3D MAP BUILDING OF STRUCTURED INDOOR ENVIRONMENTS WITH HETEROGENEOUS ROBOTS

by

RAVI K. KAUSHIK

A dissertation submitted to the Graduate Faculty in Computer Science in partial fulfillment for the requirements of degree of Doctor of Philosophy,

The City University of New York

2011

This manuscript has been read and accepted for the
Graduate Faculty in Computer Science in satisfaction of the
dissertation requirement for the degree of Doctor of Philosophy

Dr. Jizhong Xiao

Date

Chair of Examining Committee

Dr. Theodore Brown

Date

Executive Officer

Dr. Zhigang Zhu

Dr. Theodore Raphan

Dr. Yan Meng

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

Abstract

3D MAP BUILDING OF STRUCTURED INDOOR ENVIRONMENTS WITH HETEROGENEOUS ROBOTS

By

Ravi K. Kaushik

ADVISOR: Prof. Jizhong Xiao

We explore a methodology to construct 3D maps of indoor environments using multiple heterogeneous robots. The robots acquire overlapping 3D range images at contiguous locations and fuse them together to form a complete range map. Our methodology involves three major steps in map construction: the first step is to estimate an initial pose between the multiple robots using a real-time camera pose estimation algorithm, the second step involves extracting polygon features from overlapping range images and the third step involves registration of the overlapping range images using extracted polygonal features. We evaluate the performance of several approaches to camera pose estimation, which is used to determine the initial estimate of pose between the multiple robots. We introduce a fast and robust segmentation algorithm, which extracts the polygonal features from the 3D range images. A polygon-based registration algorithm accurately fuses the overlapping range images. The algorithm further refines the initial pose estimate by minimizing the geometric distance between corresponding polygonal features extracted from overlapping range images. Our polygon-based registration combined with camera pose estimation executes in real-time and much faster compared to point-based scan registration techniques. We present experimental results obtained while building indoor maps and quantitative analysis of the algorithms used in map construction of indoor environments.

Preface

This dissertation is an outcome of an effort in "advancing mobile robots to 3D", a project whose principal investigator is Prof. Jizhong Xiao. The research work was carried out at the CCNY Robotics Lab of the Grove School of Engineering, The City College of New York (CCNY), New York. Several colleagues of mine participated in this project. While we were developing robust algorithms to build maps of indoor corridor at the dept. of Electrical Engineering, CCNY, we faced a number of challenges related to hardware and the design of algorithms for 3D mapping. We have made every effort to present the solutions to the challenging problems in a clear and lucid manner. We developed the methodology from the scratch and laid a strong mathematical foundation underneath the algorithms. It was quite a challenge to perform experiments with multiple robots. Long and arduous preparations prior to data acquisition helped us collect accurate data from the robots. The light weight range scanners used in the experiments are considerably noisy and the perspective cameras were of-the-shelf webcams. Sensors were calibrated and pre-processed to eliminate outliers. We made many wise choices while choosing the route to design of mapping algorithms, although some avenues led us to dead-end approaches and we had to abandon them in the middle. This thesis describes the research work carried out in a comprehensive manner. Our analysis of the subject had innumerous discussions and brain-storming sessions. Lot of ideas presented here are outcomes of such discussions. We believe that this dissertation is a wealth of information for researchers interested in map construction with robots in 3D.

Ravi K. Kaushik

September 2011

Acknowledgments

I would like to thank several people who provided me the support while I carried out the dissertation work. First and foremost, I am grateful to my parents, Venkatesh Ramanna and Savithri Venkatesh for believing in me. Their support has been through tremendous perseverance and involved untold sacrifices. On an equal footing, my wife, Shruti has played a big role in my completion of dissertation work. I dedicate this thesis to them.

I would like to acknowledge Prof. Jizhong Xiao, my advisor for his support, Prof. Theodore Raphan, my mentor during early PhD years, Prof. Zhigang Zhu, Prof. Simon Parsons, Prof. Robert Haralick and others for sharing their knowledge with me. I would also like to thank Prof. Yan Meng from Stevens Institute of Technology for providing me insightful suggestions for improving my dissertation work. They have been a source of inspiration and the best mentors one could have. My experience at The Graduate Center, CUNY has been exceptionally good and would be incomplete without acknowledging my classmates Ms. Suzanne Tamang, Mr. Mo Reza Zahrani and others. My colleagues at the CCNY Robotics Lab, were great fun to work with and we shared our day-to-day research activities. Among them, I would like to acknowledge Mr. Narashiman Chakravarty, Mr. Flavio Cabrera-Mora, Mr. Angel Calle, Mr. William Morris, Dr. Xiaohai Li, Mr. Samleo L. Joseph, Mr. Rex Wong and many others who were visiting scholars during 2006-2011. I am grateful to my dear friends Arjun Hebbar, Rajesh Goyal and several others for being there when I needed the most.

The results reported in the thesis were made possible by the research grants supported by the organizations in part by the National Science Foundation and the U. S. Army Research Office.

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

We present a methodology to autonomously construct 3D maps of indoor environments using multiple heterogeneous robots. In this work, a 3D map refers to a depth/range map of the surrounding environment represented in Euclidean space. We deployed state-of-the-art heterogeneous robots equipped with sensors such as perspective camera, 3D range scanners, Time-of-Flight (ToF) depth and RGB-Depth cameras to construct 3D maps. These sensors digitize the surrounding environment by acquiring 3D depth maps along with color images at multiple viewpoints locally with respect to the *Sensor Coordinate Frame* (SCF). A complete map is obtained by transforming all the local maps with respect to a global *Reference Frame* (RF).

Current state-of-the-art map construction algorithms listed in [Nüchter 2009] generate 3D depth maps and their computational complexity is much higher compared to 2D mapping algorithms developed in earlier robotics research [Leonard and Durrant-Whyte 1991; Smith, Self et al. 1990]. In order to fuse overlapping range maps, it is required to compute the 6 Degree of Freedom (DoF) relative pose from the overlapping range images acquired by the robots at contiguous locations. The nonlinearity embedded in the Euclidean space poses greater challenges for 3D map construction and requires application of optimization problems for 6DoF pose estimation. The estimated relative pose from the overlapping range images is applied to transform the local range maps to geometrically align with the other maps and represented in a global RF. This thesis is a comprehensive literature on the 3D map construction, which includes sensor

data acquisition, pre-processing sensor data, polygonal feature extraction from range images, feature-based scan registration of partially overlapping range images and range map fusion.

In our approach, the robots acquire range images predominantly with 3D range scanners, which are fused together to form a complete 3D range map. We also use a perspective camera to initially estimate a pose between the multiple robots to a limited accuracy. The exact pose from the overlapping range images is determined by a scan registration algorithm, which is initialized by the camera pose estimate. We have used commercially available 2D range scanners mounted on a rotating platform to acquire 3D range images. Together, the 3D range scanner and camera acquire information that complements each other while constructing 3D maps in an efficient manner. We individually explore pose estimation algorithms that process input from the two sensors viz., camera and the range scanner. Once the relative pose between the robots is estimated, the range images are fused by applying transformation with respect to the RF. We deployed multiple heterogeneous robots (e.g. wall-climbing robot and Pioneer® ground robots) for acquiring overlapping range images. The wall-climbing robot with its top-down view point was able to acquire additional range information, which was missed by the ground robots.

1.1 Motivation

We were motivated by a number of factors while making decisions to map the indoor environment. Early in our research, we had to consider the type of sensors used, efficiency of the algorithms and type of map representations. The algorithms were developed keeping in mind the following factors.

1.1.1 Use of multiple heterogeneous robots

Ground robots acquire 3D range images, which are not complete due to occlusions. The occluded regions include floor space, top of furniture or other objects above the height of the sensor. To obtain a complete map, we made use of an additional robot, a wall-climbing robot capable of moving on ceiling, walls and floor. At an elevated level, the wall-climbing robot has an added advantage of vantage viewpoints. The 3D range scanner onboard the wall-climbing robot augments information to the 3D range scanner on the ground robot.

Indoor office corridors are sometimes difficult to map by a single robot alone when the floor is slippery and odometry becomes unreliable. This is true especially when the office corridors are highly structured, symmetric and lack visual features. With no input from odometry or vision, it may not be possible to achieve accurate registration of overlapping range images with point-based registration techniques alone. In such a scenario, a multi-robot system will be able to carry their own visual markers and estimate the relative pose between them in real-time using vision. In addition, current state-of-the-art scan registration algorithms converge accurately and fast to fuse partially overlapping range images with a good initial estimate of pose from the camera. In addition, the area covered by the multi-robot system is large and need minimal overlap to register two overlapping range images. This improves the speed of the mapping process. These factors motivated us to approach 3D mapping problem with multiple robots exploring both approaches of estimating pose with vision and range scanners.

1.1.2 Vision-based pose estimation

The motivation for using vision-based pose estimation for aiding overlapping range image registration is two-fold: 1) Almost every laser scan registration algorithm [Besl and McKay 1992; Biber and Straber 2003; Hähnel and Burgard 2002; Lu and Milios 1997;

Weib, Wetzler et al. 1994] works better with an initial estimate of pose and converges into a global minimum (accurate pose) in cases of large rotation and translation between the local frames of two range images. 2) Current laser scan registration methods are computationally intensive. And the complexity of some faster algorithms like the Normal Distribution Transform (NDT) are in the order of $O(N)$ [Takeuchi and Tsubouchi 2006], where N is the number of scan points in the input point cloud set to be matched within the reference scan (N is in the range of $10^5 - 10^7$ 3D coordinate points). There is room for improvement with regard to computational complexity of registration algorithms if a good initial estimate is provided by a real-time pose estimation algorithm. These two aspects motivated us to develop an algorithm to register the two range images with the aid of a camera. The vision-based algorithm provides the initial estimate for scan matching which executes in real-time. The output of the vision-based algorithm can be used to initialize any scan matching algorithms such as the Iterative Closest Point (ICP) or the NDT, which further refines the relative pose from the partially overlapping range images. We show in section 3.8 that, as the initial estimate is close to the real solution, the number of iterations will be decreased for these algorithms to minimize the mean square objective function to a pre-set threshold; initialization increases the computational efficiency of the registration algorithms.

1.1.3 Compact data representation with features

Range images acquired by 3D range scanners consist of large number of data points ($> 10^5$). Large amount of data processing is a significant concern when we register two overlapping range images to align with each other. The computation time of the laser scan registration algorithms is directly dependent on the number of range image data points. In addition, visualizing and storage of 3D point clouds consumes large amount of memory. A compact representation of range images proves to be useful when extracted

features can replace the 3D point cloud. We found that the feature representation is useful for both visualization and in registration of overlapping range images in real-time. In a largely planar environment, we found that we can compress large number of range images and replace them with polygons.

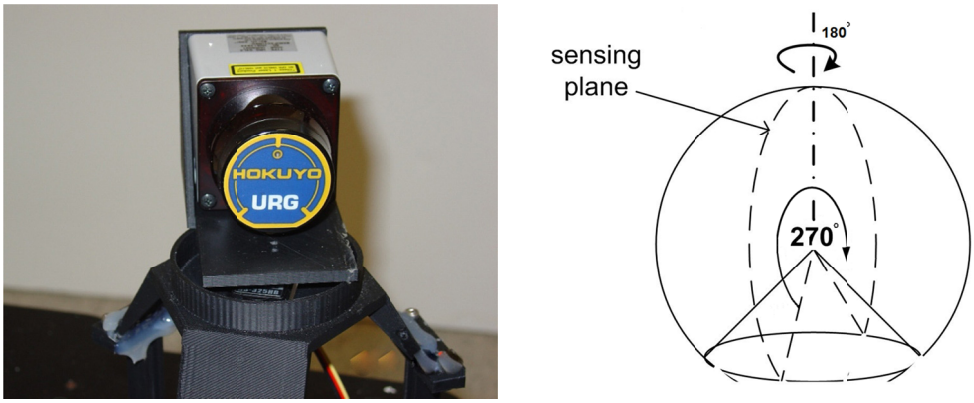
In the following section, we discuss the sensors used by the robots in the mapping process. Their advantages are discussed with regard to the mapping application.

1.2 Multi-modal sensors

The robots used in this research work are equipped with both active and passive sensors for mapping the indoor environment. The passive sensors include a perspective camera and a 3D orientation sensor. The perspective camera used in this experiment is commercial-off-the-shelf (COTS) webcam stripped to a bare minimum. The camera is capable of capturing RGB images with a resolution of 1.3 megapixels. Before the experiments were performed, the camera was calibrated to remove lens distortions and compute the internal parameters of the camera. More information about camera calibration can be accessed online [Strobl, Sepp et al. 2006; Zhang 1998].

Although, there are many applications of map building using camera, most preferred sensor for mapping environments is the range scanner. A range scanner is an active sensor that measures the depth of the surrounding environment and the resulting outcome is a 3D point cloud / 3D range image represented in a Cartesian / spherical coordinate system respectively. Hokuyo® range scanner is a distance measuring apparatus [Mori and Hino 2009], which emits a pulsed measurement light from a photodiode (PD) towards the target object to be measured. The light bounces back from the target object to the instrument. The emitted light is collimated to achieve higher signal strength at the receiver (avalanche photodiode). The electronics at the receiver end computes the delay in the time-of-flight of the pulsed light. Consequently, the distance is estimated based on the time delay between the measuring instrument

and the target object. The 2D range scanner has a rotating mirror inside the sensor which redirects the laser rays to obtain 2D range information in a sensing plane. To obtain a 3D range image, a 2D range scanner is mounted on a rotating platform and panned using an external servo motor (Fig. 1.1). The sensing plane is a 2D plane along which the range is measured by the scanner. A 180° rotation of the sensing plane results in a complete 3D range image as shown in Fig. 1.1. The main application of the 3D range scanner is to measure the depth of the surrounding environment. The range images obtained at various locations are fused together by a process known as ***laser scan registration*** discussed in Chapter 4.



(a) (b)

Fig. 1.1 shows (a) 3D range scanner - Hokuyo® URG-LX 2D range scanner mounted on a rotating platform to acquire a full 3D scan. (b) The range scanner acquires 2D range data along the sensing plane (270°). A complete 3D range image is obtained by rotating the 2D scanner by 180°.

1.3 Multi-robot system

We explored a number of multi-robot system (MRS) scenarios that have a natural advantage over single robots in mapping application. MRS included a number of Pioneer® ground robots equipped with state-of-the-art sensors to map the indoor

environment. In some scenarios, a wall-climbing robot, also known as City-Climber [Elliott, Morris et al. 2006] was part of the MRS that mapped indoor environments. The City-Climber robot adopts a novel adhesive mechanism based on aerodynamic attraction which does not require perfect sealing and enables the robot to operate on smooth and rough surfaces. The City-Climber robot is able to carry relatively large payload up to 4.2kg and fitted with light-weight sensors similar to the ground robots for mapping indoor environments. The City-Climber robots have vantage viewpoints when moving on the ceiling. Their sensor data augments the ground robot mapping data to form more detailed maps without occlusions. Ground robot has viewpoints at a low level and the sensors cannot "see" the top of objects above the viewpoints. On the other hand, the City-Climber robot acquires sensor data of most surfaces from a top-down view. This sensor data is complementary to that of the ground robot. Various functionalities of the City-Climber and their maneuvering operations are presented in [Elliott, Xiao et al. 2007]. A light-weight Gumstix® embedded system acquires data from various sensors on the City-Climber robot.

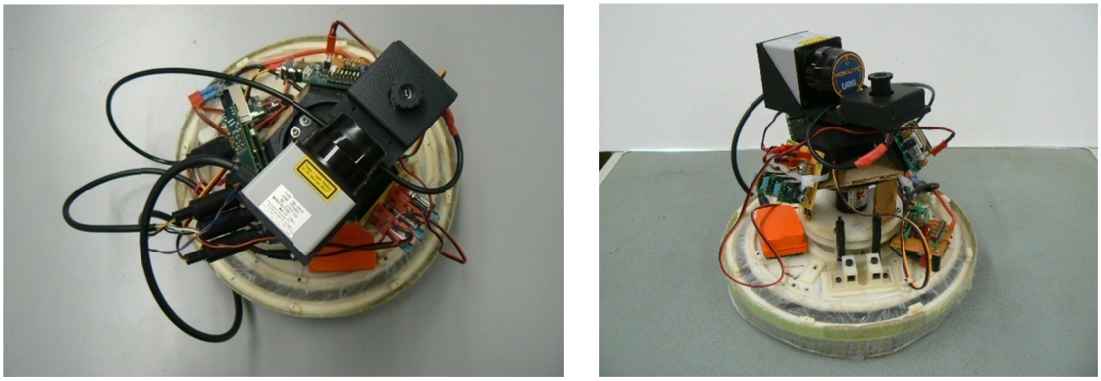


Fig. 1.2 shows City-Climber robot equipped with multiple sensors including 3D range scanner, perspective camera, orientation sensor and Gumstix® embedded processor.

(a) (b)

Fig. 1.3 (a) shows dual Pioneer® ground robots performing mapping experiment. (b) shows a ground robot and a wall-climbing robot on the ceiling mapping indoor corridors

1.4 Map building architecture

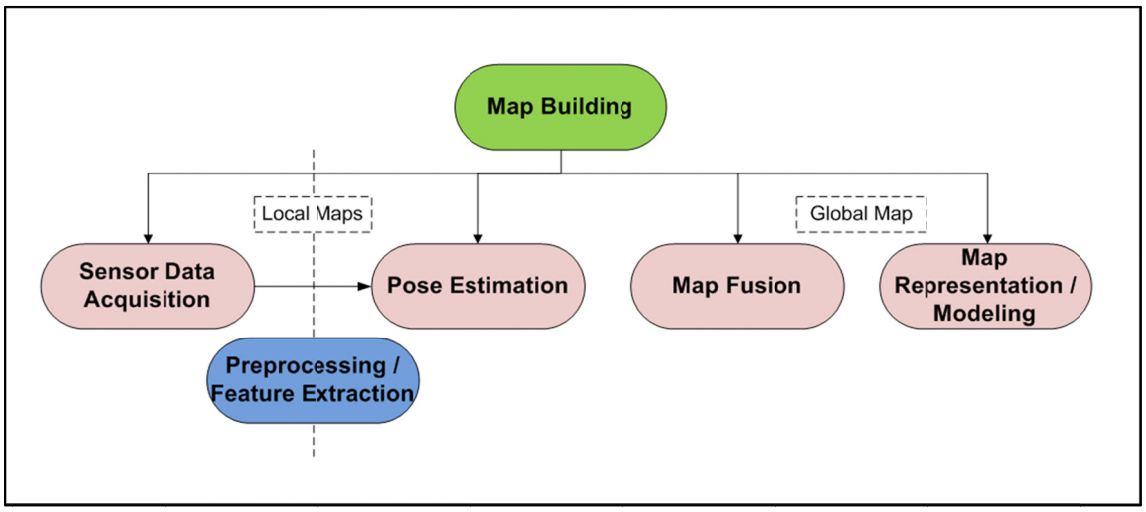


Fig. 1.4 is a schematic diagram showing important steps involved in 3D map construction.

