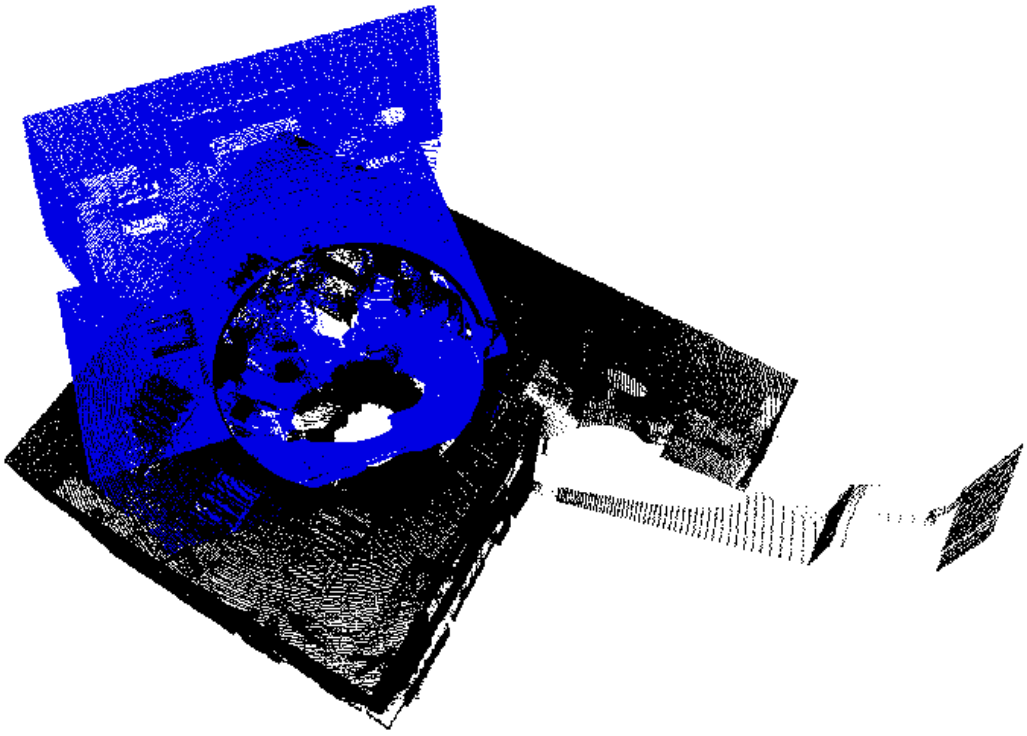
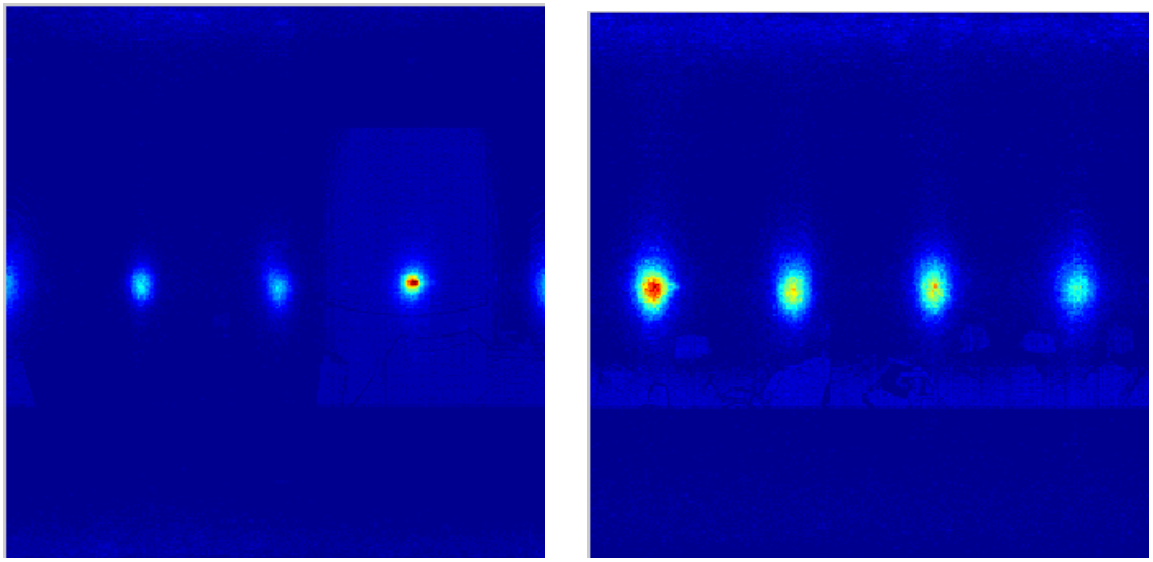
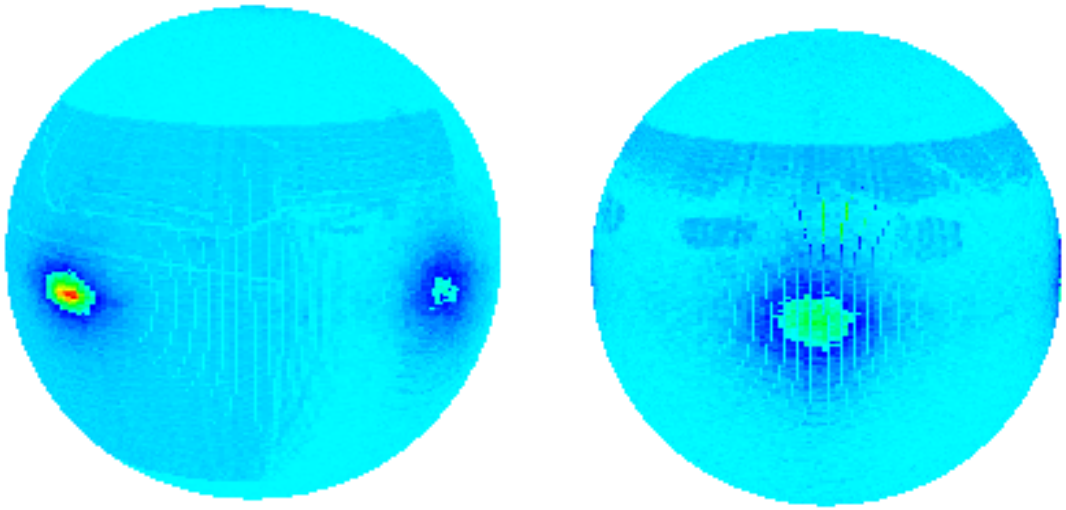


Figure 4.3 Model and data point clouds taken from two separate vantage points of a conference room shown in their initial local coordinate system



(a) Data histogram (b) Model histogram

Figure 4.4 Surface normal orientation histogram on θ φ plane



(a) Data EGI (b) Model EGI

Figure 4.5 Orientation histogram projected on EGI

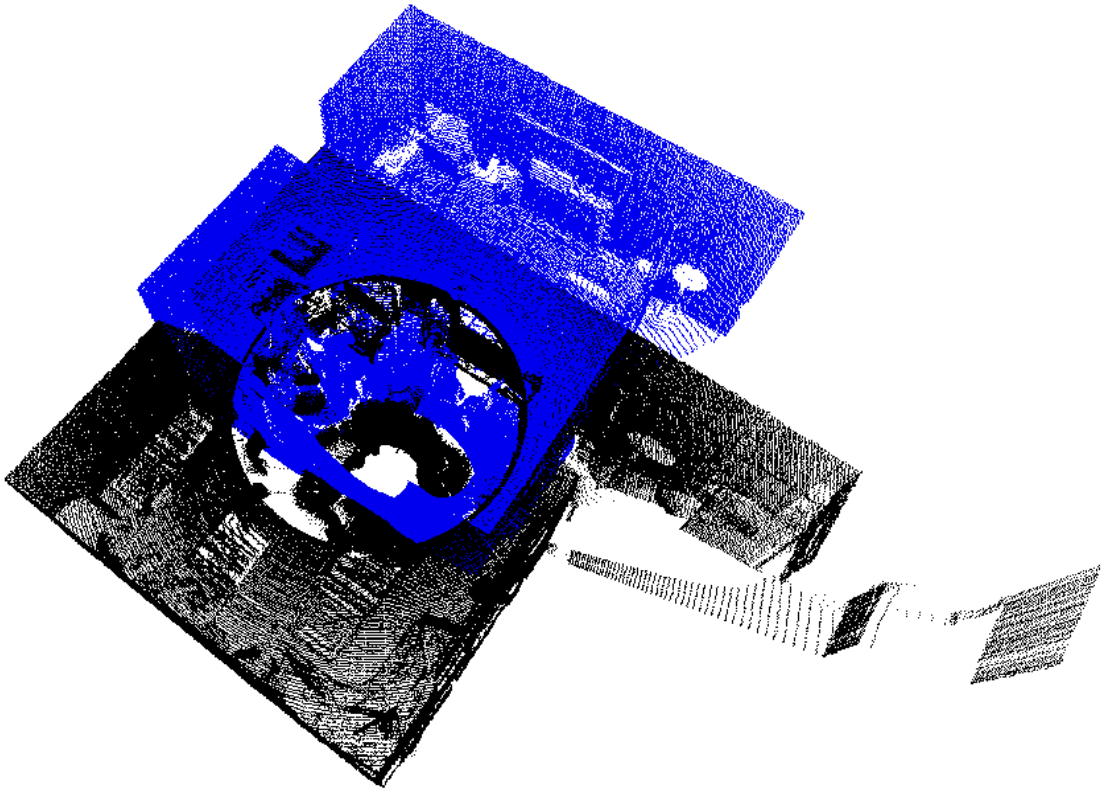


Figure 4.6. Point clouds after rotational alignment

Correlation in the frequency domain

The translation matrix is computed by solving the correlation of occupancy grid between data and model point clouds after rotational alignment [15]. The bounding box of the occupancy grid is the common bounding box between two point clouds. The occupancy function is built by

$$Og(x,y,z)=\begin{cases}1, & if \quad point\,s\ exist\\ 0, & otherwise\end{cases}$$

The 3D occupancy grid of data and model point clouds are shown as Figure 4.7. They can be stretched into 1D plane and shown as occupancy function $Og(\tau)$ in Figure 4.8. The correlation between data and model occupancy grids can be solved by

$$G(T) = \int_{\tau \in \Re^3} Og_m(\tau) Og_d(\tau + T) \tag{4.7}$$

The best transformation vector ***T*** for correlation can be solved by computing the convolution between $Og_m(\tau)$ and $Og_d(\tau)$ as:

$$T = \max(real\{fft^{-1}(fft[Og_d(t)] \cdot fft[Og_m(t)]\}) \tag{4.8}$$

The SO(3) Fourier coefficients of the correlation of two spherical function can be obtained directly from the point-wise multiplication of the individual SFT coefficients, with total computation complexity $O(L^3 log^2 L)$.

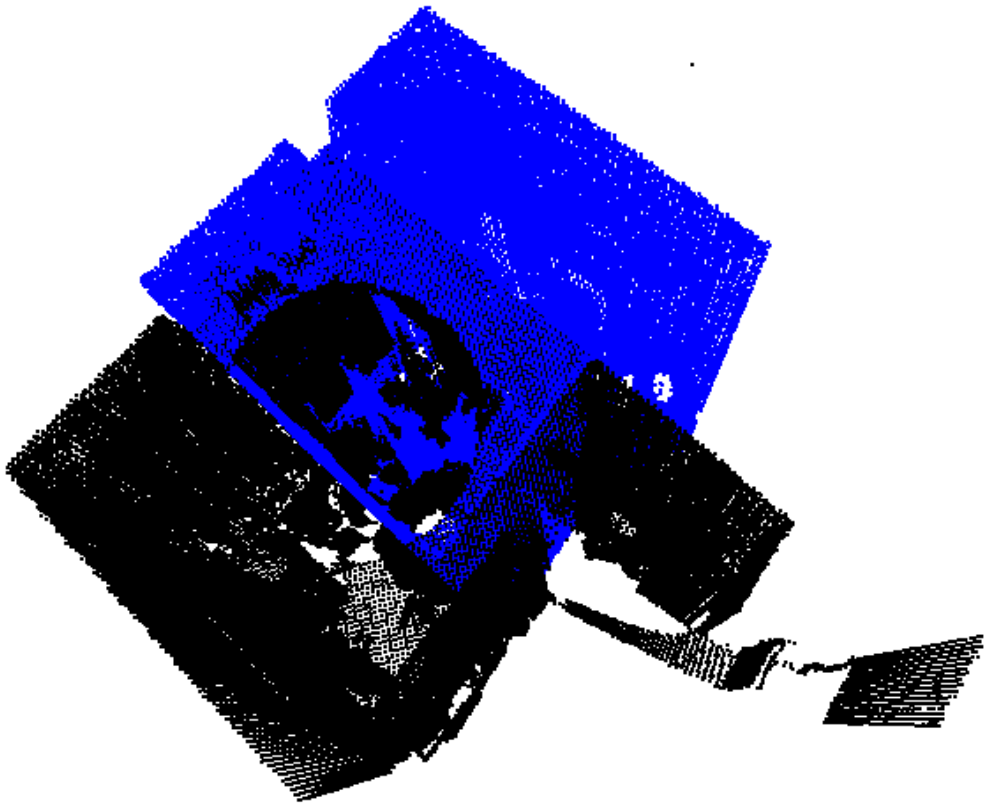


Figure 4.7 3D Occupancy grids generated from point clouds

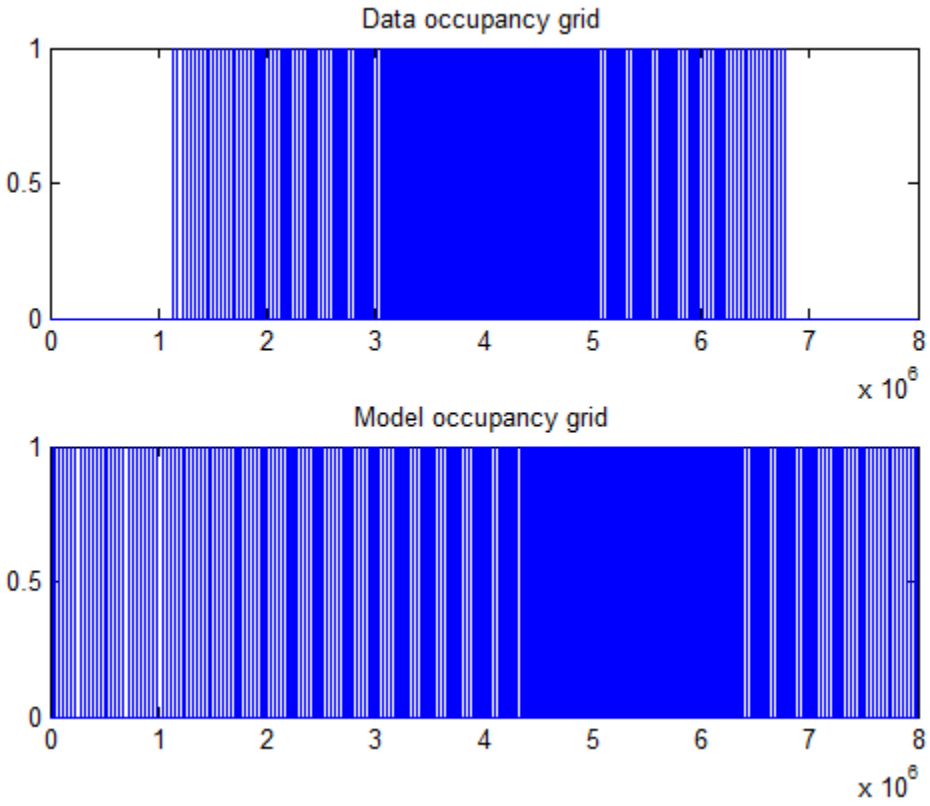


Figure 4.8 Occupancy grid shown as binary image.

The correlation results between $Og_m(\tau)$ and $Og_d(\tau)$ are shown as Figure 4.8. The data point cloud can then be translated with solved translation $\boldsymbol{T}$ into the model point cloud space, shown in Figure 4.9.

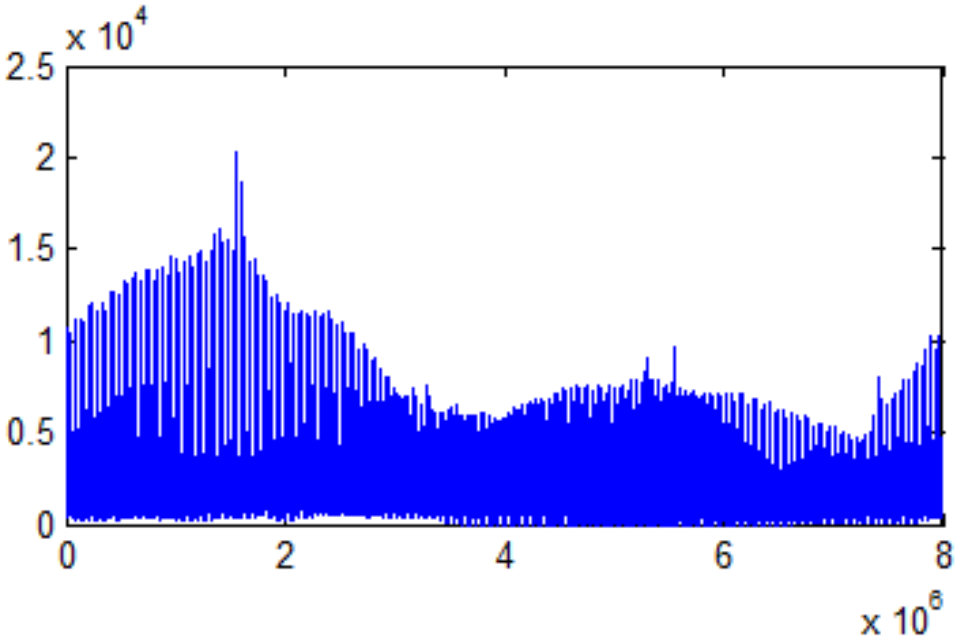


Figure 4.9 Cross corelation results of 1D occupancy grid

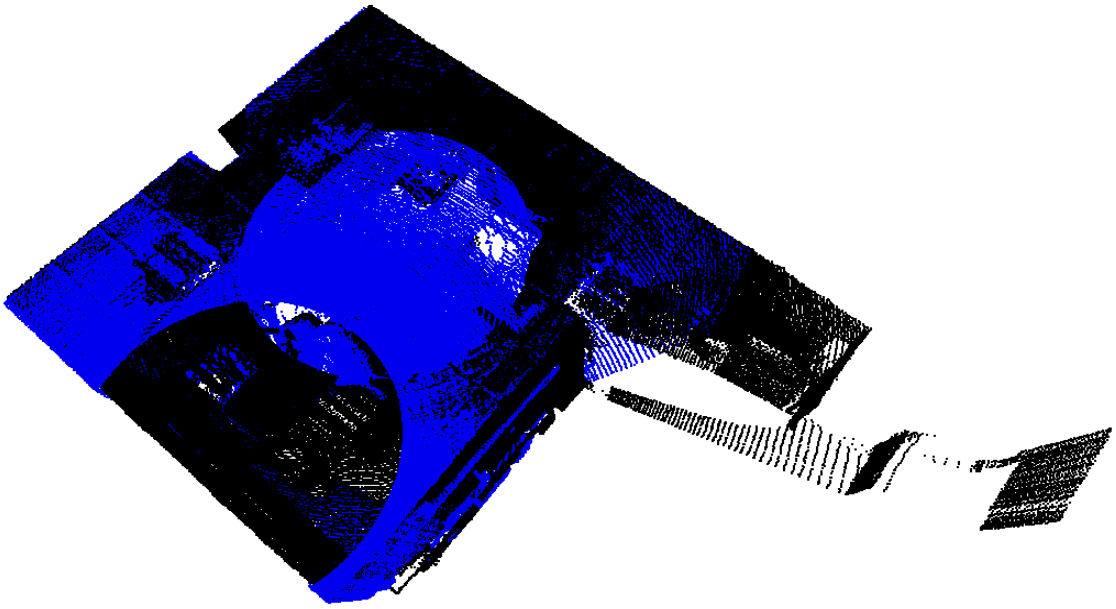


Figure 4.10 Data point cloud rotated and translated into model space – complete registration.

4.3 Hue Assisted Color Point Cloud Coarse Registration

Hue Distribution Analysis

Registration speed and accuracy can be improved by taking the advantages of color attributes from the color point cloud. Color attributes on each point represent the visual characteristics of the scene. In most cases, lighting conditions and camera orientation determine the Red Green Blue value distribution in RGB color model. Hue value from the Hue-Saturation-Lightness (HSL) model has been applied to diminish the effect of lighting conditions. In the conference room point cloud examples, the color point clouds from initial local vantage positions are shown in Figure 4.11. Color properties especially provide detail information about surface texture of the scene. In order to effectively apply the hue filter to assist the automatic registration algorithm, hue distribution of data and model point clouds is computed and shown in Figure 4.12.

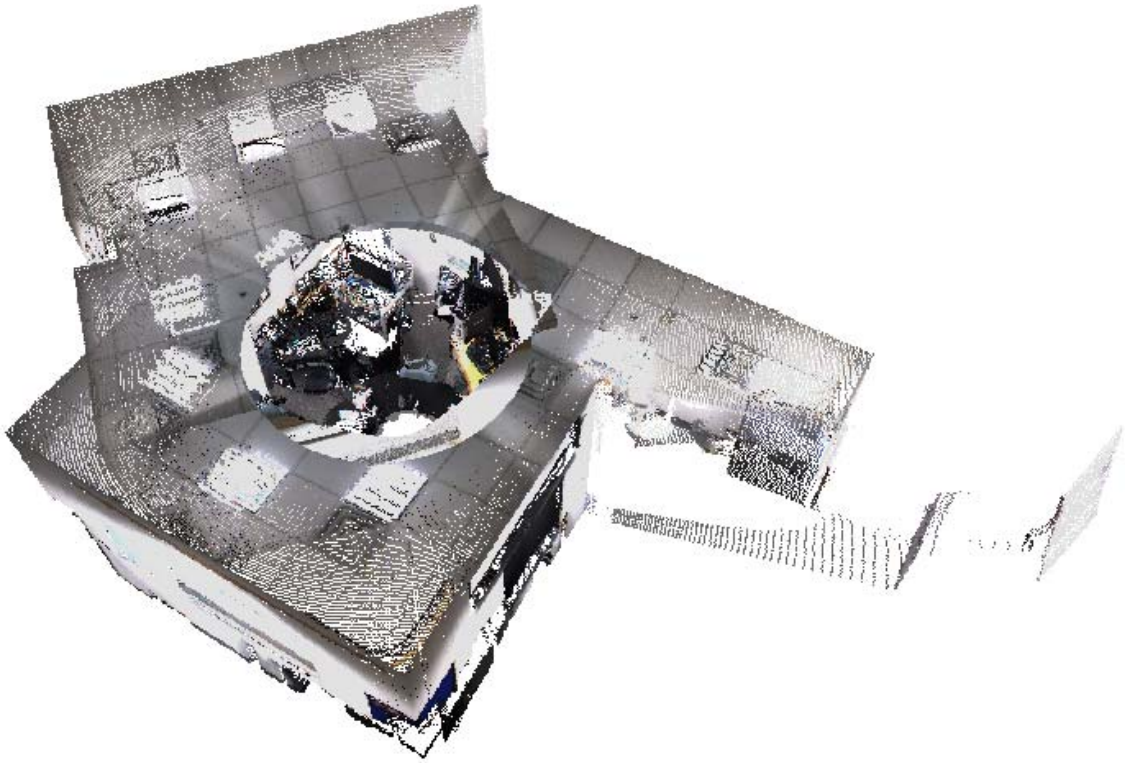


Figure 4.11. Color point clouds from 2 different vantage positions.

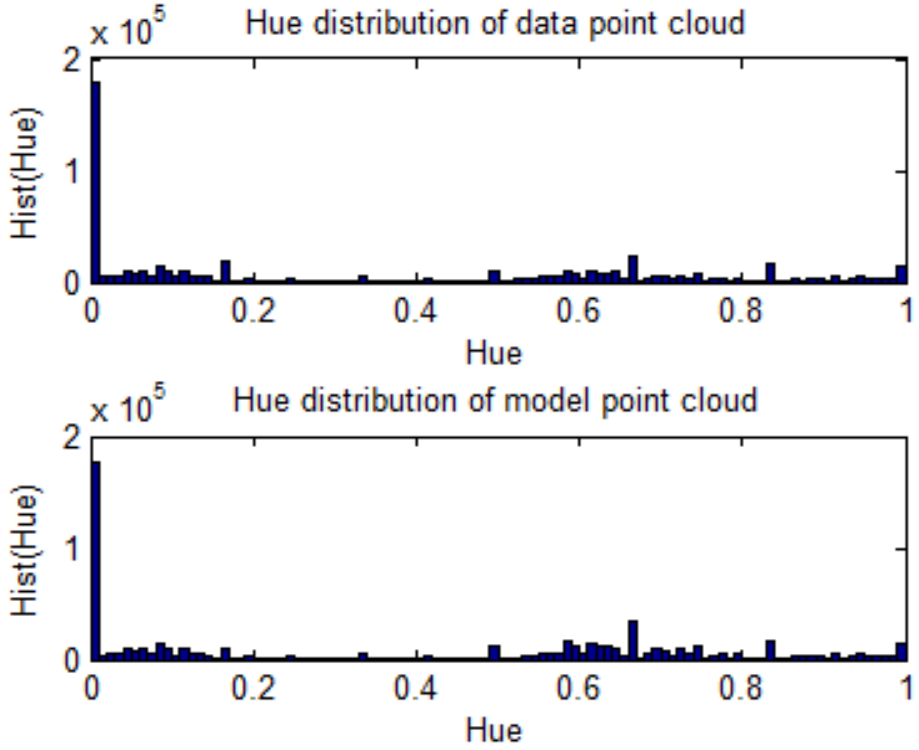


Figure 4.12 Hue distribution of 2 color point clouds.

Eight color point clouds taken at different vantage positions in a building hallway are shown in Figure 4.13. Different lignting conditions have their own effect on the rendered point clouds. The hue attribute on every point has been computed and analyzed, shown in Figure 4.14. From the hue distribution historgram in Figure 4.14, it is obvious that most points in the color point clouds have a hue value lower than 0.5. Therefore, a lower pass hue filter could be utilized to filter point clouds to improve the performance of automatic registration process.