The schematic diagram shows important steps that lead to mapping of indoor environments. ***Sensor data acquisition*** is mostly related to hardware and involves some embedded processing system built into the robots for reliable acquisition. ***Pose***

estimation is the most important step and involves a robust algorithm to determine the pose from contiguous range maps that overlap with each other. A number of algorithms were developed and tested while conducting research on 3D mapping in indoor environments. We explored two major approaches to determine the relative pose between the robots viz. camera pose estimation algorithms using a perspective camera and scan registration algorithms using 3D range scanners. We briefly discuss both class of pose estimation algorithms and list some of the previous attempts in section 1.5. Subsequent Chapter 3 & Chapter 4 will thoroughly discuss these approaches. Map fusion involves transforming local maps with respect to a Reference Frame using the estimated relative pose. This step is relatively straight-forward application of transformation to the map. Finally, map representation involves making a choice early, usually after data acquisition to convert a raw point cloud data into a more compact form by extracting features. These features could be used both in map representation and in some cases used in pose estimation, the latter is our case. This step is beneficial in 3D map construction, which reduces both the computation time of pose estimation and the memory storage space.

1.5 Literature review

3D map construction approaches can be extensively classified based on the type of sensors used for map building. Most widely used sensors for mapping the environment are *perspective cameras* and *3D range scanners*. A number of other sensors such as odometers, GPS, orientation sensors etc. provide additional information to the robots for map building and navigation. ToF depth-cameras and color-depth cameras have been used in recent years in mapping applications, although these sensors are still at their infancy and need to improve on their signal-to-noise ratio.

1.5.1 3D map construction with perspective camera

Map construction with vision sensors has been a popular approach (only next to range scanners) in computer vision and robotics community [Asai, Kanbara et al. 2005; Davison, Reid et al. 2007; Ellekilde, Huang et al. 2007; Hartley and Zisserman 2004; Jensfelt, Kragic et al. 2006; Paz, piniés et al. 2008; Tomono 2009]. Most vision-based algorithms extract features and compute their 3D locations with respect to a local frame. After establishing correspondences between the features of successive frames, the robot is localized mostly using probabilistic algorithms such as the Extended Kalman Filter (EKF) [Thrun, Burgard et al. 2005], FastSLAM [Montemerlo, Thrun et al. 2002], or Extended Information Filter (EIF) [Ellekilde, Huang et al. 2007; Thrun, Liu et al. 2004]. In [Ellekilde, Huang et al. 2007], the features are extracted from camera images using Scale Invariant Feature Transform (SIFT) [Lowe 2004] and the Random Sample Consensus (RANSAC) [Fischler and Bolles 1981] algorithm to fit 3D point sets. The EIF algorithm updates the robot state vector while navigating through indoor environment. [Se, Lowe et al. 2000] presents an approach to vision SLAM-Building using SIFT matching from Triclops® stereo camera. The 3D landmarks are tracked and their positions are estimated using Newton method for Least Square (LS) minimization. Another example of Monocular SLAM [Eade and Drummond 2006] extracts landmarks and represent them with 3D ellipsoids in the map. Their solution uses FastSLAM-based particle filter to build and store maps. This SLAM approach was classified as a subset of Structure-From-Motion (SFM) algorithms in vision. The use of a ***camera*** to obtain the ***relative pose*** from subsequent viewpoints has a distinct advantage over the use of the laser scans, which is to establish corresponding features between the two sets of images using texture information. However, the camera is a 2D representation of the 3D world and the depth of the features can be extracted from the camera only to an unknown scale factor. This limits the camera to be used as a measuring device to obtain absolute depth of its

surrounding environment. A number of researchers have found alternative techniques to overcome this problem. [Spletzer, Das et al. 2001] approached localization of multiple robots using cameras and determined the locations initially to a scale factor and ultimately determined the exact scale heuristically. [Hartley and Zisserman 2004; Thrun, Burgard et al. 2005] discussed a number of localization methods using just visual imagery and localization with visual imagery in conjunction with other sensors such as inertial sensors and GPS in a Simultaneous Localization and Mapping (SLAM) framework.

1.5.2 3D map construction with range scanners

3D range scanners are the most popular of the sensors for mapping environments in the robotics community [Borrmann, Elseberg et al. 2008; Grzonka, Grisetti et al. 2009; Hähnel, Burgard et al. 2003; Liu, Emery et al. 2001; Nüchter, Lingemann et al. 2005; Pathak, Birk et al. 2010; Takeuchi and Tsubouchi 2006; Thrun, Martin et al. 2004; Weingarten, Gruener et al. 2003; Weingarten and Siegwart 2005]. The depth measurement by the range scanners is the most accurate among its peers to build a 3D depth-map of the environment. As the robot navigates through the environment in a stop-n-go fashion, the 3D range images are acquired at multiple viewpoints. These local maps are fused together by registration techniques discussed in Chapter 4.

Following are some mapping algorithms that use planar segments as features extracted from range images. [Liu, Emery et al. 2001; Thrun, Martin et al. 2004] apply Expectation-Maximization (EM) algorithm to build low-complexity multi-planar model using 3D range scan input. The algorithm performs quadratic optimization and goes through several iterations before it stabilizes on the number of planar regions. It assumes that the pose estimate of the robot is given by the Particle Filter (PF) algorithm while transiting through the environment [Thrun 2001]. Low-complexity models [Grzonka, Grisetti et al. 2009; Hähnel, Burgard et al. 2003; Pathak, Vaskevicius et al. 2009] are preferred in robotic applications such as 3D SLAM and navigation. The current

mapping techniques aim at modeling the environment in real time versus offline modeling that result in large highly complex models by the algorithms presented in computer graphics community [Garland and Heckbert 1997; Hoppe 1996; Stamos and Allen 2000; Turk and Levoy 1994].

Many attempts have been made by robotics researchers to obtain a low complexity model sufficient to navigate through the environment while keeping the computation time of the mapping algorithms to a minimum. One such approach is by [Hähnel, Burgard et al. 2003]; they present a recursive region-growing algorithm that is used to obtain large flat surfaces from the range images in structured environment. The pose of the robot is estimated using probabilistic scan matching algorithm [Hähnel and Burgard 2002]. The region-growing algorithm clusters the range points into planar and non-planar points. Planar points are represented by large polygons and non-planar points by fine-structured polygons to display a complete 3D map. The process of adding points to a plane is randomized and whenever a point is added to a plane, the plane parameters are recomputed. The computation time of the segmentation algorithm increases exponentially with the number of new points added to a planar segment. [Weingarten and Siegwart 2006; Weingarten and Siegwart 2005] introduced 3D SLAM with EKF using planar segments. As a first step, the range points were segmented into planar segments using RANSAC [Fischler and Bolles 1981] and a region-growing algorithm [Weingarten, Gruener et al. 2003]. Further, the pose of the robot was computed by the EKF algorithm. A probabilistic data association is applied to two overlapping scans for matching planar segments. The corresponding planar segments needed to satisfy the χ^2-hypothesis test based on squared Mahalanobis distance.

Other scan matching algorithms include Iterative Closest Point (ICP) algorithm [Besl and McKay 1992]. It tries to minimize the geometric distance between the corresponding closest points. An alternative technique to establish correspondence

between two point sets is presented in Iterative Dual Correspondence (IDC) algorithm [Bengtsson and Baerveldt 2003; Lu and Milios 1997]. It improves the correspondence between range points of two scans by searching in a limited angular space. A number of researchers [Biber and Straber 2003; Magnusson, Andreasson et al. 2009; Takeuchi and Tsubouchi 2006] have used Normal Distribution Transform (NDT) representation of metric maps to fuse range images and estimate the pose of the robot. The speed of convergence depends on the number of cubes / voxels that describe the scan points and the number of iterations that estimate the pose parameters using Newton's algorithm in [Takeuchi and Tsubouchi 2006] and EM algorithm to fit Gaussian curves in [Magnusson, Andreasson et al. 2009].

Our research work on range image registration is on similar lines to the work described by [Pathak, Birk et al. 2010]. In that, the authors construct a metric map by matching large planar segments. As a first step, large planar segments are extracted using an improvised region-growing algorithm [Poppinga, Vaskevicius et al. 2008] that provides two additional recursive mathematical functions to improve the speed of the region-growing algorithm in [Hähnel, Burgard et al. 2003]. Further, the scan registration of two range images is performed by establishing the correspondences between the planar segments and minimizing the distance between the corresponding planar segments [Pathak, Vaskevicius et al. 2009]. The rotation and translation component are decoupled initially for estimating the pose. The rotation component is estimated using a closed form solution based on Quaternions [Horn 1987]. The translation component is determined via a more elaborate method using the LS minimization approach. In some cases, the robot resorts to odometry for estimating the translation to resolve the uncertainties in the pose. A graph-pose relaxation technique is presented to improve the accuracy of the maps in [Pathak, Birk et al. 2010]. [Pathak, Vaskevicius et al. 2009] describes an algorithm to establish correspondence between two plane sets (to be

registered) with a time complexity of $O(N_A^2 N_B^2)$ where N_A is the number of planes from range image set A and N_B is the number of planes from range image set B.

Our aim was to develop a real-time algorithm to establish polygon correspondence and register partially overlapping range images. We studied some of the attributes of polygonal map representation and found that it is possible to establish correspondences of polygon sets by comparing polygon attributes. This inspired us to develop a novel approach to register two sets of polygons extracted from partially overlapping polygons. To minimize the distance between corresponding polygon sets, we applied a nonlinear parameter estimation technique that estimates rotation and translation parameters simultaneously and minimizes the distance between corresponding polygons. We use the term "polygon" interchangeably with "plane / planar segment" in this paper to refer to a finite plane defined by boundary points.

1.5.3 3D map construction with multi-modal sensors

There are several advantages of using a number of sensors together for map building.

- Detailed map information (e.g. depth & texture mapping)
- Complementary information from different sensors could improve the accuracy of map registration
- Good initial estimation of pose for registration techniques

Some of the map building approaches using multi-modal sensors is as follows. [Edlinger, Puttkamer et al. 1991] approached the problem of localization and mapping using odometry inputs (Dead Reckoning) and 2D range data, which was one of the initial approaches to ***localize the robot*** as it moves on the ground. This approach provided an initial estimate of pose to fuse the 2D laser scans. The major drawback of this approach is that errors accumulate over the distance traveled. This approach was limited to 3DoF pose estimate and the map was generated in 2D space. A number of other researchers

followed a methodology of map construction with the same combination of sensors and attempted to improve the accuracy of the pose estimates. This led to the development of a new framework known as probabilistic SLAM. [Eliazar and Parr 2004; Fox, Burgard et al. 1999; Fox, Burgard et al. 1999; Howard 2006; Liu, Emery et al. 2001; Montemerlo, Thrun et al. 2002; Thrun 2001]. Probabilistic SLAM included predominantly two steps towards map building and localization. In the first step known as ***prediction***, a sensor would approximately localize the robot with the noisy data. A set of random variables are assumed to estimate the state vector (pose) of the robot/s. In the second step known as ***update***, the information obtained from another sensor is fused together with the prior knowledge of the state vector, which improves the accuracy of the 2D pose estimate of robots. However, it has been cited by many researchers that probabilistic SLAM may not be computationally feasible to scale them upwards to building 3D maps [Nüchter 2009; Surmann, Nüchter et al. 2003].

A number of recent map building techniques process the information from both the 3D depth map acquired from range scanners with the texture information acquired from camera. One advantage of such a multimodal approach is an enhanced depth image for visualization [Scheibe, Scheele et al. 2004; Sequeira, Ng et al. 1999; Stamos and Allen 2000]. In [Newman, Cole et al. 2006], the range images from the laser scanner are used to build maps incrementally as the robot travels through the outdoor environment. Sequences of images from the camera are acquired at the regular intervals of robot motion. The image dataset is then used for testing loop closure using SIFT descriptors and Harris Affine Detector for describing landmarks in the image.

Our approach also used both vision and range scanners for building 3D maps [Kaushik, Feng et al. 2008; Kaushik, Xiao et al. 2009]. The two sensors were used for different purposes. A range scanner was used to acquire 3D depth maps at multiple viewpoints. A camera pose estimation algorithm provided a good initial estimate of pose

between the two robots. This pose estimate was used to initialize the scan registration algorithm to align partially overlapping scans acquired by the range scanners. In the following chapter, we define the camera pose estimation problem and discuss some of the approaches.

CHAPTER 2

Camera-based pose estimation

The camera pose estimation problem involves computing the translation and orientation of the camera coordinate frame with respect to a reference object (e. g. visual marker), whose location is known with respect to a Reference Frame (RF). It is assumed that the camera internal parameters is known prior to the pose estimation by one of the camera calibration techniques [Heikkilä and Silvén 1997; Tan 1989; Zhang 1998]. The camera captures the image of the reference object. The image is then preprocessed to extract specific image points known as control points, whose coordinates are known with respect to the RF. The pose estimation algorithm then computes the pose estimate by taking the following inputs: the image coordinates of the control points and their 3D coordinates with respect to the RF.

We have extensively applied the camera pose estimation problem to compute the relative pose between the multiple robots in various scenarios [Feng, Zhu et al. 2006; Feng, Zhu et al. 2007; Kaushik, Feng et al. 2008; Kaushik, Xiao et al. 2009]. This approach has a number of advantages in localizing the robots with respect to each other. First, the camera pose estimation algorithms can provide an estimate of pose between the robots. In our application, it would form a rough initial pose estimate between the robot coordinate frames fixed on the range scanners of the two robots. This is used to initialize the scan registration algorithm to align 3D range images. With a good estimate, the scan registration algorithms are more likely to converge to the actual pose determined from the overlapping range images [Besl and McKay 1992; Biber and

Straber 2003; Rofer 2002]. Hence, the initial estimate is beneficial for guaranteed convergence. Second, camera pose estimation algorithm can estimate pose in real-time compared to offline iterative scan registration techniques. Hence, this mapping approach finds an advantage of supplementing the scan registration techniques with a good initial estimate in real time. In the following sections, we introduce two class of camera pose estimation algorithms based on the computational methodology: Iterative algorithms and Closed-form solution based algorithms.

2.1 Camera pose estimation using iterative algorithms

A different class of camera pose estimation algorithms are based on nonlinear iterative techniques [Kato and Billinghurst 1999; Lu, Hager et al. 2000] and estimates pose with respect to a quadrangular target. In [Kato and Billinghurst 1999], an initial estimate of the rotation and translation is obtained using a direct analytic method. The initial estimate is used to initialize the iterative algorithm to reduce the difference between the 2D measured coordinates of the marker and the marker coordinates (3D points converted to 2D). The final solution is computed based on LS minimization technique using the Newton's method. This approach is embedded in the well-known ARToolKit® software. [Lu, Hager et al. 2000] introduce an Orthogonal Iteration (OI) algorithm, which estimates the pose parameters in 3D Euclidean space. It solves the pose estimation / absolute estimation problem and computes the pose parameters by minimizing the error function, which is defined in the object-space rather than 2D-pixel space. Sometimes, the algorithm may not compute the solution accurately as it may converge to a local minimum. [Schweighofer and Pinz 2006] provide an algorithm that ensures that a correct pose is obtained when more than one local minimum exists while minimizing the error function. The iterative camera pose estimation algorithms are relatively slow compared to direct methods but they are more accurate compared to

the direct methods when using noisy images and converge to a global minimum in most cases.

2.2 Camera pose estimation by PnP algorithms

The wall-climbing robot is mounted with an overhead camera, and the robot coordinate frame is fixed at the center of perspective (O) of the camera. The vertex (O) forms a tetrahedron with the n control points on the ground robot. We then solve the PnP problem, also known as *Location Determination Problem* (LDP) [Fischler and Bolles 1981] or *Camera Pose Estimation problem* [Chang and Chen 2004], which provides the relative pose of the camera with respect to the n control points. Given $(n \geq 3)$ points in Euclidean space with respect to a marker coordinate frame and their 2D image points in the image coordinate frame of the camera, it is possible to compute the relative translation and orientation between the marker coordinate frame and the camera coordinate frame [Fischler and Bolles 1981]. For $(n = 1 \,\&\, 2)$, there exists infinite number of solutions. For $(n = 3)$, there are up to 4 real solutions [Fischler and Bolles 1981; Gao, Hou et al. 2003; Haralick, Lee et al. 1991]. For $(n = 4)$ coplanar points, there exists a unique solution and some researchers have obtained a closed-form solution to the problem [Abidi and Chandra 1995; Fischler and Bolles 1981]. For $(n = 4 \,\&\, 5)$ non-coplanar points, there exist up to two solutions. For $(n \geq 6)$, there exists a unique solution and can be solved using linear equations [Fischler and Bolles 1981; Quan and Lan 1998].

2.2.1 Multiple solutions to PnP problems

For ($n \geq 3$ and $n < 6$) control points, solving the PnP problem results in multiple solutions out of which only one solution is physically valid [Gao, Hou et al. 2003; Haralick, Lee et al. 1991]. Some researchers have presented geometric conditions where the P3P problem yields a unique valid solution [Wang, Wang et al. 2006]. But, none of

these methods are practically applicable to robot pose estimation. Consider a case where the wall climbing robot on the ceiling is trying to localize itself with respect to a ground robot. In such a scenario, it is not possible to place the robot in the special zones which yield the single solution, as the approach would look contrived. One of our efforts has been to identify a valid solution to the P3P problem while the robots are in arbitrary positions. The resulting pose estimate is used to initialize the registration algorithm to fuse multiple range images. This is most useful in digitization of environments with 3D range maps. We proposed several approaches to determine the pose of the camera with respect to a known target by moving the camera and applying the P3P problem multiple times. The multiple solutions at each instance are weighed based on the motion of the camera. In our early work [Feng, Zhu et al. 2006] to be described in Chapter 3, we proposed a deterministic algorithm for uniquely locating the wall-climbing robot with respect to a ground robot by making use of the straight line motion constraint on the wall climbing robot. Later, we attempted to determine the pose of the wall climbing robot moving in arbitrary directions. The multiple solutions were weighed based on the motion of the robot, which was inspired by the Particle Filter (PF) method [Thrun, Burgard et al. 2005]. The algorithm follows the prediction-update rules to determine the valid solution to the P3P problem based on the movement of the climbing robot and entirely abandoned the linear constraint motion of the robot. We now consider the P3P problem in detail.

2.3 Perspective Three Point (P3P) problem

The P3P problem is a subset of the PnP problem with least number of required control points defined as:

"Given the relative spatial locations of n control points and given the angle to every pair of control points from an additional point called the Center of Perspective (O), find the lengths of sides joining the (O) to every control point. This problem is defined as Perspective-n-Point (PnP) problem" [Fischler and Bolles 1981]

Proposition 1: Given the positions of three non-collinear points and distances to a particular point from them, the position of the particular point can be determined by the intersection of three spheres, centered at the three points, with the radius of the three distances, respectively.

Definition 1: As shown in Fig. 2.1, let O be the optical center of the camera, let P_1, P_2 and P_3 be the three non-collinear points with respect to coordinate frame $(X_1Y_1Z_1)$; let q_1, q_2 and q_3 be the projection of the three points P_1, P_2, P_3 in the camera coordinate frame (XYZ). The **P3P problem** is to estimate the position of O and the rotation matrix $R_{(3x3)}$ of the camera frame (XYZ) with respect to the frame $(X_1Y_1Z_1)$.

Given the image location and the 3D location of the control points with respect to the robot coordinate frame, we can estimate the pose of the overhead camera by solving the P3P problem. The P3P problem [Fischler and Bolles 1981; Haralick, Lee et al. 1991] provides the relative pose of the camera with respect to the three control points. The CP and the three control points form a tetrahedron (Fig. 2.1). The first step in the camera pose estimation process is to determine the sides (s_1, s_2, s_3) of the tetrahedron. The next step is to determine the relative pose that defines the geometric relation between the camera coordinate frame and the reference frame (control points are assumed to be known with respect to this frame).

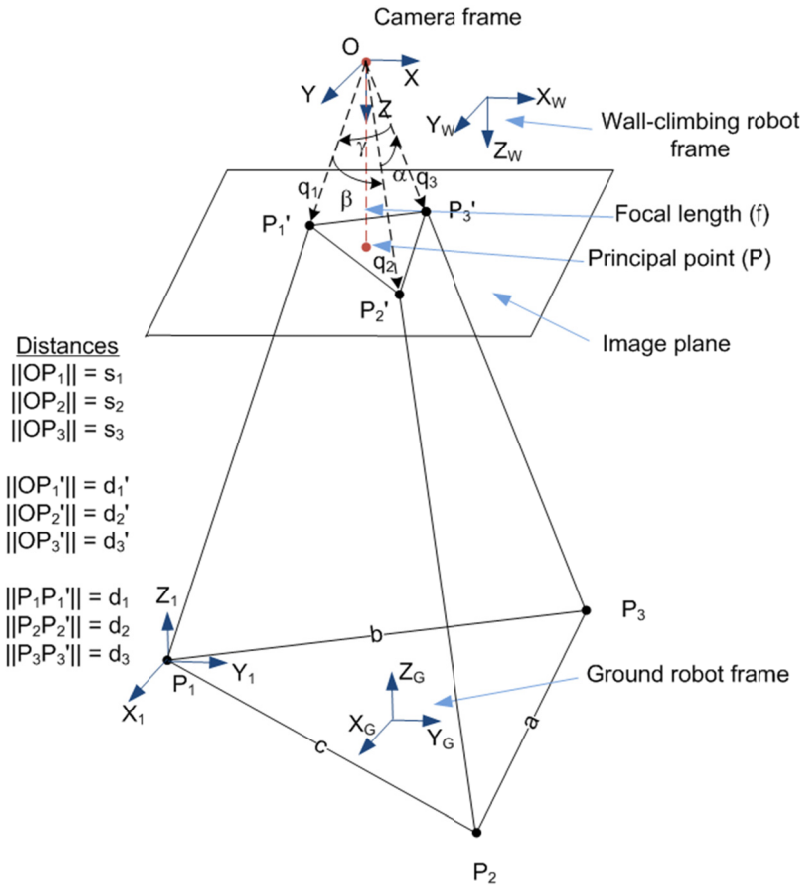


Fig. 2.1 shows formation of tetrahedron whose vertices are the three markers on the ground mobile robot and the center of perspective camera (O) on the wall-climbing robot. The vision-based algorithm determines the position matrix (T) and orientation matrix (R) of the camera on the wall-climbing robot with respect to the coordinate frame fixed on point P_1.

We define an application of the P3P problem in the context of a dual-robot system scenario and introduce the mathematical background to the P3P problem. The camera is mounted on an overhead robot (e.g. wall climbing robot) and the control points (markers) are located on a different robot. In this context, the pose between the two robots can be established. Fig. 2.1 shows a tetrahedron formation between the points ($OP_1P_2P_3$). The vertex O represents the origin of the camera coordinate system

mounted on the wall-climbing robot. The other three vertices represent the three control points given by $P_i = [x_i, y_i, z_i, i = 1,2,3]$ with respect to the ground robot coordinate frame fixed at P_1. Let the side length between the 3D coordinate points be $a = \|P_2 - P_3\|$, b= $\|P_1 - P_3\|$, $c = \|P_1 - P_2\|$ which are assumed to be known a priori. The 2D projections of the control points on the image plane of the camera is given by $q_i = [u_i, v_i]^T, i = \{1,2,3\}$. By perspective pinhole camera model, we have $u_i = \frac{x_i}{z_i}$ and $v_i = \frac{y_i}{z_i}$. Given the focal length of the camera (f), the unit vectors are computed as, $j_i = \frac{1}{\sqrt{u_i^2+v_i^2+f^2}}.(u_i, v_i, f), i = \{1,2,3\}$, which originate from the Center of Perspective (O) in the direction of P_1, P_2 and P_3 control points respectively. The first step is to determine the distance of the sides (s_1, s_2, s_3) joining the CP and the three control points. Appendix A.1 & A.2 provide steps to compute the pose parameters using an analytical method.

2.3.1 Grunert's solution to the P3P problem

Applying the law of cosines to the tetrahedron, we have

$$a^2 = s_2^2 + s_3^2 - 2s_2 s_3.\cos(\alpha) \quad 2.1$$

$$b^2 = s_1^2 + s_3^2 - 2s_1 s_3 \cos(\beta) \quad 2.2$$

$$c^2 = s_1^2 + s_2^2 - 2s_1 s_2 \cos(\gamma) \quad 2.3$$

Let,

$$s_1 = u.s_2 \text{ and } s_1 = v.s_3 \quad 2.4$$

$$s_1^2 = \frac{a^2}{u^2 + v^2 - 2uvcos(\alpha)} = \frac{b^2}{1 + v^2 - 2vcos(\beta)} = \frac{c^2}{1 + u^2 - 2ucos(\gamma)} \quad 2.5$$

From Eqn. 2.5, we can obtain two equations with two unknown variables u and v.

$$u^2 + \frac{b^2 - a^2}{b^2}.v^2 - 2uv.\cos(\alpha) + \frac{2a^2}{b^2}v.\cos(\beta) - \frac{a^2}{b^2} = 0 \qquad 2.6$$

$$u^2 - \frac{c^2}{b^2}v^2 + 2v\frac{c^2}{b^2}\cos(\beta) - 2u.\cos(\gamma) + \frac{b^2 - c^2}{b^2} = 0 \qquad 2.7$$

From Eqn. 2.6, we have

$$u^2 = -\frac{b^2 - a^2}{b^2}v^2 + 2uv.\cos(\alpha) - \frac{2a^2}{b^2}v.\cos(\beta) + \frac{a^2}{b^2} \qquad 2.8$$

Substituting Eqn. 2.8 in Eqn. 2.7, we obtain an equation, which expresses u as a function of v.

$$u = \frac{\left(-1 + \frac{a^2 - c^2}{b^2}\right)v^2 - 2\left(\frac{a^2 - c^2}{b^2}\right)\cos(\beta).v + 1 + \left(\frac{a^2 - c^2}{b^2}\right)}{2(\cos(\gamma) - v.\cos(\alpha))} \qquad 2.9$$

Further, Eqn. 2.9 is substituted back into Eqn. 2.6 to obtain a fourth order polynomial in v.

$$A_4v^4 + A_3v^3 + A_2v^2 + A_1v^1 + A_0 = 0 \qquad 2.10$$

Solving the above quartic function results in up to four solutions of u and v, some roots are real numbers and the rest are imaginary. Ignoring the imaginary roots, we calculate the sides $\left(s_1^i, s_2^i, s_3^i\right), i = \{1..n\}$, n is the number of real roots as solution to the quartic Eqn. 2.10. (multiple solutions to the P3P problem).

Given the possible solutions of sides $\left(s_1^i, s_2^i, s_3^i\right)$ and the prior information $P_i = [x_i, y_i, z_i, i = 1,2,3]$, it is possible to determine the location of the center (O) of the camera (on the wall-climbing robot) and rotation matrix (R); this matrix describes the relative orientation between the camera coordinate frame with respect to the ground robot frame at P_1.

2.4 Performance evaluation of camera-based pose estimation algorithms

In this section, we compare the iterative algorithms and a direct approach to camera pose estimation. Prior to executing the algorithms, vision-based pose estimation involves camera calibration, which is a critical step in improving the accuracy of the estimated pose. The camera calibration is an elaborate process of determining the internal camera parameters [Heikkilä and Silvén 1997] of the camera, namely effective focal length in pixels (f_x, f_y), principal point of the camera (o_x, o_y), radial and tangential distortions (k) and skew co-efficient (α). The pixel position error in the camera can be attributed to improper calibration. Assuming that the radial and tangential distortion of the lens are corrected, the pixel coordinates are given by,

$$x_{pix} = \frac{f_x.X_c}{Z_c} + o_x, y_{pix} = \frac{f_y.Y_c}{Z_c} + o_y \qquad 2.11$$

We can observe that the pixel error becomes more prominent as the marker is placed far from the camera. In our simulation, we perturbed the position of the pixels of the input image with uniformly distributed noise (to recreate positional error). The signal-to-noise ratio as a distance measure of pixels is given by,

$$SNR = 20.log\frac{\mu_{pix}}{\sigma_{pix}} = 20.\log\frac{\sum_{i=0}^{n}\overline{X} - X_i}{(\overline{X} - \mu)(\overline{X} - \mu)^T} \qquad 2.12$$

The transformation between the virtual camera frame and the reference frame of the control points is known. The image points were computed based on the pin-hole camera model. The noise was added to the 2D image points of the four control points. The noisy image points and the four control point locations with respect to the reference frame formed the input to the camera pose estimation algorithms. We tested the accuracy of four algorithms namely, closed-form P3P, image-space iterative algorithm (ARToolkit®), orthogonal iteration algorithm (object-space iterative) and pose ambiguity resolver. Some characteristic differences between the algorithms are noted in Table 2-1.

Table 2-1. Comparison of characteristics of Iterative and Closed-form pose estimation algorithms

PNP-BASED (CLOSED-FORM)	IMAGE-SPACE ITERATIVE	ORTHOGONAL ITERATION	POSE AMBIGUITY RESOLVER
[Fischler and Bolles 1981; Haralick, Lee et al. 1991]	[Kato and Billinghurst 1999]	[Lu, Hager et al. 2000]	[Schweighofer and Pinz 2006]
Algebraic solution to the problem	Initialization by an analytical solution	Absolute orientation by Quaternions, Singular Value Decomposition	Search for local minima of an error function in 3D space
Multiple solutions, resolved by a particle filter by probabilistic approach [Kaushik et. al., 2009]	Gauss-Newton (GN), Levenberg-Marquardt algorithm (LMA). Iterations in 2D image space	Orthogonal Iteration algorithm that iteratively computes pose in 3D object space	Iterations are performed multiple times to determine a global minimum
Fast, non-iterative solution and guarantees unique solution	Converges to the global minimum with good initialization	Known to converge faster than LMA, but may not converge to a global minimum	Search for multiple local minima and identify the best pose after initialization

The pose estimation is evaluated for two important attributes: translation accuracy, along three axis (tx, ty, tz).

1. Positional error of pose estimation algorithms for fixed marker coordinate size, fixed orientation and varying position of marker along the optical axis of the camera (Fig. 2.2).

2. Pose (6DoF) error of pose estimation algorithms with varying pan angle between the camera and the marker coordinate frame (Fig. 2.3).

Fig. 2.2 indicates the translational accuracy of the pose estimation algorithms. The distance is varied between the camera coordinate frame and the control points, while

the algorithms estimated the translation in each case. The results shown are averaged over 1000 runs with random noise. The accuracy of LMA-based iterative pose estimation algorithm performed the best compared to other algorithms. All algorithms displayed translational accuracy as the control points moved away from the camera. This is expected due to the nature of the pin-hole camera model. We note an axiom here that "the accuracy of the distance measure of the objects using a monocular camera is a function of the distance between the camera and the objects and other camera attributes as a result of sampling".

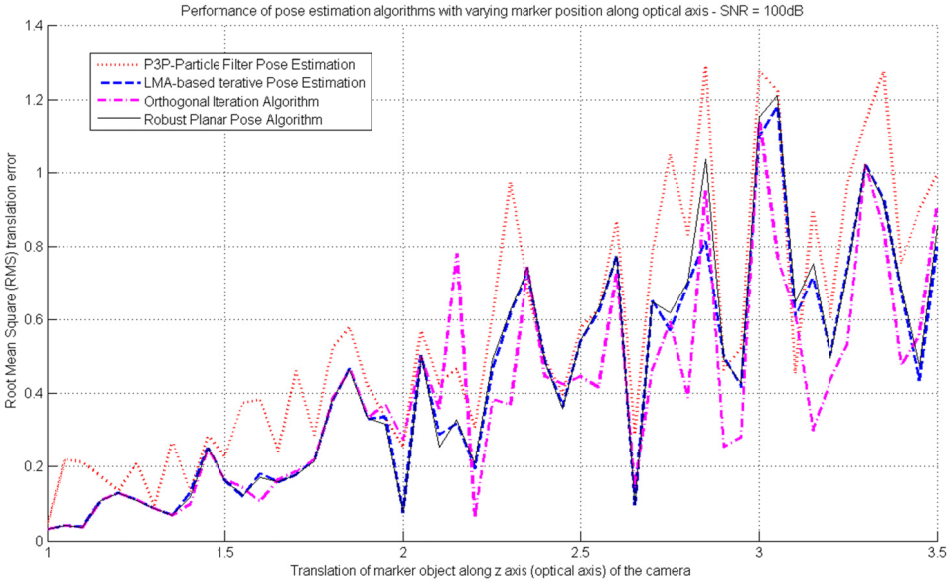


Fig. 2.2 Translation accuracy of the pose estimation algorithms as the marker is placed at different distances from the camera. Pixel position error measured in SNR is set to 100dB.

Below, we consider another real experiment, where the camera is panned along the gravity vector. As the camera steps through pan angles, the images of the control points are acquired. The control points appear to move from left to right in the image plane. The iterative pose estimation algorithms are tested for angular and translational accuracy using the same images. It appeared that the ARToolkit® has a higher accuracy

in estimating the pan angles. There was relatively no difference in estimating the translation by all the iterative algorithms at a distance of approximately $1m$.

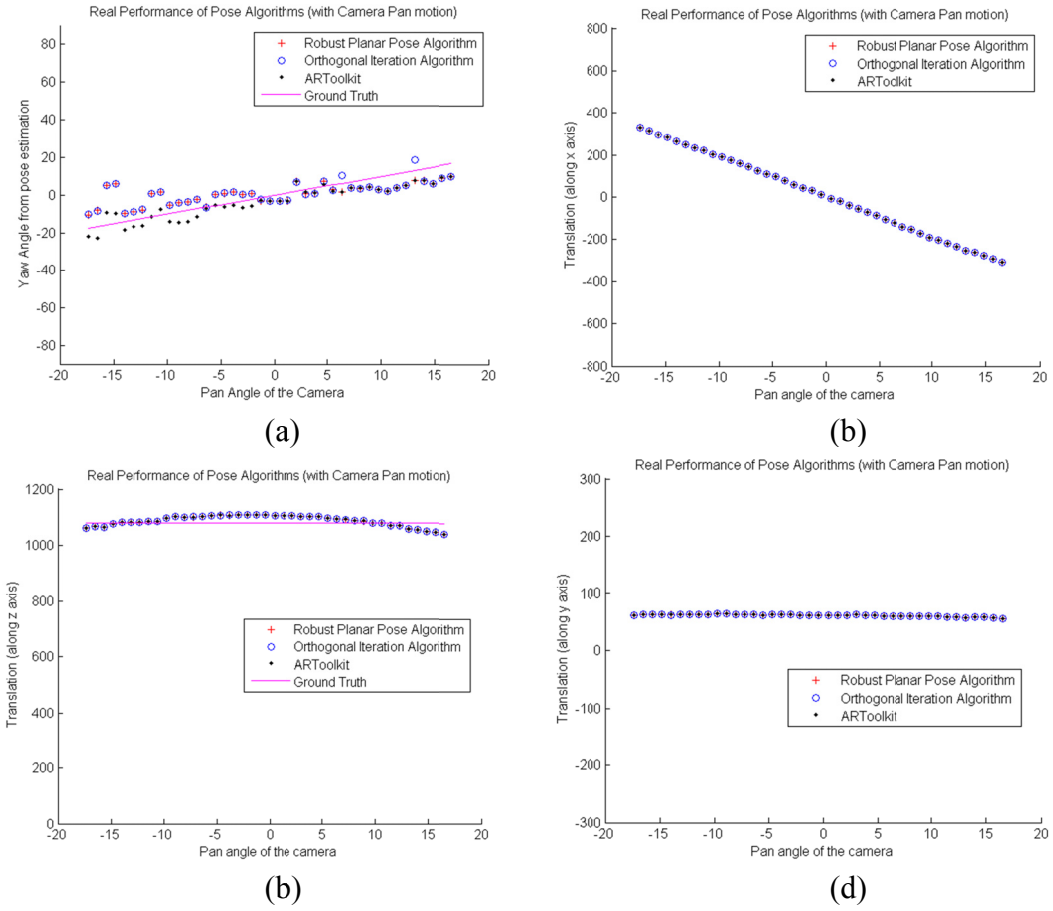


Fig. 2.3 Performance of camera pose estimation algorithms as the camera is panned with respect to the marker. The distance between the marker and camera are fixed. (a) indicate the pan angles obtained by the pose estimation algorithms (b), (c) and (d) are translation (t_x, t_y, t_z) respectively estimated by the algorithms

The ARToolKit® camera pose estimation algorithm performed relatively well compared to other algorithms and was used in our experiments. The final result is the relative pose between the camera coordinate frame and the marker coordinate frame assumed to be located at the center of the rectangular marker. The rigid transformation between the

camera coordinate frame and the 3D range scanner of the first robot, the marker coordinate frame and the 3D range scanner of the second robot are known prior to the experiments. This transformation is applied to the result from the camera pose estimation algorithm. The final resulting pose is an estimate between the two 3D range scanners on the two robots. In the next chapter, we use the camera pose estimation algorithm to fuse two sets of overlapping range images. A number of multi-robot scenarios are discussed that lead to a coordinated effort in building a 3D map of an indoor environment.