Figure 4.13 Eight Color Point clouds generated for the hallway

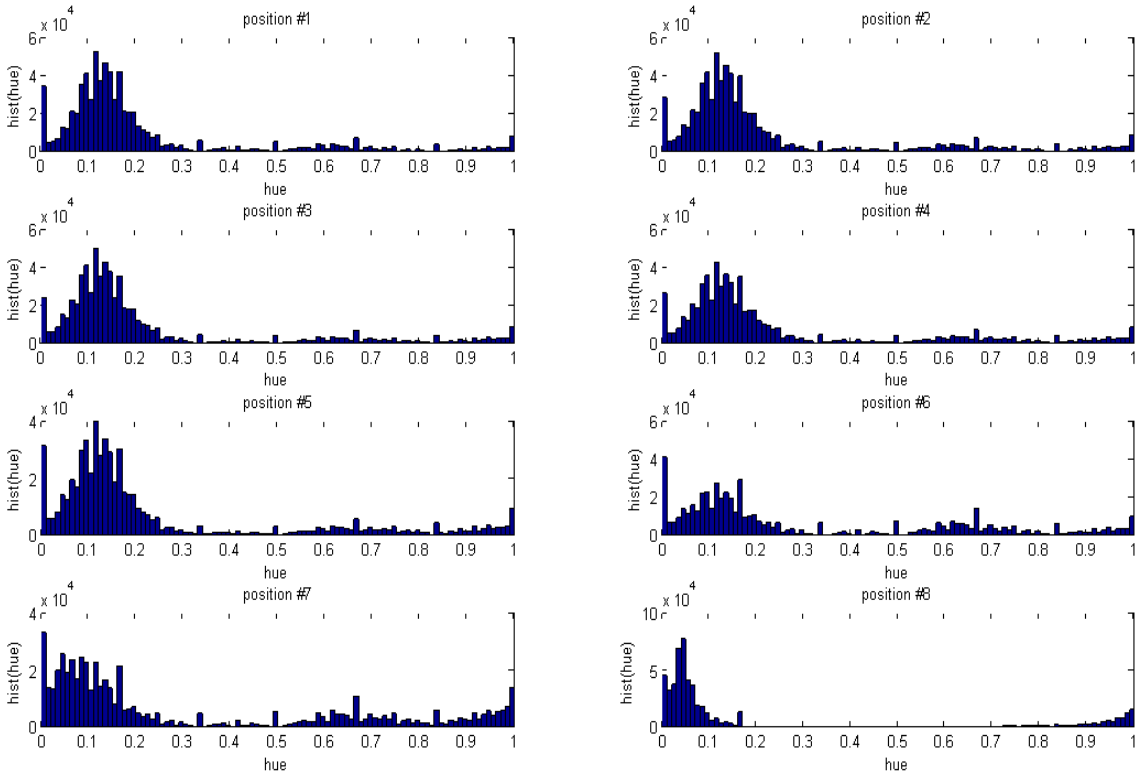
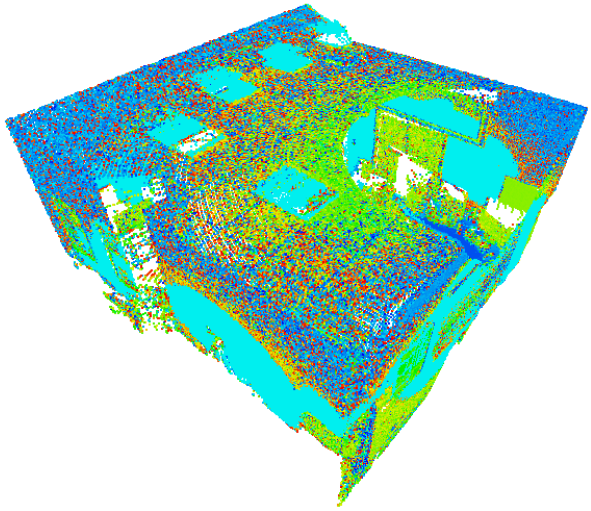


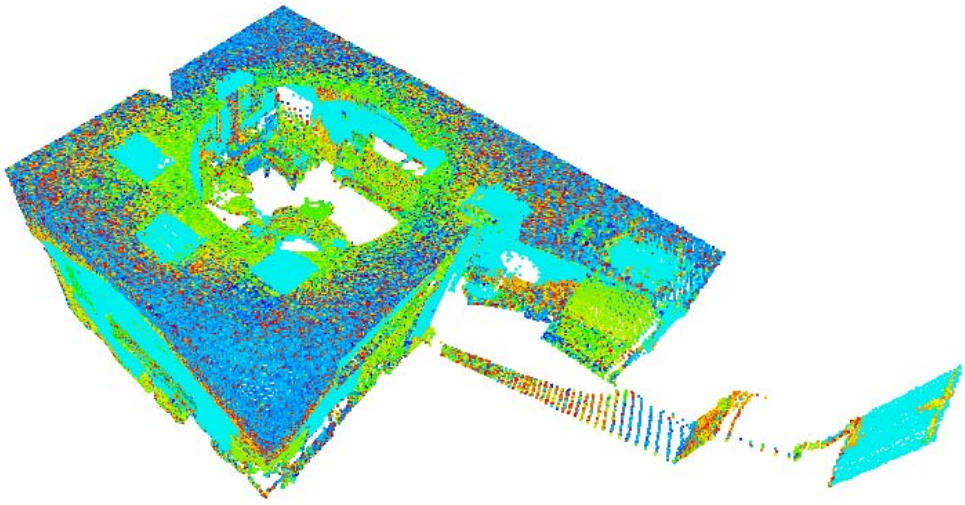
Figure 4.14 Hue distriution histogram for the eight color point clouds in shown Figure 4.18

Most points have a low hue value from the distribution analysis. Point cloud color was rendered by its hue value and both point clouds were re-plotted in Figure 4.15. The hue property on same object surfaces between different point clouds after hue filtering remains stable. Therefore a low pass hue filter (h_f<=0.5) is applied, filtered point clouds of the conference room are shown in Figure 4.16. The number of points in data point cloud has been reduced from 597270 to 353386, and the number on model point cloud has been reduced from 631400 to 3308321. The surface geometry of environment still remains so that the automatic registration algorithm can be utilized after fitering.The orientation histogram after hue filtering is shown in Figure 4.17, projected EGI is shown in Figure 4.18. Therefore a rotational and translational alignment after computing the

spherical correlation between orientation histogram and occupancy function is shown in Figure 4.19.

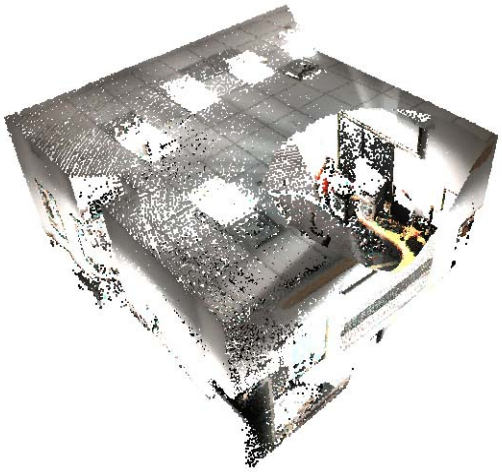


(a) Data point cloud colored by the hue value

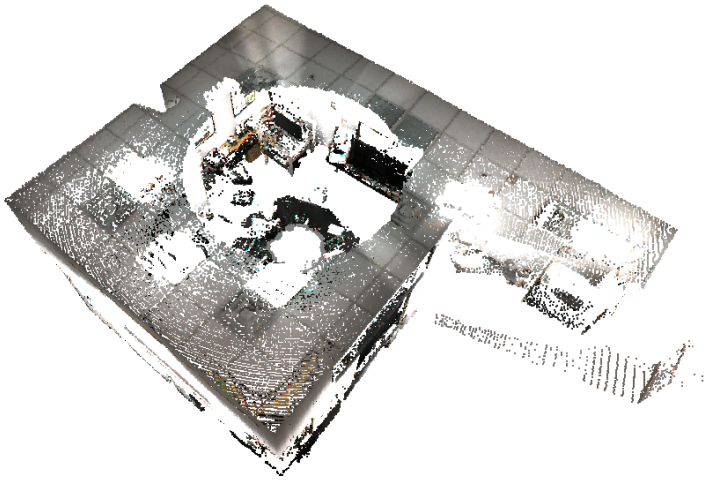


(b) Model point cloud colored with hue value

Figure 4.15 Hue rendered point clouds at 2 different vantage positions



(a) Data Point cloud after the hue filter



(b) Model Point cloud after the hue filter

Figure 4.16 Hue filtered data and model point clouds

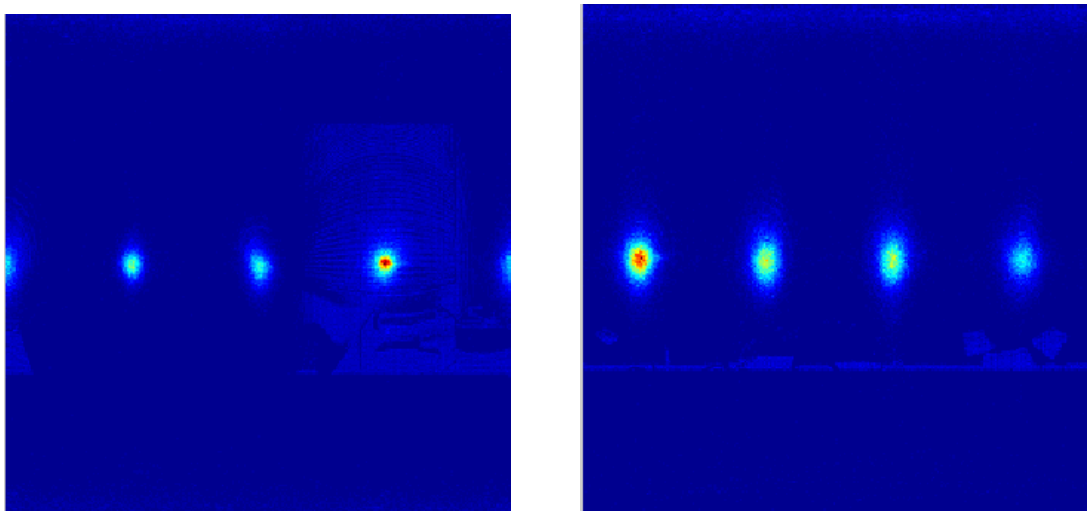
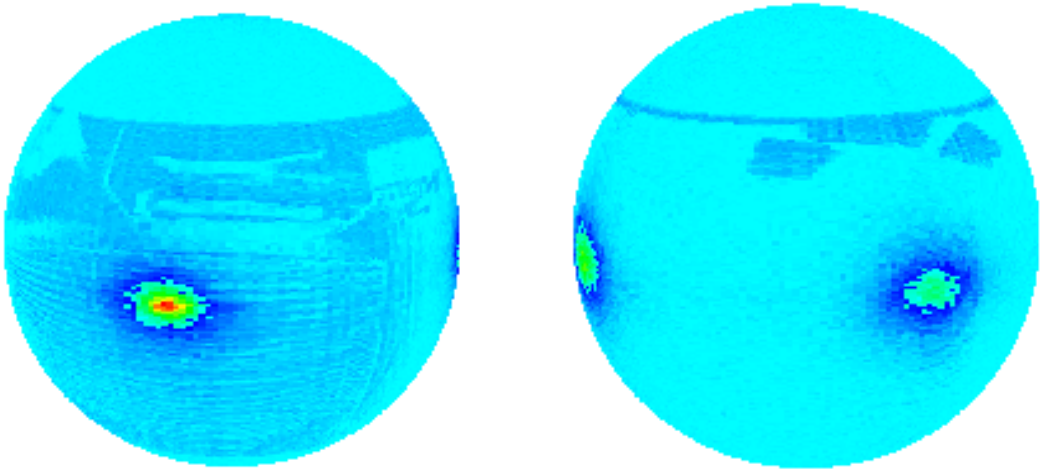


Figure 4.17 Orientation histogram of filtered point clouds



(a) (b)

Figure 4.18 EGI of filtered point clouds (a) Model (b) Data

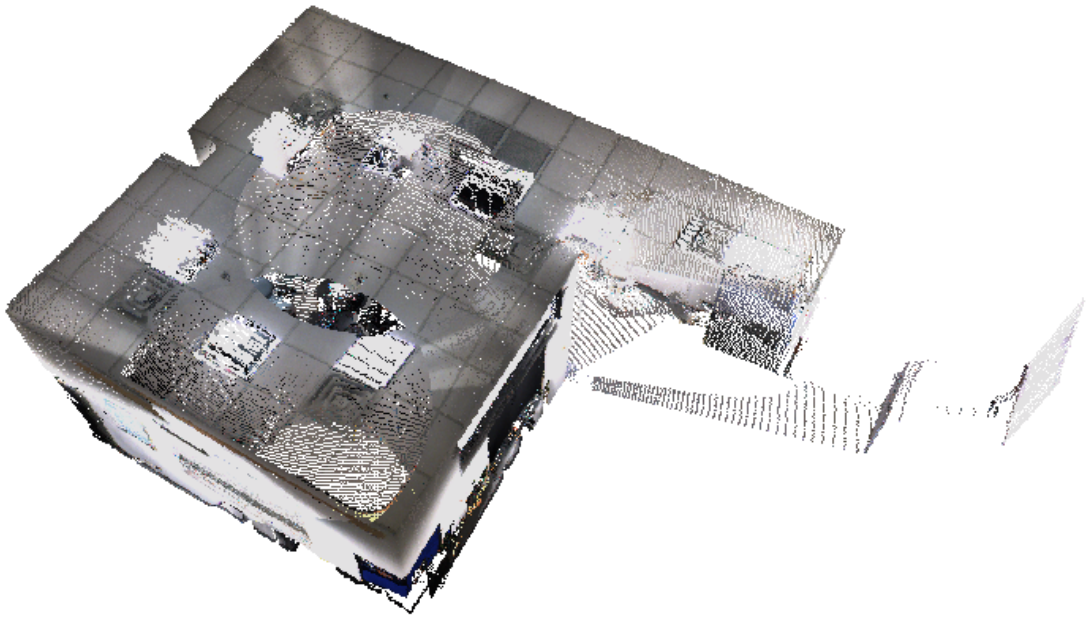


Figure 4.19. Registration results of filtered color point clouds

Hue constrained Automated Registration

The hue assisted automated color point cloud registration algorithm can be described as follows:

1. Load data point cloud $\boldsymbol{P_d}$ and model point cloud $\boldsymbol{P_m}$;
2. Compute hue on every points in $\boldsymbol{P_d}$ and $\boldsymbol{P_m}$;
3. Filter points based on hue distribution;
4. Filter outlier and voxel de-sampling on both $\boldsymbol{P_d}$ and $\boldsymbol{P_m}$;
5. Compute point surface normal $\boldsymbol{n_d}\{1\ldots N_d\}$ on data point cloud $\boldsymbol{P_d}$ and $\boldsymbol{n_m}\{1\ldots N_m\}$ on model point cloud $\boldsymbol{P_m}$;
6. Construct orientation histogram $\boldsymbol{H_d}$ and $\boldsymbol{H_m}$ based on extracted point normal;

7. Solve rigid rotation matrix $\boldsymbol{R}$ by solving the correlation between $\boldsymbol{H_d}$ and $\boldsymbol{H_m}$ with SOFT(3);
8. Rotate $\boldsymbol{P_d}$ with **R, get $\boldsymbol{P_d}$';**
9. Building occupancy grid with common bounding box in $\boldsymbol{P_d}$*' and* $\boldsymbol{P_m}$*;*
10. Solving rigid translation $\boldsymbol{T}$ by solving the convolution of $\boldsymbol{P_d}$' and $\boldsymbol{P_m}$,
11. Save transformed $\boldsymbol{P_d}$' for fine registrations.

Error Measurement

The registration error is measured by average distance between 2 nearest points from data to model point clouds, defined as:

$$\varepsilon = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_{id}-x_{im})^2+(y_{id}-y_{im})^2+(z_{id}-z_{im})^2} \tag{4.9}$$

In which, point $\boldsymbol{p_{id}}\{x_{id},y_{id},z_{id}\}$ in point cloud $\boldsymbol{P_d}$ looks for its nearest neighbor point $\boldsymbol{p_{im}}\{x_{im},y_{im},z_{im}\}$ in point cloud $\boldsymbol{P_m}$. The two nearest points can be paired when the distance between $\boldsymbol{p_{id}}$ and $\boldsymbol{p_{im}}$ dist($\boldsymbol{p_{id}}$, $\boldsymbol{p_{im}}$)<r_l. r_l is a constant value to ensure two nearest points are matched within a certain distance. N is the total number of paired points.

The hue assisted automatic registration on conference color point clouds takes 22.599 seconds and error is 0.001482. The automatic registration on 3D point clouds takes 31.572 seconds to complete with error 0.001501.

4.4 Experimental Results

Eight color point clouds generated at different vantage positions on a building hallway have been registered together by applying both automatic registration algorithms in 3D and with hue filter. The vantage positions are numbered in Figure 4.20. Coordinate system of position 3 generated point cloud has been considered as global space. Rest point clouds are registered into position 3 space. The computation time and registration error have been measured, results are compared and shown in Table 4.1. Experimental results prove that hue assisted automatic registration algorithm provides a reliable improvement on both computation time and registration accuracy.

Figure 4.20 8 scans about a building hallway

Position	**3D registration**		**Hue assisted registration**	
	Time (sec)	***error***	***Time (sec)***	***error***
1	35.270	0.003326	32.742	0.003098
2	35.445	.0.002398	32.251	0.002326
4	35.464	0.001586	30.845	0.001581
5	36.982	0.002108	29.838	0.002089
6	34.743	0.003378	28.353	0.001943
7	32.798	0.003087	27.971	0.003016
8	34.440	0.008274	29.822	0.008274
Average	**34.877**	**0.00363**	**30.260**	**0.00319**

Table 4.1 Registration Results Comparison

A global color point clouds map is constructed after registerging 7 color point clouds into position #3's space. A 3D view after registration is shown in Figure 4.21. These coarse registration results can then be supplied for fine registraton such as ICP to produce higher quality results.

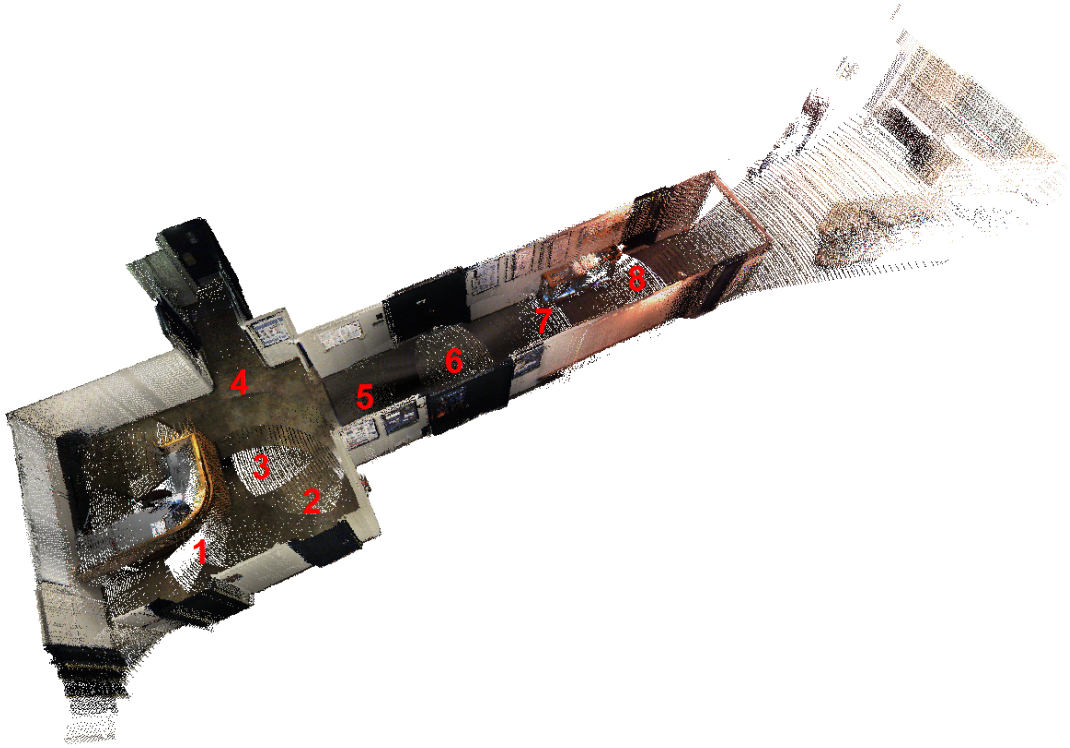


Figure 4.21 Map after registration result

4.5 Concluding Remarks

A Hue assisted automatic coarse registration algorithm that solves correlation for spherical rotation and 3D translation in Fourier space is described in this paper. The Hue value has been analyzed and utilized as a filter without the need for position and orientation information. A set of eight point clouds of a hallway and a conference room have been registered using a hue assisted automatic registration algorithm. Use of the hue value is shown to accelerate the registration speed and error minimization is shown to be effective.

Chapter 5 Hue Assisted Iterative Closest Point Algorithm

The Hue assisted Iterative Closest Point (HICP) algorithm brings color attributes from imagery into point cloud map fine registration process. The primary hypothesis of this algorithm is that the hue value can be applied to increase the accuracy of point association and accelerate the registration process. The major time and computation cost during ICP is finding the correct points pairs. A k-d tree—based Nearest Neighbor Search (NNS) is used to associate common points {x, y, z, hue} in 4-D space. The unknown rigid translation and rotation matrix required for registration is iteratively solved using the Singular Value Decomposition (SVD) method. A mobile robot with integrated 3-D scanning station generated color point cloud is used for verification and performance measurement of various map registration techniques. Numerical results on the generated map segments show that the HICP method resolves ambiguity in registration and converges faster than the 3-D ICP.

5.1 Color Point Cloud Registration

The most well known point cloud fine registration techniques is the ICP algorithm that have been applied to stitch two neighbor 3-D point cloud maps together into one map based on their common coverage area [7]. Upon convergence, the ICP algorithm terminates at a minimum. Several algorithms exist for calculating the minimum average distance between two point clouds. The Singular Value Decomposition (SVD) method [9], Eigensystem methods that exploit the orthonormal properties of the rotation matrices,

and unit and dual quaternion techniques were adopted in the ICP process. Quaternion—based algorithms have been used in ICP for map fusion [7]. SVD—based algorithms are widely used in ICP and 6DOF SLAM [9, 10, 13] as they have enough strength to reach local minimum and are easy to implement. Several variants of ICP are reported to increase the speed and precision for point cloud map registration. [11]. Corresponding points sampling, matching, weighting and rejecting are some methods used to accelerate the ICP algorithm. In the ICP algorithm, associating corresponding points in two point cloud data sets is the most critical step. Nearest Neighbor Search in 2-D or 3-D space is commonly used for associating the corresponding points. Parallel ICP algorithms have been developed to accelerate computation speed [14]. Point-to-plane registration method accelerates the ICP iteration and convergence [12, 13, 15].

Registration of color point clouds has been considered [31, 32, 83] after imagery information has been added into point cloud for better visualization. By applying proper calibration on the hybrid sensor system [48, 84], range measurement and visual information can be integrated together to construct a visually accurate representation of the scene. Color mapped 3D data was used in map registration by weighted red, green, blue data. The corresponding point search during the ICP is conducted on both the coordinate and color data. Hue filters were also used to constrain the closest point search in every ICP iteration[83]. Color data can be used to estimate initial alignment of pair wise scans using Scale Invariant Feature Transform (SIFT) techniques. Color attributes transferred in YIQ color model can also be weighted to construct new variant together with range information for ICP fine registration [85]. Depth-interpolated Image Feature (DIFT) algorithm solves corresponding points between two images and registers color point clouds based on extracted correspondences [86].

The Hue assisted ICP algorithm that takes input from a 3-D color scanner [3]. The key idea is to apply weighted hue value with coordinate data to accelerate the registration speed. The criteria for association are defined on a 4-D space rather than 3-D geometric space. The fourth-dimension is the hue, representing the intrinsic color values of the pixel. While achieving the effect of a hue-based filter, hue-association reduces the NNS burden considerably.

5.2 Introduction to ICP Algorithm

ICP algorithm is an iterative process that calculates the rigid transformation based on associating two point clouds. The point cloud defined about a known reference frame is termed as the model point cloud and that being registered as the data point cloud. A Singular Value Decomposition (SVD) based mean square error minimization method is applied to solve the transformation matrix that registers the data point cloud into the model reference frame [3]. A transformation error function, ***E(R,T)*** is defined as Eq. (5.1) that defines the mean distance between associated points.

$$E(R,T) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{ij} \left\| \vec{m}_i - (R\vec{d}_j - T) \right\|^2 \tag{5.1}$$

R and ***T*** are the relative 3D rotation and translation matrix for data point cloud transformation into the reference frame. N_m and N_d are the number of related points in model and data point clouds, respectively. $\vec{m}_i = \{m_{ix}, m_{iy}, m_{iz}\}$ represents the coordinates of the i^{th} point in the model point cloud and $\vec{d}_j = \{d_{jx}, d_{jy}, d_{jz}\}$ is the j^{th} point in data

point cloud. W_{ij} is weight for the association, which is assumed unity in this effort indicating that each associated point contributes equally to the error.

The point association of correspondence between the data and model point clouds is based on a nearest neighbor search using a k-d tree. The transformation matrix R and T are then updated by minimizing Eq. (5.1). The point association and update of the transformation matrices, R and T is iteratively performed until the specified convergence criterion is reached, in most cases the convergence criteria is defined as E reaches minimum. This variant of the ICP algorithm has been proved to be convergent [2].

5.3 k-d Tree Based Closest Point Association

The ICP computation speed and precision are highly dependent on association process. Use of a k-d tree for closest point search and association increases the speed and efficiency of the search. The k-d tree is a spatial partitioning data structure that stores and organizes data in a k dimensional space. The k-d tree is a generalized type of binary tree, with every leaf node is a k-dimensional data point that splits the hyperspace into two subspaces. Splitting is done sequentially from the first dimension to the k^{th} dimension. A typical k-d tree in 2D space is shown in Fig. 5.1. Each point in the 2D space divides the space sequentially into a left-right spaces (about x-axis) or into a top-bottom spaces (about y-axis).

Closest point association can be accomplished very efficiently based on k-d trees. For a given point with known coordinates in the data point cloud and a search radius, the

algorithm recursively moves down the tree and follows the same procedure as insertion. Search stops at a leaf node of the tree and the points in the model tree within the search radius are identified. The nearest point is obtained using distance computation. Figure 5.2 shows the nearest neighbor (red square) for the search point at the center of the circle. The nearest point is then regarded as the point associated with the search point.

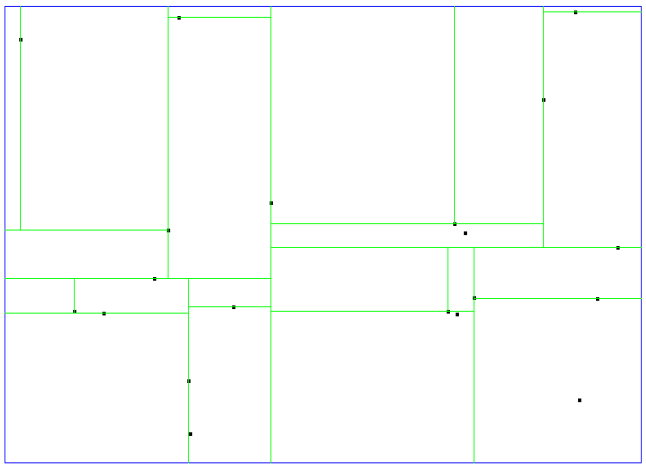


Figure 5.1 k-d tree construction in 2D space

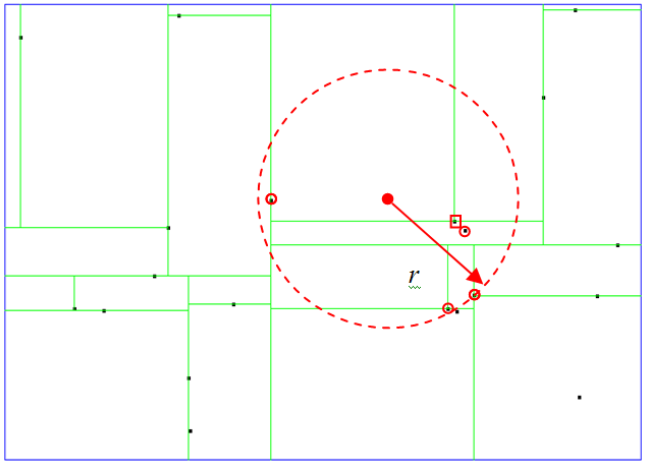


Figure 5.2 2D Space nearest neighbor search in k-d tree

5.4 Hue Invariance with Vantage position

Hue value remains consistent about the same point between images taken from two vantage positions, while the color values represented in red, green and blue quantities usually differ because of variation in light conditions. In order to effectively apply color to improve the association process, lighting effect should be removed. Color raw data are transformed into representation of separate chroma, lightness and brightness value. Figure 5.1 shows two camera images of different angles of a color palette on a Rubik's cube, four colors are used on the same surface. Figure 5.3 also shows the color pixels with the background and black frame removed. Histograms showing the red, green and blue value in RGB space for all the pixels are shown in Figure 5.5. In the RGB histogram, R, G, and B distributions of the image vary considerably with the vantage position. When the RGB color space is transformed into HSL space and histograms of hue, lightness and saturation are plotted Figure 5.5, the hue values remain relatively invariant with the position of the camera. Therefore, hue value of the pixel taken from the Hue-Saturation-

Lightness (HSL) model is used as the fourth dimension in the color point association process. In Figure 5.6, the picture about the building is shown in Figure 5.6(a), laser reflection rendered point cloud is shown in Figure 5.6 (b), the hue point cloud map of color point cloud map in Figure 5.6(c) is shown. Hue values are normalized between 0 and 1. The hue distribution is typically similar to the color distribution in Figure 5.6(d).