CHAPTER 3

Vision-aided multiple heterogeneous robot mapping

Multi-robot systems used in mapping can be advantageous in several ways over a single robot. Multiple robots can estimate the pose between them using real-time camera pose estimation algorithms. Let us consider a scenario where multi-robot coordination is useful, such as in mapping / modeling an environment devoid of features and where odometry readings fail to register accurate readings. Both are extreme conditions and likely found in many indoor environments. An example of such a scenario would be long symmetric corridors with slippery floors. Another example would be an underground mine. Mine floors are extremely rugged, dark and devoid of specific features that can be used by vision algorithms for robot localization. It is possible that the use of a single robot equipped with a range scanner and/or a camera will be insufficient to map such an environment. The scan registration algorithms are more likely to diverge without a proper initial estimate of pose between the two robots. If a single robot is replaced by two or more robots that can localize with respect to each other, the overlapping range scans can be fused together accurately.

This chapter exclusively deals with the 3D map construction of structured indoor environments using a multi-robot system consisting of heterogeneous robots such as ground robots and a wall-climbing robot equipped with a range scanner and a camera. The ground robot can generate range images of the region above and its surrounding regions. Due to occlusions, the range scanner on the ground robot cannot capture the

floor and in some cases the top surface of the objects (e.g., furniture tops) that are above the height of the range sensor positioned in the environment. A wall-climbing robot operating on the ceiling can acquire range images complementary to the ground robot with a partial overlap of the surrounding region. This vantage viewpoint of the wall-climbing robot is another important advantage of using multiple heterogeneous robots in mapping. The two range images are later fused together by the scan registration algorithm to obtain an enhanced 3D map without occlusions. We tried several approaches on how to deploy multiple heterogeneous robots for mapping. The trial-and-error process helped us understand the complexities involved in localization of the robots with respect to a global RF. In a multi-robot system, localization of the robots with respect to a RF deals firstly with estimating the relative pose between any two given robots and then updates the previous transformation of pose with the current relative pose.

The use of multiple robots for mapping provided us an opportunity to use a vision-based algorithm that executes in real-time compared to an offline scan registration algorithm. The quality of map fusion is dependent on estimation of the relative pose between the robots, which is the main goal of this research work. The primary sensors for mapping and localization are vision and range scanners. We explored pose estimation algorithms based on both the sensors individually and in a combined mode. In the following section, we discuss some of the scenarios of multi-robot systems where the algorithms were tested.

3.1 Multiple heterogeneous robot mapping

Fig. 3.1 shows experimental setup that includes three ground robots and a wall-climbing robot on the ceiling.

In our first attempt to construct a map of an indoor environment, we deployed three ground robots and a wall-climbing robot, which formed a heterogeneous robot team mapping in collaboration (Fig. 3.1). Each ground robot is equipped with a LED cluster (marker), a perspective camera and a 3D range scanner. The three LED clusters on the ground robot act as markers and is visible to other robots. All four robots are static

while acquiring the sensor data. The rotary range scanner on each robot is programmed to acquire a 3D range image with respect to the sensor coordinate frame. It is assumed that the three ground robots are in the field of view of camera placed on other robots. We acknowledge that this is a very tight constraint indeed for robot maneuvers. Multi-robot maneuvers are always complex, especially without losing sight of other robots and localizing with respect to each other at all times. A concurrent maneuver of three or more heterogeneous robots is presented in [Feng, Zhu et al. 2006]. In later experiments, we relax the number of robots and their motion constraints by considering a dual-robot system in the section 3.5.

An approach to cooperative 3D localization of three or more ground robots using primary modality as vision is described in [Feng, Zhu et al. 2007; Feng, Zhu et al. 2006; Spletzer, Das et al. 2001]. For now, we will consider three ground robots (located by visual markers placed on each robot) for our mapping application, which is the minimum number of required robots for cooperative localization and mapping. With regard to the map construction using multiple heterogeneous robots, we assumed that the locations of all three ground robots can see each other using a perspective camera. With this information, we can compute the intra-distance between any two robots using the algorithm presented in [Spletzer, Das et al. 2001]. At first, the intra-distance between the robots is obtained up to a scale factor and the absolute distance is measured using the geometric configuration of the robots while they are statically placed in the environment. Our goal was to localize the wall-climbing robot operating on the ceiling with respect to the three ground robots. With the aid of the camera and the application of P3P problem to the scenario, the wall-climbing robot can self-localize with respect to the ground robots in the team by solving the P3P problem and identifying the unique solution (section 2.3). Based on the relative pose information among the robots, the partial maps constructed by individual robots can be fused together to generate a

complete 3D map in real-time. To construct 3D maps of relatively large scale indoor environments, the robots need to move around in the environment and acquire many range maps. Our approach can be extended to incrementally construct a large map by taking one robot in the team as a reference point and move the other robots to designated positions by applying maneuver procedures [Feng, Zhu et al. 2006]. The known issue with solving the P3P problem was to identify the unique solution. We proposed two algorithms to identify the unique solution after solving the P3P problem by a known movement of the wall climbing robot. The P3P problem was applied twice and solved, once before the movement of the robot and once after the movement of the robot. This allows us to numerically examine each solution. The unique solution has to comply with the known movement of the wall-climbing robot. Initially, the wall-climbing robot was restricted to the linear motion and solved deterministically (section 3.2). Later, we propose an algorithm (section 3.6), to obtain the unique solution to P3P problem by any arbitrary movement of the wall climbing robot using a probabilistic approach.

3.2 Deterministic linear movement algorithm

If we know the relative distances among the three ground robots, we can use the Grunert's solution to P3P problem to estimate the pose of the perspective camera on wall-climbing robot with up to four solutions, but only one of them is a true solution. In this section, we introduce a deterministic linear movement algorithm that eliminates pseudo-solutions. The idea of this algorithm is straight forward: if the wall-climbing robot moves along a straight line, the genuine solutions should also keep along a straight line. On the other hand, pseudo-solutions geometrically will not align in a straight line. Thus, we eliminate the latter and identify the unique solution.

The algorithm is composed of following steps:

1. The three ground robots localize with respect to each other by co-operative localization and then remain stationary. Camera on the wall-climbing robot takes a snapshot image of the three ground robots and estimates its own pose with respect to the ground robots using P3P-pose estimation algorithm. It may result in up to four solutions of pose.
2. Wall-climbing robot moves in steps along a linear trajectory to two different positions and acquires two additional snapshot images of three ground robots at both locations. At each step, the robot computes the relative pose at all three locations and obtains (up to four) of solutions at each location of the wall climbing robot.
3. Now we have three groups of valid solutions, which provide up to $4^3 = 64$ solution combinations. By using the linear motion constraint of the robot movement, we perform steps 4 & 5 on all these solution combinations.
4. For each solution combination, determine a line by the first camera pose solution and the third camera pose solution. Calculate the distance of the second camera pose to the line, record it as the *error* of current solution combination.
5. Take the solution option with the smallest *error* as the set of genuine solutions along the robot motion.

We conducted a simulation experiment to verify the algorithm which is reported in [Feng, Zhu et al. 2007]. Fig. 3.2 shows three ground robots (+), position (o) and orientation (-) of the wall-climbing robot at three instances (blue, red and green). There are three (blue), two (red), and three (green) solutions at the three time instances respectively generated by Grunert's solution to the fourth order equation in solving the P3P problem. The deterministic algorithm can identify the unique solutions that agree with the linear trajectory.

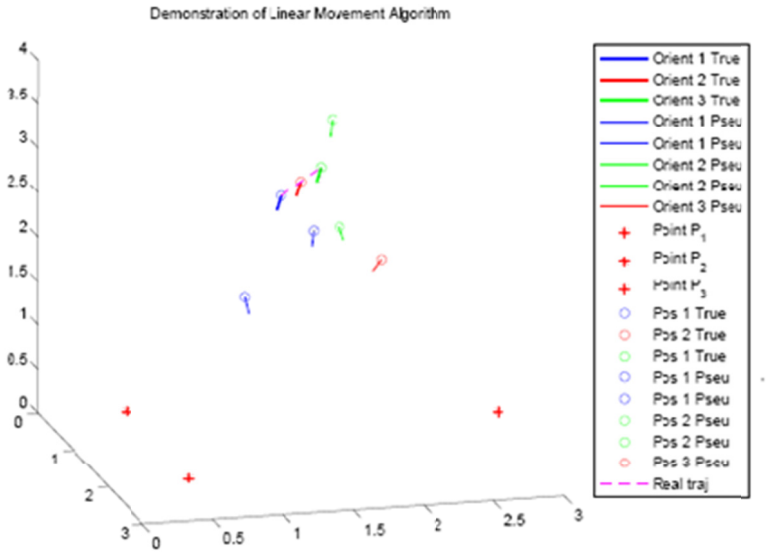


Fig. 3.2 Deterministic algorithm shows three instances of solutions to the P3P problem and identification of three unique solutions at each instance following the linear path. (Three circles (Blue, Red, Green) passing through the line is the real trajectory)

3.3 Map construction

This section enumerates steps involved in map construction using multiple heterogeneous robots (three ground mobile robots and one wall-climbing robot on the ceiling), which is composed of four basic steps. The steps are repeated to extend the map to other locations as the robots navigate in the environment.

1. The three ground robots perform intra-robot localization using the Pan-Tilt-Zoom (PTZ) cameras. It is assumed that the robots can mutually view each other (robots lie in each other's FoV). The intra-distance between the multiple robots can be calculated using the computer vision algorithm [Spletzer, Das et al. 2001] based on mutual visibility.

2. The wall-climbing robot captures a snapshot image of the ground robots with the markers. The wall-climbing robot follows a linear motion while capturing images of ground robots at multiple instances. At each instance, the P3P-pose estimation algorithm yields multiple solutions of relative pose of which only one is the correct solution. By using the deterministic linear movement algorithm, we can obtain the unique solution to the P3P problem, i.e., the relative pose (rotation-translation) of the camera with respect to the four robots can be determined at each step of the wall-climbing robot.
3. After the robots know their relative geometric formation, each robot keeps stationary and generates a point cloud map in their local coordinate system using the 3D range scanner.
4. The fixed relative pose between the camera and the range scanner on each robot is calibrated beforehand (see section 3.4). Final transformation is computed between the range images and applied to the range images to fuse the four local point cloud maps into a global map.

3.4 Calibration between the camera and the range scanner

The map construction using range scanners in section 3.3 involves estimating the relative pose between the two robots (Sensor Coordinate Frame (SCF) of the range scanners). However, the camera pose estimation algorithm computes the relative pose between the camera coordinate frame (CCF) on one robot and the RF (also the SCF) of the other robot. The CCF and the SCF on the same robot have a small transformation between them. This transformation is obtained by a calibration technique between the camera and range scanner. To acquire the exact translation-rotation relationship between the camera and the 3D range scanner pair on each robot, a preliminary one-time calibration process has to be carried out. Because we are calibrating two different

types of sensors, we need to establish a similar metric for measuring the depth of several points in Euclidean space. We designed a “calibration cube” for this specific application shown in (Fig. 3.3). The depth image of the 3D range scanner is acquired and converted into an image whose intensity is directly proportional to the depth. Also, the perspective camera image of the calibration cube is acquired. The corresponding points in both intensity images and the camera images were manually selected. A minimum of 6 or more corresponding points between the two images was used to compute the homography, which leads to the relative translation and orientation between the two inherently different sensors.

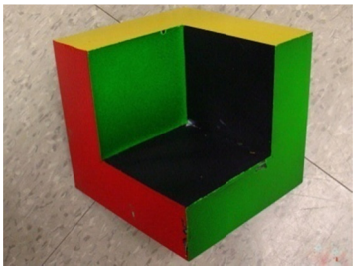


Fig. 3.3 Calibration cube to obtain relative pose of the camera and 3D range scanner

3.5 Dual-heterogeneous robot mapping

The maneuvering complexity of multiple heterogeneous robots grows with the number of robots present in a team. A solution to this problem is to use a minimum number of robots and taking advantage of the pose estimation techniques using the camera. This led us to a scenario, which utilizes just dual robots consisting of a ground robot and a wall-climbing robot to map indoor environments. This scenario is an improvement in terms of maneuvering freedom for the robots over previous attempt to map the indoor environment using three ground robots and a wall-climbing robot. The robots are constrained to a lesser extent and the only requirement is for the ground robot to lie in the FOV of the wall-climbing robot while computing the transformation between the

range images acquired by the two robots. All three LEDs placed on a single ground robot also eliminate the need for multiple robots and the intra-group localization step between the ground robots.

Fig. 3.4 shows an experimental setup of the dual-robot system consisting of a ground robot and a wall-climbing robot operating on the ceiling to map the indoor environment.

The camera pose estimation algorithm in section 2.3 computes the geometric relationship (i.e., the transformation matrix) between the wall-climbing robot and the ground robot. The overhead camera on the wall-climbing robot captures 2D images of the 3 blinking LED lights (control points) that are placed on the ground robot. To date, all attempts to solve the P3P problem have resulted in multiple solutions, out of which only one solution is valid [Quan and Lan 1999]. Earlier, we presented an algorithm that

determines a valid solution to the P3P problem when the wall-climbing robot moves in a linear trajectory [Feng, Zhu et al. 2007]. In this section, we present a novel particle filter algorithm that probabilistically determines the valid solution to the P3P problem when wall-climbing robot moves with an arbitrary trajectory. The error analysis of the camera pose estimation algorithm reveals that the solution to the P3P problem is bound to have errors if noisy images are considered. However, the result from the P3P problem is accurate enough to be considered as a good initial estimate for the scan registration algorithm to further refine the accuracy of the transformation matrix. Initially, we used the ICP algorithm [Besl and McKay 1992], which is guaranteed to converge into the global minimum with a good initial estimate. Another advantage of initialization is that the number of iterations of the ICP algorithm to register the laser scans is reduced compared to ICP algorithm with no initial transformation input. Other scan matching algorithms can be used for refining the transformation matrix such as the Normal Distribution Transform, which requires the initial position to be found by preview or odometry [Takeuchi and Tsubouchi 2006], and the Angle Histogram technique [Weib, Wetzler et al. 1994]. It is known that these scan matching algorithms including the ICP algorithm fail to converge to a global minimum without a good initialization especially when there is a large rotation and translation transformation between the partially overlapping range images. This is the case when we consider scan registration of two range images acquired by the ground robot and the wall-climbing robot (one scan is inverted with respect to the other scan if the wall-climbing robot is on the ceiling).

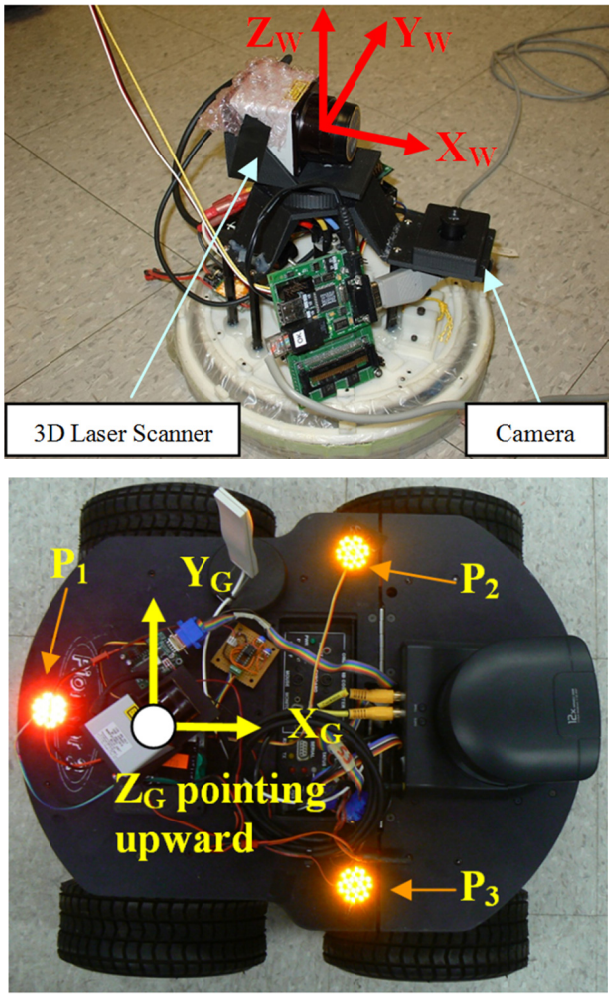


Fig. 3.5 (Left) Wall-climbing robot equipped with a 3D range scanner, a camera and a Stargate® embedded processor. (Right) Ground robot equipped with 3D range scanner and 3 LED clusters (markers).

We obtain the solution to the P3P problem, which is the pose estimate of the camera coordinate frame $(X_w Y_w Z_w)$ with respect to a reference coordinate frame $(X_G Y_G Z_G)$. The control points are spatially known with respect to the reference coordinate frame. The image location of each of these three points is obtained from an image snapshot acquired by the camera. The spatial location of the control points and its image

locations form the input to the Grunert's algorithm (which provides solutions to the P3P problem) in addition to camera calibration [Strobl, Sepp et al. 2006] parameters, namely focal length, principal point and scale factor. Grunert's solution [Haralick, Lee et al. 1991] to P3P problem leads to solving a fourth order polynomial

$$A_4 v^4 + A_3 v^3 + A_2 v^2 + A_1 v^1 + A_0 v^0 = 0 \qquad 3.1$$

From Eqn. 3.1, we obtain up to four solutions, which are the length (s_1, s_2, s_3) of the sides of the tetrahedron (Fig. 3.6) formed by connecting the center of perspective (O) of the camera and the three control points (P_1, P_2, P_3). After we compute the length of the sides, given the three point locations with respect to ground robot coordinate frame, we can determine the position of the camera center (O) and the camera's orientation as described in [Fischler and Bolles 1981] (Refer to Appendix A.1, A.2 for mathematical steps).

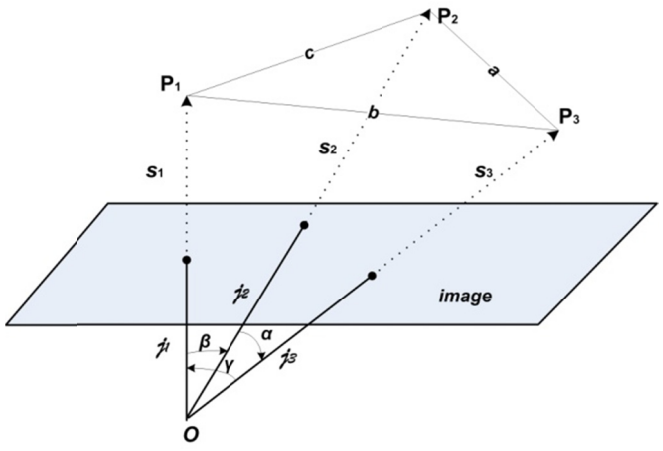


Fig. 3.6 Tetrahedron formed by connecting the center of perspective of camera (O) and the three control points (P_1, P_2, P_3) are spatially known with respect to a reference coordinate system (In our case, the ground robot frame).

At this stage, we have up to four sets of poses of the camera, of which only one of them is a valid solution. In the following section, we obtain a unique solution using a probabilistic algorithm.