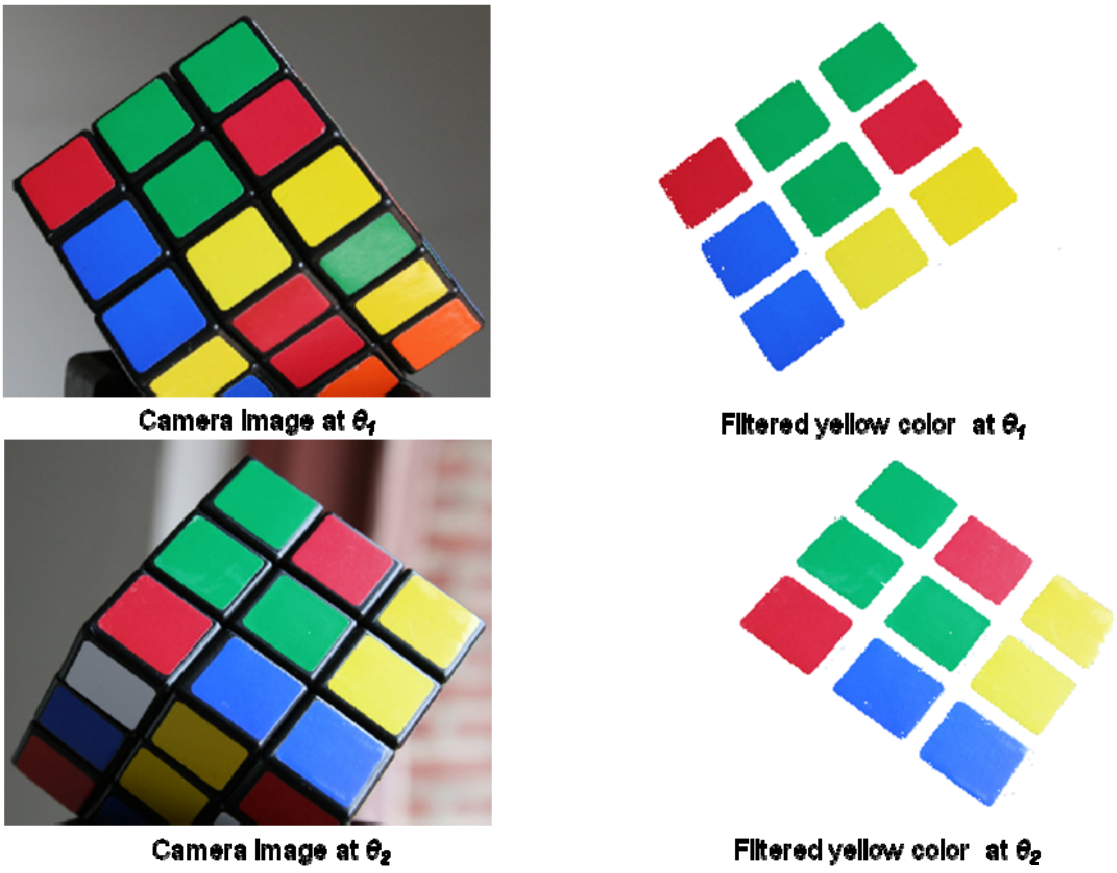


Figure 5.3 Rubik's cube camera images take from 2 different angles

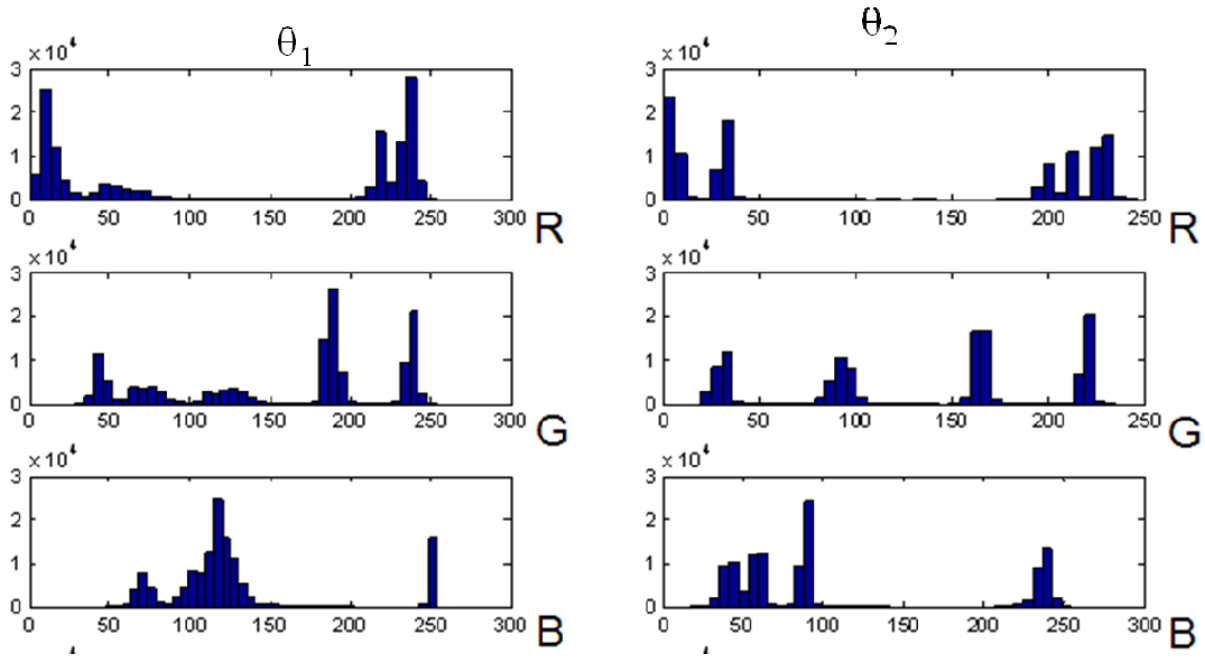


Figure 5.4 RGB distributions change with camera positions (θ_1, θ_2)

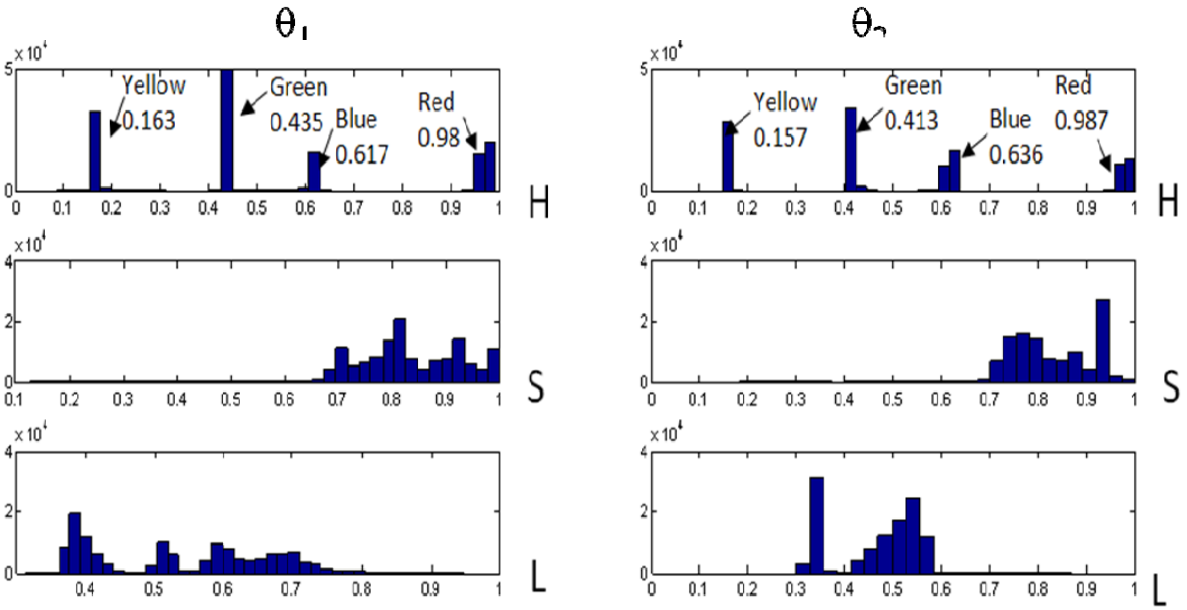
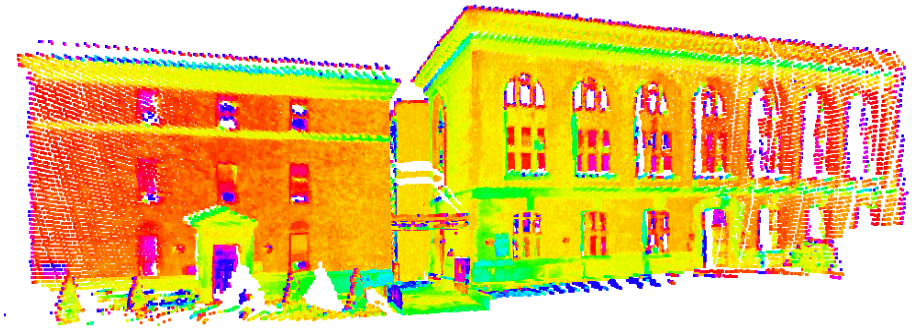


Figure 5.5 HSL distribution: hue remains invariant

(a): Image of an urban building



(b) Laser reflection intensity rendered point cloud of urban building

(a) Color point cloud map of urban building

(b) Hue map of the scene shown in (a)

Figure 5.6 Color point cloud map and its hue map about urban building

5.5 k-d Tree Based Color Point Cloud Association

In 3D ICP algorithm, corresponding points are searched according to the closest distance rule. This may cause incorrect matching during single iteration loop as Figure 5.7. Dashed line circle illustrates range based nearest point association results, in which all points in data set look for nearest neighbor in 3D space. It takes more than 1 iteration to pair correct nearest neighbor points for given data points set. Grey circle explains HICP nearest point search, based on correct hue property, the best neighbor in the model can be found in one iteration. Depending on the correct color information, corresponding point can locked with less iteration.

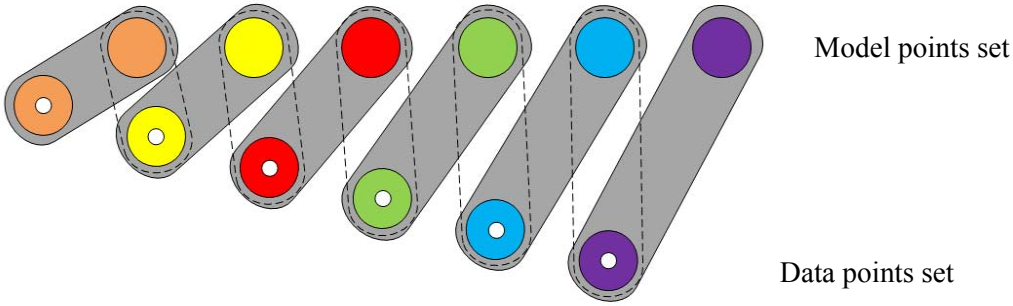


Figure 5.7 Comparison between color point association and range point association

The ICP computation speed and precision are highly dependent on association process. For Hue assisted ICP process, weighted hue combine with x, y, z range data can be utilized to construct 4D k-d tree for closest point search and association. In 3D closest point search, the distance between 2 points between 2 point clouds is:

$$r_{ij} = \sqrt{(m_{ix} - d_{jx})^2 + (m_{iy} - d_{jy})^2 + (m_{iz} - d_{jz})^2} \tag{5.2}$$

In which, $\boldsymbol{d_i}\{d_{ix},d_{iy},d_{iz}\}$ and $\boldsymbol{m_j}\{m_{jx},m_{jy},m_{jz}\}$ are point spatial coordinates in data and model point cloud map respectively.

Both hue and range values have to be combined together in the HICP variant as $\{x_o, y_o, z_o, h_w\}$ for point association. x_o, y_o, z_o are the original coordinate values with distance units, and h_w is the weighted hue value. Hue values are calculated from 0 to 1 and then scaled to xyz space based on the point cloud bounding box dimension; hue also must be weighted during the closest point search in the four-dimensional space. The new variant for point association is $\{x, y. z, h_w\}$, where x, y, z are the coordinates and h_w is the weighted hue.

The weight value for the hue dimension should be properly selected for point association since both range and hue values are scaled according to the point cloud bounding box. Weight for hue represents its influence in the Nearest Neighbor Search process. Low weight inclines the point association towards the range data and a high weight towards the hue values. Small weight values for the hue correspond to the traditional 3-D-ICP. Error in HICP is evaluated by the average mean square root distance of associated point pairs.

In 4D space, the 4th dimension for each point should be weighed hue value d_{hw} or $m_{hw.}$ The spatial value of points should be normalized by 3D search radius r_{ij} as mentioned in section 5.2. In order to accomplish closest point search in 4D space, the distance between two normalized points $\boldsymbol{d_i}\{d_{ix},d_{iy},d_{iz,},d_{ihw}\}$ and $\boldsymbol{m_j}\{m_{jx},m_{jy},m_{jz},m_{jhw}\}$ should be:

$$r_{ij}^{'} = \sqrt{(m_{ix} - d_{jx})^2 + (m_{iy} - d_{jy})^2 + (m_{iz} - d_{jz})^2 + (m_{ihw} - d_{jhw})^2} \tag{5.3}$$

or

$$r_{ij}' = \sqrt{r_{ij}^2 + \Delta h_{ijw}{}^2} \tag{5.4}$$

In the ICP process, search radius effects the computation time and final result. A constant search radius is applied for all iteration loops. If the search radius is large, too many points will be included as candidates during association. On the other hand, if the search radius is small, the points may not be associated and more iteration loops will be required. The optimal search radius depends upon the density of point cloud. In 4D k-d tree search, the search radius is based on both the coordinate data as well as the weighted hue as shown in Eq. (5.3). As a rule of thumb, search radius is typically selected to yield about 50 candidate points. If a substantial weight is used in the construction of 4-D space, the k-D search will bias toward hue dimension and the HICP algorithm will behave close to a applying a hue-filter to the system.

Strictly coordinate based association may result in non-unique registration. For example, if the points in the model and the data point clouds belong to a plane, coordinate based ICP results in non-unique association of points. In such cases using the hue value (if differences exist in this dimension in the matched features) may result in unique registration of the points. The computation complexity of 3D ICP algorithm is $O(n^2)$, where n is the number of points in data point cloud. As the hue-value is unaffected by the coordinate transformations applied during the ICP process, specifying weighted hue value into the k-d tree nearest point search, the computation complexity remains as $O(n^2)$.

5.6 Error Minimization

If $m_i = \{m_{ix}, m_{iy}, m_{iz}\}$ represent the coordinates of the i^{th} point in the model point cloud and $d_j = \{d_{jx}, d_{jy}, d_{jz}\}$ are coordinates of the j^{th} point in the associated or paired point set, a distance

error is defined as given in Eq. (5.2). Similar as error evaluation during 3D point cloud registration process, error in HICP iteration is computed as:

$$E(R,T)=\frac{1}{N}\sum_{i=1}^{N}\left\|m_i-(Rd_i+T)\right\| \tag{5.5}$$

Centroids are computed for the associated points in both model and data point clouds as shown in Eq.5.5. The coordinates are translated to have the origin at the centroid as given inEq.5.5. An orthonomal transformation matrix of associated points can be constructed (Eq.5.7). Rotation ***R*** and translation ***T*** are decoupled. Using Singular Value Decomposition (SVD), ***R*** can be solved from the orthonormality matrix (Eq.5.8). Translation ***T*** is computed based on the translating the centroids of model and data point sets (Eq.5.9).

$$\overline{m}=\frac{1}{N}\sum_{i=1}^{N}m_i \quad , \quad \overline{d}=\frac{1}{N}\sum_{i=1}^{N}d_i$$

$\overline{m}=\{\overline{m}_x,\overline{m}_y,\overline{m}_z\}$ and $\overline{d}=\{\overline{d}_x,\overline{d}_y,\overline{d}_z\}$ are the centroids of associated points in model and data point clouds. N is the total number of associated points. The coordinates after transformation are:

$$m_i'=m_i-\overline{m} \quad , \quad d_i'=d_i-\overline{d} \tag{5.6}$$

$m_i'=\{m_{ix}',m_{iy}',m_{iz}'\}$ and $d_i'=\{d_{ix}',d_{iy}',d_{iz}'\}$ are the i^{th} associated point about the transformed coordinate system. The orthonormality matrix H can be constructed based on $m'\{m_i', i=1\ldots N\}$ and $d'\{d_i', i=1\ldots N\}$.

$$H=\begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}$$

Where

$$S_{xx} = \sum_{i=1}^{N} m'_{ix}\, d'_{ix} \quad , \quad S_{yy} = \sum_{i=1}^{N} m'_{iy}\, d'_{iy} \quad , \quad S_{zz} = \sum_{i=1}^{N} m'_{iz}\, d'_{iz} \quad , \quad S_{xy} = \sum_{i=1}^{N} m'_{ix}\, d'_{iy} \tag{5.7}$$

Singular value decomposition is performed for the ***H*** matrix to determine the rotation matrix **R,** that minimizes the error as:

$$H = U \Lambda V^T \tag{5.8}$$

Where optimal rotation $R = VU^T$. The translation ***T*** can be calculated as:

$$T = \overline{m}^T - R\,\overline{d}^T \tag{5.9}$$

5.7 Surface Normal based Error Evaluation

In order to compare the registration accuracy between H-ICP and 3-D ICP algorithm, point cloud surface normal based error is utilized to measure the model point cloud surface normal variance before and after registration. The surface normal based error is defined as

$$\varepsilon_n = \frac{1}{N} \sum_{i=1}^{N} \left| n_{ix} n'_{ix} + n_{iy} n'_{iy} + n_{iz} n'_{iz} \right| \tag{5.10}$$

In which, ε_n is the normal based error, N represents the number of points in model point cloud, $\{n_{ix,}\ n_{iy,}\ n_{iz}\}$ is the surface normal vector of i^{th} point in model point cloud before registration, $\{n_{ix}',\ n_{iy}',\ n_{iz}'\}$ is the surface normal vector of the i^{th} point in model point cloud after registration. After data point cloud has been registered into model point cloud space, both model and data point clouds are merged together and point cloud surface normal are re-computed on newly merged point cloud. Therefore the normal based error

describes point surface normal variance before and after registration. The normal based error ε_n goes higher when better registration quality is reached.

5.8 Algorithm for Hue-Assisted ICP

Both hue and range value have to be combined together in the HICP variant as $\{x_o, y_o, z_o, h_w\}$ for point association. x_o, y_o, z_o are the original coordinate values with distance units and h_w is the weighted hue value. Hue values are normalized from 0 to 1 and must be weighted during the closest point search in the four-dimensional space. In order to normalize the coordinates, we find the bounding box for each map segment and the coordinate space is rescaled to a 0-1 range. The normalized variant for point association is $\{x, y. z, h_w\}$, where $x=x_o/r_x$, $y=y_o/r_y$, $z=z_o/r_z$. r_x, r_y, r_z are the dimensions of the bounding box in x, y, z directions.

The weight value for the hue dimension should be properly selected for point association. Since both range and hue value are normalized from 0 to 1. Weight for hue represents its influence in the nearest neighbor search process. Low weight biases the point association towards the range data and a high weight towards the hue values. Small weight values for the hue correspond to the traditional 3D-ICP. Hue weight should be selected between 10% and 35% for accurate point association. Error in HICP will be evaluated by the average mean square root distance of normalized associated point pairs.

The Hue-assisted ICP algorithm entails the following steps:

1. Estimate the initial values for the matrices ***R*** and ***T*** that transform points in the data point cloud into the model point cloud's coordinate system.
2. Construct k-d tree of model point cloud $\boldsymbol{M_i m_1, m_2, m_3 \ldots m_M\}}$, with weighted hue value as the 4th dimension;

3. ***While*** merging error ε > *preset tolerance*

3.1 Use ***R*** and ***T*** to transfer data point cloud $\boldsymbol{D\{d_1,d_2 \ldots d_N\}}$: $\overrightarrow{D} = R\overrightarrow{D} + T$

3.2 Nearest Neighbor Association Step:

For *i*=1 ***to*** Number of points in the data point cloud

Set Number of Associated Points N = 0

Search closest point for point $\boldsymbol{d_i}\{d_{ix}, d_{iy}, d_{iz}, d_{ih}\}$ in model *k*-d tree

If closest point $\boldsymbol{m_j}$ exists within a specified search range, *r*

Associate $\mathbf{d_i}$ and $\mathbf{m_j}$ as $\{\mathbf{d_k}, \mathbf{m_k}\}$;

Increment number of associated points: *N* ++;

End If

End For

3.3 Distance error Computation: For each associated point pair, calculate normalized mean square root distance ε *as* error,

$$\varepsilon = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(d_{ix}-m_{ix})^2+(d_{iy}-m_{iy})^2+(d_{iz}-m_{iz})^2}$$

3.4 Solve for R and T that minimize: Construct orthonormality matrix ***H*** (Eq.5.7) and solve rigid rotation ***R*** and translation ***T*** (Eq.5.8 & 5.9);

End While

4. ***Post-Registration error estimates:*** Compute any post registration errors such as planarity or curvature continuities.

5.9 Convergence Criteria

Three separate criteria are established for convergence of the HICP iteration. First, a measure called the association stability is applied. Association stability(S) is defined as the number of points that changed their paired points in the previous iteration of the ICP algorithm. A large value of *S* indicates that the point association is not stable and a small or zero value indicates that the point pairing has stabilized. Secondly we use the convergence of number of points associated during the NNS search. Second convergence criterion used is the change in error, HICP algorithm is terminated when the following three measures converge:

a) Error value: $\varepsilon \rightarrow 0$

b) Number of associated points: $N \rightarrow 0$

c) Association stability measure: $S \rightarrow 0$.

To illustrate the convergence of the HICP and ICP algorithms, a point cloud has been utilized and shown in Figure 5.6, with 122,409 points as the model point cloud. The data point cloud is obtained from model point cloud with a known rotation about one axis. For this case, the exact rotation required to register the data and model point clouds is known. Figure 5.9 (a) shows the comparisons of convergence of the error measures in HICP and traditional 3D ICP. The number of associated points converges after the 5^{th} iteration (Figure 5.9(b)) and the stability of the association converges after several more iterations. Figure 5.9(c) shows that the association stability ($S \rightarrow 0$) after 15^{th} and 26^{th} iteration for HICP and 3D ICP respectively. Figure 5.10 shows the development of the elements of the transformation matrix required to register the two data sets and the known transformation matrix converges with the convergence of error, the number of the associated points and the stability measures.

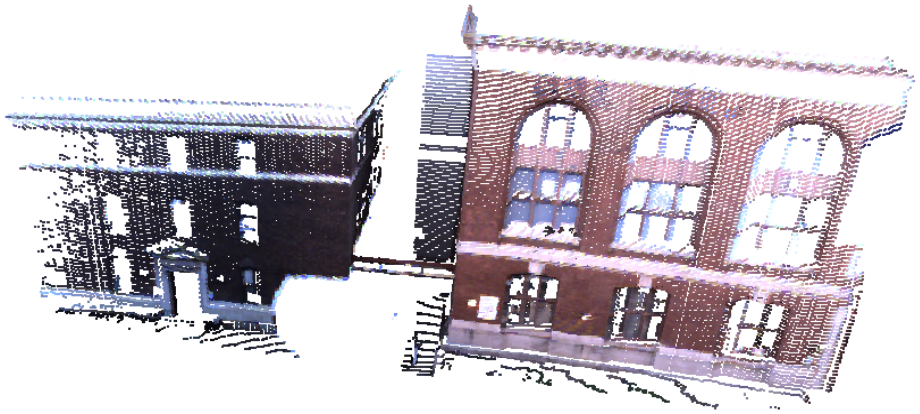
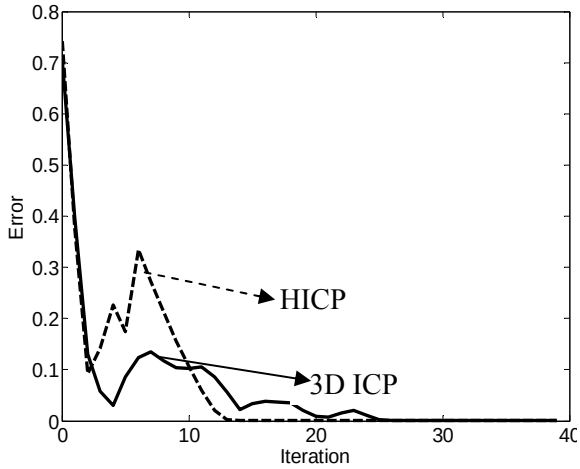
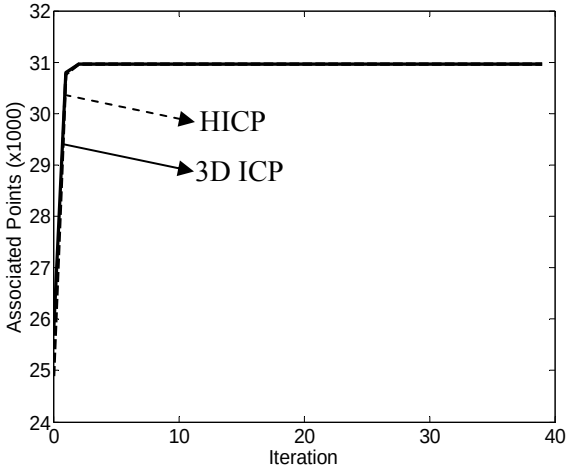


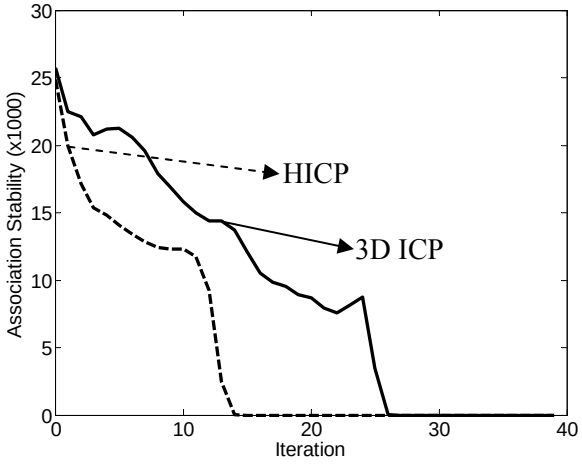
Figure 5.8 Color point cloud map about urban building



(a) Mean square error



(b) Number of associated points



(c) Stability metric

Figure 5.9 Evolution of error metrics for HICP and 3D ICP in building map registration

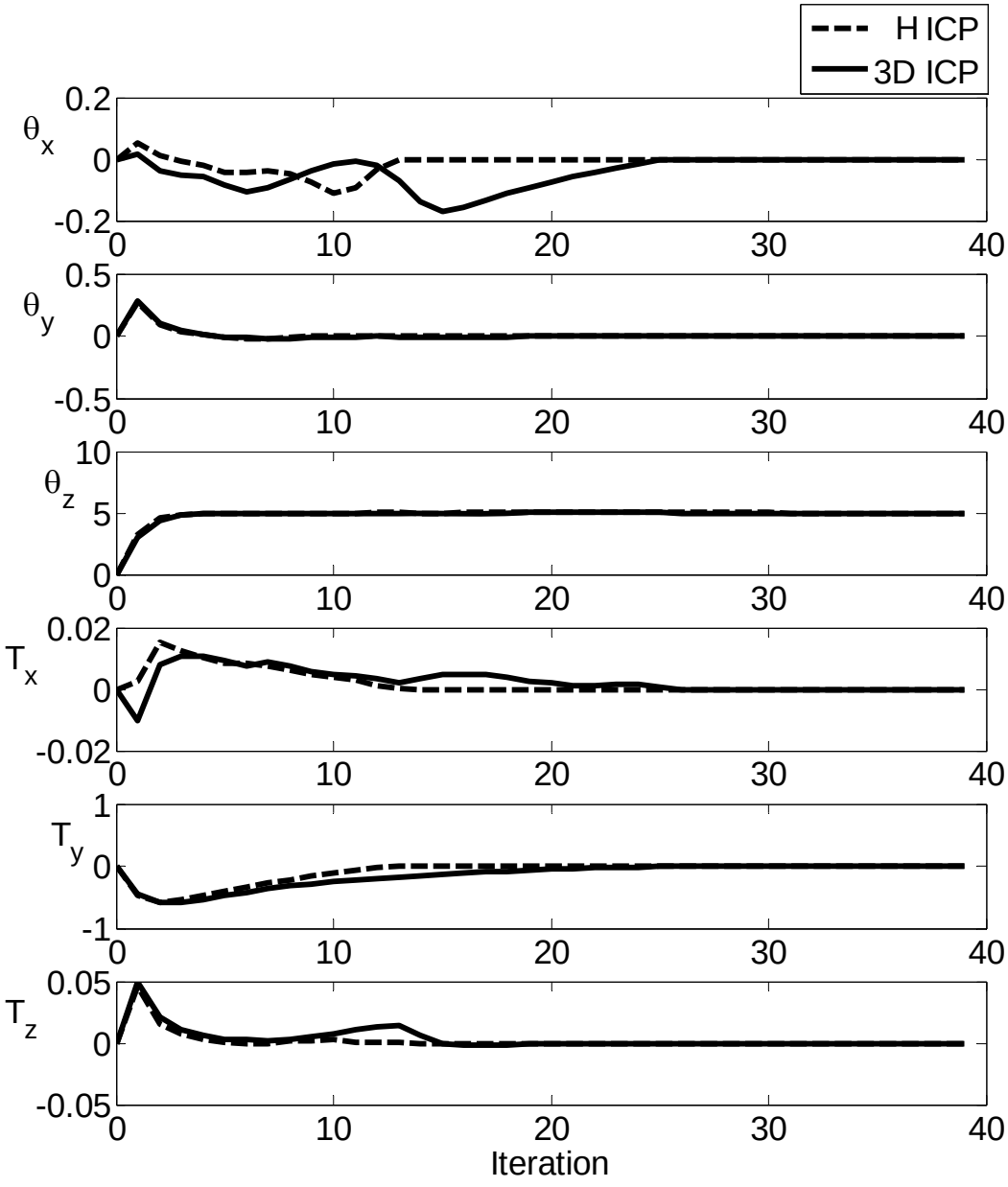


Figure 5.10 Evolution of registration transformation matrix during HICP and 3D ICP

5.10 Map Registration with ICP and HICP

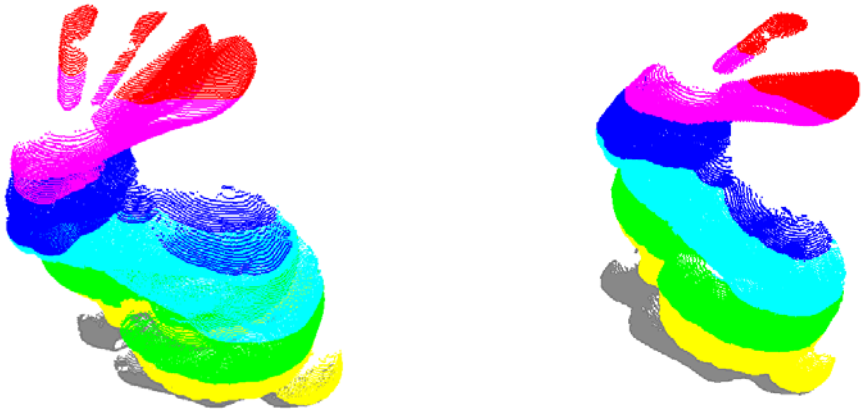
This section compares the performance of the ICP and HICP algorithms under various hue distribution scenarios on the same geometric model --- the Stanford bunny point cloud. Table 5.1 shows colors, corresponding R, G, B values and the hue value on 0 -360 scale.

5.10.1 Point clouds with Fixed Hue Distributions

For the first experiment, we colorized the Stanford bunny point cloud model as shown in Figure 5.11(a). In this model, the hue varies from 0 to 360 from bottom to top at Z direction in seven segments. Figure 5.11(b) also shows the initial registration of the model and data point clouds used for this simulation.

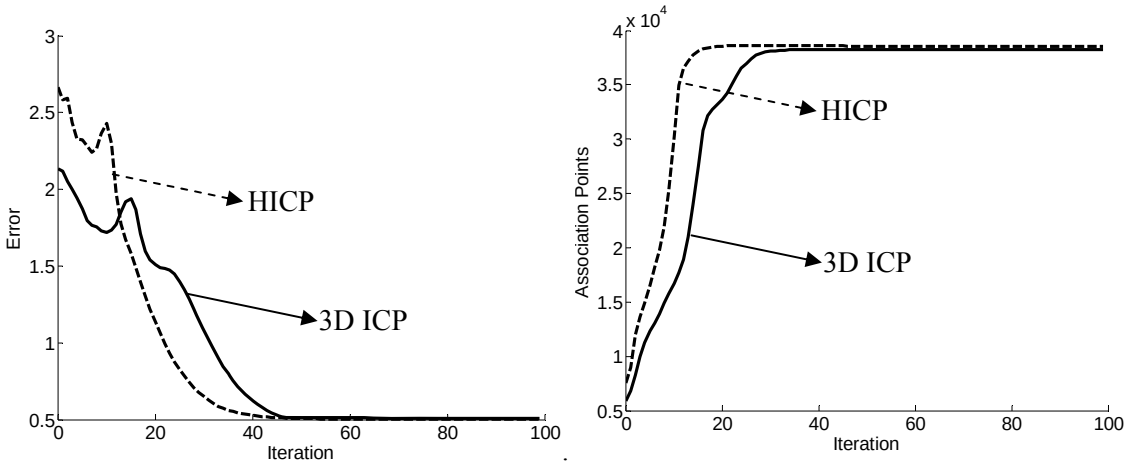
	R	G	B	Hue
Gray	128	128	128	0
Yellow	255	255	0	60
Green	0	255	0	120
Cyan	0	255	255	180
Blue	0	0	255	240
Magenta	255	0	255	300
Red	255	0	0	360

Table 5.1 Varied hue and corresponding color in RGB space



(a) Data and model point clouds (b) After registration

Figure 5.11 Seven-Segment hue rendered bunny model before and after registration



(a) Mean square error (b) Number of associated points

Figure 5.12 Comparison of convergence between HICP and 3D ICP algorithm

Hue assisted ICP registration progress is shown in Figures 5.12(a) and (b). Figure 5.12(a) shows the mean square error during the ICP process and Figure5.12 (b) shows the number of points associated during iteration loops. Both data and model point cloud after registration is shown in Figure 5.11(b). The hue-assisted ICP registers the point and data clouds faster than the traditional coordinate based ICP.

5.10.2 Continuously Varied Hue along One Dimension

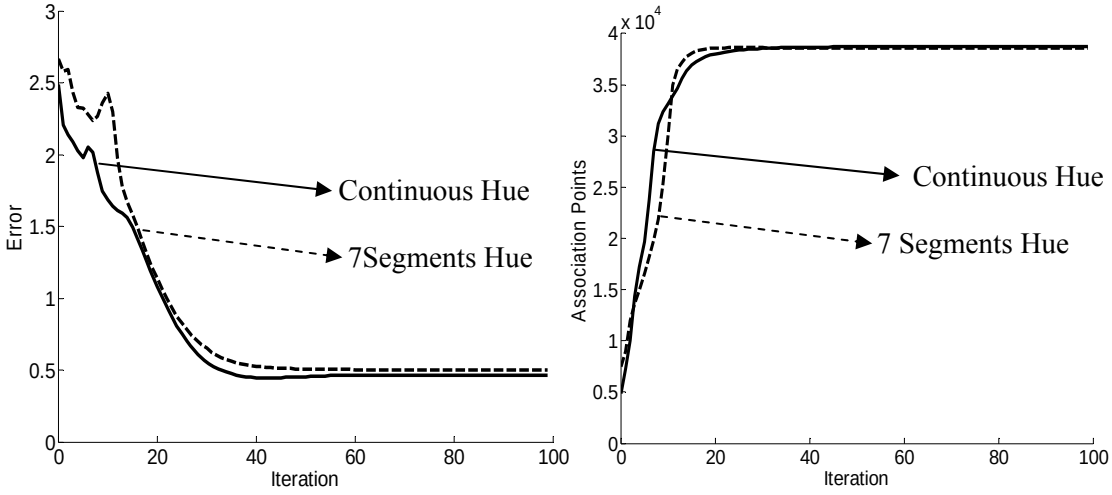
In the second simulation, a continuous hue distribution is assigned to the bunny model. The hue value varies from 0 to 360, smoothly, along the z (vertical) direction. The resultant model and data clouds are shown in Figure 5.13 (a, b). Saturation and lightness value have been set as constant at every point inside dataset. Hue value can be calculated by Eq. (5.10).

$$h = 360\frac{z_i - z_{min}}{z_{max} - z_{min}} \tag{5.10}$$

h is the hue value at range point i, z_i is the coordinate distance for i^{th} point at z direction, z_{max} and z_{min} are maximum and minimum coordinate of the point cloud at z direction. Continuous hue distribution on point cloud data is registered together (Figure 5.13 (c)) and the results are shown in Figure 5.15. A comparison of model performance on discrete and continuous distribution of hue on the same model shows the expected acceleration in performance due to uniform distribution of hue on the model.

(a) Data point cloud (b) Model point cloud (c) Merged View

Figure 5.13 Bunny model with continuous hue variation in one axis



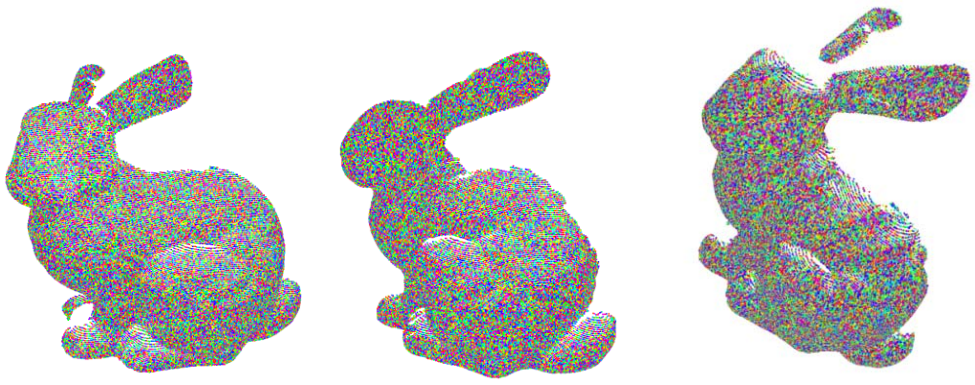
(a) Mean square error (b) Number of associated points

Figure 5.14 Convergence of HICP for 7-segment hue model and continuous hue model

5.10.3 Randomized Hue on the Model

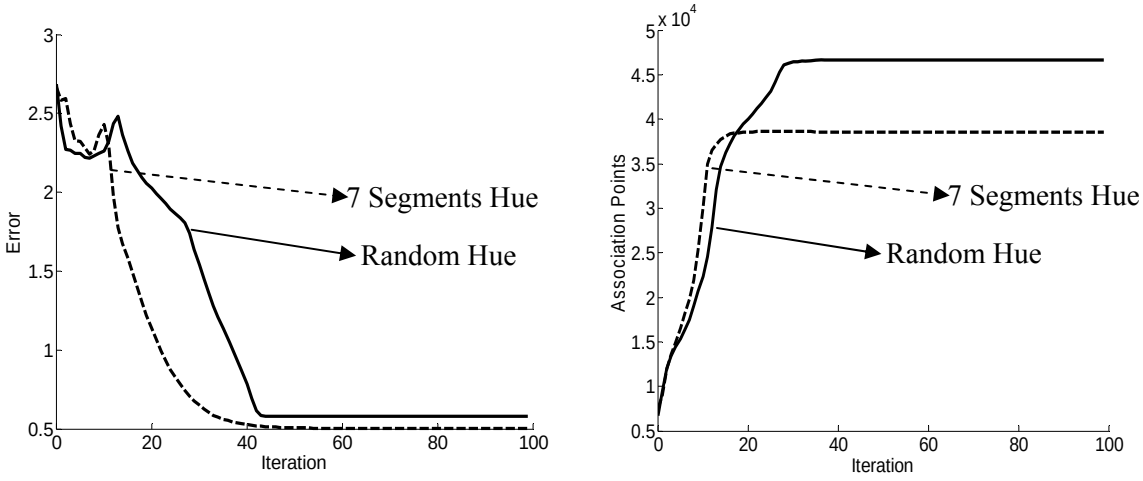
In this case, the model considered has a continuously distributed hue but with a randomized and noisy pattern. In this case, there is no geometric pattern for the color on the object. The color point clouds are rendered in Figure 5.15 (a, b). The merged cloud point cloud after registration is

shown in Figure 5.15(c). Figure 5.15 shows the error minimization iteration and comparison with the seven-segment hue distribution model. In this case the hue confuses the nearest neighbor search. The registration accuracy is also not as good as a patterned hue case.



(a) Data point cloud (b) Model point cloud (c) Merged view

Figure 5.15 Random hue rendered bunny

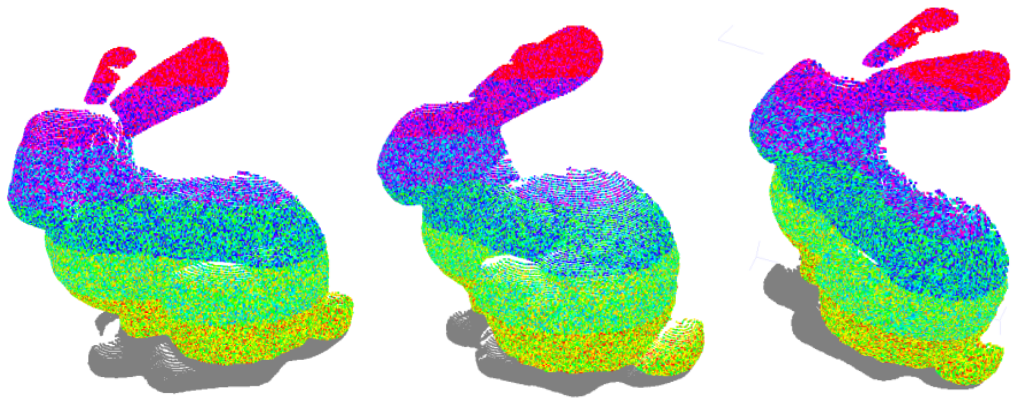


(a) Mean square error (b) Number of associated points

Figure 5.16 Comparison between discrete and random hue distribution case

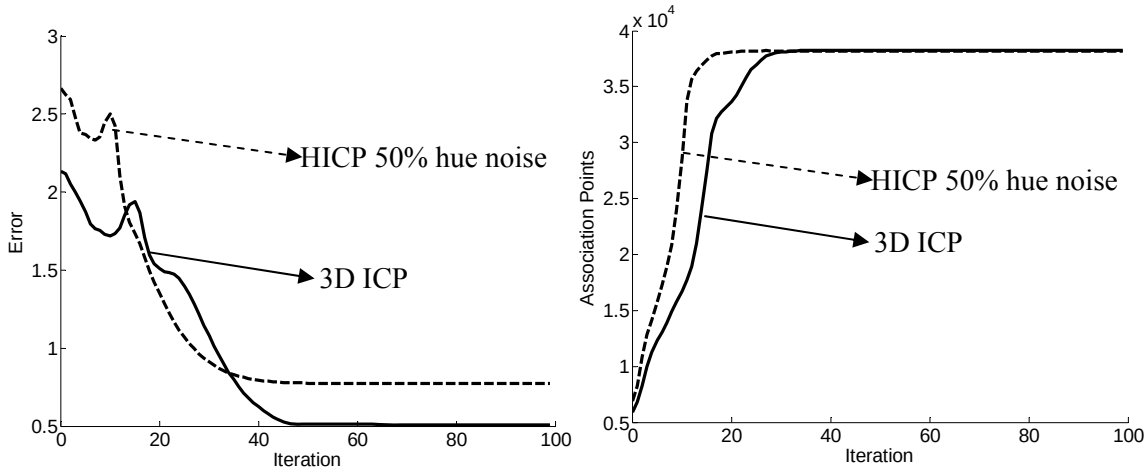
5.10.4 Effect of Camera Noise

In the previous simulation, the imaging sensor is assumed perfect. The hue on a point is assumed to be recorded by the imaging sensor perfectly in both model and data clouds. Some noise in the color measurement can be expected when the point clouds are generated from two vantage positions. Considering this situation, we colorized the bunny model but with 50% noise in the sensor. The points in the model and data clouds differ in color by as much as 50%. The resulting point clouds are shown in Figure 5.17 (a, b). The merged color point cloud is shown in Figure 5.17(c).



(a) Data point cloud (b) Model point cloud (c) Merged View

Figure 5.17 Varied hue with 50% noise rendered bunny model



(a) Mean square error (b) Number of associated points

Figure 5.18 Comparison between HICP in noise hue data and 3D ICP results

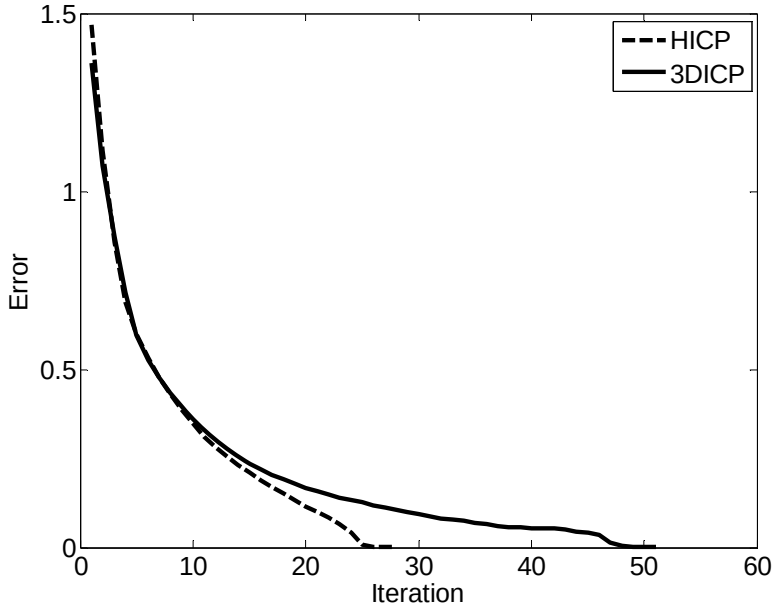
HICP matching result in camera noise color point cloud is compared with 3D ICP matching performance. From Figure 5.18, noise in hue decreases the matching accuracy and reduces the iteration efficiency. Two groups of cloud point clouds are selected to evaluate the performance of HICP algorithm compared with typical 3D ICP. A known transformation point cloud data pair was generated by transforming model point cloud at 6DOF to compare the convergence speed and registration accuracy as the rigid transformation is already known.

5.11 Registration with Known Six DOF Transformation

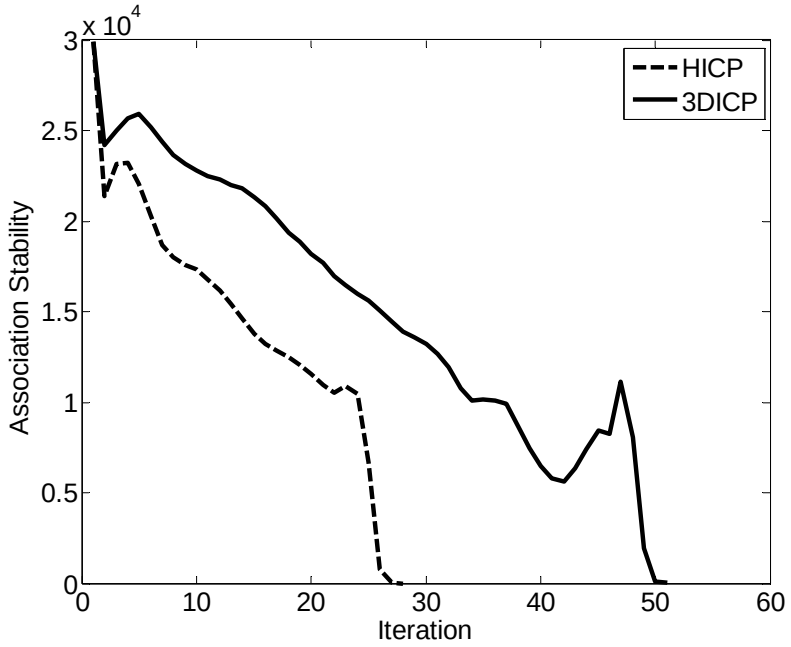
This experiment compares registration speed between 3D ICP and HICP for two point clouds whose registration transformation is known a priori. Both HICP algorithm and 3D ICP algorithm were applied on the map obtained from mobile mapping robot [48]. The same point cloud has been transformed to a new viewpoint at 6DOF. New view position is selected with 5° off around the X, Y, and Z axes in original space. Translation is selected as distance 1.014969, 2.00582 and 0.706715 on X, Y, Z axes, respectively. Error comparison is shown in Figure 5.19(a) and

association stability via iteration is shown in Figure 5.19(b) to illustrate convergence of the process. Rigid motion is compared in Figure 5.20. The HICP completes registration after the 28th iteration, which range ICP accomplishes after the 51st iteration. This registration comparison proves that hue—assisted ICP algorithm works well on larger off—alignment point cloud registration. The 6DOF map registration is accelerated by the HICP algorithm.

The building registration comparison demonstrates that the HICP algorithm increases the speed of the registration process. Because the HICP applies hue to be associated in the fourth—dimension, more accurate point association can conduct convergence earlier than the 3-D ICP method. Merged color point cloud about the building is shown in Figure 5.21. Three-dimensional ICP takes 166.889 seconds and 51 iterations to complete the registration process, while HICP takes 87.764 seconds and 28 iterations to register point cloud together when weight is selected as 20%.



(a): Mean square error comparison



(b): Stability Comparison

Figure 5.19 Registration comparison between 3-D ICP and HICP algorithm

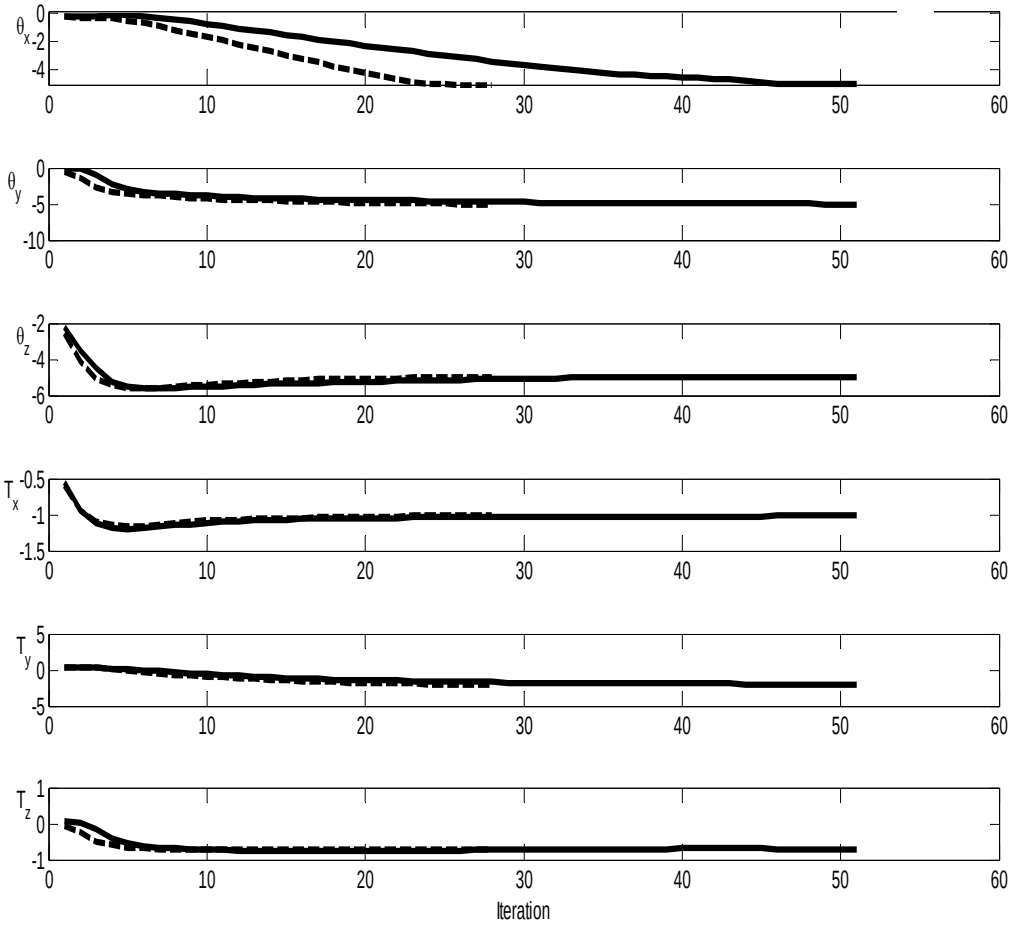


Figure 5.20 Rigid transformation comparison during building map registration process

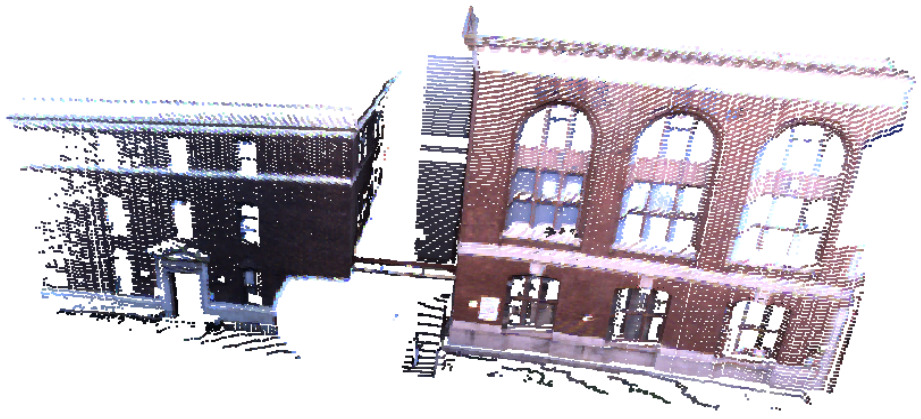
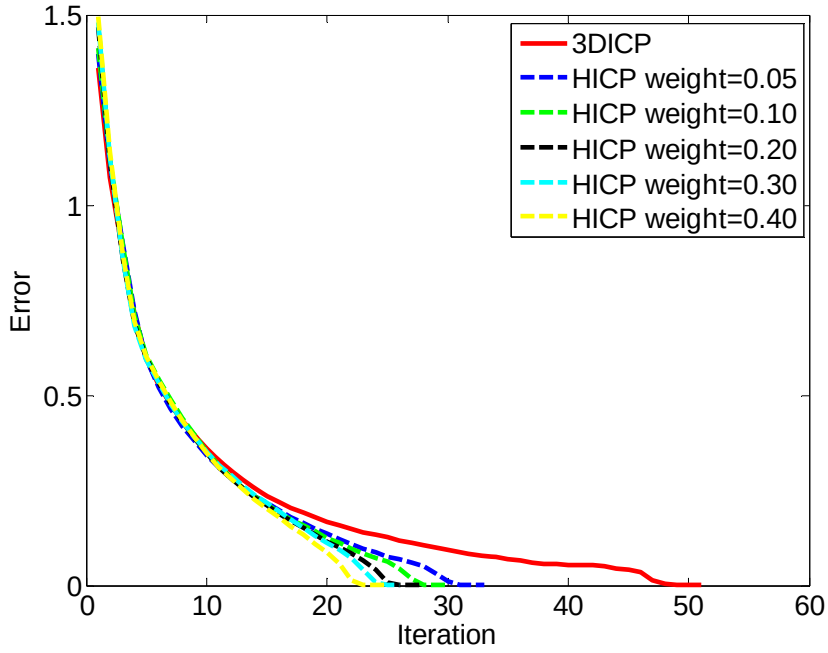


Figure 5.21 Registered building point cloud view

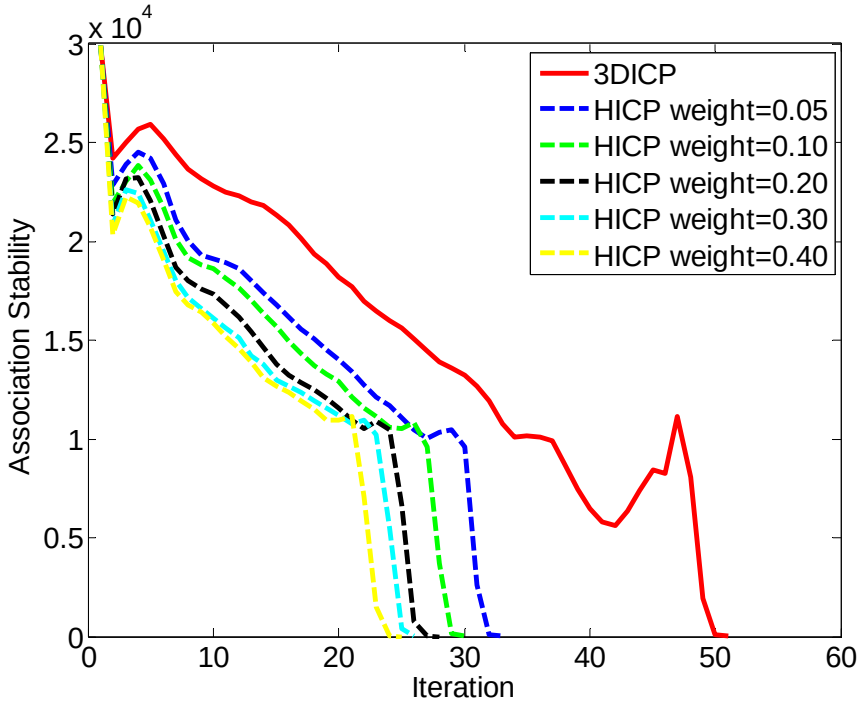
Different hue weights including: 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 1.0, 2.0, 5.0, 10.0, 100.0 in HICP have been compared with 3-D ICP. All different hue weights in HICP conduct the same registration results with same rigid transformation. The time comparison and number of iterations are compared in Table 5.2. Both data and model point clouds have 100% overlap, the hue value on associated points are equal without any noise. Normal based error ε_n went to 1 at all different weights. In this case, higher weight of hue brings faster registration speed and less iteration loops based on more accurate point association in every iteration. The registration process including stability and error comparison is shown in Figure 5.22. In Figure 5.22(b), higher weighted hue in HICP enables faster convergence, which is proven in the error comparison in Figure 5.22(a).

Methods	Time (seconds)	Iterations
3-DICP	166.89	51
HICP weight=0.05	98.05	33
HICP weight =0.10	93.07	30
HICP weight=0.20	87.76	28
HICP weight=0.40	73.11	25
HICP weight=1.00	56.06	21
HICP weight=2.00	52.42	19
HICP weight=5.00	42.25	16
HICP weight=10.00	36.21	14
HICP weight=100.00	28.08	12

Table 5.2 Time and iteration comparison between weighted HICP and 3-D ICP (100% overlap)



(a) Mean square error comparison



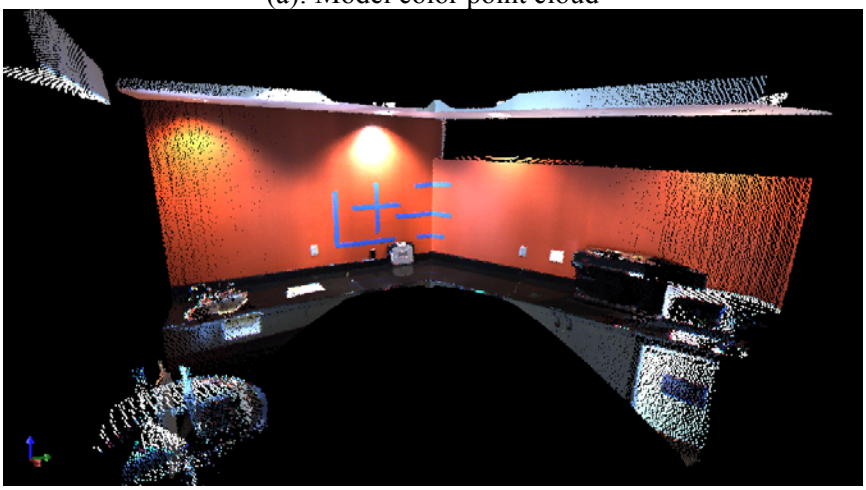
(b) Stability Comparison

Figure 5.22 Registration comparison between 3-D ICP and different weighted HICP algorithm

5.12 Indoor Point Clouds Registration with Different Hue Weight

Two color point clouds generated at different vantage positions about indoor environments have 90.19% overlap have been applied for HICP registration, as shown in Figure 5.23. Figure 5.23 (a) illustrates the model color point cloud and Figure 5.23(b) is the date the point cloud will to register into model space.