3.6 P3P-Particle Filter (PF) algorithm

The number of solutions obtained by solving the P3P problem using Grunert's algorithm always results in more than one solution and up to four solutions. The algorithm presented in this section tackles the problem of identifying the valid solution which represents the real pose of the camera with respect to the RF of the ground robot. In the first instance, the wall-climbing robot acquires first set of images of the three control points (blinking LED lights) on the ground robot. The image is then processed (image subtraction algorithm identifies image locations of the three blinking LED lights) to locate three control points. After solving the P3P problem using the pixel information of the three control points, we obtain multiple solutions (i.e., the pose of the overhead camera) to the P3P problem. Next, the wall-climbing robot is commanded to move to a new position in any arbitrary direction and the odometry records the new position with respect to its previous pose. At the second instance (time after wall-climbing robot moves to a new position), the wall-climbing robot captures another set of images of the ground robot and is processed to obtain second set of solutions by applying the P3P problem. At this time, we have two sets of solutions to P3P problem and the odometry data of the wall-climbing robot. Below, we describe an algorithm inspired by particle filter algorithm [Thrun, Burgard et al. 2005] to determine the valid solution at two instances from the two sets of multiple solutions (Refer to **Algorithm 1**).

Prior Distribution: At first instance $(t-1)$, multiple solutions to the P3P problem are considered as *particles* and each of them is assigned an equal weight. If the number

of solutions obtained is m, each of the particles is assigned an equal weight of $1/m$. At this instance, we don't know which among them is the valid solution.

Prediction stage: At second instance (t), the prediction stage incorporates the odometry data from the wall-climbing robot and predicts the new location of (O) for all particles from its previous pose at time $(t-1)$. The odometry motion model of the wall-climbing robot follows the probabilistic approach where the new prediction samples are drawn from the distribution $p(x_t|u_t, x_{t-1})$, where u_t is the control input (odometry), x_{t-1} and x_t are the pose of the wall-climbing robot at time $(t-1)$ and t respectively.

Odometry motion model: Typical odometry measurements are influenced by rotational error and translational error [Fox, Burgard et al. 1999]. The odometry measurements are represented in the internal robot coordinate frame. The control input is given by

$$u_t = \begin{bmatrix} \bar{x}_{t-1} \\ \bar{x}_t \end{bmatrix} = \begin{bmatrix} (\bar{x}, \bar{y}, \bar{z}) \\ (\overline{x'}, \overline{y'}, \overline{z'}) \end{bmatrix} \quad 3.2$$

The hypothesized pose of the robot at time $(t-1)$ and t is given by $x_{t-1} = (x, y, z)$ and $x_{t-1} = (x', y', z')$.

The algorithm for computing the new samples using the posterior distribution $p(x_t|u_t, x_{t-1})$ is in **Algorithm 1**. The inputs to the algorithm are odometry input u_t and previous pose x_{t-1}. In algorithm 1, the lines 1 and 2 compute the measured rotation and translation of the wall-climbing robot on the ceiling using odometry input. Lines 3 and 4 add random noise to the odometry based on experimental (systemic error) evaluation. α_1 and α_2 are decided after measuring systemic error (assumed as Gaussian) of the odometry. Lines 5, 6 & 7 output the new position of the wall-climbing robot's camera center. Notice in line 7 that the z-coordinate remains the same since the wall-climbing robot is assumed to move only on a 2D plane (ceiling). The new sampled particles are denoted by $\bar{x}_t^{[m]}$.

Algorithm 1 Odometry Sample Motion Model (u_t, x_{t-1})

1. $\bar{\delta}_\theta = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x})$
2. $\bar{\delta}_T = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2}$
3. $\hat{\delta}_\theta = \bar{\delta}_\theta + sample(\alpha_1.|\bar{\delta}_\theta| + \alpha_2.\delta_T)$
4. $\hat{\delta}_T = \bar{\delta}_T + sample(\alpha_1.|\bar{\delta}_\theta| + \alpha_2.\delta_T)$
5. $x' = x + \hat{\delta}_T.\cos(\hat{\delta}_\theta)$
6. $y' = y + \hat{\delta}_T.\cos(\hat{\delta}_\theta)$
7. $z' = z$
8. **return** (x', y', z')

Update Stage: At time t, the wall-climbing robot generates a new set of multiple solutions ($z_t^{[n]}$) using Grunert's algorithm to P3P. n is the number of solutions at time t. The weight of each particle is obtained based on the proximity of the predicted particle. We choose the Radial Basis Function (RBF) kernel as weighing function Eqn. 3.3, often used in non-parametric estimation techniques. The kernel generates a non-negative weight (ranging from 0 to 1), and is dependent on the Euclidean distance, a measure of proximity between the prediction particles $x_t^{[m]}$ and the new estimate $z_t^{[n]}$ from the camera sensor.

$$w_t^{[m]} = K\left(z_t^{[n]}, x_t^{[m]}\right) = exp\left(\frac{-\left(z_t^{[n]} - x_t^{[m]}\right)}{2\sigma^2}\right) \qquad 3.3$$

It provides a simulation result of the particle filter algorithm to determine the valid solution to P3P problem where the wall-climbing robot moves along a sinusoidal trajectory. At four instances, the pose of the camera is calculated by solving the P3P

problem and applies the particle filter algorithm to identify, which is the valid solution at each instance.

3.7 Limitations of the P3P-PF algorithm

The accuracy of camera pose estimated by the P3P-Particle Filter algorithm is limited since it depends on the closed-form solution as computed by the Grunert's method. The pixel noise in the 2D image propagates into an inaccurate estimate of pose if the camera is not accurately calibrated. The closed-form solution accuracy is comparable to that of the iterative camera pose estimation algorithms, but not better than them. However, the closed-form solution is faster than all iterative algorithms available for camera pose estimation. The pose accuracy is a function of the size of the target and the distance from the camera coordinate frame to the target. Some factors that affect pose accuracy are the quality of camera lens and the camera sensor resolution (error in quantization). The closed-form solutions are mostly analytical in nature and computed using the geometric configuration of the camera frame and the target. In these cases, there is no way to minimize the pixel positional noise such as the use of least square minimization technique in iterative algorithms. The particle filter algorithm also depends on initial estimate from odometry of the mobile robot to predict initial state of pose. This could be a drawback in cases where the odometry of the robot is not accurate e.g., robot moving on a slippery surface.

Algorithm 2 Unique Solution to P3P (X_{t-1}, u_t, z_t)

Input X_{t-1}: Pose of the robot at time $t-1$

u_t: Odometry readings at time t

z_t: New pose readings from grunert's solution to P3P problem.

Output x_t: Valid unique solution to P3P at time t

1. $X_t = 0$ (Empty set of new particles)

Initialize particles weight (at time T =0)

2. $w_{t-1}^{[m]} = \frac{1}{m}$

while (Wall-climbing robot steps**) do**

for $\boldsymbol{j = 1\ to\ M}$ (at time $T = t$)

3. $x_t^{[m]} = p(x_t^{[m]}|u_t, x_{t-1}^{[m]})$

(State transition model)

4. $w'^{[m]}_t = p\left(z_t^{[n]}|\bar{x}_t^{[m]}\right)$

(Sensor model-compute importance factor)

5. $\bar{w}_t^{[m]} = w_{t-1}^{[m]} + w'^{[m]}_t$ (Merge the weights)

6. $w_t^{[m]} = \frac{\bar{w}_t^{[m]}}{\sum_{j=1}^{m} \bar{w}_t^j}$ (Normalize the weights)

7. $X_t = X_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$ (Add to new set)

end for

end while

(Importance sampling: pick the particle with max weight)

8. **return** X_t, $x_t = x_t^j \in X_t, where\ w_t^j = \max(w_t^{[m]})$

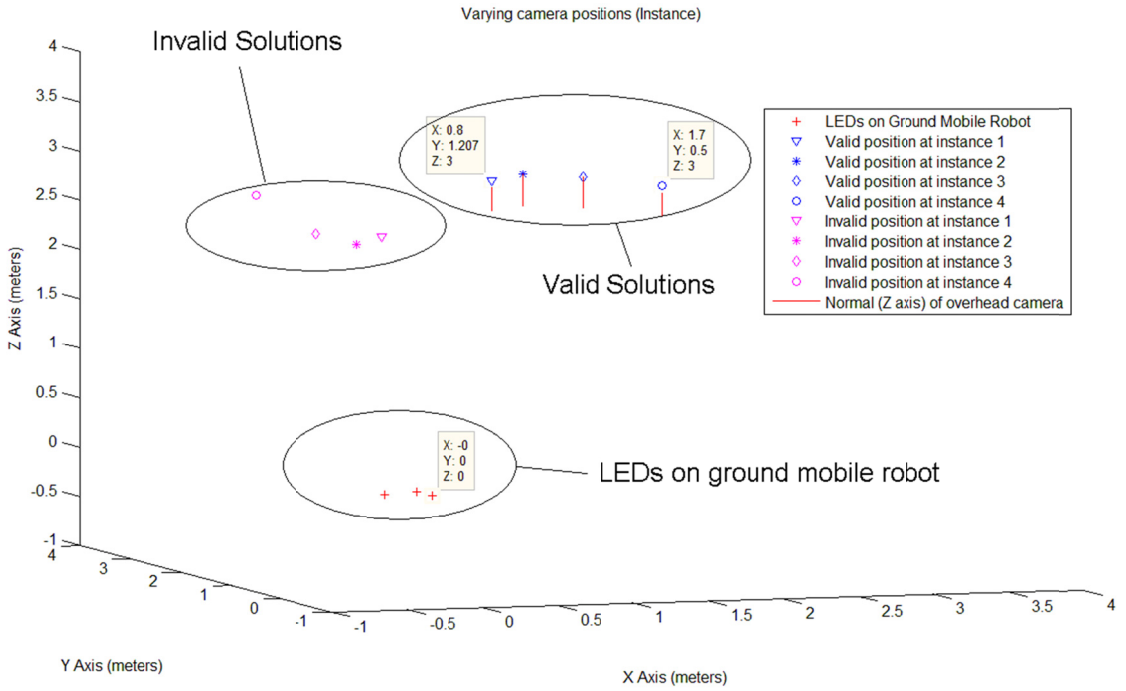


Fig. 3.7 Plot of solutions (pose estimate) to the camera pose estimation algorithm (P3P-PF). Blue markers (∇, *, ◊, o) indicate wall climbing robot's real position as it moves in an arbitrary trajectory and green markers (∇, *, ◊, o) indicate invalid position at four instances. (+) indicates the three blinking LEDs on the ground robot. The wall climbing robot movement can be seen following a sinusoidal wave in agreement to programmed input.

3.8 Error analysis of camera pose estimation algorithm

In this section, we consider the effect of noise (related to pixel positions) on images and how it affects the vision-based algorithm's performance. The camera pose estimation algorithm computes the relative pose based on two inputs: pixel positions of the control points and their Euclidean positions with respect to the RF. Based on these two inputs, the error can be classified into two kinds that can affect the final solution: one is the pixel position error in the images due to lens distortions, which is assumed as the Gaussian noise, and the second type of error is related to the locations of the three

control points in Euclidean space. The latter with the slightest error leads to erroneous results in pose estimation i.e. the Grunert's method to solving the P3P problem tends to be unpredictable without an accurate input of measurement of the distance between the control points. We simulate the pixel position error[1] by adding Gaussian random noise to the pixel location of three control points as seen by the camera. The error is measured by Root Mean Square (RMS) estimation error, which is used to represent the difference between the true position and the estimated position, given by

$$\varepsilon_{est} = E\left[\frac{\left\|\hat{X} - \bar{X}\right\|^2}{N}\right]^{1/2} \qquad 3.4$$

where $\hat{X} = (\hat{x}, \hat{y}, \hat{z})$ is the estimated position and $\bar{X} = (\bar{x}, \bar{y}, \bar{z})$ is the true position, N is the number of simulation runs ($(N = 1000)$ in experiments). Table 3-1 lists RMS estimation error for varying pixel error and focal length of the camera. It also indicates that the error in focal length also affects the final solution and is inversely proportional to the RMS estimation error. The pixel positional error contributes directly and proportional to the RMS estimation error.

3.9 Range image fusion

The solution obtained by the camera pose estimation algorithm is slightly error-bound due to pixel positional noise and therefore provides a good approximation of the real pose as described in Section II. This P3P-particle filter algorithm has two distinct advantages: 1) It runs extremely fast (it took about 0.001919 sec to execute in MATLAB 7.6.0 on PC with Intel Core 2 Duo Processor 6600 @ 2.4GHz, 2GB RAM); and 2) It

[1] The pixel positional error stems from the lens distortion, which is reflected in the images. Camera calibration is an effective way to minimize the positional error.

provides an initial pose estimate (approximating real pose) between dual robots to fuse overlapping range images using the ICP algorithm.

3.9.1 Preprocessing range image data

We obtain (220×680) 3D range data points from the range scanner on the ground robot and the wall climbing robot. The ICP algorithm is used to register the two overlapping range scans and derive the relative transformation between them. The time complexity of the ICP algorithm is $O(n^2)$, where n is the number of scan points. We used an improved ICP algorithm that applies k-D tree search in establishing closest point correspondence, which reduces the computational complexity to $O(n.logn)$. The algorithm is still slow due to the size of the input data (large point clouds) and needs a good initial estimate for guaranteed convergence. Sub-sampling of range images leads to a slightly faster convergence.

In addition, the range images contain noisy data points and outliers that inhibit the ICP algorithm from converging to the global minimum. Hence, it was necessary to de-noise the range images. The noisy data appears in the scan array as the smallest possible distance (0.19m) and sometimes as the maximum distance (4m) among the other range data points obtained by Hokuyo URG-LX® range scanner. We employed a Gaussian filter to remove the noisy data points and the outliers. The Gaussian filter assumes that the neighboring points do not change position drastically. It identifies the outliers based on the proximity to its nearest neighbors and classifies a data point as an outlier if it is far away from all its neighbors. After eliminating the outliers, empty bins are created in the scan array, which are replaced by median interpolation. After data pre-processing, we compute the transformation matrix using the ICP algorithm. Note that the ICP algorithm eliminates the need to perform camera / range scanner calibration. Following section presents experimental results after fusing range images.

Table 3-1. RMS estimation error of the solution to the P3P problem with varying focal length of the camera and pixel position error defined by (μ,σ)

Focal Length (m)	Estimated Error (μ)	Estimated Error (σ)
Pixel Error: Gaussian noise with σ^2 = 3		
0.016	0.0815	0.0407
0.026	0.0522	0.0260
0.036	0.0370	0.0178
Pixel Error: Gaussian noise with σ^2 = 5		
0.016	0.1072	0.0551
0.026	0.0641	0.0334
0.036	0.0481	0.0230
Pixel Error: Gaussian noise with σ^2 = 10		
0.016	0.1447	0.0712
0.026	0.0917	0.0462
0.036	0.0674	0.0338

3.9.2 Experimental Results

In our experiments, the wall-climbing robot operates on the ceiling and the ground mobile robot operates on the floor as shown in Fig. 3.4. In the first instance, the wall-climbing robot takes a set of images of the ground robot and is processed to obtain the locations of the three control points. The Grunert's method is used to compute multiple solutions of the pose of the overhead camera. In the second instance, the wall-climbing robot moves to a new position and takes another set of images of the ground robot and is processed. Grunert's method computes another set of solutions at the second instance. The particle filter algorithm picks the solution that is valid in both instances.

Finally, the ground mobile robot and the wall-climbing robot each acquire a 3D range image of the indoor environment.

We present the experimental results in the form of a 3D point cloud. The red and the blue dots indicate the 3D point clouds acquired by the ground mobile robot and the wall-climbing robot respectively. Fig. 3.8 indicates no transformation applied to the laser scans. The wall climbing robot is inverted with respect to the ground mobile robot. As seen in Fig. 3.8, it appears that the two range images represented in their respective local coordinate frames (SCF) acquired by the two robots appears to be inverted with respect to each other. The range images are shown with respect to their respective local coordinate frames without any application of transformation results obtained by the pose estimation algorithms. It can be noticed that the table top remains at the top since the two scans are displaced. With no P3P initialization, the ICP algorithm performs poorly (Fig. 3.9) and considers closest fit of the points. It ignores the rotation of 180° about the Y axis (inverted) between the two scans and converges to a local minimum.

It is possible to acquire an initial orientation estimate of a robot using orientation sensors. However, there is the problem of perpetual drift with orientation sensors with magnetic interference. A more viable option of camera pose estimation provides a good pose estimate between a ground robot and the wall climbing robot, which is an example of vision-based localization. Further, the ICP algorithm refines the transformation matrix based on the initial estimate provided by the vision-based algorithm. After applying the resulting transformation matrix, the two laser scans are fused together as seen in (Fig. 3.10). shows the numerical results of pose estimation; It summarizes the ground truth and the ICP output with and without P3P initialization. A qualitative evaluation of the fused range image data is performed manually to ensure that the two scans were fused well. The numerical results indicate a translational error noticeable along z-axis. Rotational error was small (1~2°) in scan matching in one axis as seen in Fig. 3.11, which

can be attributed to sensor noise. A quantitative measure by comparing closest points would not be appropriate as the real correspondences between the range image points cannot be established.