(a): Model color point cloud



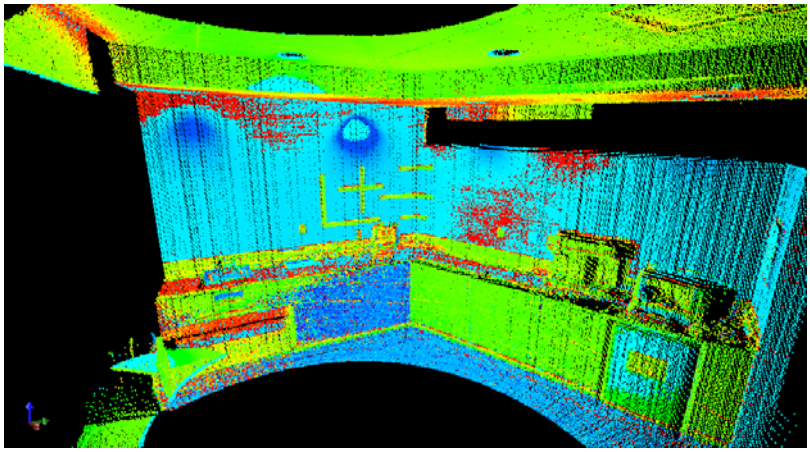
(b): Data color point cloud

Figure 5.23 Two color point clouds about indoor environment

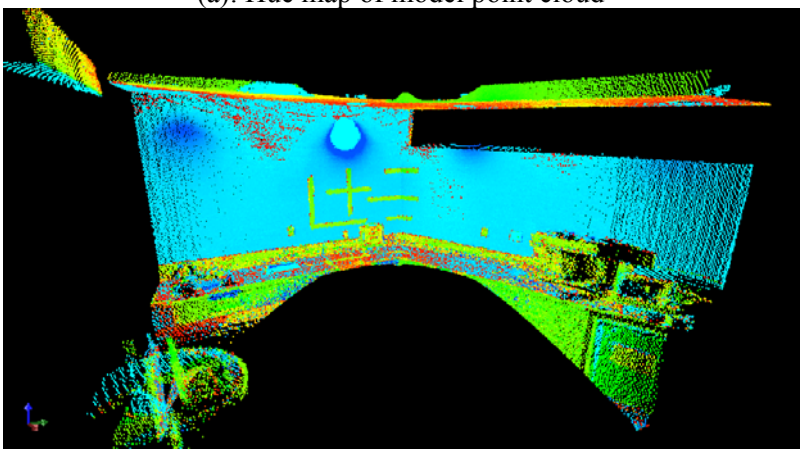
Noise of hue exists because of sensor noise and reflection on object surface, the hue map of color point clouds are shown in Figure 5.25. The red points on the wall in both point clouds are noise of hue, a band filter was applied in order to take out the noise effect from hue. Hue has been computed between 0 to 1 range and in this case a low pass filter at hue=0.5 was applied so that points with hue value lower than 0.5 are applied for the registration. HICP algorithm has been applied with different hue weight to register point cloud together, the point clouds before and after registration is shown in Figure 5.25 (a, b). The results are shown in Figure 5.26. Error in Figure 5.26(a) shows that weight=0.05 produces best registration accuracy. Stability plot in Figure 5.26(b) illustrates higher weight of hue generate faster convergence and less iteration loops, also included as the time consumption in Table 5.3. Due to the deviation of hue on associated points, registration accuracy will decrease if the hue weight goes too high. On contrast, the registration speed will increase significantly if very low weight is selected, the accuracy remains close to 3D ICP registration because hue value is not well utilized as the 4^{th} dimensional element. From the comparison in Table 5.3, hue weight as 0.05 satisfies both accelerating registration speed and increasing accuracy. The normal based error ε_n at weight 0.05 is 0.9969, which means the registration result at weight =0.05 may not be the best rotational alignment orientation. However, considering the translational alignment, this weighted H-ICP cloud conduct best registration results.

Methods	Time (seconds)	Iterations	Error	Normal Error
3-DICP	678.02	108	0.066	0.9957
HICP weight=0.01	437.02	93	0.041	0.9968
HICP weight=0.02	432.58	94	0.041	0.9969
HICP weight=0.05	332.69	74	0.040	0.9969
HICP weight=0.10	366.60	81	0.042	0.9969
HICP weight=0.20	306.15	70	0.043	0.9971
HICP weight=0.30	277.61	74	0.045	0.9971

Table 5.3 Registration results comparison between weighted HICP and 3-DICP (90.19% overlap)



(a): Hue map of model point cloud



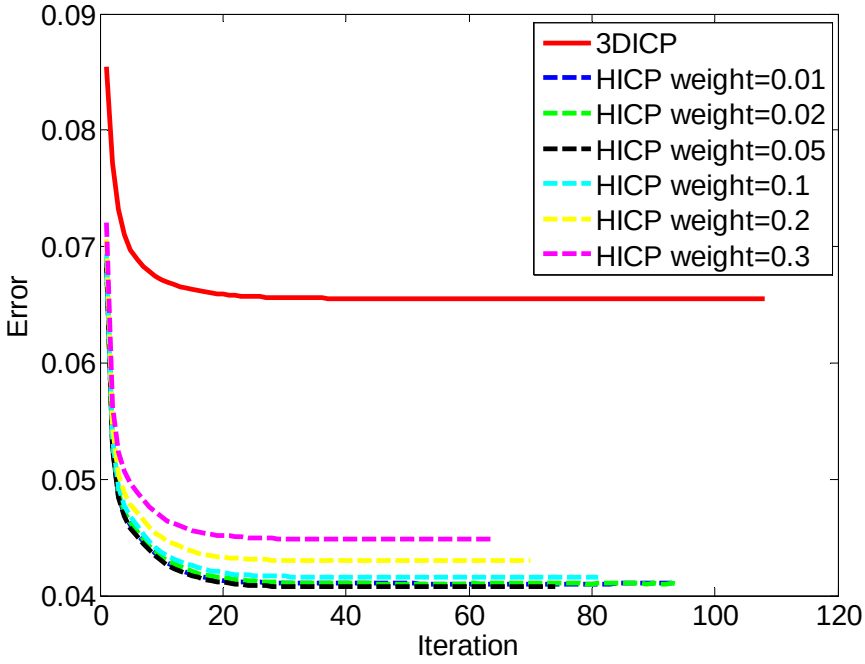
(b): Hue map of data color point cloud

Figure 5.24 Hue map about given point clouds before registration

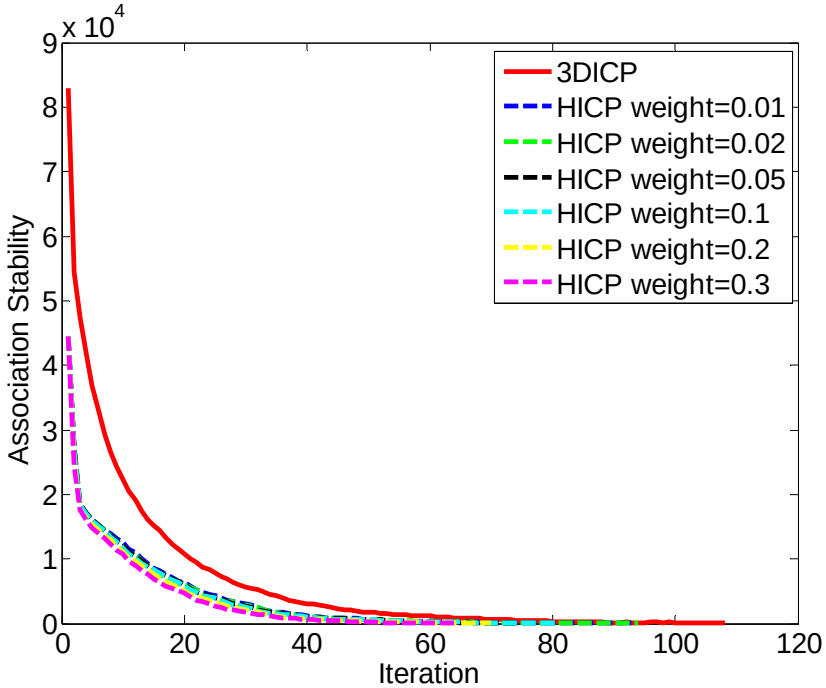
(a): Point Clouds before HICP

(b): Indoor color point clouds after registration

Figure 5.25 Indoor color point clouds before and after registration



(a): Mean square error comparison



(b): Stability Comparison

Figure 5.26 Registration comparison between 3-D ICP and different weighted HICP algorithm

5.13 Sequential Registration with Unknown Transformation

Three-dimensional range ICP and HICP algorithms are applied to pair—wise indoor color point cloud map registrations. Color point clouds taken from six different places are pair—wisely registered together to construct a map about the whole floor. Figure 5.27 shows the top view of the floor with vantage positions on registered point cloud maps. This scene includes walls, stairs, and desks. Pair—wised registration is applied between the current position color range map and previous position range map. Color point clouds are filtered using a low pass filter at hue=0.5, and the weight for hue is selected based on the previous experimental result: weight =0.05. The error is evaluated by the paired points' mean square root distance and number of iterations to converge. Final error is illustrated in Table 5.4.

Position	3-D ICP Iteration #	3-DICP Time(seconds)	HICP Iteration #	HICP Time(seconds)	3-D ICP Error	HICP Error
2	81	360.24	74	263.36	0.379	0.370
3	83	203.14	37	63.43	0.303	0.301
4	84	386.54	67	199.88	0.472	0.461
5	133	393.46	68	329.76	0.568	0.549
6	120	465.85	71	156.69	0.947	0.849

Table 5.4 Multiple color point clouds pair wise registration comparison

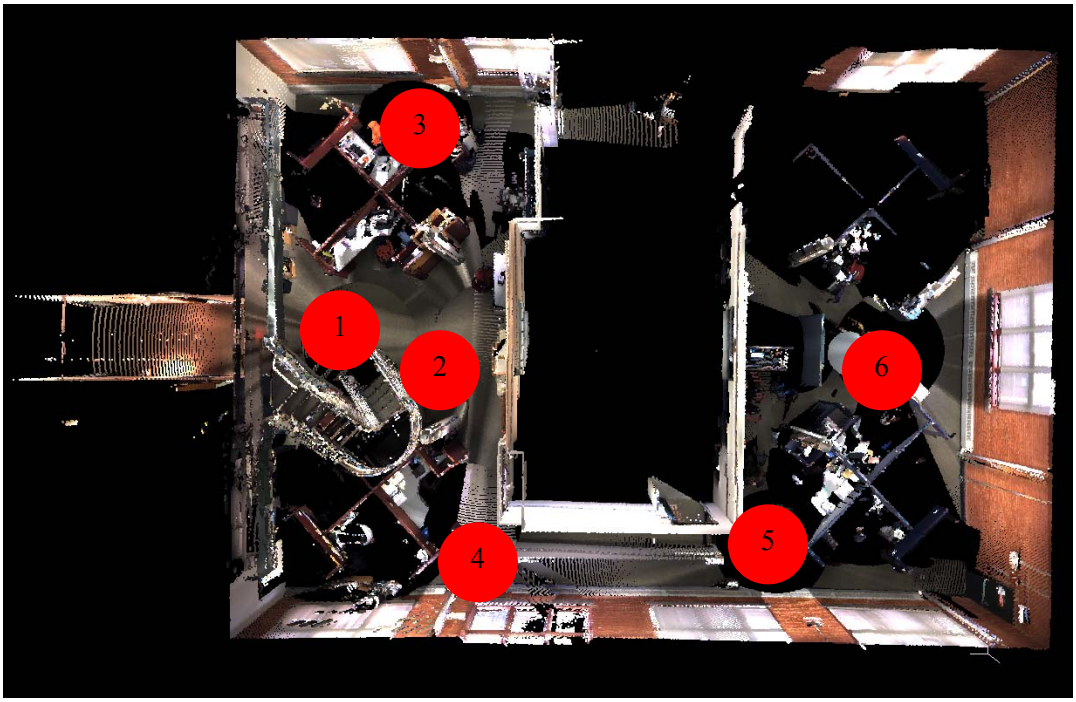
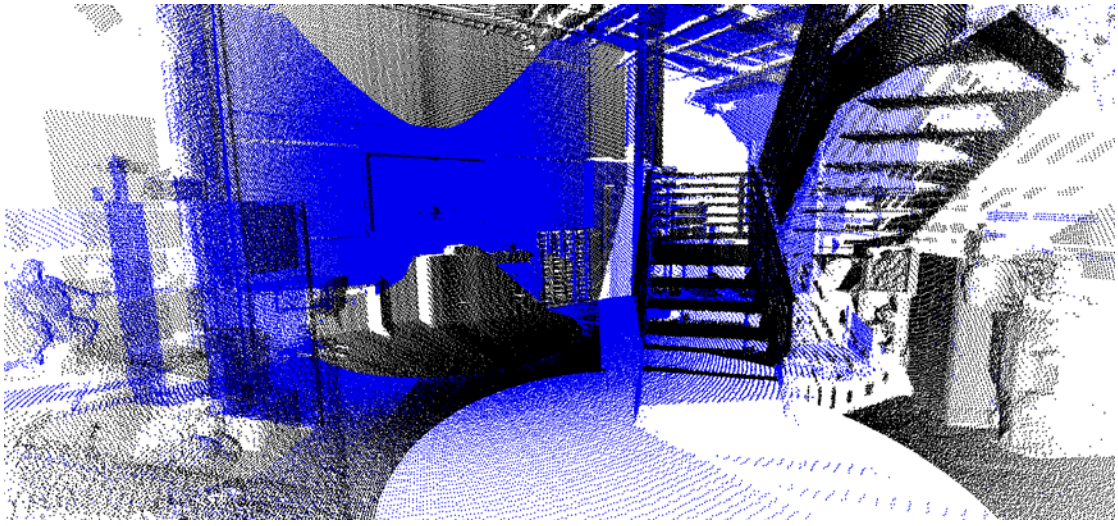
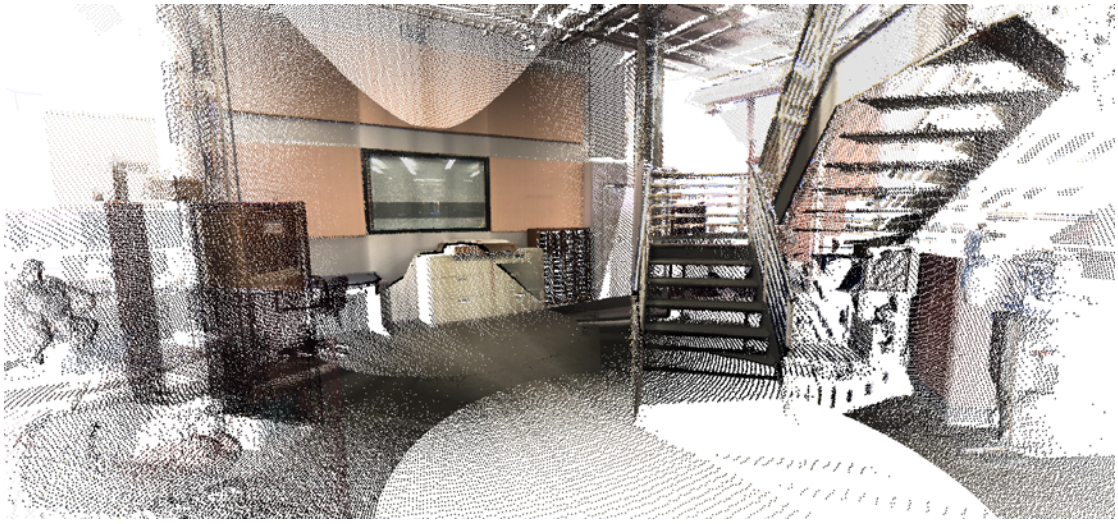


Figure 5.27 Top view of registered color range maps

Hue—assisted ICP takes less iteration loops than 3-D range ICP to converge for color point cloud registration in Table 5.4. This experiment proves that faster registration will be conducted by adding color value into the registration process. HICP algorithm reduces registration time and error. Position 1 and 2 acquired point cloud maps are registered together and are shown in Figure 5.28. Figure 5.28(a) describes two different point cloud maps with two different color point clouds; position 2 (blue)is registered into position 1 point cloud (black). Combined point clouds with color are shown in Figure 5.28 (b).



(a): Registered position 2 (blue) point cloud into position 1 (black) point cloud.



(b): Registered position 1&2 color range map.
Figure 5.28 Vantage position 1 & 2 registered map view.

5.14 Concluding Remarks

This chapter describes an algorithm to introduce color attributes into ICP point cloud registration process and fundamental algorithms for autonomous robotic mapping. Both range data and weighted hue value are applied during the registration process and quantitatively evaluate the effect of hue weight for the point association process. A building data set, pair—wise indoor environment, and multiple scan indoor color point maps are registered using an image data assisted algorithm. Use of the hue value to assist the point association and error minimization is effective during the ICP iteration schemes. Higher dimensional point association based on properly selected weighted hue and range data provides more accurate point matching results and conducts earlier convergence of the ICP process and reduces computation time. When rigid transformation is applied in every iteration loop during the ICP period, hue value does not change in space transformation. However, in HSL data space, lightness should change according to the view angle and light position. Corresponding point search using additional lightness values could be a further research field to increase the HICP algorithm.

Chapter 6 Path Planning for Exploration and Mapping

The path planning algorithm is desired to produce a next stop vantage position for mapping based on acquired map knowledge of the environment. The incomplete map should be utilized to compute explored and unexplored areas. The path planning process is driven by map completeness and exploration, therefore the vantage generation algorithm is a frontier in the next best view position generation process. However, due to the overlap requirements for map registration and intrinsic blind zone from the mapping sensor, vantage position computation is constrained. The path planning utilized a point distribution histogram at attitude direction on a horizontal 2D occupancy grid. Experimental results prove the robustness and reliability of the path planning algorithm.

6.1 Introduction

Path planning strategies are critical when a robot only has partially acquired knowledge about the environment. The efficiency and performance of robot exploration can be improved by adaptively utilizing their knowledge about the travelled area, or so-called map [26, 69, 72]. In most autonomous mapping applications, robots are required to be navigated to reach their destination with minimum travel and time consumption.

Essentially, path planning for mapping missions includes three processes [65, 87, 88]: *map learning, localization, and path-planning*. The map learning process requires memorization of the map information in a proper form of representation, construction of the map with a global reference frame, recognition and extraction of objects and landmarks. Localization derives

current position and orientation from observations of the scene and a map reference for association of landmarks. On board velocity and position sensors estimate and correct the estimation by associating landmarks from previous observations. Path planning selects a course of action for reaching a goal position from the current position. Map learning and localization are dependent on each other because global map construction requires localization data and localization requires a mapping reference.

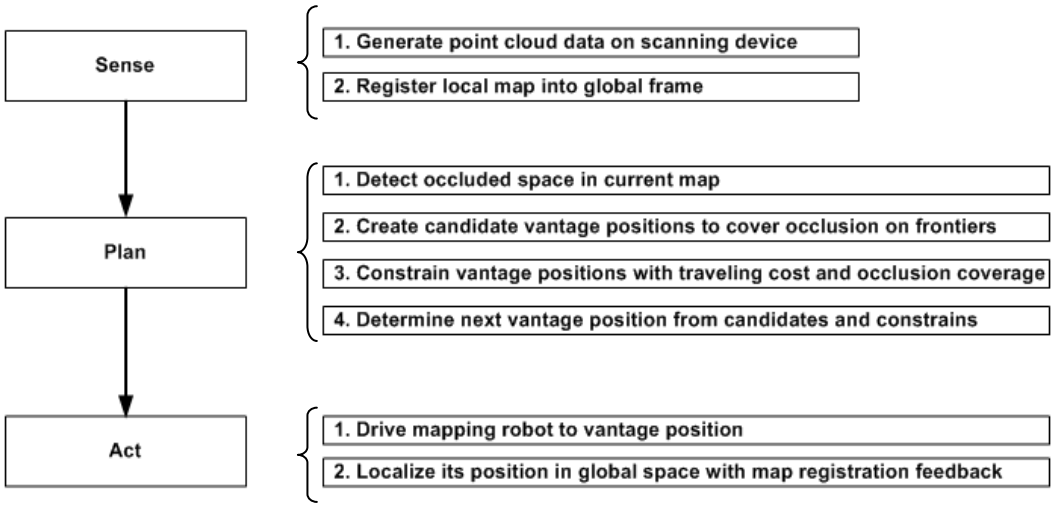


Figure 6.1 Autonomous map generation process

In order to accomplish autonomous mapping, a robot deployed at an initial (start) position must go through three phases of the mapping process, typically following a Sense-Plan-Act model. Figure 6.2 shows the work flow. The robot must be able to localize itself so it can navigate the scene. This can be accomplished by well-known SLAM (Simultaneous Localization and Mapping) techniques. Methodologies for establishing the map completeness and detection of occluded areas are necessary. Determination of the optimal vantage position for filling in the occluded areas and exploring unmapped areas is a critical step. As the navigation is based on imprecise mapping and

localization information, the map segment registration based on 3D color point clouds is the last, but crucial, step in building the complete map of a given area.

The use of exploration for mapping processes is incremental and relies on the overlap between the scans. As scans are generated at each vantage point, the global map is memorized in the form of features and occupancy level [44, 74, 89]. Fuzzy logic is employed to represent uncertainty in the feature position [90]. A similar scheme, but without orthogonal walls assumption, has been developed for complete mapping [30]. Instead of using a value iteration algorithm [34], exploration is directed toward the closest frontier between explored and unexplored areas. The path to this frontier is computed based on depth-first search in the explored area. Probabilistic techniques using Bayes theory have been applied to determine the likelihood of each grid occupied [42, 74, 89].

Minimum exploration cost computation and the map completeness evaluation are two major challenges to design an efficient exploration strategy for complete mapping. For complete coverage mapping some hypothesis have been made [11, 75, 76], an exploration agenda memorizes detected but not explored corridors. Opportunity scripts based on short sequences of actions for mapping error detection and correction [75] are also evident in the literature. When multiple paths are possible, A* or D* algorithms have been used to determine the shortest path between two robot positions.

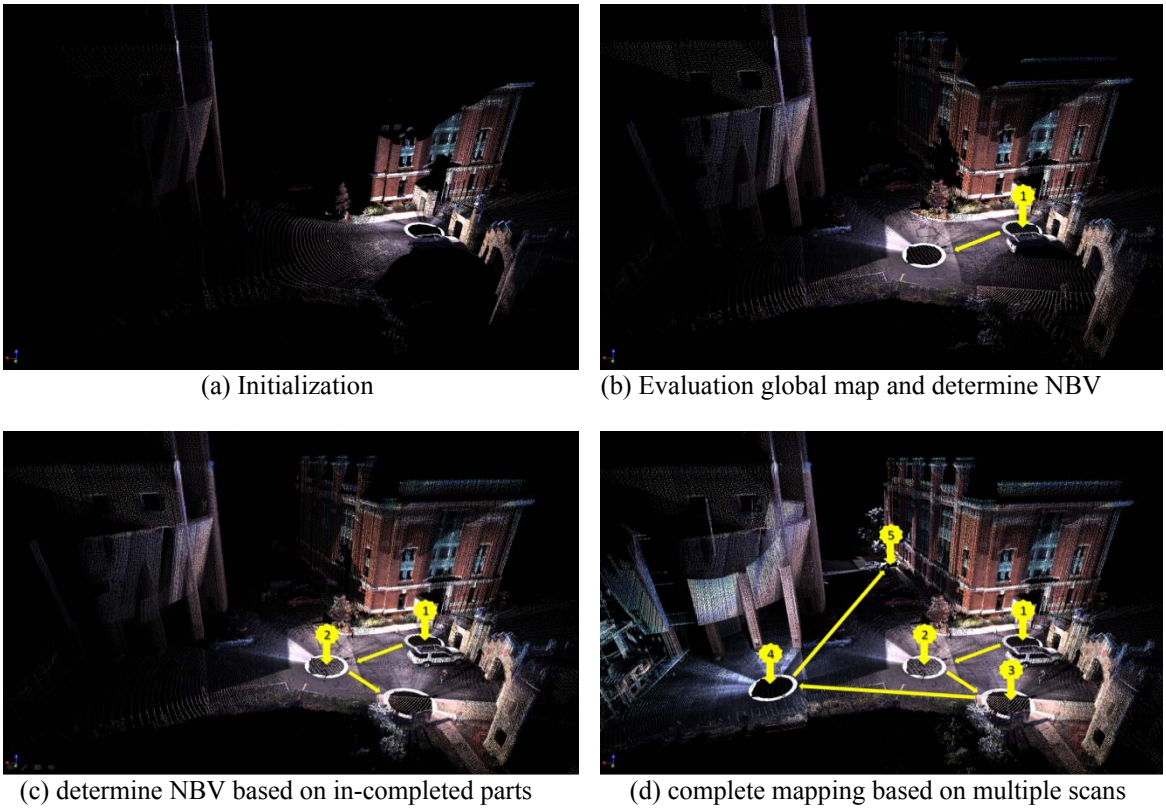
Other considerations for the determination of the next vantage point are the field of view and coverage of unexplored spaces. Computing the vantage position for mapping is commonly known as the Next Best View (NBV) problem [11, 81]. Determination of the next vantage position based on the current map data can generally be based on following criteria:

- Maximum coverage of unmapped area;
- Ascertaining that overlaps in map data

- Minimum accessibility risks and traveling costs;

The next vantage position is determined in two steps. The first step is candidate position generation and the second step is the selection of optimal vantage position. The candidate vantage positions can be created based on the frontier exploration algorithm [70] considering obstacles, position, and terrain conditions. The vantage position is selected between candidate positions that have the best view of coverage and shortest traveling cost. The next vantage position should be decided based on the best views for filling occluded regions.

Figure 6.3 illustrates the path planning process used for generating the 3D map described in Chapter 2. Figure 6.3(a) shows the 3D map from the first vantage position, which is used as the global reference coordinates frame. A point cloud density and grid occupancy evaluation decides the next best vantage view position as shown in Figure 6.3(b). After scanning is completed at the second vantage point the process is repeated to find the third position, and the scan taken from this position is shown in Figure 6.3(c). This process is repeated until all of the occluded areas have been filled (Figure 6.3(d)).



(a) Initialization (b) Evaluation global map and determine NBV

(c) determine NBV based on in-completed parts (d) complete mapping based on multiple scans

Figure 6.2 Complete mapping process with ROAMS color mapping robot

6.2 Exploration Strategy for Mapping

In this section a multiple step algorithm is presented to generate complete maps. The strategy includes the use of discrete exploration grid, the scan processing algorithm, scan registration, robot navigability analysis, preferred heading generation, identification of optimal vantage position at frontier, and path planning to that position.

- ***Exploration grid***: The notion of exploration grid was presented in Chapter 3. The purpose of exploration grid is to identify the broad area of interest and to discretize the space to enable classification. The space is classified into mapped and unmapped regions.

The mapped region is further classified into obstacles and navigable ground cells. The exploration grid is a 2D surface grid, and for each grid cell a *Z*-height distribution describes the elevation of points.

- ***Scan processing algorithms***: Several scan processing algorithms were described in Chapter 3. These allow determination and analysis of voxel occupancy sub-sampling, noise reduction, and navigability analysis of the scan points. The occlusion detection is accomplished based on scan pattern analysis.
- ***Scan Registration***: Since the robot has no positioning information, the generated scans must be registered into global exploration grid. The required rigid transformations are computed based on H-EGI(coarse) and H-ICP(fine) registration methods. Chapters 4 and 5 described these algorithms in detail.
- ***Robot navigability analysis***: The robot navigability analysis entails robot characteristics with point distributions in a cell. The result is the classification of the cell either as navigable or as having obstacles.
- **Preferred heading generation:** A novel force based technique has been developed to generate a preferred heading after each scan. Each unexplored cell contributes an attractive force based on its distance to the current position. While each cell with a wall or obstacle generates a repulsive force, the preferred heading direction is determined by a weighted vector sum of individual forces. Preferred heading direction is interpreted in a fuzzy fashion and a zone is selected for the optimal vantage position search (Figure 6.9).
- **Optimal vantage position identification**: In the search zone, frontier cells that are traversable are identified. A cost function is used to select optimal vantage position. Since the next vantage position stays with current scan, the overlap between two maps is guaranteed. However, two more candidate positions are marked for eventualities that

require back tracking. Back tracking may be required when the scan generated from the current vantage position does not yield a significant addition to the exploration grid. If the new scan adds sufficiently to the occupancy of exploration grid, the next preferred heading is computed.

- ***Path Planning:*** A known path planning algorithm such as the A* algorithm is invoked to navigate the robot to the next optimal vantage position.

The work flow for the exploration algorithm is shown in Figure 6.3. The process begins with initializing the exploration grid and classifying all cells as unmapped ***U***. Next, a scan is generated at its current location and projected on the existing global exploration grid. Elevation distribution is computed for each cell with point occupancy. The mapped areas are classified as obstacles/walls ***W*** and ground ***G***. The current cell of the mapping robot is marked as ***I***. Preferred heading direction is computed and candidate vantage positions are generated. A cost function is evaluated to determine the optimal vantage position and path planning is invoked. For subsequent scans, scan registration with H-EGI and H-ICP is required to transform the scan from local space to global exploration grid. Before the preferred heading directions are computed, we analyze if backtracking is necessary. This process is continued until all cells are classified as mapped, or there is no navigable path to the zone determined by preferred heading direction.