Table 3-2. Experimental results showing relative pose between the ground robot and wall-climbing robot laser scans

Rotation Matrix - R			Angles (deg)		Translation (m)	
No Initialization						
0.3327	-0.9429	0.0170	α_x	+1.306	X	-0.1279
0.9429	0.3324	-0.0203	β_y	-0.773	Y	0.1247
0.0135	0.0228	0.9996	γ_z	+70.564	Z	-0.1742
P3P Initialization						
-0.1418	0.9897	-0.0211	α_x	-178.808	X	0.1166
0.9899	0.1419	-0.0002	β_y	-0.154	Y	0.0897
0.0027	-0.0208	-0.9998	γ_z	+98.152	Z	2.0160
Ground Truth						
-0.1392	0.9903	0.0000	α_x	-179.719	X	-0.1000
0.9897	0.1391	-0.0349	β_y	+1.982	Y	-0.0500
-0.0346	-0.0049	-0.9994	γ_z	+98.006	Z	1.9500

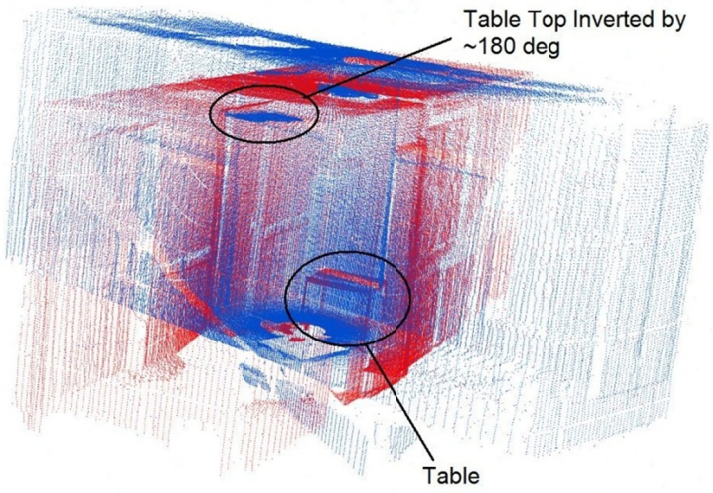


Fig. 3.8 Unmatched scans from the wall-climbing robot (blue) and the ground mobile robot (red) shown in the coordinate frame of the laser scanner

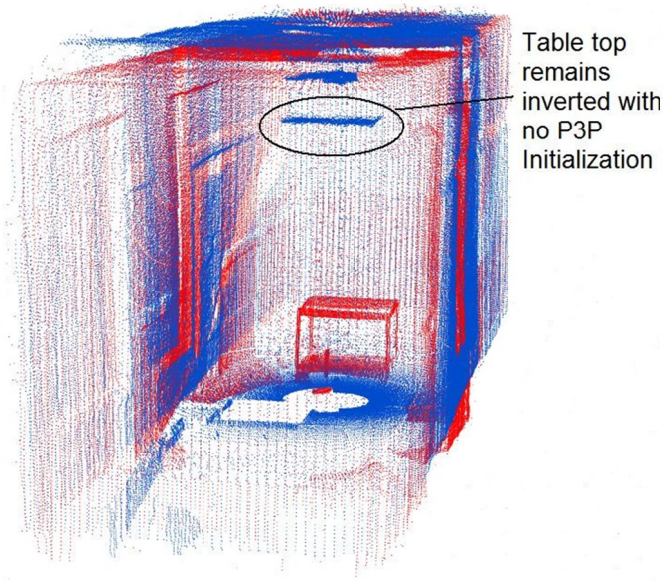


Fig. 3.9 Fused point clouds using the ICP algorithm with no P3P initialization. The scans are still inverted by 180° about Y axis in a right hand coordinate system

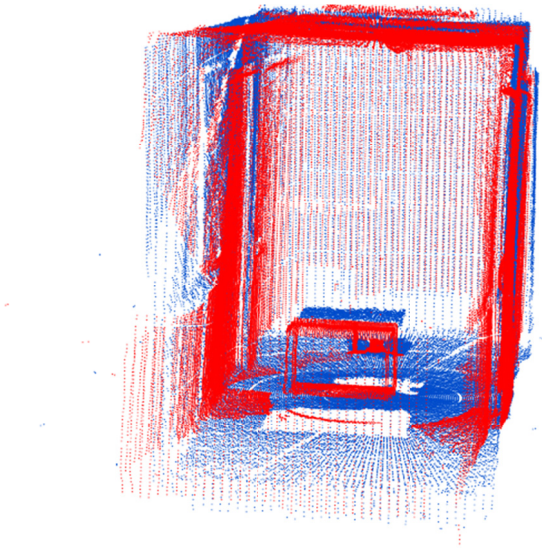


Fig. 3.10 Transformation (R, T) obtained from vision-based algorithm brings the two laser scans fused accurately

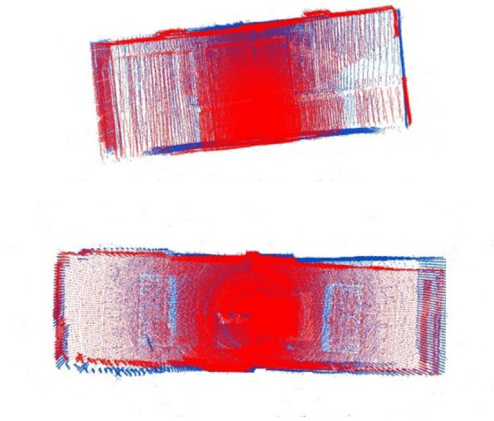


Fig. 3.11 Side view (left) and top view (right) of the matched scans after applying transformation (obtained from ICP algorithm initialized by the vision-based algorithm).

3.10 Analysis of scan registration with the ICP algorithm

In section 3.9, we presented fused range images after applying the transformation results obtained by the ICP algorithm in two cases, with and without initialization using

the pose estimates computed from the camera pose estimation algorithm. In this section, a detailed analysis of the ICP algorithm indicates the algorithm's performance in both cases and how the initialization of pose affects the performance of the scan registration algorithm. Especially, we look at the number of iterations that ICP algorithm performs to reach a particular threshold.

In the case with no initial estimation, the ICP algorithm converges to a local minimum (Fig. 3.12). The solid line indicates the ground truth. The small dotted line indicates the rotation and translation parameters obtained by the iterations of the ICP algorithm without the initial estimate. The rotation matrix is set to the diagonal identity matrix and translation vector as zeros. We can see that the rotation angle γ (small green dotted line in Fig. 3.12) is diverging from the global minimum and reaches (γ = -70.56°) versus the ground truth (82°). In the second case, we initialized the ICP algorithm with the relative pose obtained by the camera pose estimation algorithm. The rotation angles (α, β, γ) in the large dotted lines during the iterations performed fairly well and converged to the ground truth. Fig. 3.12 also shows the convergence of the ICP algorithm to the global minimum with P3P initialization. A good initial estimate to the ICP algorithm not only guarantees the convergence of the algorithm to a global minimum but also improves the computational speed of the ICP algorithm. The ICP algorithm takes a smaller number of iterations to converge with a good estimate. We designed software to simulate a 3D laser scan using the ray-polygon intersection and generated two mismatched point clouds (~ 16200 scan points) transformed by a large rotation (α_x = 90°, β_y = 10°, γ_z = 180°). With an initial estimate that is close to the real pose, the ICP algorithm converged faster. As seen in Fig. 3.13, the ICP algorithm converges in 29 iterations with a good initial estimate and converges slowly in 39 iterations with an average initial estimate.

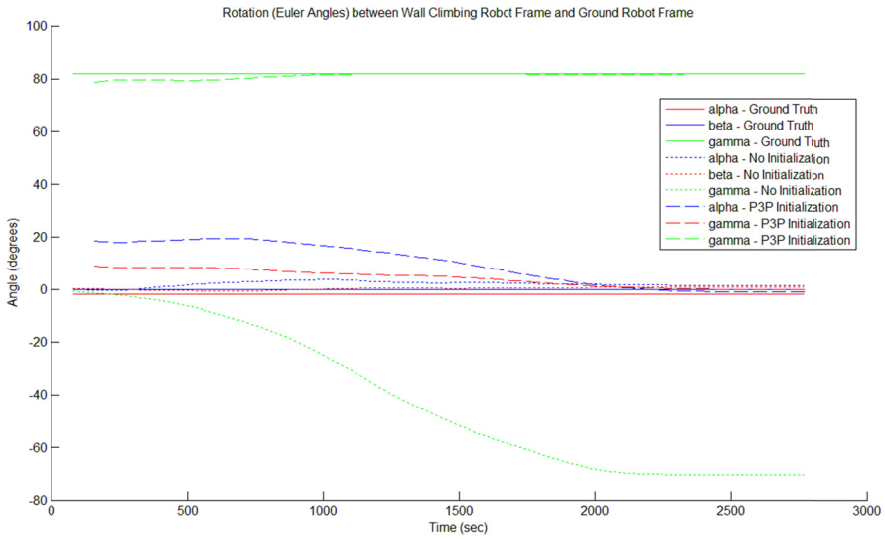


Fig. 3.12 Convergence of the rotation angles as computed by the ICP algorithm with and without initialization

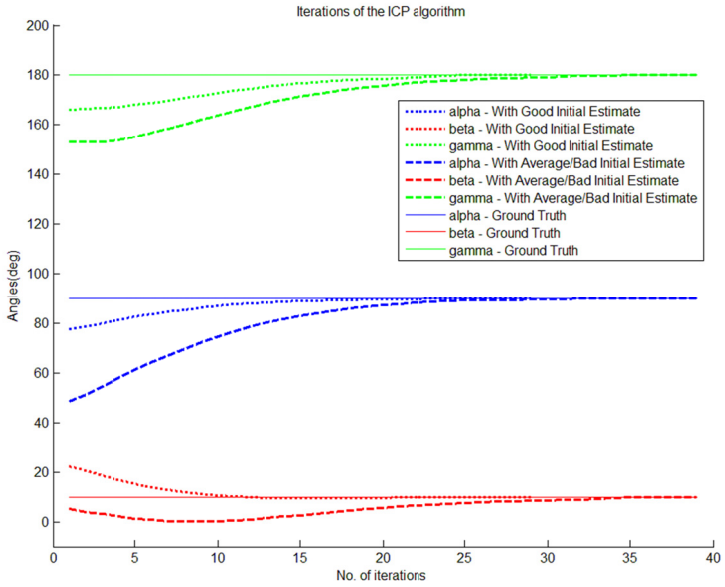


Fig. 3.13 Convergence of the rotation parameters as computed by the ICP algorithm in a simulated environment initialized by different pose values

The benefit of a good initial estimate for initializing the scan registration algorithm is evident from the above analysis. The initialization of relative pose guarantees convergence to a global minimum. The final transformation between the two range images is thus more accurate. The added advantage is that the speed of convergence improves with a better estimate. This chapter presented a real-time camera pose estimation algorithm that could be used in estimating the pose between multiple robots equipped with a camera and the visual markers. The accuracy of pose was sufficient to initialize scan registration algorithm to fuse multiple range images. In the next chapter, we consider different types of scan registration algorithms including point-based and feature-based approaches. In the process, we study the usefulness of extracting features from range images and scan matching with features such as polygons.

CHAPTER 4

Laser scan registration

3D range scanners are the most popular choice among the current state-of-the-art sensors for mapping / modeling in robotics and photogrammetry. These sensors acquire a 3D range image, which is a depth map of the surrounding environment and stored in a spherical coordinate system (r, θ, ϕ) with respect to the Sensor Coordinate Frame (SCF). The robots acquire a number of range images as the robot traverses through the environment. The most challenging aspect of map building is to fuse these local maps with respect to a Reference Frame (RF). It is assumed that the robot acquires range images in a stop-n-go fashion and the consecutive range images have a minimum overlap between them. A pair of consecutive range images is considered at a time for transforming them into a global RF. It is required to compute the transformation / relative pose between the SCFs at locations where the range images were acquired. Once the relative pose is determined, it is straight forward to transform the range images with respect to the RF using the relative pose information. The problem of computing the 6DoF relative pose $(x, y, z, \alpha, \beta, \gamma)$ between the partially overlapping scans is known as laser scan registration.

4.1 Point-based scan registration

The most popular algorithm for registering partially overlapping range images is the Iterative Closest Point (ICP) algorithm [Besl and McKay 1992]. This approach can used to register various representations of geometric data including point sets, line segment sets, implicit curves, parametric curves, triangle sets and free-form surfaces. The ICP

algorithm reduces the Least Square (LS) distance between the paired corresponding sets. Solution to pose by LS minimization (the ICP algorithm) can be classified into direct or indirect methods [Nüchter 2009]. A number of direct methods provide closed-form solution to the ICP algorithm, which are quicker compared to the indirect methods based on nonlinear LS minimization. A comparison of variants of the ICP algorithm and their performance evaluation can be found in [Rusinkiewicz and Levoy 2001]. The success or failure of registering range images using the ICP algorithm is mainly dependent on establishing correspondences between the pair of overlapping range images. ICP algorithm is an iterative steep descent algorithm. The computation time of the algorithm is directly dependent on number of input scan points.

4.2 Features-based scan registration

One alternative way to scan registration and reduce the computation time is to reduce large data points in the range image to a more compact form (discussed in section 4.3) and then register the range images using compacted data set. A compact form of representation is an act of feature extraction from range images. Some useful features previously explored are normal distribution, planes, spin-images, lines, edge and triangular meshes. A detailed comparison of registration techniques using two prominent forms of compact representation viz. edge representation and triangular mesh representation is presented in [Specht, Sappa et al. 2005]. Another approach deals by minimizing the distance between the points and the tangent plane of the overlapping digital surfaces [Chen and Medioni 1991]. This approach does not need to establish corresponding point-pairs and thus eliminate false correspondences leading to slower convergence. Feature-based registration of overlapping range images is proven to an effective way to reduce computation time compared to point-based approaches [Chen

and Medioni 1991; Sequeira, Ng et al. 1999]. In [Carmichael and Herbert 1998], spin-images [Johnson 1997] were extracted from images and used for scan matching.

4.3 Range image segmentation

Some common features that can be easily extracted from range images in indoor environments are edges and planar segments (polygons). The most difficult representation of range images involves representation of range data points that represent non-planar surfaces. Benefits of feature extraction include a compact representation of range images, which lead to efficient storage and visualization. Further, these compact representations can be used as input to the scan registration algorithms based on feature matching. Some examples of the feature extraction include [Coleman, Scotney et al. 2010; Kasvand 1988; Ye and Hegde 2009]. They deal with extraction of edges from 3D range images. Edge detection in range images is a non-trivial problem unlike in image datasets. The range images store depth as information and needs to be scaled appropriately at different places and needs to be adaptive. This is unlike the intensity images as they do not need any preprocessing. Most recent attempt [Coleman, Scotney et al. 2010] applied Laplacian operators on range data to extract edges. In the following sections, we discuss state-of-the-art edge extraction techniques. We also introduce an efficient map representation using polygons as features. This type of feature is especially beneficial in indoor environments with large planar surfaces. We discuss the polygon extraction while focusing on mapping indoor environments as our main application. We extend our multi-robot mapping techniques work using a dual robot system that maps the interiors of an office corridor.

4.4 Application: Mapping with homogeneous dual-robot systems

In the context of a multi-robot system, 3D map building involves computation of relative pose parameters in 6 DoF namely; the three Euler angles $(\alpha_z, \beta_y, \gamma_x)$ and the translation vector (t_x, t_y, t_z) between the robots sensor coordinate frames. Further, as the robots navigate in the 3D space, the 3D range map and the pose parameters of the robots are transformed with respect to the global reference frame. The range maps acquired by the robots at different locations are fused together by applying the pose transformation to form a complete 3D map. The dual robots include two ground robots. Each ground robot is equipped with a 3D range scanner, a perspective camera and cube with visual markers on all sides, which is useful in tracking the robots and their localization using vision.

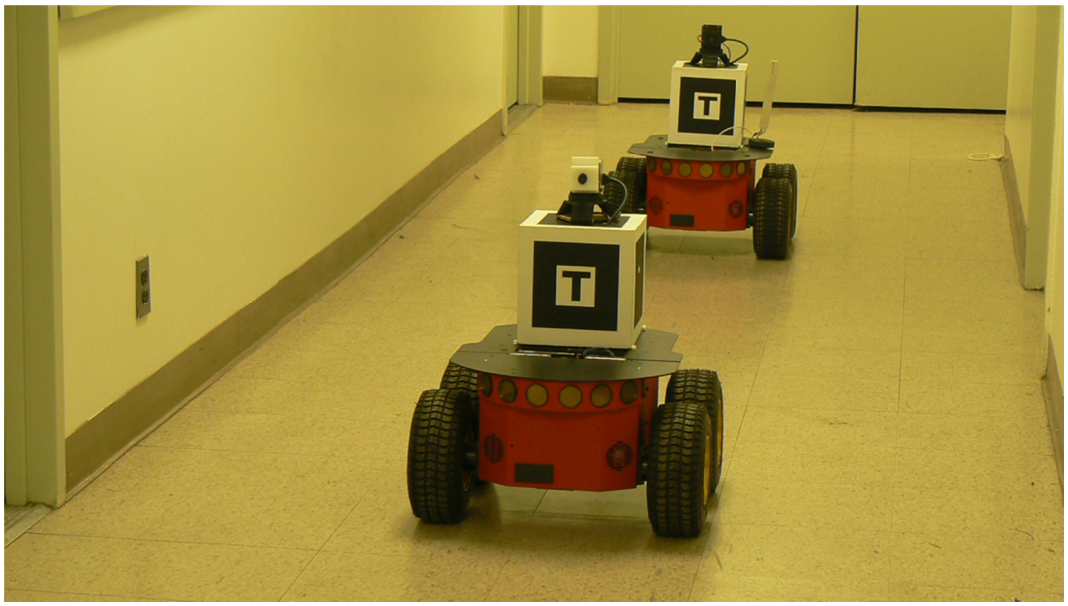


Fig. 4.1 Two Pioneer® ground robots mapping the indoor environment using the vision-aided laser mapping algorithm.

An indoor environment with featureless long corridors is difficult to map by single robot alone assuming the initial odometry estimate on smooth surfaces is error-prone. The odometry data becomes unreliable over long distances and most visual odometry algorithms need features that can matched with respect to each other. In such a scenario, a dual-robot system would be ideal for solving the 3D mapping problem where each robot can carry the marker and visually identify each other. In addition, we obtain the initial estimate of pose between the robots by vision-based relative pose estimation, which is accurate and can be computed in real time as discussed in section 2.4. The robots carry the markers and do not have to rely on the surrounding environment for features for self-localization. The robots are commanded to move in tandem and their global pose is updated over various scan locations with respect to a global coordinate frame.

4.4.1 Scenario and system architecture

The two ground robots acquire 3D range images in stop-n-go fashion and move in tandem without visually losing sight of the robot in the front as shown in Fig. 4.2. At any given instance, *trailing* robot holds its position and the *forward* robot moves to a new position in the environment. At this moment, both ground robots acquire the 3D laser scan, which will have a partial overlap of the surrounding environment. The trailing robot acquires an image of the forward robot with the marker. The relative pose between the two robots is computed using the two-step approach: camera pose estimation and scan registration. Then, the pose of both the robots are updated with respect to a global coordinate frame. After these steps, the trailing robot and forward robot switch their duties and the relative pose is estimated all over again. The whole process is repeated until the entire environment is mapped. Fig. 4.3 shows a schematic

flow chart of the steps involved in 3D mapping and is a snapshot of the processes used in estimating the pose between two robots.

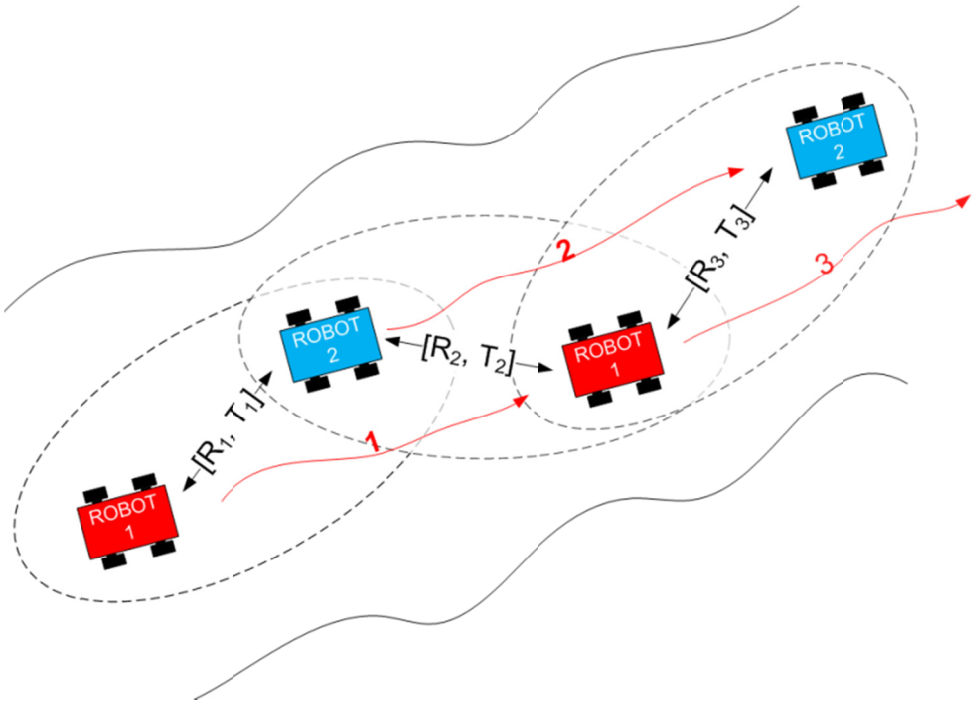


Fig. 4.2 Dual Robots, namely Robot1 (red) and Robot2 (blue) alternately take new positions after their relative pose (R_i, T_i) is established by both vision and scan registration algorithm. Once their global positions are updated, the trailing robot moves to the forward post and process is repeated. The robots move in tandem while updating their global pose at every step.

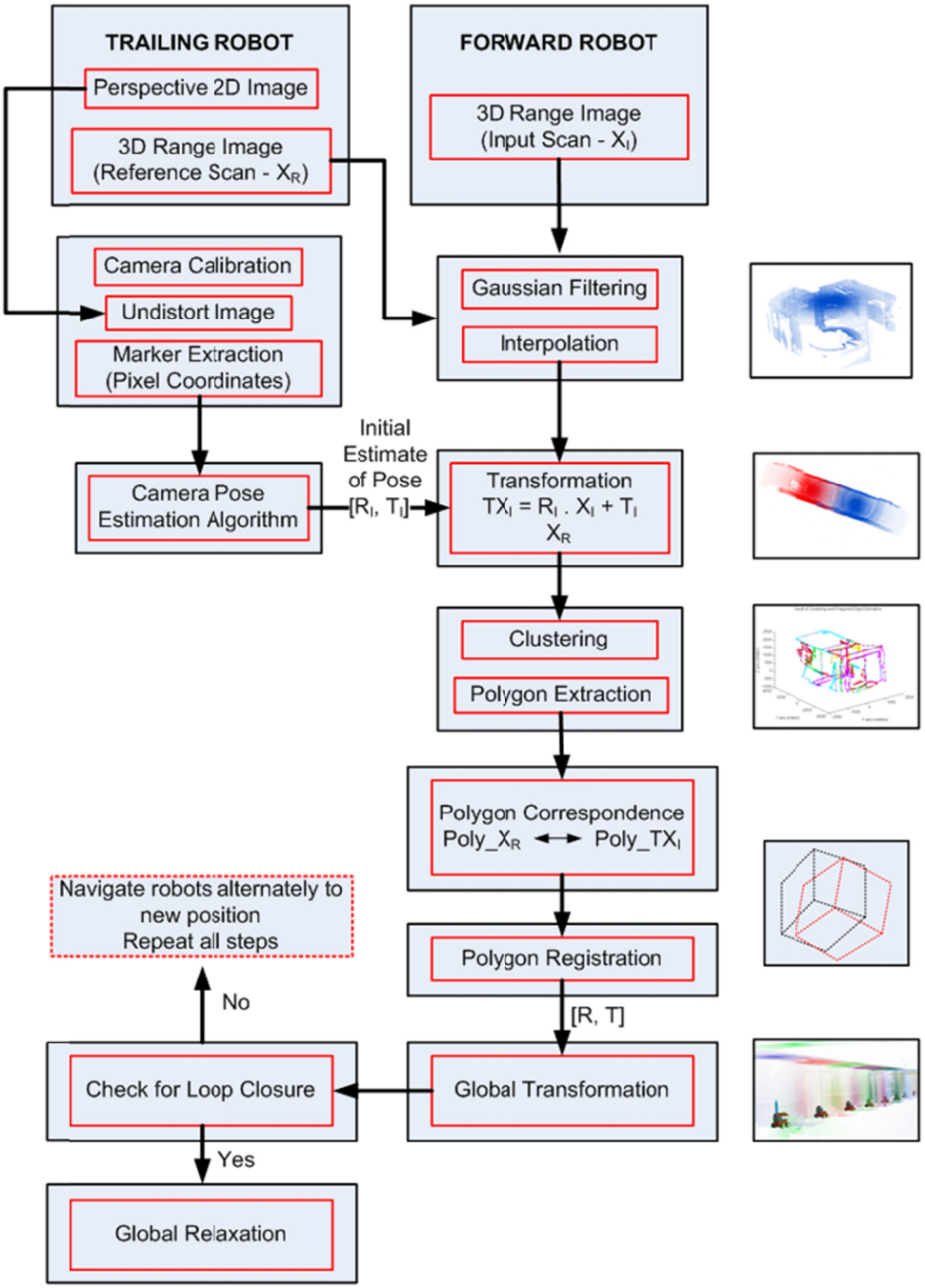


Fig. 4.3 Schematic diagram of numerous steps involved in the 3D mapping of structured indoor environment using dual-robot system

4.4.2 Complementary dual sensor registration

In this framework, relative pose estimation between the two robots is computed in two steps:

- Initialization with camera pose estimation algorithm
- Refinement with range image-based scan matching algorithm

The two range images acquired by the dual robots have less than 50% partial overlap between them. We take account of all kinds of occlusions and noisy range data, which may appear due to reflections and refraction of IR pulses emitted by the range scanner while scanning the surrounding environment. As shown earlier in section 3.10, it is unlikely that these two range images can be registered together by existing state-of-the-art scan matching algorithms without a good initial estimate of pose. This leads to couple the vision and laser scan registration approaches in to a combined effort to estimate the relative pose. The first estimate of initial pose is obtained using the camera pose estimation algorithm, which computes the pose in real-time. The pose estimate describes translation and rotation between the two scans to a good accuracy. This visual pose estimate is far more accurate than the existing 2D odometry approaches. This approach would be one viable solution in feature-less and odometry-less environment such as slippery long corridors or find applications in a rugged terrain such as cave exploration or mars exploration. This approach especially holds good when the robots in the multi-robot system are not ground vehicles, such as a wall climbing robot or an autonomous flying aircraft. Further, there is a need to refine the pose estimate between the two scans for building maps and hence the need for a reliable scan matching algorithm. The range images store accurate depth information of the surrounding environment. There are numerous scan matching algorithms that can compute accurate estimate of pose by registering two or more range images with overlapping regions. In

Chapter 6, we introduce a fast polygon-based registration algorithm that computes the pose estimate between the robots by registering two polygon sets extracted from partially overlapping range images. We performed both simulation and experiments to test the algorithm and we discuss the results in terms of both speed and accuracy compared to state-of-the-art scan matching algorithms.

Traditionally, scan registration is performed based on points matching where the relative pose is computed by minimization of distance between corresponding points from the two range images using Quaternions [Besl and McKay 1992; Surmann, Nüchter et al. 2003] and Singular Value Decomposition [Nüchter, Lingemann et al. 2005]. These algorithms have a time complexity of *O(n.logn)* and higher, where n is the number of points in a 3D range image (in the order of $10^4 - 10^6$), resulting in high computation time. One way to speed up the process is to extract polygonal surfaces from range images and perform feature-feature registration. We introduce a novel polygon registration algorithm where the features matched would be irregular polygons extracted from the successive range images. The polygon extraction leads to high level of data compression [Kaushik, Joseph et al. 2010]. In addition, the polygon registration algorithm in an indoor setting used only ≈ 5% data points from the range image, which are mainly the outlining boundary points of the extracted polygons in a highly planar environment.

The laser scanner on the ground robots acquire 3D scan mounted on a panning device at a given pose. The 3D range image is a point cloud in spherical coordinate system (220×680) array, where each element in the array is uniquely defined by (r, θ, φ) where r is the radius, θ is the pan angle and φ is the tilt angle. A number of algorithms like ICP [Besl and McKay 1992; Rusinkiewicz and Levoy 2001] , NDT [Biber and Straber 2003; Magnusson, Andreasson et al. 2009; Takeuchi and Tsubouchi 2006], Angle Histogram [Rofer 2002] make use of 3D coordinate points to compute the relative

pose after obtaining correspondences between the point scan. Most of these processes are iterative with a large number of input elements (the corresponding point-pairs, grid-based Gaussian estimate or angle histogram correlation techniques). Hence, the computation time is usually high. The methodology presented to fuse multiple range images reduces the computational complexity by following steps.

- ***Range image clustering and polygon extraction***: the scan points are clustered into planar regions. Non-planar points are discarded in the scan registration process. This approach works well when the environment is at least partially planar. The outer boundary points are extracted that define the polygons representing a planar segment. The improvised clustering algorithm [Kaushik, Joseph et al. 2010] executes in $O(n.logn)$ and implements patch-based clustering (Chapter 5).
- ***Polygon Correspondence***: First step to polygon registration is to identify the corresponding polygons extracted from two partially overlapping range images that represent the same planar region in the environment. The scan matching is performed by using a small number of corresponding polygon pairs instead of using a large number of corresponding point-pairs (Chapter 6).
- ***Polygon Registration (PR)***: A novel registration algorithm based on non-linear LS minimization using Levenberg-Marquardt algorithm minimizes the distance between boundary points of the polygon in a given set and boundary point projection on each of its corresponding polygons. The pose parameters are estimated over several iterations until the minimization error meets certain preset threshold (Chapter 6).

CHAPTER 5

Patch-based Plane Clustering (PPC) and polygonal extraction

This chapter introduces a methodology to cluster noisy range images into planar regions acquired in indoor environments. The noisy range images are segmented based on a Gaussian similarity metric, which compares the geometric attributes that satisfy the coplanarity conditions. The algorithm is designed to cluster coplanar noisy range data by means of patch-based sampling from range images. We discuss the advantages of patch-based clustering over point-based clustering of noisy range images that includes minimizing the clustering error and accelerating the clustering process. The final output of the algorithm is a set of polygons, where each polygon is defined by a set of boundary points that replaces large number of coplanar data points in a given planar region. The 3D range image is acquired by a rotating 2D range scanner and stored in a 2D array. Each element in the array is explicitly stored as the range distance; the indices of the array implicitly retain neighborhood and angular information. The array is grouped into mutually-exclusive patches of size $(k \times k)$ and the Hessian plane parameters are computed for each patch. We propose a graph-search algorithm that compares the plane parameters of neighboring patches by searching breadth-wise and clusters the coplanar patches into respective planes. We compare the proposed Patch-based Plane Clustering (PPC) algorithm with the point-based Region Growing (RG) algorithm and the RANSAC plane segmentation method to analyze the performance of each of the algorithms in terms of speed and accuracy. Experimental results indicate that the PPC

algorithm shows a significant improvement in computational speed when compared with the state-of-the-art segmentation algorithms while maintaining a high accuracy in segmenting noisy range images. The polygonal map representation finds use in modeling the indoor environment, both for visualization purposes and multiple map fusion. A structured environment such as office corridors consists of large number of planar surfaces. In such a scenario, an ideal form of representation of range images would be irregular polygons interspersed with non-planar points for modeling the environment. Applications of plane clustering and polygon extraction from range images are abound. The range data acquired by 3D range scanners consists of large amounts of digital information that needs memory storage space and should be further processed in real-time for a number of robotic applications such as 3D scan registration with polygons [Kaushik, Xiao et al. 2010; Pathak, Birk et al. 2010], digitization of indoor environments [Surmann, Nüchter et al. 2003], 3D map construction [Kaushik, Joseph et al. 2011] and robotic navigation. Feature-based representation such as polygon representation of range images is advantageous over point-based representations in a number of ways. The polygonal maps occupy less storage space compared to 3D point clouds/range images. Efficient memory storage is achieved by replacing large numbers of data points in the range image with the polygons (defined by boundary vertices) that represent planar regions, which results in high data compression. In addition, polygons extracted from range image can be rendered in computer graphics tools faster than 3D point clouds with the aid of Graphical Processing Units (GPUs).These two aspects become prominent while rendering large number of fused range maps. The polygons are extracted as features as a preliminary step to build 3D range maps. Most recently, a number of researchers have worked on registration of multiple range images with polygons as features [Kaushik, Joseph et al. 2011; Pathak, Birk et al. 2010; Sequeira, Ng et al. 1999]. These approaches have been experimentally shown to be much faster than

iterative point-based 3D registration techniques [Besl and McKay 1992; Takeuchi and Tsubouchi 2006]

3D range scanners acquire range images, which essentially outputs range distance measurements between the sensor and the surrounding environment. This paper deals with real-time processing of range images acquired in indoor environments, which consists of mostly planar surfaces. A range image represented in the spherical coordinate system can be converted to a 3D point cloud representation in the Cartesian coordinate system and vice-versa. This conversion is even applicable to multiple overlapping 3D point clouds, which are an outcome of 3D registration [Besl and McKay 1992]. For robotics applications, it is desirable to have a more compact representation of range images, which can be achieved by extracting polygon features defined by a set of boundary points leading to compression of range maps. Polygon extraction from noisy range images is a challenging task and the polygonal model of the environment is useful in robotic applications such as 3D mapping and robot navigation in indoor environments. The current commercial 3D range scanners used in robotics applications are capable of acquiring large number of data points ($N > 10^4$) in a short period of time($< 30\ s$). This data acquisition is achieved by spinning the 2D range scanner on a rotating platform. The angular resolution of the range image is limited by the resolution of the motors spinning the range scanner. Most conventional range scanners used in robotics are inexpensive but noisy. Their range measurement accuracy is dependent on a number of factors including range sensor limitations, embedded processors in the scanning equipment, step accuracy of internal and external motors driving the range sensor and reflection/refraction of the laser beam on different surfaces. The high-resolution range scanners are accurate but they are normally bulky and expensive. Our research makes use of less expensive light-weight range scanners, namely the Hokuyo® URG-LX and the UTM-30LX 2D range scanners that can be mounted on wall climbing

robots [Xiao, Sadegh et al. 2005] and miniature Quadraptor helicopters [Morris, Dryanovski et al. 2010], which have constraints in payload and on-board computational power.

The range data acquired by 3D range scanners consists of large amounts of digital information that needs memory storage space and should be further processed in real-time for a number of robotic applications such as 3D scan registration with polygons [Kaushik, Joseph et al. 2010; Kaushik, Joseph et al. 2011; Kaushik, Xiao et al. 2010; Pathak, Birk et al. 2010], digitization of indoor environments [Surmann, Nüchter et al. 2003], 3D map construction [Kaushik, Joseph et al. 2011] and robotic navigation. Feature-based representation such as polygon representation of range images is advantageous over point-based representations in a number of ways. The polygonal maps occupy less storage space compared to 3D point clouds/range images. Efficient memory storage is achieved by replacing large numbers of data points in the range image with the polygons (defined by boundary vertices) that represent planar regions, which results in high data compression. In addition, polygons extracted from range image can be rendered in computer graphics tools faster than 3D point clouds with the aid of Graphical Processing Units (GPUs). These two aspects become prominent while rendering large number of fused range maps. The polygons are extracted as features as a preliminary step to build 3D range maps. Most recently, a number of researchers have worked on registration of multiple range images with polygons as features [Pathak, Birk et al. 2010; Sequeira, Ng et al. 1999]. These approaches have been experimentally shown to be much faster than iterative point-based 3D registration techniques [Besl and McKay 1992; Takeuchi and Tsubouchi 2006].

Planar segmentation is a widely studied topic in computer graphics, photogrammetry and robotics catering to numerous applications. Many algorithms have been developed in the field of computer graphics to obtain polygonal models of 3D

point clouds for graphical rendering / visualization [Bernardini, Mittleman et al. 1999; Turk and Levoy 1994]. These algorithms are designed to model the objects with complex surfaces resulting in large number of polygons. The polygonal mesh can be further decimated or simplified by approximation based on the requirements to satisfy a certain level of detail for visualization as in [Garland and Heckbert 1997; Hoppe 1996]. We can identify two major steps in these approaches: mesh formation/triangulation and mesh approximation. The best known time complexity of mesh formation using Delaunay triangulation is $O(n.logn)$ for an unstructured 3D point cloud using divide-and-conquer strategy [Cignoni, Montani et al. 1998]. Further, [Hoppe 1996] present an iterative process for mesh optimization and is based on minimization of a cost function. [Garland and Heckbert 1997] employs constructing an error quadric metric that involves computing the plane parameters for each triangle in the mesh. These algorithms focus more on visualization with polygonal models and do not address real-time processing of range images.

The requirements for 3D mapping in robotics community are different from a computer graphics perspective. The latter focuses on building complex and accurate models of architectural objects at the expense of speed of the algorithms. The range scanners used in the field of computer graphics produce dense and highly accurate range images. The former focuses on building simpler 3D models using less accurate range scanners while they minimize the computation time of the feature extraction algorithm to satisfy the real-time applications such as Simultaneous Localization and Mapping (SLAM). A number of methodologies developed for robotics application are available to extract simplified polygon models from 3D range images [Hähnel, Burgard et al. 2003; Hoover, Jean-Baptiste et al. 2002; Kurogi, Wakeyama et al. 2008; Poppinga, Vaskevicius et al. 2008; Triebel, Burgard et al. 2005; Weingarten, Gruener et al. 2003]. The complexity of the planar model is simple yet efficient to be useful in ground robot

navigation [Pathak, Birk et al. 2010; Weingarten and Siegwart 2006] and aerial robot navigation [Grzonka, Grisetti et al. 2009].