6.3 Exploration Boundaries and Discretization of Space

The path planning algorithm for exploration is driven by unexplored area within a given boundary. An exploration occupancy grid is initialized before exploration and mapping processes start. Once the point cloud generated at the latest vantage position is registered into the global frame,

the exploration occupancy grid is updated so that next vantage position for mapping can be determined. The exploration occupancy grid is defined by a 2D occupancy grid on a horizontal plane with an integration of elevation point distribution description. Though the occlusion detection algorithm for single point clouds mentioned in a previous chapter helps to identify occlusion boundaries in 3D space, it is difficult to apply the same algorithm in the global frame because of its limitation in local coordinates and the size of point cloud data. The point distribution at elevation direction can be computed and updated in a global frame without significant computation payload increase. Occlusion and traversable area can also be identified based on the distribution pattern. The evaluation process could be completed on the same occupancy grid until all of the exploration and mapping steps are accomplished.

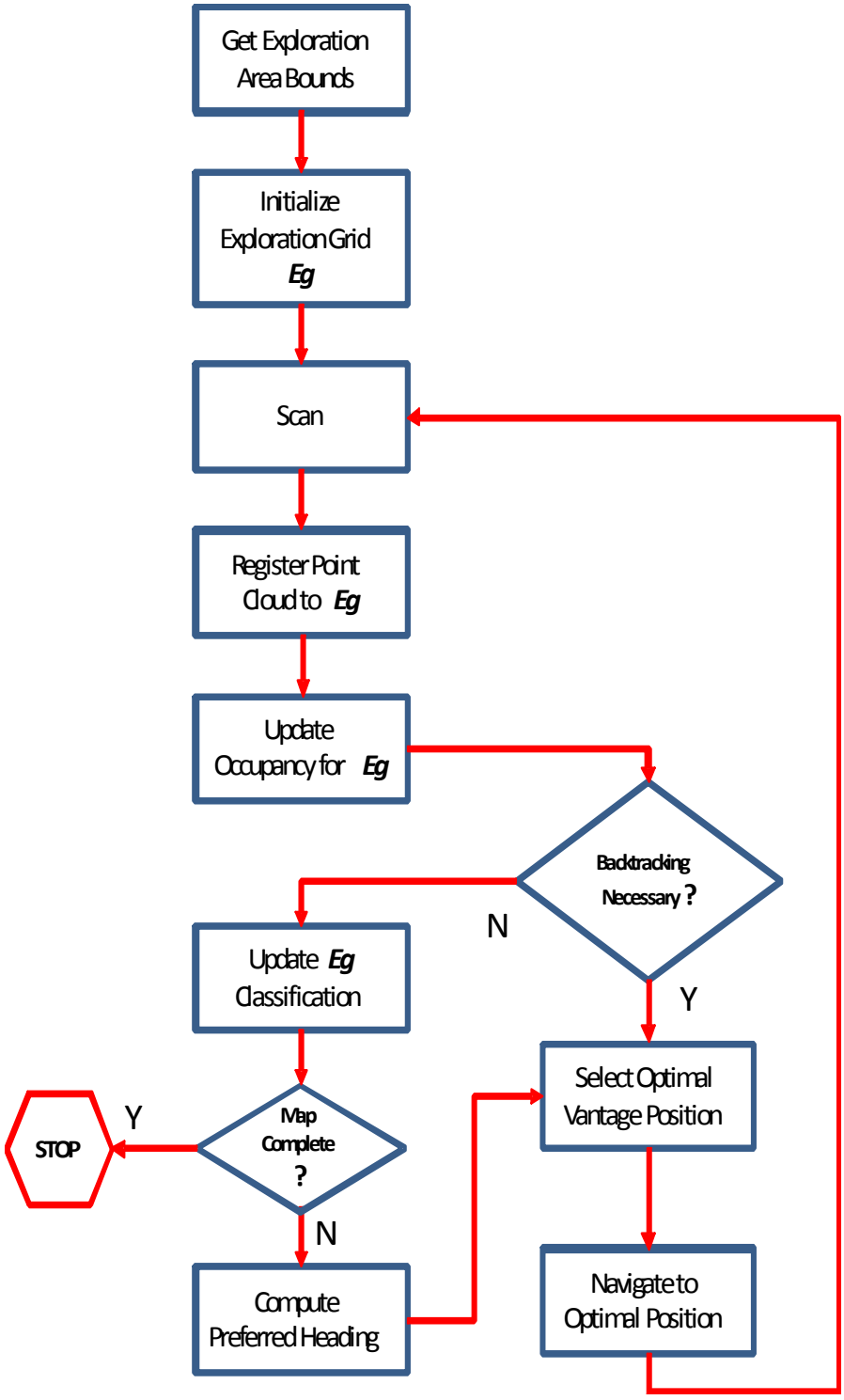


Figure 6.3 Flow chart for the exploration and mapping strategy

Initialization

The exploration occupancy grid ***Eg*** is defined by the boundary of the given area, which is defined as $\boldsymbol{B}\{x_{max},x_{min},y_{max},y_{min},z_{max},z_{min}\}$. Chapter 3 introduced the algorithm for exploration occupancy grid construction. In the exploration tasks resolution of the gird is defined by moving step size $\boldsymbol{d_m}$ of the robotic mapping platform, and the boundary is given before exploration starts. The point elevation distribution on every grid is predefined with $\boldsymbol{N_z}$ levels. The exploration occupancy grid can be initialized in global space after the destination area is given. Usually the first vantage observation generated point cloud belongs to global space. The exploration occupancy grid updates by computing point elevation distribution on newly observed point clouds and mapping them on initialized grids.

The initlization of the exploration occupancy grid is defined as:

1. Load $\boldsymbol{B}\{x_{max},x_{min},y_{max},y_{min},z_{max},z_{min}\}$ and $\boldsymbol{d_m}$;
2. Compute exploration occupancy grid size:

   $\boldsymbol{N_x}=(\boldsymbol{x_{max}}-\boldsymbol{x_{min}})/\boldsymbol{d_m}$;

   $\boldsymbol{N_y}=(\boldsymbol{y_{max}}-\boldsymbol{y_{min}})/\boldsymbol{d_m}$;
3. Compute dz by $\boldsymbol{d_z}=(\boldsymbol{z_{max}}-\boldsymbol{z_{min}})/\boldsymbol{N_z}$;
4. ***for*** *i =0 to* $\boldsymbol{N}$, $\boldsymbol{N}=\boldsymbol{N_x}$ x $\boldsymbol{N_y}$ x $\boldsymbol{N_z}$

   Eg[i]=0;

   end for

Once the exploration occupancy grid ***Eg*** is initialized, every point cloud is going to be computed and projected on the grid with elevation distribution. For a mapping task, the first vantage position generates point cloud such as in Figure 6.4. The point cloud can be projected on exploration an occupancy grid as shown in Figure 6.5.

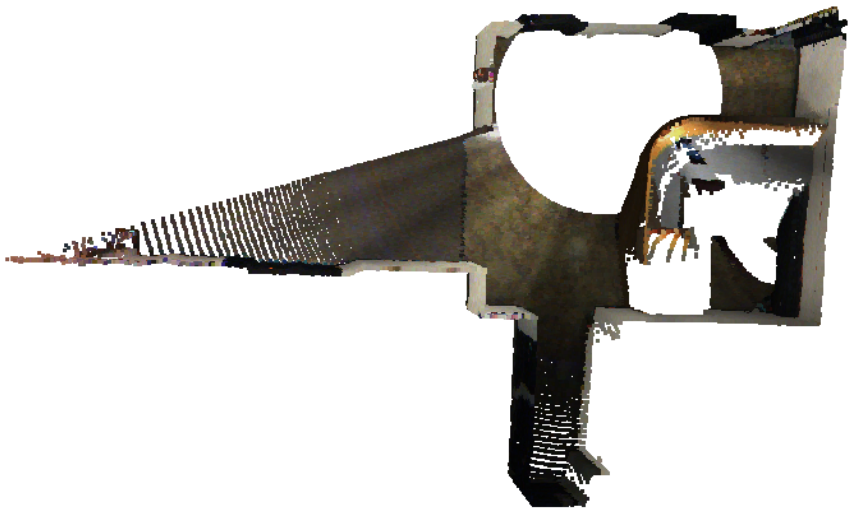


Figure 6.4 Point cloud generated at the first vantage position

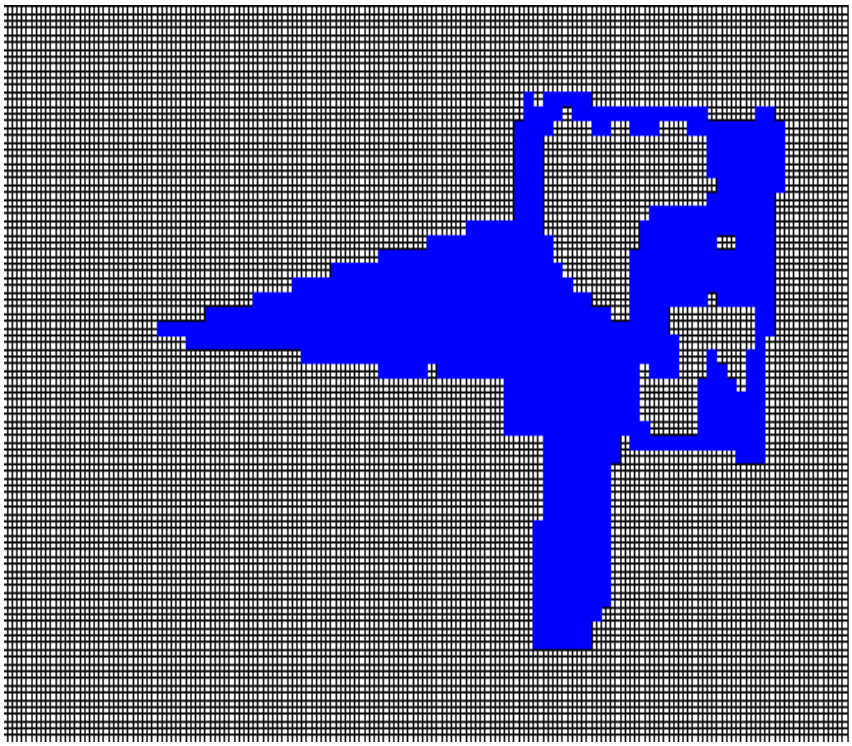


Figure 6.5 Point cloud projected on global exploration occupancy grid. All cells with sampled data are identified in blue color. Unexplored and occulded areas are unfilled.

Update

Once the point clouds are registrated into global space, the exploration occupancy grid should be updated by:

for i^{th} point $\boldsymbol{p_i}\{x_i,y_i,z_i\}$

$index\_x=(x_i-x_{min})/d_m;$

$index\_y=(y_i-y_{min})/d_m;$

$index\_z=(zi-zmin)/d_z;$

Eg$[index\_x \times N_y \times N_z+index\_y \times N_z+index\_z]$++;

End for

The exploration should update itself as soon as new point clouds are generated at the latest vantage positions. In this case, new color point clouds are created and registered with the first point cloud, shown in Figure 6.6. The exploration occupancy grid can be updated based on the algorithms shown above. The size of the exploration occupancy grid is fixed and defined before the exploration task starts, therefore the compuation cost for updating will not incrementally increase. The updated exploration occupancy grid map is shown in Figure 6.7.

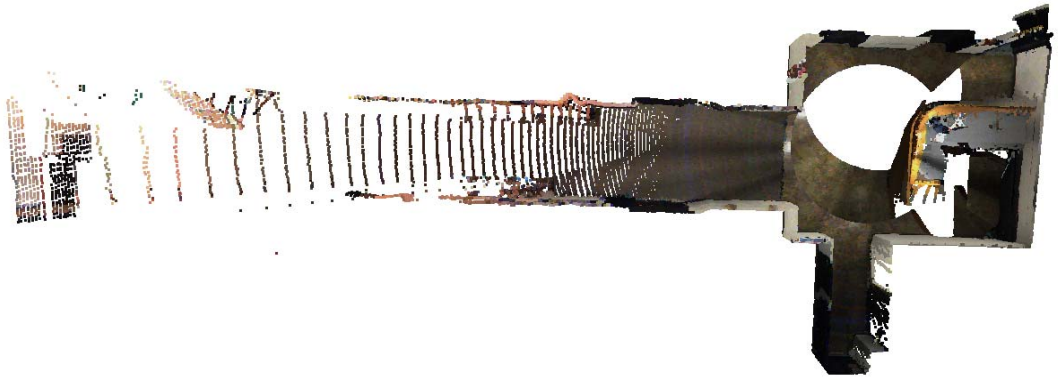


Figure 6.6 Updated color point cloud registered into global space

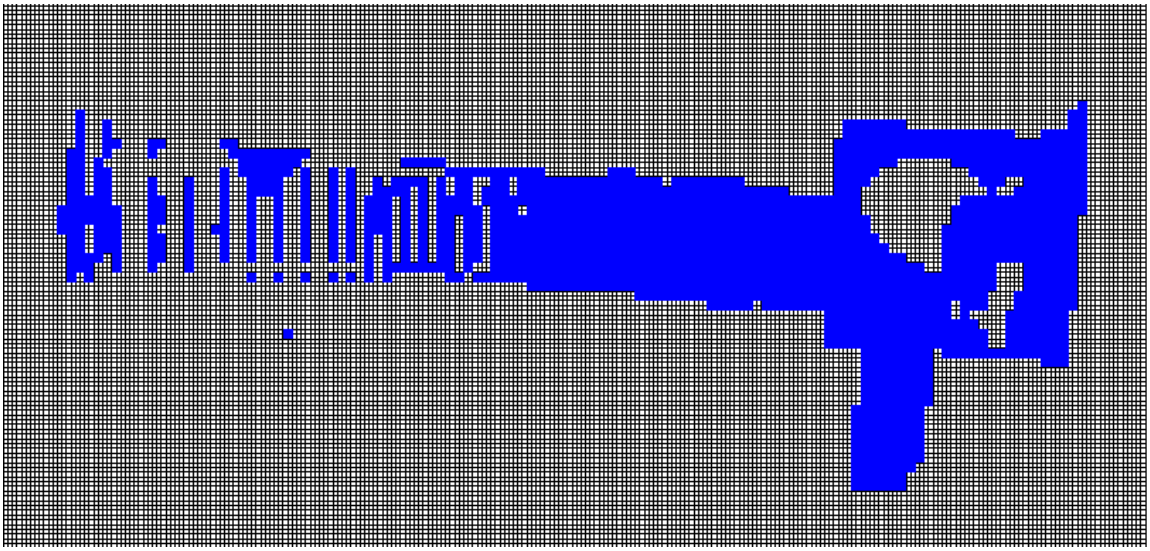


Figure 6.7 Updated exploration occupancy grid with two scans projected on to it. The heterogeneous distribution of points can be noted.

The exploration occupancy grid provides a fundermental framwork to determine obstacles and traversable ground surfaces for exploration. The map evaluation process can also be accomplished based on this framework.

6.4 Robot Navigability Analysis

The exploration occupancy grid needs to be processed to provide useful information for mapping robot navigation. The elevation distribution on each grid can be utilized to distinguish traversable ground and obstacles. The criteria is defined based on the point density. The points are evenly distributed after noise filtering and voxel grid based de-sampling. According to the algorithms discussed in Chapter 3, the ground surface should have point distribution at -1.6m. Point distribution above an elevation of -1.3m is regarded obstacle surface and not suitable for travelling. The first vantage stop generated exploration occupancy grid can then be processed and is shown in Figure 6.8. In this figure, yellow grids are not mapped area. Red grids are obstacle

surfaces while green grids are ground surfaces that the mapping robot can travel through to complete the mission.

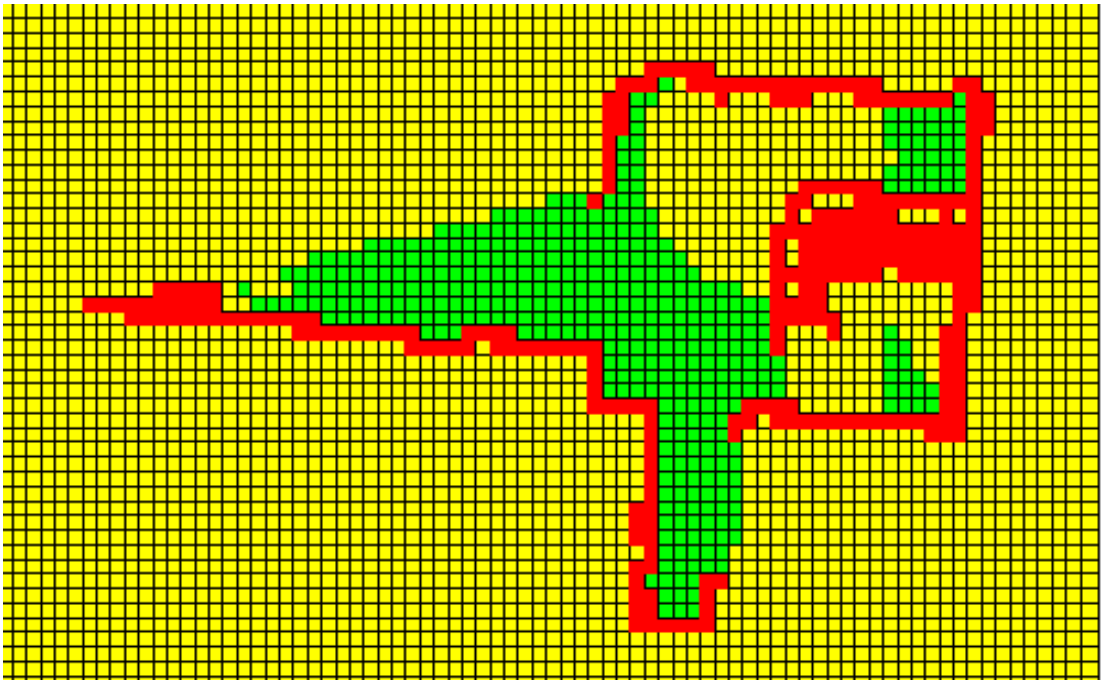


Figure 6.8 Classifed exploration occupancy grid for exploration and mapping

The explioration occupancy grid map can be udpated with the latest generated data, based on the criteria of:

$$O = U \cup O \cup G \tag{6.1}$$

$$G = U \cup G \not\subset O \tag{6.2}$$

In this, U is un-mapped area, G is ground surface, and O is obstacle surface. Obstacle is identifed once there is point detected above certain elevation level. Ground surface is the rest of the occupied grid except for obstacles. The rest of the grid is marked as un-mapped . The exploration

occupancy grid created by the first point cloud is updated with other scan data as shown in Figure 6.9, as they become available.

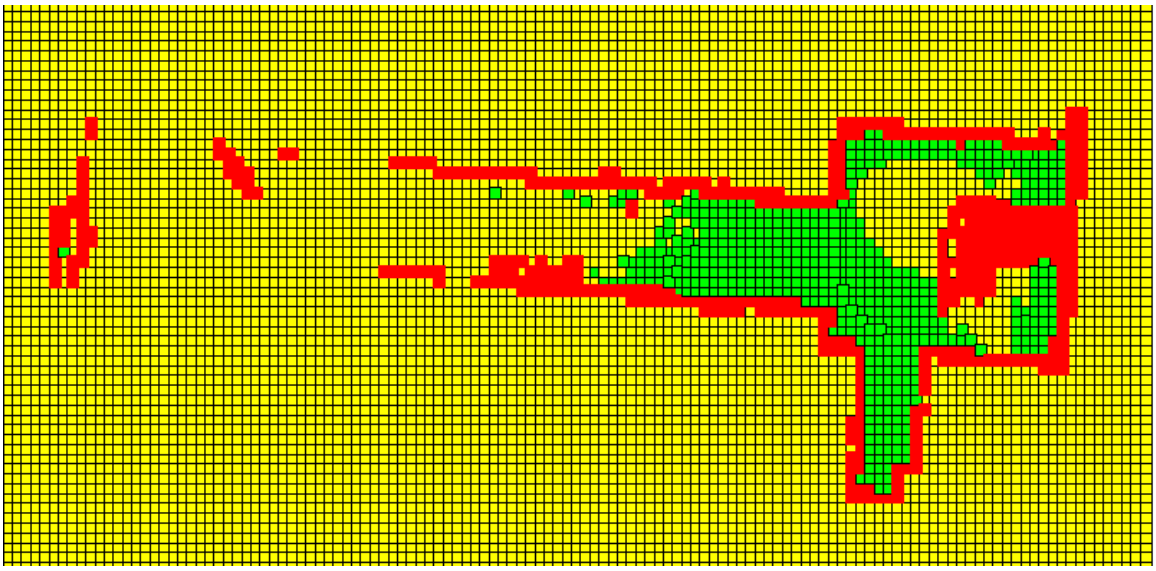


Figure 6.9 Updated exploration grid with navigable areas marked in green. The cells that are frontiers are marked as larger squares.

6.5 Preferred Heading Direction

In this section we determine a preferred heading direction $\boldsymbol{H}$ based on the information available and locations of unexplored area. As in a greedy method, the unexplored space farthest from current position has the highest attraction. However, walls and obstacles must repel the robot in order to have a clear unobstructed area around the robot. We determine the preferred heading direction as a weighted sum of scale vector contributions from every cell of the exploration grid. Eq(6.3) shows the composition of the heading vector. The summation in the first term of Eq. 6.3 describes the attraction to unmapped cells while the second summation is for walled (un-navigable occupied) cells. The weight of $\boldsymbol{W_u}$ and $\boldsymbol{W_w}$ can be manipulated to control the

contribution of attractive and repulsive forces. The scale factors $\boldsymbol{K_{kI}}$ and $\boldsymbol{K_{lI}}$ are determined using quadratic attraction/repulsive functions as given in Eq 6.4 and Eq. 6.5. The major diagonal of exploration grid as defined in Eq. 6.6 is denoted by $\boldsymbol{D}$.

$$\overrightarrow{H} = W_u \sum_{k=1}^{N_u} K_{kI} \left(\frac{\overrightarrow{x_k} - \overrightarrow{x_i}}{\left\| \overrightarrow{x_k} - \overrightarrow{x_i} \right\|} \right) - W_w \sum_{l=1}^{N_w} K_{lI} \left(\frac{\overrightarrow{x_l} - \overrightarrow{x_i}}{\left\| \overrightarrow{x_l} - \overrightarrow{x_i} \right\|} \right) \tag{6.3}$$

$$K_{kI} = \frac{1}{D^2} \left\| \overrightarrow{x_k} - \overrightarrow{x_i} \right\|^2 \tag{6.4}$$

$$K_{lI} = \frac{D^2}{\left\| \overrightarrow{x_l} - \overrightarrow{x_i} \right\|^2} \tag{6.5}$$

$$D = \sqrt{(x_{\max} - x_{\min})^2 + (y_{\max} - y_{\min})^2} \tag{6.6}$$

The attraction scale factor $\boldsymbol{K_{kI}}$ increases with the distance between unmapped cells and current location. The repulsive scale factor $\boldsymbol{K_{lI}}$ is high for walls that are close by current location. Therefore, close-by walls steer the robot away from them. This is particularly useful for hallways and close spaces. In many cases when mapping robot is closed to the obstacle, a tangent force is applied while repulsive force and attraction force are on opposite direction. The tangent force is defined as:

$$\overrightarrow{F_t} = \overrightarrow{F_w} \times \overrightarrow{Z} \tag{6.7}$$

In which $\boldsymbol{Z}$ ={0 0 1}. Tangent force $\boldsymbol{F_t}$ would help robot get out from the concave environment.

6.6 Candidate Position Generation

The candidate positions are generated based on two steps. First, principal mapping direction has to be determined based on the un-mapped area. The mapping orientated will bring robot to un-

mapped spaces with the best direction selection. The second step is the frontier position generation, which selects the top 10 positions with the farthest distance from its current vantage position. The determination of best view position will be passed to next step.

The mapping robot position on exploration occupancy grid map is surrounded by 8 neighbor grids. There are 8 potential principle directions for next movement. The 8 principle directions can be described in polar coordinates, $\{\ _1,\ _2,\ _3,\ _4,\ _5,\ _6,\ _7,\ _8\}$.shown in Figure 6.10. The unmapped area on each principle zone in global exploration occupancy grid can be computed, and the principle direction with highest number of unmapped grids should be the principle moving direction for mapping, shown in Figure 6.11. Candidate vantage positions are then generated on the frontier shown in Figure 6.12.

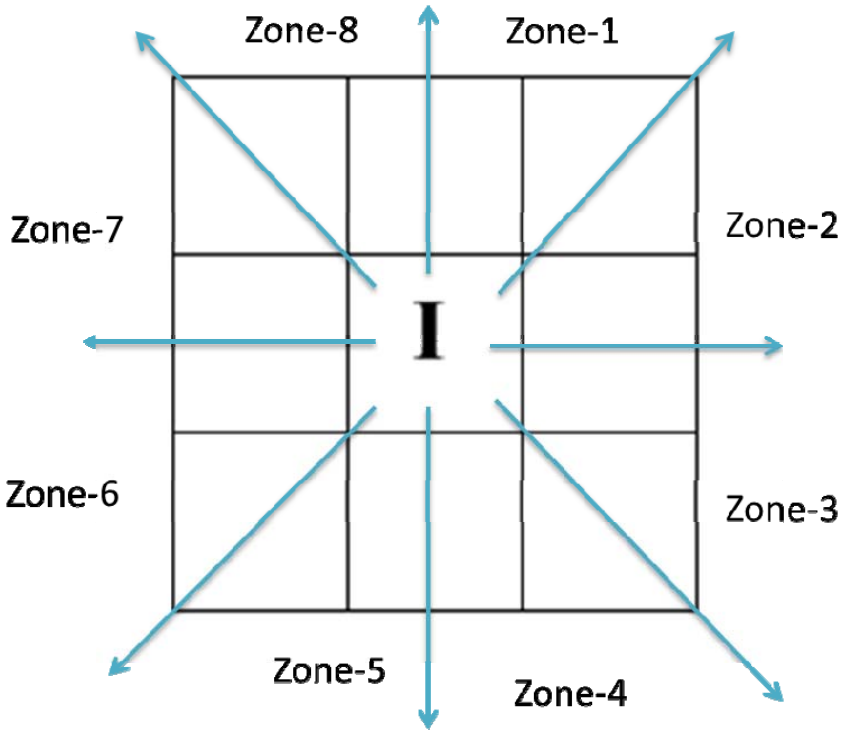


Figure 6.10 Eight possible heading (discrete) directions and search zones for optimal vantage points from the current position (I) mapping robot current position

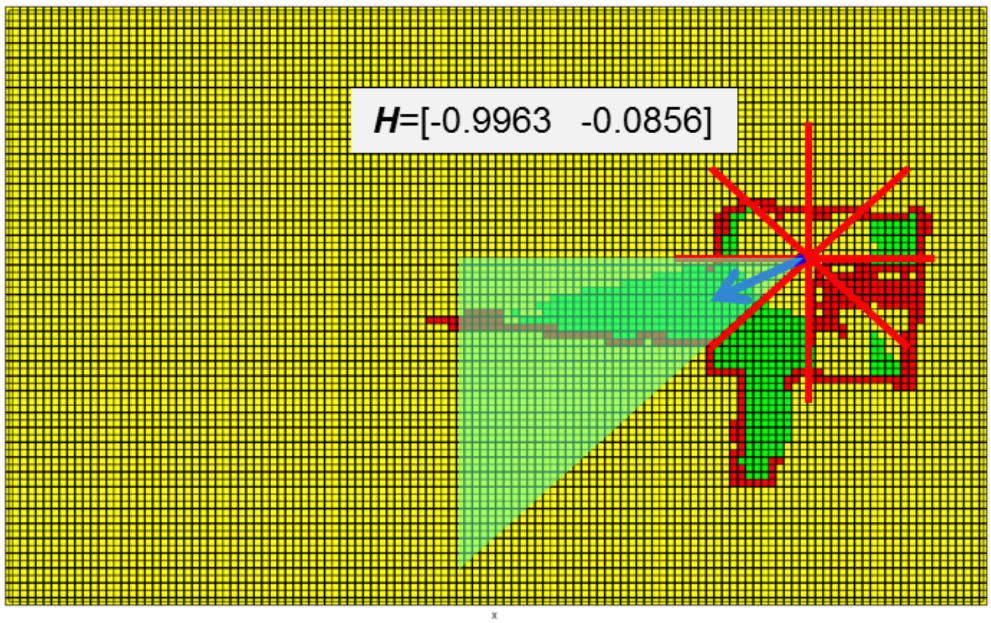


Figure 6.11 Figure illustrates the search in zone-6 and the possible frontier cells that can provide the optimal vantage positions.

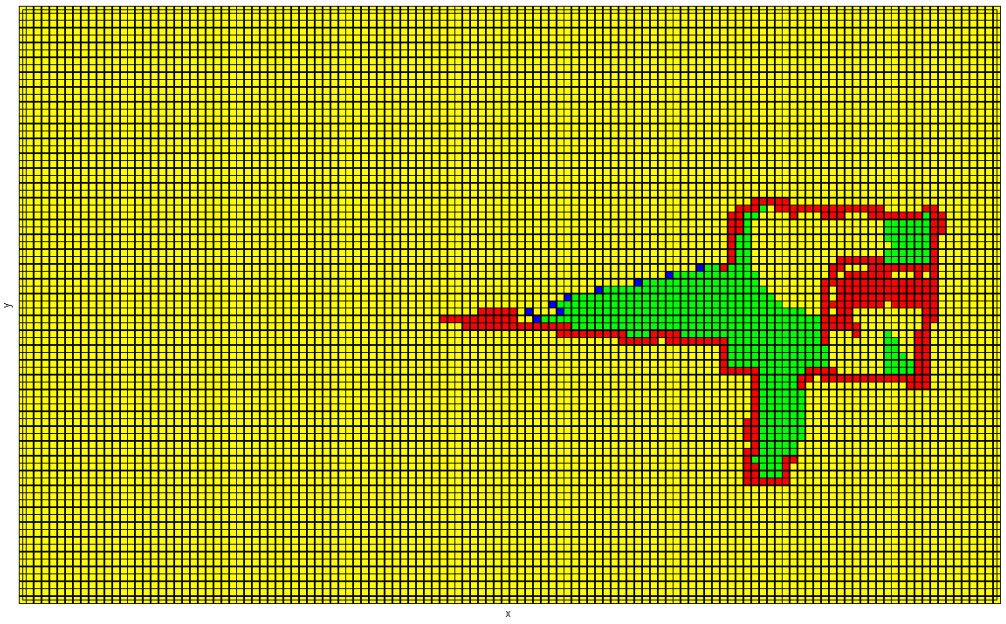


Figure 6.12 Candidate vantage positions generated on the frontier mapping area.

The distance between all ground surface grids to the current vantage position grid I are computed according to the frontier exploration strategy. The distance computation is accomplished based on the A* algorithm.

6.7 Next Vantage Position Determination

Efficient exploration strategy has to be applied for path planning during the mapping process. The perimeter observed in the map covered from a single position is taken to be a. Given the range r of the laser range scanner, approximately considered $a = 2\pi r$. Each scan is comprised of a set of points that represent the outline of the world's objects within the range r of the laser range scanner. A single scanned point cloud map includes blind zones and occlusions which need to be filled by point cloud maps from other observation vantage positions. The whole exploration process is desired to accomplish with minimal travel costs, and also need to ensure complete map generation. Therefore, the robot observation vantage position for Next Best View (NBV) needs to be determined within the currently known free space based on the principles illustrated in this section.

The frontier based exploration strategy for the next best view mapping is usually applied on grid maps [67, 89]. The objective of the frontier based exploration strategy is to navigate the mapping robot to frontiers between explored and unexplored cells. With known perceptive mapping area constrains from the map evaluation process, the robot is expected to travel the shortest trajectory to the frontier and acquire the most information about the unexplored space during the next observation. However, this observation position on the frontier is constrained by covering occlusions in the current observed point cloud map. Points of the current scene in a 3D occupancy

grid are utilized for map coverage evaluation. The next best view position is determined close to the frontier, but must cover most occlusions. Once the next vantage position is determined and all objects in the current scene are explored, a path planning algorithm can be applied to generate a path for the robot to move to the next vantage position.

A* algorithm searches all possible paths and travel costs on trajectory nodes on the initial map, from destination position (next best view position) to current position, and selects the path with minimum cost. A* algorithm is robust because travel costs on each node toward the destination are pre-computed and only related nodes will update once the environmental configuration is changed.

In order to satisfy the requirement for complete mapping, the vantage position determination process need to ensure enough overlap between point clouds generated at one vantage position and previous position.

The next vantage position determination process is divided into two steps. First the direction from candidate grid towards current grid should be computed and compared with principle mapping direction. The candidate grid that has the least offset with mapping principle direction is selected. A path from the current vantage position towards the candidate position can be created by A* shortest path planning algorithm. Every grid on the generated path is saved for evaluation. The mapping robot has a coverage range of 8 meters. To ensure enough map overlap for map registration, the overlap radio needs to meet the 50% requirement. The farthest grid on the generated path that has 50% map overlap with the previous map is selected as the next best view vantage position for exploration and mapping as shown in Figure 6.13. The potential coverage area on the next vantage position can be evaluated based on current mapping information, shown in Figure 6.14.

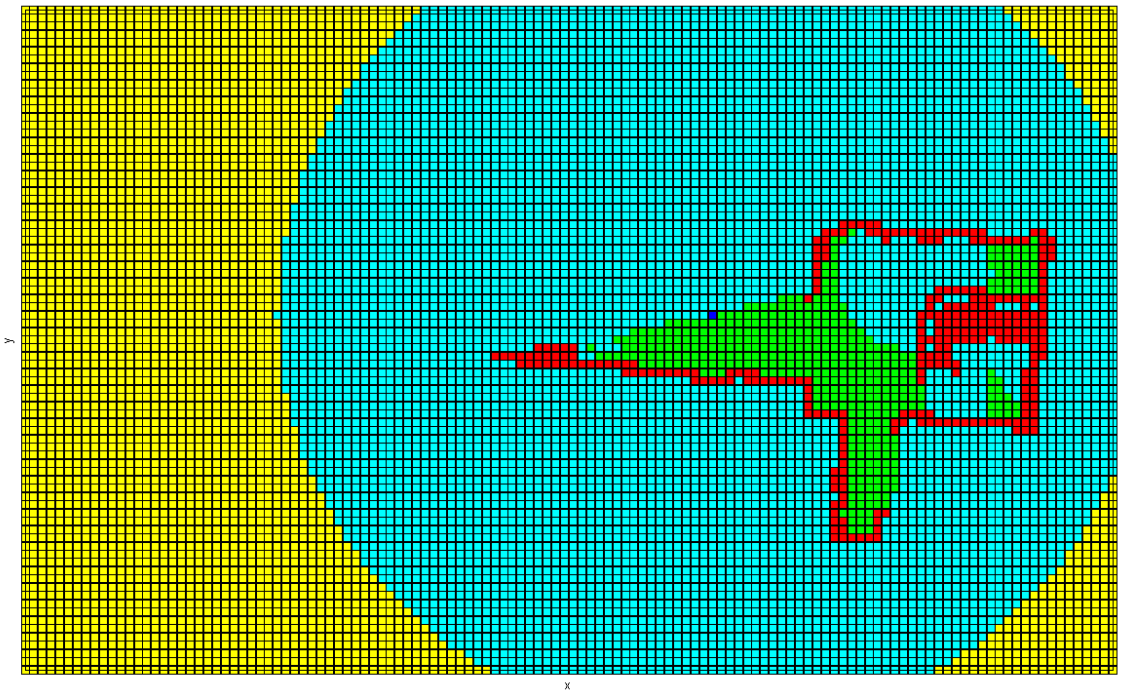


Figure 6.13 Next mapping position determination

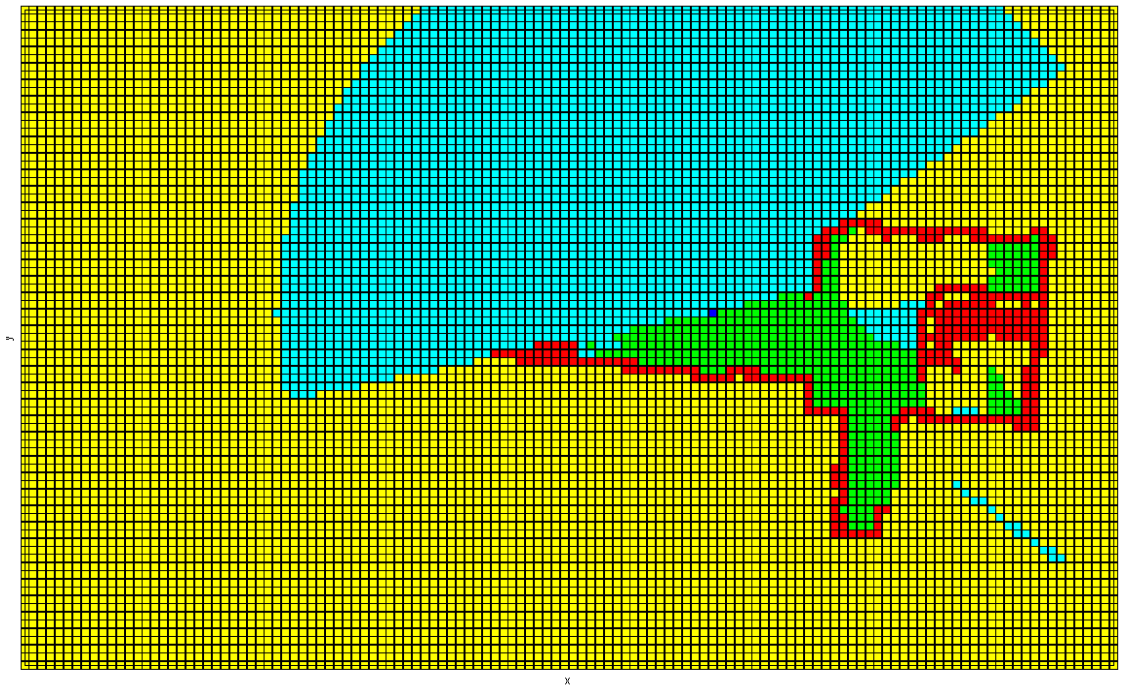


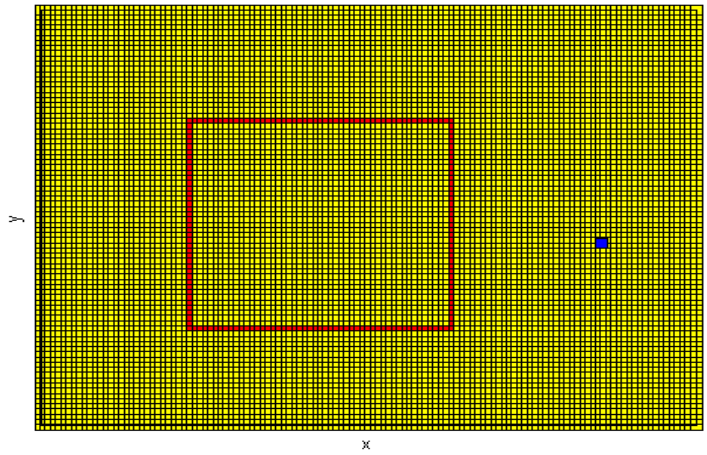
Figure 6.14 Potential mapping coverage area based on current mapping data

In order to complete the autonomous exploration for mapping missions, efficient path planning algorithms are required due to limited map information. The described path planning algorithms rely on the exploration occupancy grid to determine the next vantage position, as no map information exists beyond the frontier of the current scan. As the objective for path planning in this case is to generate a series of vantage positions to completely map all surfaces in a given space, the problem is different from traditional navigation missions where the objective to move from one place to other.

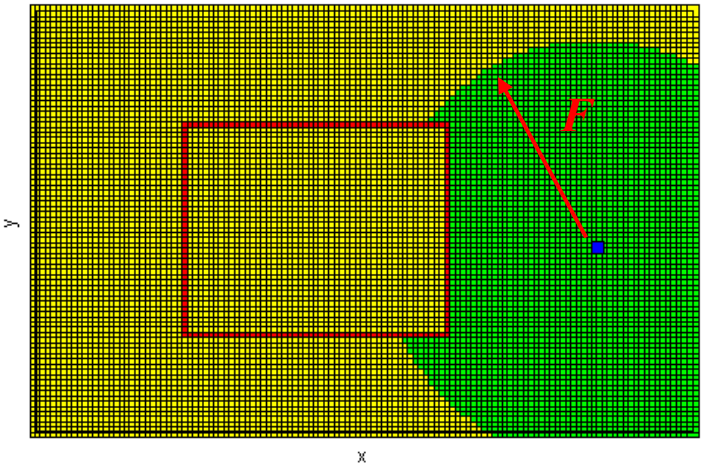
6.8 Simulation and Experimental Results

Simulation Results

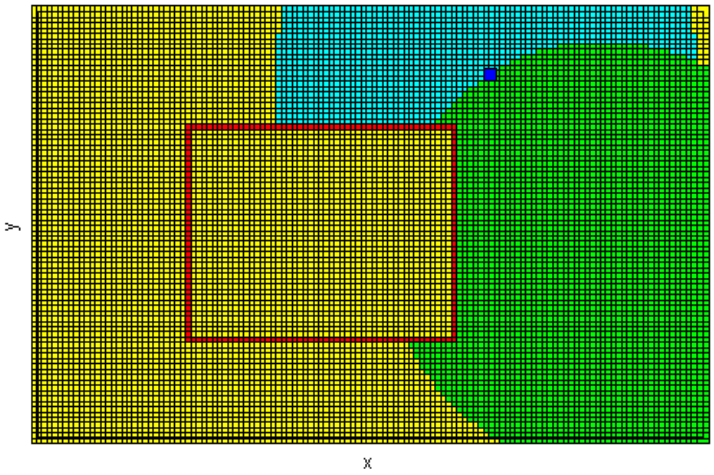
Three different scenes are generated to evaluate the performance of path planning strategy. Mapping robot is placed in a closed connected wall scene, a concave scene and a convex scene, shown in Figure 6.15, Figure 6.16 and Figure 6.17 respectively. Robot is marked as blue block, unmapped area is marked as yellow color and walls are marked as red. When robot is close towards wall, repulsive and attraction force are on opposite direction at the same time, tangent force $\boldsymbol{F_t}$ can be utilized based on Eq.(6.7). Heading direction is determined based on tangent force, attraction force from unmapped area and repulsive force from the obstacle, shown in Figure 6.15(b), Figure 6.16(b) and Figure 6.17(b) respectively. On the heading direction, observation vantage position can be determined and shown in Figure 6.15(c), Figure 6.16(c) and Figure 6.17(c). Potential mapping area can be evaluated based on current map and mapping sensor coverage, marked as cyan in respective figures. The simulation results prove that the heading direction determination process could sufficiently guide mapping robot to frontiers and keep the exploration process continues.



(a) Mapping robot in a close connected wall scene

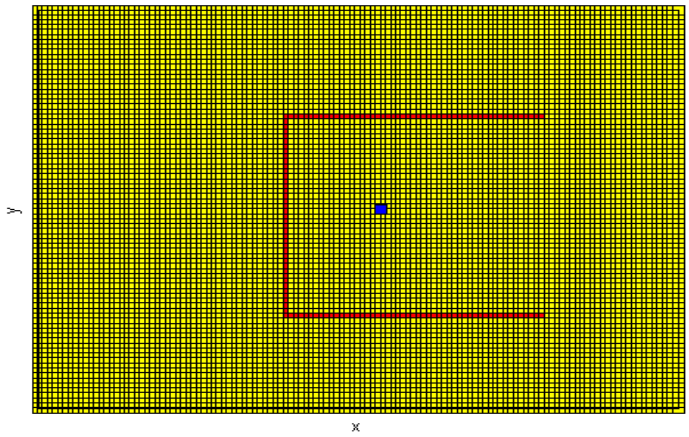


(b) Mapped area at before path planning

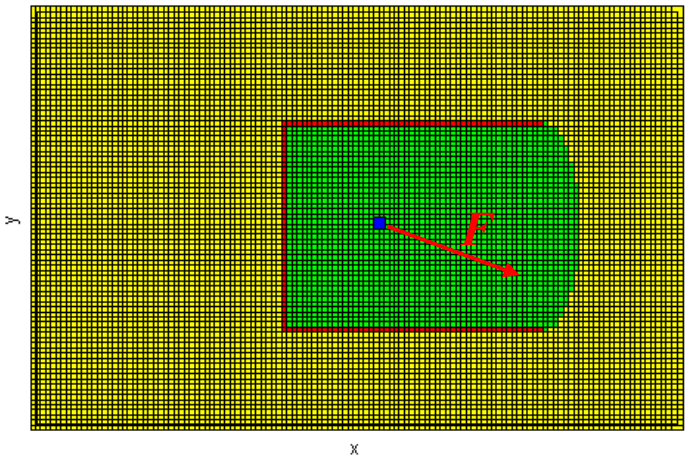


(c) Potential mapping area based on the frontier exploration strategy

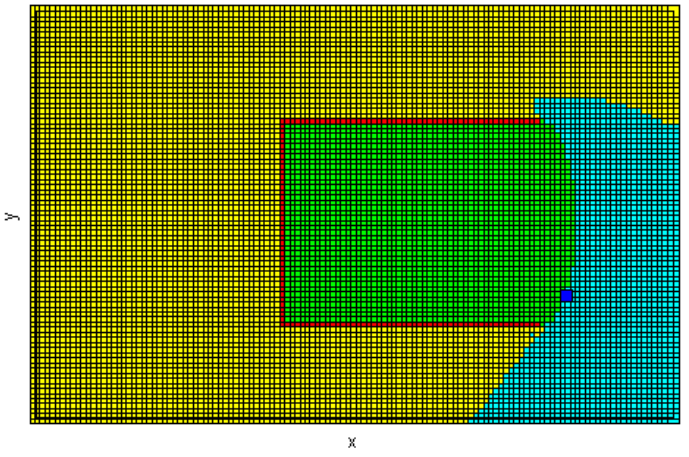
Figure 6.15 Frontier exploration strategy in closed connected wall environment



(a) Mapping robot in concave scene

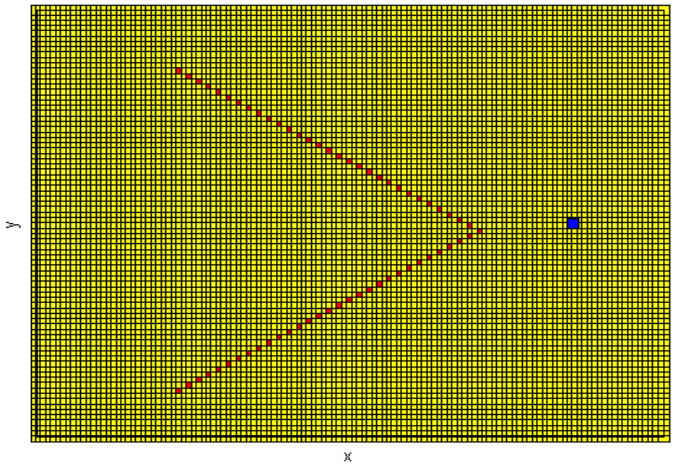


(b) Mapped area at before path planning

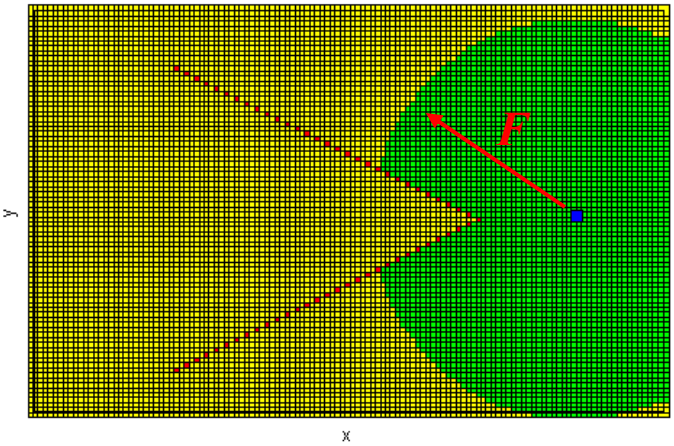


(c) Potential mapping area based on the frontier exploration strategy

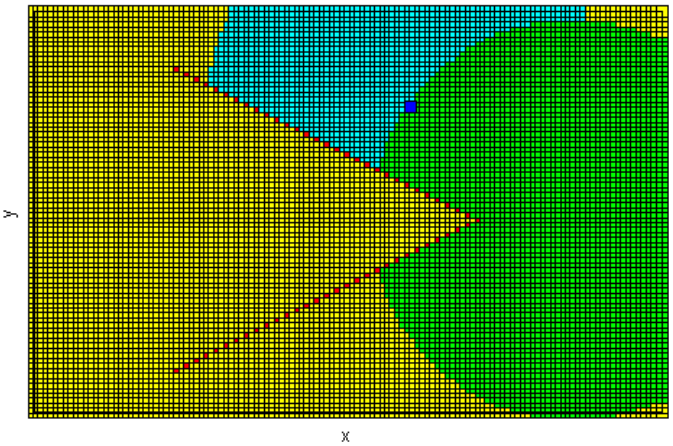
Figure 6.16 Frontier exploration strategy in concave environment



(a) Mapping robot in convex scene



(b) Mapped area at before path planning



(c) Potential mapping area based on the frontier exploration strategy

Figure 6.17 Frontier exploration strategy in convex environment

Experimental Results

Observation vantage positions are generated sequentially based on the frontier exploration strategy. Point cloud generated at most recent vantage position is registered into previous point clouds in global frame. The exploration grid is then updated with most recent point cloud data. The exploration and mapping process following Figure 6.3 assist mapping robot successfully travels through the hallway and updated occupancy grid at every vantage position is shown in Figure 6.18.

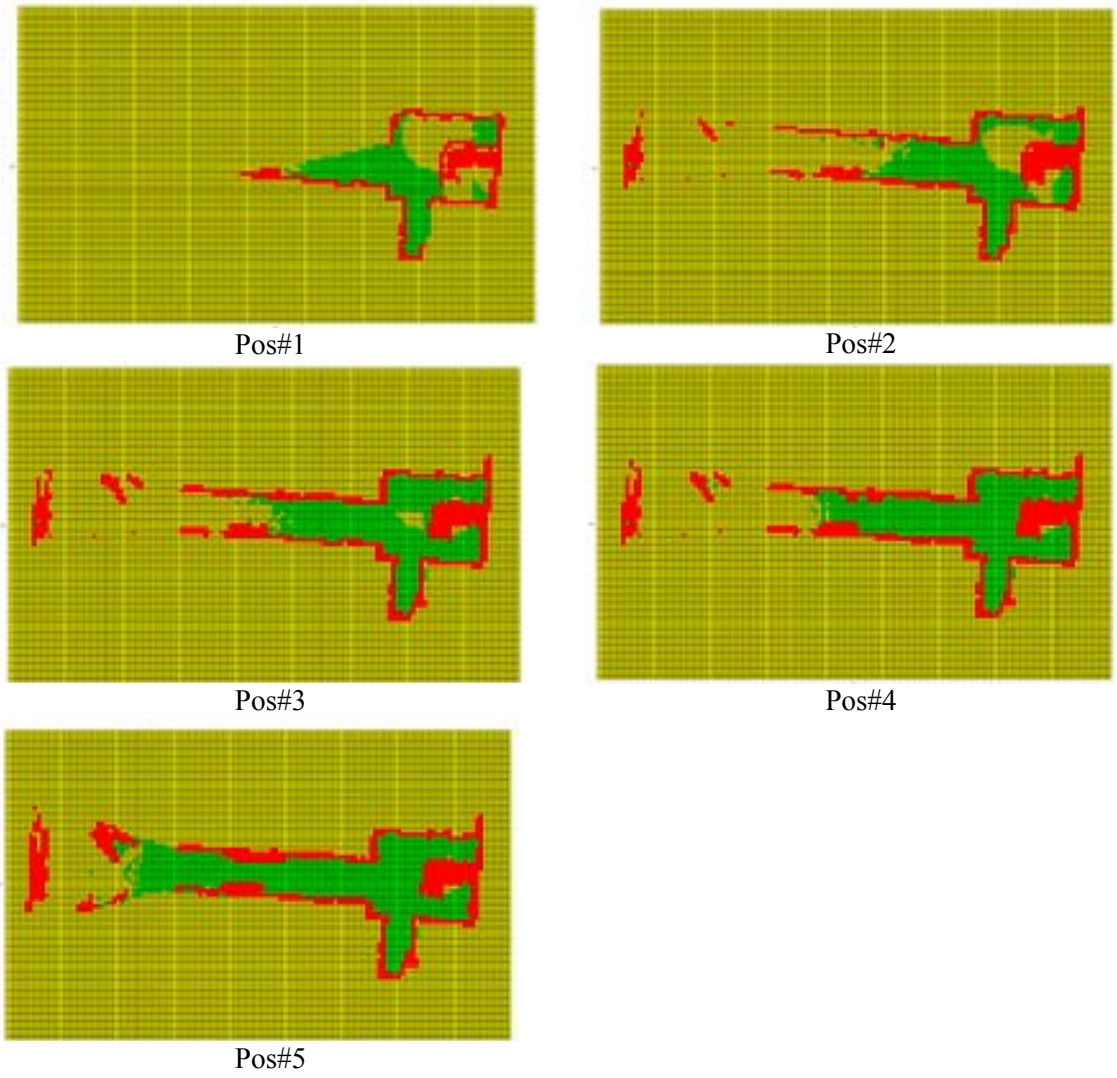


Figure 6.18 Exploration and mapping results in Carnegie building hallway

6.9 Concluding Remarks

An exploration strategy for mapping is described in this chapter. The point cloud map created at different vantage positions has been projected into the global exploration grid. Analysis is completed on the exploration occupancy grid to distinguish ground surface for movement, obstacles to avoid and un-mapped areas to cover. The exploration strategy is determined based on selecting the vantage candidate positions generated on the frontier based exploration algorithm. A cost function includes travel cost, and coverage has been proposed for the vantage position determination process.

Chapter 7 Conclusions, Contributions and Future Work

7.1 Concluding Remarks

The main objective of this research is to develop robotic systems and algorithms to perform autonomous and complete mapping in uncharted areas. As precise and dynamic position as well as attitude information is quite expensive, both computationally and cost-wise, a scan-stop-scan mapping methodology was developed without the need for any positioning information. Once a map is generated the robot navigation in the explored areas can be performed with well-known SLAM algorithms, again without the need for precise position information. The contributions of this thesis are summarized as follows:

1. **Design and Prototyping the Robotic Mapping System**

   A mobile robotic platform was designed and built with a four-wheel driven, all-terrain vehicle body, drive by wire system, electrical power supply, management module, and sensor and communication modules. A LIDAR is installed on a 3DOF rotational platform on top of the robotic body frame. A 3D accurate point cloud map is created by rotating the LIDAR 2D scanning plane, Points on the 2D LIDAR scanning plane are projected into 3D space with the rotational angles provided by potential meters mounted with the rotation platform. Two color cameras are installed on the 2D scanning plane and rotating together with LIDAR. Camera images and 3D point clouds are precisely merged together by calibrating the relative position between camera and LIDAR.

   The mobile robotic mapping system could be able to remotely or autonomously operate dependeing on the positioning sensors, communication modules, drive by wire system

and other environmental perception sensors. This robotic mapping system is able to generate rendered color point clouds of the environment, which provide not only distance information but also object surface textures in environments. Due to the coverage limitations of the mapping sensor, including intrinsic blind zones from LIDAR and the occlusions from the scene, the mobile mapping system is able travel through urban environments at different vantage positions to complete the exploration and mapping tasks.

2. Point Cloud Processing

Point clouds created by the robotic mapping system have been processed to detect occlusions in 3D space based on LIDAR scanning pattern. The point cloud is analyzed in polar coordinate space and the occlusion boundary in 3D can be identified by the depth difference. The points on the object surface have depth continuity while the depth on occlusion edges is not. The occlusion detection in 3D space is critical for surface reconstruction and navigation.

The concept of the exploration occupancy grid is introduced so that point clouds can be project onto it. The size of the exploration occupancy with given area is limited and much smaller than the size of the point cloud itself. Therefore the computation cost for exploration occupancy grid update is constant and will not increase. The exploration occupancy grid contains occupancy signals with point distribution at elevation direction on each grid. Point clouds generated at each vantage position can be projected on the grid and updated after the mapping. Objects between ground surface and obstacle surface can be distinguished by comparing the elevation distribution pattern on each grid.

3. High Performance Map Registration

Both coarse and fine registration algorithms for color point cloud registrations have been introduced and discussed. Color attributes on points have been utilized to improve the performance of current 3D point cloud registration algorithms. Color model has been studied and hue from HSL space has been selected to assist in the point cloud registration process. A case study was provided to prove that under different lighting conditions hue value remains constant while RGB value varies significantly. Therefore, hue attribute computed from the color property of point cloud can be utilized to improve the registration process.

The coarse registration does not require pre-alignment from the positioning sensor, which is usually provided by GPS but not available in indoor conditions. The point surface normal vectors have been computed and projected on an Extended Gaussian Image. The coarse registration is accomplished by solving the rigid rotation after spherical correlation between multiple EGIs. Rigid translation is solved afterwards by computing the correlation of discrete occupancy grid function in Fourier domain. The voxel grid de-sampling algorithm has been introduced to ensure even point distribution in 3D space. The hue attribute has been adapted as a filter to reduce noise and computation cost. The color assisted coarse registration algorithm has provided a shorter time cost and higher accuracy than the registration algorithm in 3D point clouds.

The Iterative Closest Point algorithm has been discussed as the fine registration for point clouds after coarse registration. Points between different cloud areas are associated as pairs based on the principle of nearest neighbor search in 3D space. Point clouds area iteratively registered together at the end. This algorithm requires a pre-alignment, which

can be provided by coarse registration steps. Hue value has been weighted and combined together with 3D coordinates during nearest neighbor association process. The weight of hue is critical for the HICP algorithm in fine registration and the impact of scaling for the hue-dimension was studied. Experimental results prove that hue could assist current 3D point cloud registration techniques to gain a better performance in terms of time cost and accuracy.

4. **Path Planning for complete exploration**

   Point clouds are processed after map registration. The obstacle surface and ground surface identified from previous steps are utilized for path planning. The path planning is determined by frontier-based approaches, which have the next best potential view for mapping. The path planning results are computed based on an exploration occupancy grid which contains both exploration history and terrain surface information so that reversibility can be decided. The next vantage position is also constrained by scanning range and overlaping areas with previously acquired point clouds so that map registration can be performed.

7.2 Suggestions for future work

7.2.1. Hybrid Image-Point Cloud Techniques for Up-scaling Resolution

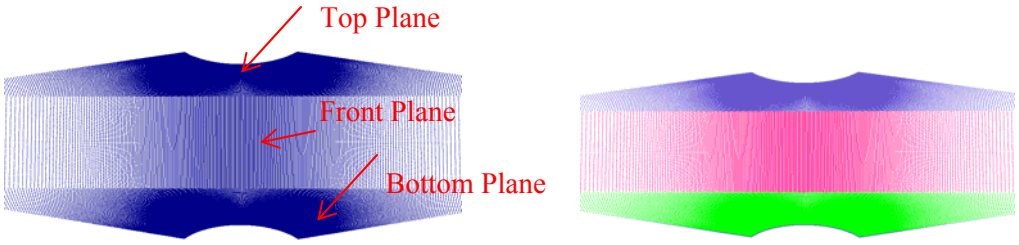
Most modern LIDAR scanners have associated camera imagery. Panoramic images generated during the mapping process can be utilized for overlap area estimation and for up-scaling the resolution of the point clouds. A Scale Invariant Feature Transform (SIFT) algorithm can be applied to find associated feature points between images and the point clouds – cross-registering the two domains. This allows functional mapping of coordinates onto high resolution images.

Considering a typical 12 Mega pixel camera produces 4000 pixel wide images and the lidar used produces only 401, considerable opportunities exist for up-scaling the point cloud resolution. The images can also be used to smooth out the areas of discontinuities in the point clouds.

7.2.2. Surface Segmentation and Real-time Boundary Detection

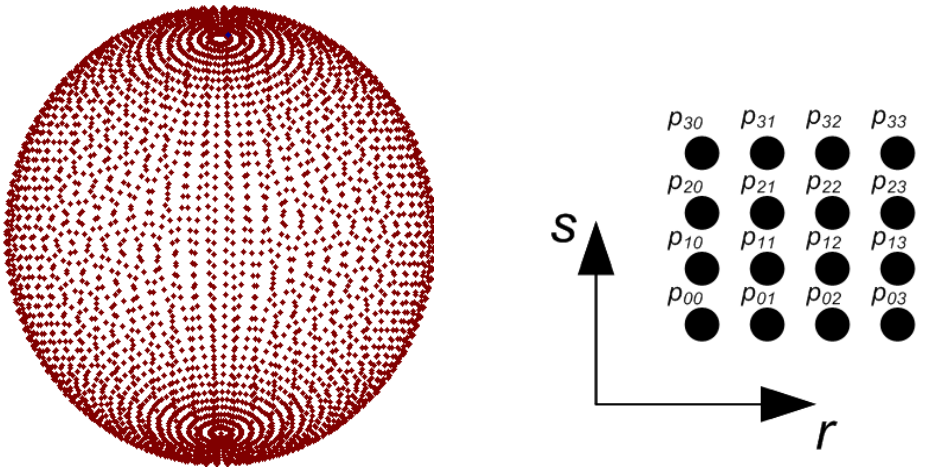
Higher order geometric representation of objects and surfaces are required for the construction of CAD surfaces and solid models. Fitted surface models can then be utilized for semantic maps and labeling. In order to reduce the burden of surface identification we again use an analysis of distortion of scan patterns. The absolute orientation angle of an object surface patch is known about reference coordinate system.

Consider the point cloud of an interior scene shown in Figure 7.2. The figure shows two planes (floor and ceiling) with a convex wall. Using a surface orientation based algorithm and plotting the angle between the surface normal and the incident ray of the LIDAR, the local surface orientation angles for every point in the scan are obtained. The angle is color contoured to classify the scene into various planes and other forms of geometric surfaces. Figure 7.2(a) shows the classification along the N-S (variation in mirror angle). The figure shows that both the top and bottom planes are classified accurately. The curved front surface can also be identified, but it requires further processing before its surface can be recognized. Figure 7.2(b) shows the classification about the scanner head (East-West) direction and once again, it can be seen that both the planes and the surface can be identified in the scene.



(a) Point cloud includes 3 different planes (b) Identified planes marked by unique color

Figure 7.1 Object surface plane identification in point cloud



(a) Scanning pattern of 3D LIDAR (b) 4x4 matrix for surface identification

Figure 7.2 Three dimensional LIDAR scanning pattern for surface identification

The 3D LIDAR scan pattern forms a sphere of discrete points as shown in Figure 7.2. The resolution or point spacing is defined by the speed of the scanning mirror and head rotation. When the scanning pattern shown in Figure 7.3(a) is projected onto object surfaces with different curvatures and orientations, the distortion in the pattern can be used to determine the geometric characteristics of the surface. Figure 7.3(b) shows the 4x4 grid of 16 points patch. For the 4x4 points patch, a bi-cubic surface patch can be fitted uniquely. Any point $\boldsymbol{p}(r, s)$ on this patch can be calculated by its parametric coordinates (r, s). A surface net can be fitted and smoothed for the 16 points as shown in Figure 7.3.

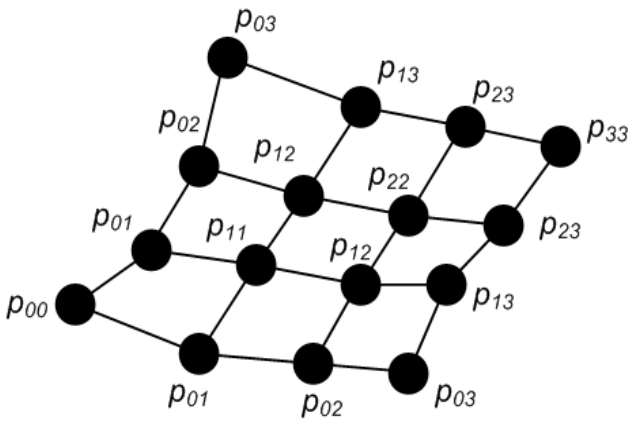


Figure 7.3 Deformed scanning patch on object surface

Using a bi-cubic representation with the algebraic form of the patch defined as follows:

$$\begin{aligned} p(r,s) = \quad & a_{33}r^3s^3 + a_{32}r^3s^2 + a_{31}r^3s \;\; + a_{30}r^3 \\ & + a_{23}r^2s^3 + a_{22}r^2s^2 + a_{21}r^2s \;\; + a_{20}r^2 \\ & + a_{13}rs^3 + a_{12}rs^2 + a_{11}rs \;\; + a_{10}r \\ & + a_{03}s^3 + a_{02}s^2 + a_{01}s \;\; + a_{00} \end{aligned} \tag{7.2}$$

In which r and s are the normalized parametric coordinates from 0 to 1, a_{ij} are the coefficients. In matrix form, Eq(7.2) becomes

$$\boldsymbol{P=EA} \tag{7.3}$$

$$[P]_{1\times 3} = [E]_{1\times 16}[A]_{6\times 3}$$

$$\begin{bmatrix} P_x & P_y & P_z \end{bmatrix} = \begin{bmatrix} r^3s^3 & r^3s^2 & r^3s & r^3 & r^2s^3 & r^2s^2 & r^2s & r^2 & rs^3 & rs^2 & rs & r & s^3 & s^2 & s & 1 \end{bmatrix}.$$
$$\begin{bmatrix} a_{33}^x & a_{32}^x & a_{31}^x & a_{30}^x & a_{23}^x & a_{22}^x & a_{21}^x & a_{20}^x & a_{13}^x & a_{12}^x & a_{11}^x & a_{10}^x & a_{03}^x & a_{02}^x & a_{01}^x & a_{00}^x \\ a_{33}^y & a_{32}^y & a_{31}^y & a_{30}^y & a_{23}^y & a_{22}^y & a_{21}^y & a_{20}^y & a_{13}^y & a_{12}^y & a_{11}^y & a_{10}^y & a_{03}^y & a_{02}^y & a_{01}^y & a_{00}^y \\ a_{33}^z & a_{32}^z & a_{31}^z & a_{30}^z & a_{23}^z & a_{22}^z & a_{21}^z & a_{20}^z & a_{13}^z & a_{12}^z & a_{11}^z & a_{10}^z & a_{03}^z & a_{02}^z & a_{01}^z & a_{00}^z \end{bmatrix}^T$$

In which $\boldsymbol{A}$ is the 16x3 vector of coefficients a_{ij}, $\boldsymbol{E}$ is a 1x16 matrix of rs product defined by the path formation, and $\boldsymbol{P}$ is a 1x3 coordinates of given data points $\boldsymbol{p}$. $\boldsymbol{E}$ and $\boldsymbol{P}$ are known from scan

data and $\boldsymbol{A}$ is solved from Eq. (7.3). The coefficient vector **A** represents the attribute of the patch geometry. $\boldsymbol{A}$ vector on different patches can be used to classify geometry into plane, convex, concave quadratic, and higher order geometrics. Surfaces in the 3D scene can be classified by swiping this 4x4 point patch along all possible positions. Points with similar $\boldsymbol{A}$ attributes are categorized together and fitted into a surface pattern. The extracted geometric surfaces will be applied on higher level CAD model construction and extraction of contours for object map completion.

Surface Identification: Both scanning pattern deformation and global orientation angle are applied to detect surfaces in point cloud. Points on the same surface are identified to fit the surface model. The output from this part could reduce computation costs for point cloud processing and accelerate the 3D model reconstruction process.

7.2.3. Exploration Trajectory Optimization

The completeness of exploration and mapping results can be conducted to evaluate the performance of path planning strategy and mapping results. Global optimized strategy can be implemented based on current occlusion driven exploration strategy. The global path planning should consider occlusion, unmappable areas and exploration costs. Moreover, the blind zone from scanning bind spot and overlap area requirements for map registration should be considered in this specific situation.

7.3 Last Words

Construction of 3D models from point clouds requires new methodologies for point cloud processing. If the scanners generate and segment higher order geometric entities on site, in real-time and with embedded hardware, CAD model generation will be much simpler. We see these advancements in digital cameras, it will only be a matter of time before the 3D scanners catch up.

References

1. Thrun, S., *Robotic Mapping: A Survey*, in *Exploring Artificial Intelligence in the New Millennium*. 2002, Morgan Kaufmann.
2. Thrun, S., *A Probabilistic online mapping algorithm for teams of mobile robots.* The International Journal of Robotic Research, 2001. **20**: p. 335-363.
3. Thrun, S., D. Hahnel, and D. Fergnuson, *A system for Volumetric Robotic Mapping of Abandoned Mines*, in *IEEE International Conference on Robotics and Automation*. 2003: Taipei, Taiwan.
4. Choi, Y.-W., et al., *Three-Dimensional LiDAR Data Classifying to Extract Road Point in Urban Area.* IEEE Geoscience and Remote Sensing Letter, 2008. **5**(4): p. 725-729.
5. Lefsky, M.A., et al., *Lidar Remote Sensing for Ecosystem Studies.* BioScience, 2002. **52**(1): p. 19-30.
6. Streutker, D.R. and N.F. Glenn, *LIDAR Measurement of Sagebrush Steppe Vegetation Heights.* Remote Sensing of Environment, 2006. **102**: p. 135-145.
7. Nüchter, A., *3D Robotic Mapping*. Vol. 52. 2009: Springer Tracts in Advanced Robotics.
8. Cherkaeva, E. and A.C. Tripp, *Optimal survey design using focused resistivity arrays.* IEEE Transaction on Geoscience and Remote Sensing, 1996. **34**(2): p. 358-366.
9. Chin, J.L., F.L. Wong, and P.R. Carlson, *Shifting Shoals and Shattered Rocks-How Man Has Transformed the Floor of West-Central San Franscisco Bay*. 2004, U. S. Geological Survey.
10. Fong, S., et al., *Robotic Site Survey at Haughton Creater*, in *9th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS)*. 2008: Los Angeles, CA.

11. Pavlovskaya, M.E., *Mapping Urban Change and Changing GIS: Other views of economic restructuring.* Gender, Place and Culture, 2002: p. 281-289.

12. Veen, A.V.D., et al., *Mapping urban air pollution using GIS: a regression-based approach.* International Journal of Geographical Information Science, 1997. **11**(7): p. 699-718.

13. Biswas, R., et al., *Towards Object Mapping in Dynamic Environments With Mobile Robots*, in *IEEE International Conference on Intelligent Robots and Systems*. 2002: Lausanne, Switzerland.

14. Chatila, R. and J. Laumond, *Position referencing and consistent world modeling for mobile robots*, in *IEEE International Conference on Robotics and Automation*. 1985.

15. Erdody, T.L. and L.M. Moskal, *Fusion of LIDAR and Imagery for Estimating Forest Canopy Fuels.* Remote Sensing of Environment, 2010. **114**(4): p. 725-737.

16. Liu, Z., et al., *Validating Lidar depolarization calibration using solar radiation scattered by iced clouds.* IEEE Geoscience and Remote Sensing Letters, 2004. **1**(3): p. 157-161.

17. Mutlu, M. and S.C. Popescu, *Mapping Surface Full Models Using LIDAR and Multispectral Data Fusion for Fire Behavior.* Remote Sensing of Environment, 2008. **112**(1): p. 274-285.

18. Lee, H.S. and N.H. Younan, *DTM Extraction of LIDAR Returns via Adaptive Processing.* IEEE Transaction on Geoscience and Remote Sensing, 2003. **41**(9): p. 2063-2069.

19. Li, Y. and E. Olson, *Extracting general-purpose features from LIDAR data*, in *IEEE International Conference on Robotics and Automation*. 2010: Anchorage, Alaska.

20. Costa, B.M., T.A. Battista, and S.J. Pittman, *Comparative Evaluation of Airborne LIDAR and Ship-based Multibeam SoNAR Bathymetry and Intensity for Mapping Coral Reef Ecosystems.* Remote Sensing of Environment, 2008. **113**(5): p. 1082-1100.

21. Thrun, S., W. Burgard, and D. Fox, *A Real –Time Algorithm for Mobile Robot Mapping with Applications to Multi-Robot and 3D Mapping*, in *IEEE International Conference on Robotics and Automation*. 2000: San Francisco. p. 321-328.

22. Thrun, S., W. Burgard, and D. Fox, *A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots.* Machine Learning and Autonomous Robots, 1998. **31**(5): p. 1-25.

23. Cheng, Y.Q., et al., *Three dimensional reconstruction of points and lines with unknow correspondence across images.* International Journal of Computer Vision, 2000. **45**(2): p. 355-367.

24. Stachniss, C., G. Grisetti, and W. Burgard, *Information Gain-based Exploration using Rao-Blackwellized Particle Filters*, in *Robotics: Science and Systems*. 2005: Cambridge, MA, USA.

25. Burgard, W., et al., *Coordinated Multi-Robot Exploration.* IEEE Transaction on Robotics, 2005. **21**(3): p. 376-387.

26. Joho, D., C. Stachniss, and P. Pfaff, *Autonomous Exploration for 3D Map Learning*, in *Autonomous Mobile Systems*, K. Berns and T. Luksch, Editors. 2007, Springer: Kaiserslautern, Germany.

27. Kümmerle, R., et al., *Monte Carlo Localization in Outdoor Terrains Using Multi-Level Surface Maps*, in *International Conference on Field and Service Robotics*. 2007: Chamonix Mont-Blanc, France.

28. Thrun, S., et al., *Robust Monte Carlo Localization for Mobile Robots*. 2000, School of Computer Science, Carnegie Mellon University.

29. Castellanos, J.A., et al., *The SPmap: A probabilistic framework for simultaneous localization and map building.* IEEE Transaction on Robotics and Automation, 1999. **15**(5): p. 948-952.

30. Castellanos, J.A. and J.D. Tardos, *Mobile Robot Localization and map building: A multisensory fusion approach*. 2000: Springer.

31. Men, H. and K. Pochiraju, *Color Assisted Iterative Closest Point (ICP) Algorithm*, in *The 2nd International Conference on Computer and Automation Engineering*. 2010: Singapore.

32. Strom, J., A. Richardson, and E. Olson, *Graph-based Segmentation for Colored 3D Laser Point Clouds*, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2010: Taipei.

33. Choset, H., *Sensor based Motion Planning: The Hierarchical Generalized Voronoi Graph*. 1996, California Institute of Technology.

34. Kuwata, Y., et al., *Real Time Motion Planning with Applications to Autonomous Urban Driving.* IEEE Transaction on Control Systems Technology, 2009. **7**(5): p. 1105-1118.

35. You, S. and J. Hu, *Statement of Requirements for Urban Search and Rescue Robot Performance Standards*. 2005, National Institute of Standards and Technology.

36. Sujan, V.A. and S. Dubowsky, *Efficient Information based Visual Robotic Mapping in Unstructured Environments.* International Journal of Robotics Research, 2005. **24**(4): p. 275-293.

37. Thayer, S., et al., *Distributed robotic mapping of extreme environments*, in *SPIE: Mobile Robots XV and Telemanipulator and Telepresence Technologies VII*. 2000.

38. Adams, M.D., *Coaxial range measurement- current trends for mobile robotic applications.* IEEE Sensors Journal, 2002. **2**(1): p. 2-13.

39. Hu, X., D.F. Alarcon, and T. Gustavi, *Sensor based navigation coordination for mobile robots*, in *42nd IEEE Conference on Decision and Control*. 2003: Maui, HI.

40. Johnson, A., et al., *Lidar-based hazard avoidance for safe landing on Mars*. Vol. 25. 2002, Reston, VA, ETATS-UNIS: American Institute of Aeronautics and Astronautics. 9.

41. Li, Y. and E. Olson, *A General Purpose Feature Extractor for Light Detection and Ranging Data.* Sensors, 2010. **10**(11): p. 10356 -10375.

42. Moraver, H.P. and D.W. Cho. *A bayesian method for certainty grids*. in *AAAI Spring Symposium on Robot Navigation*. 1989. Stanford, CA.

43. Jefferies, M.E., W.-K. Yeap, and J. Baker, *Robot Mapping with a topological map of local space representations*. 2002.

44. Szabo, R., *Topological Navigation of Simulated Robots using Occupancy Grid.* Inernational Journal of Advanced Robotic Systems, 2004. **1**(3): p. 201-206.

45. Men, H. and K. Pochiraju, *Algorithms for 3D Map Registration*, in *Depth Map and 3D Imaging Applications: Algorithms and Technologies*. 2010, Accepted, IGI-Global.

46. Bajcsy, R., et al., *3D reconstruction of environments for virtual reconstruction*, in *4th IEEE Workshop on Application of Computer Vision*. 1998: Princeton, NJ.

47. Tarabanis, K.A., P.K. Allen, and R.Y. Tsai, *A Survey of Sensor Planning in Computer Vision.* IEEE Transaction on Robotics and Automation, 1995. **11**(1): p. 86-105.

48. Gebre, B., H. Men, and K. Pochiraju, *Remotely Operated and Autonomous Mapping System (ROAMS)*, in *2009 IEEE International Conference on Technologies for Practical Robot Applications*. 2009: Woburn, MA.

49. Nüchter, A., K. Lingemann, and J. Hertzberg, *Cached k-d tree search for ICP algorithms*, in *Sixth International Conference on 3-D Digital Imaging and Modeling*. 2007: Montreal, QC.

50. Men, H. and K. Pochiraju, *Hue Assisted Registration of 3D Point Clouds*, in *ASME 2010 International Design Engineering Conferences & Computers and Information in Engineering Conference*. 2010: Montreal, Quebec, Canada.

51. Bsel, P.J., *A Method for Registration of 3D Shapes.* IEEE Trans on Pattern Analysis and Machine Intelligence, 1992. **14**: p. 239-256.

52. Jez, O., *Navigation of Mobile Robots Using 6DOF SLAM Accelerated by Leveled Maps*, in *Novel Algorithms and Techniques in Telecommunications, Automation and Industrial Electronics*, T.E. Sobh, K., A. Mahmood, and M.A. Karim, Editors. 2008, Springer.

53. Rusinkiewicz, S. and M. Levoy, *Efficient Variants of the ICP Algorithm*, in *Third International Conference on 3D Digital Imaging and Modeling (3DIM)*. 2001.

54. Langis, C., M. Greenspan, and G. Godin, *The Parallel Iterative Closest Point Algorithm*, in *Third International Conference on 3-D Digital Imaging and Modeling (3DIM '01)*. 2001: Quebec City, Canada.

55. Collier, J. and A. Ramirez-Serrano, *Environment Classification for Indoor/Outdoor Robotic Mapping*, in *Canadian Conference on Computer and Robot Vision*. 2009: Kelona, British Columbia.

56. Druon, S. and M.J.C. Aldon, A., *Color Constrained ICP for Registration of Large Unstructured 3D Color Data Sets*, in *IEEE International Conference on Information Acquisition*. 2006: Weihai, China.

57. Sirmacek, B. and C. Unsalan, *Urban area and building detection using SIFT keypoints and Graph theory.* IEEE Transaction on Geoscience and Remote Sensing, 2009. **47**(4): p. 1156-1167.

58. Montemerlo, M., et al., *Winning the DARPA Grand Challenge with an AI Robot*, in *In Proceedings of the AAAI National Conference on Artificial Intelligence* 2006. p. 17-20.

59. Thrun, S., W. Burgard, and D. Fox, *Probabilistic Robotics*. 2005, Cambridge, Massachusetts: MIT Press.

60. Montemerlo, M., et al., *FastSLAM: A factored solution to the simultaneous localization and mapping problem*, in *AAAI National Conference on Artificial Intelligence*. 2002: Edmonton, Canada.

61. Herold, M. and M.E. Gardner, *Spectral Resolution Requirements for Mapping Urban Areas.* IEEE Transaction on Geoscience and Remote Sensing, 2003. **41**(9).

62. Schempf, H., et al., *Pandora: Autonomous Urban Robotic Reconnaissance System*, in *IEEE International Conference on Robotics and Automation*. 1999: Detroit, Michigan.

63. Schneider, A., M.A. Friedl, and D.K. Mclver, *Mapping Urban Areas by Fusing Multiple Sources of Cource Resolution Remotely Sensed Data*, in *IEEE International Geoscience and Remote Remote Sensing Symposium*. 2003: Toulouse, France. p. 2623-2625.

64. Rusu, R.B., et al., *Model-based and Learned Semantic Object Labeling in 3D Point Cloud Maps of Kitchen Environments*, in *IEEE/RJS International Conference on Intelligent Robotis and Systems*. 2009: St. Louis, MO, USA.

65. Anousaki, G.C. and K.J. Kviakopoulos, *Simultaneous Localization and Map Building for Mobile Robot Navigation.* IEEE Robotics and Automation Magazine, 1999. **6**(3): p. 42-53.

66. Baumgartner, E.T., P.S. Schenker, and C. Leger, *Sensor Fuzed Navigation and Manipulation from a Planetary Rover*, in *Symposium on Sensor Fusion and Decentralized Control in Robotic Systems*. 1998: Boston.

67. Elfes, A., *Using Occupancy Grids for Mobile Robot Perception and Navigation.* Computer, 1989. **22**(6): p. 46-57.

68. Thrun, S., *Probabilistic Robotics.* Communication of the ACM, 2002. **45**(3): p. 52-57.

69. Stachniss, C., *Robotic Mapping and Exploration*. Springer Trats in Advanced Robotics. Vol. 52. 2009.

70. Yamauchi, B., *Frontier-Based Exploration Using Multiple Robots*, in *The Second Inernational Conference on Autonomous Agents*. 1998: Minneapolis,MN.

71. Elfes, A., *Occupancy Grids: A probabilistic Framework for Robot Perception and Navigation*. 1989, Carnegie Mellon University.

72. Thrun, S., *Exploration and model building in mobile robot domains*, in *IEEE International Conference on Neural Networks*. 1993: San Francisco, CA, USA.

73. Mei, Y., et al., *Energy-efficient mobile robot exploration*, in *IEEE international conference on robotics and automation*. 2006.

74. Thrun, S., *Learning Occupancy Grid maps with forward sensor models.* Autonomous Robots, 2003. **15**(2): p. 111-127.

75. Tovery, C. and S. Koenig, *Improved Analysis of Greedy Mapping*, in *IEEE/RSJ International Conference on Intelligent Robotics and Systems*. 2003: Las Vegas, Nevada.

76. Thrun, S., Y. Liu, and A.Y. Ng, *Simultaneous Localization and Mapping with Sparse Extended Information Filters.* International Journal of Robotics Research, 2004. **23**(7): p. 693-716.

77. Lorusso, A., *A Comparison of Four Algorithms for Estimating 3D Rigid Transformations*, in *British Machine Vision Conference*. 1995.

78. Arun, K.S., *Least Square Fitting of Two 3D-Point Sets.* IEEE Transaction on Pattern Analysis and Machine Intelligence, 1987. **9**(5): p. 698-700.

79. Chua, C.J.R., *Point Signature: A New Representation for 3D Object Recognition.* International Journal of Computer Vision, 1997. **25**: p. 63-85.

80. Johnson, A., *Spin-Images: A Representation for 3D Surface Matching*. 1997, Carnegie Mellon University.

81. Chung, D. and Y.D.S. Lee, *Registration of Multiple Range Views Using the Reverse Calibration Technique.* Pattern Recognition, 1998. **31**(4): p. 457-464.

82. Tarel, J., H. Civi, and D. Cooper, *Pose Estimation of Free-form 3D Objects without Point Matching using a Algebraic Surface Models.* Proceedings of IEEE Workshop on Model-based 3D, 1998: p. 13-21.

83. Druon, S., M.J. Aldon, and A. Crosnier, *Color Constrained ICP for Registration of Large Unstructured 3D Color Data Sets*, in *IEEE International Conference on Information Acquisition*. 2006: Weihai, China.

84. Pochiraju, K., B. Gebre, and H. Men, *Method and Apparatus for Adaptive Transmission of Sensor Data with Latency Controls, pending.* 2009: U.S.A.

85. Joung, J.H., K.H. An, and J.W. Kang, *3D Environment Reconstruction Using Modified Color ICP Algorithm by FUsion of a Camera and a 3D Laser Range Finder*, in *IEEE International Conference on Intelligent Robotics and Systems*. 2009: St. Louis, U.S.A.

86. Andreason, H. and A.J. Lilienthal, *6D Scan Registration using Depth-interpolated Local Image Features.* Robotics and Autonomous Systems, 2010. **59**: p. 157-165.

87. Desouza, G.N. and A.C. Kak, *Vision for mobile robot navigation: A survey.* IEEE Trans of Pattern Recognition, 2002. **24**(2): p. 237-267.

88. González-Banos, H.H. and J.C. Latombe, *Navigation Strategies for Exploring Indoor Environments.* International Journal on Robotic Research, 2002. **21**(10-11): p. 829-848.

89. Jakuba, M.V. and D.R. Yoerger, *Autonomous Search for Hydrothermal Vent Fields with Occupancy Grid Maps*, in *Australasian Conference on Robotics and Automation*. 2008: Canberra, Australia.

90. Saffiotti, A., *The Uses of Fuzzy Logic in Autonomous Robot Navigation*, in *Soft Computing*. 1997, Springer. p. 180-197.

91. Koenig, S., *Fast Replanning for Navigation in Unknown Terrian.* IEEE Transaction on Robotics and Automation, 2002. **1**(20).

Vita

Date of Birth: January 3, 1983
Place of Birth: Xianyang, Shaanxi Province, People's Republic of China

Education:
Stevens Institute of Technology, Hoboken, NJ
PhD in Mechanical Engineering. May 2012

Beijing University of Technology, Beijing, China
Master of Science in Mechanical Engineering, June 2006

Xi'an Jiaotong University, Xi'an, China
Bachelor of Science in Mechanical Engineering, July, 2003

Patents:
1. K, Pochiraju, Biruk Gebre, Hao Men, Method and Apparatus for Adaptive Transmission of Sensor Data with Latency Controls, pending, 2010
2. K.Pochiraju, Hao Men, Biruk Gebre, Adaptive Mechanism Control and Scanner Positioning for Improved Three Dimensional Laser Scanning, pending, 2011

Book Chapter:
1. Men, Hao and Kishore Pochiraju. "Chapter 4: Algorithms for 3D Map Segment Registration" in "Depth Map and 3D Imaging Applications: Algorithms and Technologies" IGI Global, 2012, pp. 56-86

Publications:
1. H. Men, K. Pochiraju, *Hue Assisted Registration of Color Point Clouds, Robotics and Autonomous Systems,* Submitted 2011.
2. H. Men, K. Pochiraju, Automatic Registration of Color Point Clouds with Hue-Filters and Extended Gaussian Image Correlation, IEEE Transactions of Pattern Analysis and Machine Intelligence, Submitted 2012.
3. B. Gebre, H. Men, M. Manzione, K. Kim & K. Pochiraju, A Tele-Operated Robotic Mapping System for Generation of Color Point Clouds, Journal of Intelligent & Robotic Systems, Submitted, 2012.

Conference Proceedings:
1. H. Men, B. Gebre, K. Pochiraju, *Color Point Cloud Registration with 4D ICP Algorithm*, IEEE International Conference on Robotics and Automation, Shanghai, 2011. pp1511-1516.
2. H. Men, K. Pochiraju, *Hue Assisted Registration of 3D Point Clouds*, ASME 2010 International Design Engineering Conferences & Computers and Information in Engineering Conference, Montreal, Quebec, Canada, August 15-18, 2010. pp1075-1083.

3. H. Men, K. Pochiraju, *Color Assisted Iterative Closest Point (ICP) Algorithm*, The 2nd International IEEE Conference on Computer and Automation Engineering, Singapore, Feb 26-28, 2010
4. B. Gebre, H. Men, K. Pochiraju, *Remotely Operated and Autonomous Mapping System (ROAMS),* 2009 IEEE International Conference on Technologies for Practical Robot Applications, Woburn, MA, Nov 9-10, 2009. pp173-178.
5. H. Men, K. Pochiraju, *Coupled Lateral and Lane Separation Control for 2-D Vehicle Groups*, ASME 2008 International Design Engineering Conferences & Computers and Information in Engineering Conference, New York City, U.S.A., 2008. pp809-817.

Honors:

Stevens Graduate Student Conference Travel Award	August 2010

Professional Affiliations:

American Society of Mechanical Engineers	2005~current
Institute of Electrical and Electronics Engineers	2010~current