We compare our algorithm with two popular plane segmentation algorithms, one is point-wise Region Growing (RG) algorithm and the other is RANSAC-based plane segmentation algorithm. An example of a RG algorithm to extract polygons from range data is presented in [Hähnel, Burgard et al. 2003]. The range image data are converted into polygons in two steps. First, a computationally expensive nearest neighbor search is employed to cluster the data points belonging to individual planes. Second, the boundary points of each cluster are extracted to form polygons. The RG algorithm has a time complexity of $O(n^2)$, where n is the number of coordinate points in the range image. In [Poppinga, Vaskevicius et al. 2008], the authors proposed two alternative steps to optimize the RG algorithm in [Hähnel, Burgard et al. 2003] to improve the execution speed. Initially, three random points (seed) are chosen and their plane parameters are computed, which form the plane model. These three random points are added to a queue (cluster). The remaining data points are added to the cluster one point at a time if they satisfy the plane segmentation criteria. The plane parameters of the cluster are recomputed whenever a data point is added to the cluster. The first element in the queue is popped after checking if any of its neighboring points belong to the same cluster. The loop continuously executes until the queue is empty. The above steps are repeated until all data points are placed in respective clusters. Another example of plane segmentation algorithm that uses point-wise RG methodology is presented in [Chen and Stamos 2007]. The algorithm segments the range images into planar, smooth non-planar and non-smooth surfaces. The normal vectors are computed for each data point from a set of data points belonging to the local neighborhood. The RG algorithms in [Chen and Stamos 2007; Hähnel, Burgard et al. 2003; Poppinga, Vaskevicius et al. 2008] are empirically fine-tuned with a set of parameters to segment planar regions

within a tolerance limit. The RG methodology is not robust to noisy range images, which makes it difficult to fine-tune the parameters to segment range images with higher noise levels. Plane segmentation of noisy range images with RG algorithm sometimes leads to bleeding of one plane into adjacent planes resulting in over-segmentation. The major drawback of the RG algorithm is that it is computationally slow. The normal vector is repeatedly computed while each data point is evaluated for the planarity test. This is an expensive operation and performed repeatedly on the covariance matrix of the data points. A well-known approach of extracting the normal vectors for a given set of data points is by solving the Eigen problem [Dyranovski, Morris et al. 2010] for the covariance matrix of data points belong to the cluster, with a lower bound of $\Omega(n^3)$, where $(n = 3)$ is the size of the matrix.

In addition, we compare our algorithm with another popular form of planar segmentation, which belongs to the class of model fitting using RANdom SAmple Consensus (RANSAC) [Fischler and Bolles 1981]. Some recent efforts on planar segmentation from range data using RANSAC is presented in [Schnabel, Wahl et al. 2007; Tarsha-Kurdi, Landes et al. 2007; Tarsha-Kurdi, Landes et al. 2008]. In general, a small number of data points are randomly chosen to establish the "hypothesis" set. The parameters of the planar model are computed from this minimal set of random points. Further, the "verification" step involves testing the remaining data points to see if it fits the model defined by the parameters computed earlier. If the error function evaluates to a value below a set threshold, then the data points are added to the "consensus set". The RANSAC plane segmentation algorithm stops looking for a better planar model once the probability of finding a better ranked consensus set drops below a certain threshold. The number of trials can be set heuristically as an algorithm input or evaluated based on the probability. If there is more than one plane in the range image, the above steps are repeated after removing previously segmented data points until the remaining points

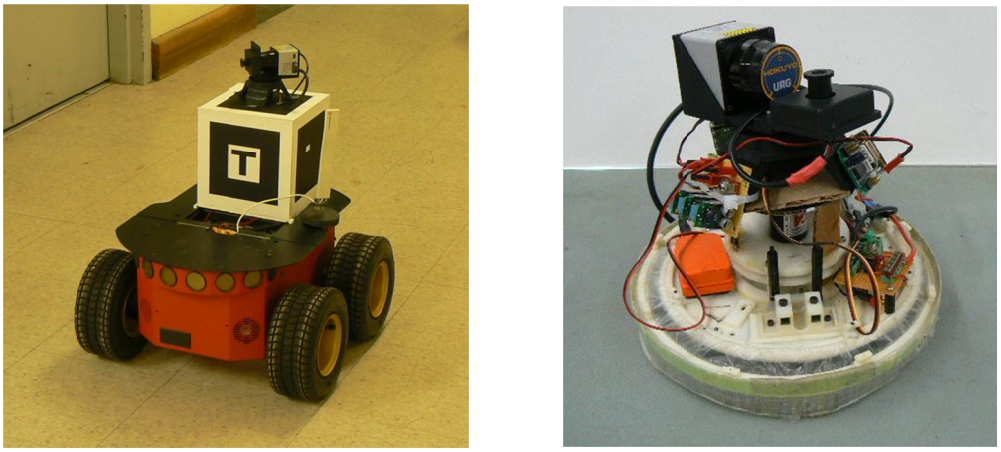
are reduced to a small percent of initial value. The time complexity of a general RANSAC plane segmentation algorithm is given by $O(T_{iter}C_{hyp}(k) + NC_{ver}(k))$, where T_{iter} is the number of iterations determined by the probability of finding the best ranked consensus set, $C_{hyp}(k)$ is the cost of hypothesis or computing the parameters of a planar model with minimum number of data points (k), $C_{ver}(k)$ is the cost to verify if the data point belongs to a particular consensus set and N is the number of data points in the 3D data set. The RANSAC is slow for several reasons. The randomized approach in choosing the data points for establishing the hypothesis sometimes leads to picking an outlier, which is imminent while establishing the hypothesis set. In addition, the number of iterations is related to the noise levels in the range image. "*While the RANSAC algorithm is effective in the presence of noise and outliers, it has two significant disadvantages, namely, its efficiency and the fact that the plane detected by RANSAC may not necessarily belong to the same object surface; that is, spurious surfaces may appear, especially in the case of parallel-gradual planar surfaces such as stairs." [Awwad, Zhu et al. 2010].* Alternatively, [Awwad, Zhu et al. 2010] presents a modification of the RANSAC plane segmentation algorithm known as the Seq-NV-RANSAC. This algorithm checks the Normal Vector (NV) of each data point (computed from a local neighborhood at any given point) and that of the hypothesized RANSAC plane. This process is shown experimentally to improve the accuracy of RANSAC planar segmentation. However, it will also lead to additional computation time in determining the normal vectors from the 3D point cloud and performs slower than the traditional RANSAC plane segmentation algorithm.

There are a few other plane segmentation approaches that emerged recently. A probabilistic approach presented in [Triebel, Burgard et al. 2005] extract planes from range images based on Expectation Maximization (EM). The disadvantage of this algorithm is that it needs to be initialized with the number of planes for a given range

image and their "main" directions, prior to the Least Square (LS) minimization. This number is usually unknown and a heuristic approach is used to determine the number of planes during the clustering process. This approach follows patch-based range image segmentation and effectively clusters points based on co-planarity conditions using a probabilistic approach. There are few other algorithms [Biosca and Lerma 2008; Hofmann, Maas et al. 2003] that employ clustering techniques to range image plane segmentation. However, these type of algorithms have not been proved to be practical when dealing with large 3D point clouds in the presence of noise and outliers and are computationally expensive [Awwad, Zhu et al. 2010].

5.1 Range image acquisition and pre-processing data

We obtain the 3D range image by rotating a commercial 2D laser scanner (Hokuyo® URG-LX) shown in Fig. 1.1(a) on a platform using an external motor. The 2D laser scanner acquires a 2D scan along the sensing plane $(270°)$ as shown in Fig. 1.1(b). The 2D range scanner is rotated by $180°$ (pan angle) on the platform to obtain a complete 3D scan. A pioneer robot and a wall climbing robot (Fig. 5.1) were used in the experiments to acquire 3D range images. The step of the external motor is given by $\Delta\theta \cong 0.81°$, considered as the "azimuth" angle. The 2D range scanner has an internal mirror that is rotated in steps by $\Delta\varphi \cong 0.36°$ to acquire a flat 2D scan, considered as the "elevation" angle. The range distance acquired by the laser scanner at (θ_i, φ_j) is stored in a 2D array indexed by (i, j), where $\theta_i = (\Delta\theta * i), i \in \{1,2 \dots 220\}$ and $\varphi_j = (\Delta\varphi * j), j \in \{1,2 \dots 680\}$. Each element in the array points to a (r, θ_i, φ_j), which together form the spherical coordinates. The indices of the array provide neighborhood information for each element. A time-consuming nearest neighbor search of elements can be avoided by comparing only indexed neighboring elements of the array.



(a) (b)

Fig. 5.1 Pioneer® robot (a) and a wall climbing robot (b) equipped with a light weight rotating 2D range scanner for acquiring 3D range images

A 3D range image can be easily transformed into a 3D point cloud represented in Cartesian coordinate system and vice-versa [Rusu and Cousins 2011] in $O(n)$ time. We dealt with multiple overlapping 3D point clouds in [Kaushik, Xiao et al. 2010], resulting in a 3D polygonal map. Multiple overlapping 3D point clouds are highly redundant since large percentage of data points overlaps each other. The multiple overlapping 3D point clouds can also be transformed into a range image by quantizing the 3D data points in spherical coordinate system within a limited radius from the center of the coordinate system. A 3D range image is a compact representation of surrounding environment in spherical coordinate system and stores fixed amount of data points. We can ensure that every data point in the 3D point cloud is accounted for by quantizing the spherical coordinate system with small azimuth and elevation step angles. Another advantage of range image representation is that we can access the surrounding neighbors of a given data point in constant time of $O(1)$ and eliminates the need for nearest-neighbor search algorithms, which have a lower bound time complexity of $\Omega(n.logn)$ to access the neighboring elements.

5.2 Pre-processing range images

The range image data obtained by spinning the commercial 2D range scanners as in our experiments are fairly noisy. The range measurements acquired by the 3D range scanner can be described by an approximative physical model as presented in [Thrun, Burgard et al. 2005] and is described as a mixture of several kinds of distributions. The model is considered as a mixture of Gaussian distribution defined by $\mathcal{N}(0, \sigma)$ centered around the "true" range between the 2D range scanner and the object being measured. In addition, the range measurement model in dynamic environment can be described by an exponential distribution, defined by an intrinsic parameter (λ_{short}) of the measurement model. We do not consider exponential distribution as our environment is assumed to be static and we can ignore the effects cause by moving objects while scanning the surrounding region. The range scanner occasionally records "phantom" readings or outliers. This range measurement is modeled as uniform distribution with a range of $[0, Z_{max}]$. This kind of outliers needs to be eliminated as they can significantly affect the final outcome of segmentation. We present a simple yet effective method to remove the outliers with a Gaussian filter function (Eqn. 5.1), that describes the distance measure between a point and its neighbor.

$$f(X, X_i) = e^{-\frac{(X-X_i)^2}{2\sigma^2}} \qquad 5.1$$

The Gaussian filter function is unit-less and yields a value between 0 and 1, where 1 stands for closest proximity and 0 stands for the farthest distance of the point from its neighbor. The Gaussian filter function is the distance measure between a point and its i^{th} neighbor. If the Gaussian filter function yields an output between the given element and all its neighbors and is less than a certain threshold ($0 < \lambda < 1$), the given element is considered as an outlier and discarded from the range image leaving behind a hole in the 2D array. This threshold can be set heuristically and is dependent on the noise levels

of the range scanner. In our experiments, we arrived at a preset threshold ($\sigma = 0.01m$, λ = 0.7) to eliminate all outliers while we retained the rest of the range measurements. This filtering process does not modify the range data, but successfully eliminates all "phantom" measurements. The holes left behind by removing the "phantom" readings in the 2D array are filled by applying a median interpolator that computes the values based on its horizontal and vertical neighbors.

After noise removal and interpolation, the range image can be grouped into mutually exclusive patches in two ways; one way is to break it down into rectangular grids of a fixed size $(k \times k)$, $k = 2..N$, another way is circular extension from a given data point, where $k = (2 * r + 1)$. r is the radius in pixels from the data point and extends into its neighboring pixels. If the patch radius is $r = 1$, then the patch includes 8 surrounding neighbors along with the element at (i, j) as shown in Fig. 5.2. If the patch radius is $r = 2$, then the patch includes a total of 24 neighboring data points and so on.

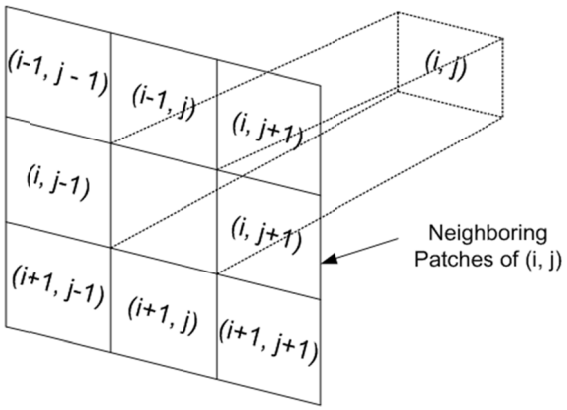


Fig. 5.2 Index of neighboring cells of given data point (i, j), which forms the patch of size (3×3). Each cell in a 2D range image contains the range data (r).

5.3 Orthogonal Distance Plane Regression (ODPR)

After the range image is grouped into patches of fixed size, each patch is fit with a plane model and resulting outcome is the Hessian plane parameters: the normal vector $\left(\hat{n}_{(3\times1)}\right)$ and the distance (d) from the patch to the origin. The normal vector is computed from the covariance matrix of all the data points $(X_i, i = 1 \dots N)$ in a given patch. This process is known as Orthogonal Distance Plane Regression (ODPR). Hessian normal form is a convenient way to represent plane equation using normal vectors. The general plane equation is given by

$$ax + by + cz + d = 0 \tag{5.2}$$

The normal vector $\hat{n} = (n_x, n_y, n_z)$ and the scalar constant which defines the distance from the plane to the origin (p) can be computed using the co-efficients of the plane equation as

$$n_x = \frac{a}{\sqrt{a^2 + b^2 + c^2}} \tag{5.3}$$

$$n_y = \frac{b}{\sqrt{a^2 + b^2 + c^2}} \tag{5.4}$$

$$n_z = \frac{c}{\sqrt{a^2 + b^2 + c^2}} \tag{5.5}$$

$$p = \frac{d}{\sqrt{a^2 + b^2 + c^2}} \tag{5.6}$$

The Hessian normal form is then given by the dot product as follows:

$$\hat{n} \cdot (X_i) = -\mathrm{p} \tag{5.7}$$

In addition, the algorithm computes the mean $(\bar{X})$ of all the data points in the patch, which is an additional attribute that describes the patch position relative to the coordinate frame of the range scanner. The range data points are affected by noise. We perform plane regression by minimizing the least square orthogonal distance (β)

between the points X_i, $i = \{1..n\}$ and the plane model, which is defined by the normal vector ($\hat{n}$) and the mean point of the patch ($\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$). The error function $E(\bar{X}, \hat{n})$ to be minimized is the sum of the squared point-plane distance for (n) data points that satisfy the hyper-plane equation

$$E(\bar{X}, \hat{n}) = \sum_{i=1}^{n} \beta^2 = \sum_{i=1}^{n} \left(\hat{n} \cdot (X_i - \bar{X})\right)^2 \qquad 5.8$$

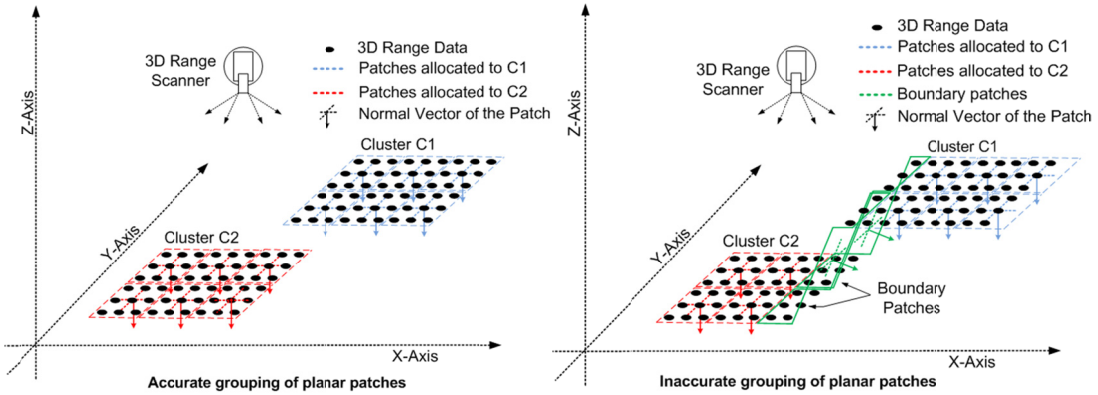
Prior to obtaining the normal vector $(\hat{n})$ from the set of data points, the (3×3) symmetric covariance matrix of all data points is computed, which is given by,

$$M(X_i, \bar{X}) = \sum_{i=o}^{n} (X_i - \bar{X})(X_i - \bar{X})^T \qquad 5.9$$

The eigenvalues and eigenvectors of the matrix $M(X_i, \bar{X})$ are obtained as described in [Pan and Chen 1999]. The normal vector of the best-fitted plane to the points X_i for a given patch is the eigenvector corresponding to the minimum eigenvalue of the covariance matrix $M(X_i, \bar{X})$. If we can assume that the matrix M row-wise form a set of three linear equations, then the three eigenvalues are one of the characteristic solutions to the set of equations. The three eigenvectors correspond to each of the eigenvalues and are orthogonal to each other. Two of the three eigenvalues correspond to eigenvectors that lie on the plane and is greater than the third eigenvalue. The third smallest eigenvalue corresponds to the normal vector which is orthogonal to the plane determined by the other two eigenvectors. The other Hessian parameter is the scalar constant, which is the distance from the planar patch to the origin of the coordinate system given by,

$$p = -\hat{n} \cdot \bar{X} \qquad 5.10$$

5.4 Distinction of patches and patch size



(a) (b)

Fig. 5.3 show two different kinds of patches after grouping range image into mutually exclusive grids. (a) Planar patches (red and blue) that fall into two separate clusters. (b) Non-planar patches are visible on the boundary of two clusters (green).

We distinguish between a *planar* patch and *non-planar* patch based on the criterion (Eqn. 5.11) when grouping the data points into patches. The data points in the planar patch (patches marked in red and blue, Fig. 5.3) lie on the same plane defined by the Hessian parameters $(\hat{n}, p)$. To accommodate the range measurement error, we define an orthogonal distance measure, which intuitively represents the spread of the data points of a given patch along the normal vector of the plane. If the distance measure is less than the preset distance threshold (S_T), then the patch is considered as planar. This criterion is given by the dot product of the diagonal of covariance matrix, $diag(M(X_i, \bar{X}))$ computed for X_i data points of the patch and the normal vector $\hat{n}$ of the patch.

$$diag\big(M(X_i, \bar{X})\big) \cdot \hat{n} < S_T \qquad 5.11$$

Non-planar patches do not satisfy the above criterion. They usually lie on the boundary of two planar segments or contain data points that do not form a plane (patches marked

in green, Fig. 5.3(b)). The Patch-based Plane Clustering (PPC) algorithm computes plane parameters of each patch as a preliminary step. A single planar patch is selected as the seed for plane model. The rest of the planar patches are added to the cluster depending on whether they satisfy the plane segmentation criteria described in section 5.5. The plane model is not recomputed when new data points are added to the cluster as in the region-growing algorithm.

In this section, we study the effects of patch size on the plane segmentation. A simulation study was conducted to evaluate the accuracy of normal vectors computed for patches of a simulated range image. The study indicates that the plane parameters computed for each patch is accurate enough to complete the clustering process if appropriate patch size is selected. Fig. 5.4 shows five planar segmented regions (4 wall surfaces and the floor plane represented by different colors) in a simulated environment as the ground truth. The noise level in the range image is set at 1% of the range distance measured as is the case for the range scanner used in our simulations. The Hokuyo® Range scanner URG-LX has an error of $\pm$0.01m for a range measurement between 0.2m to 1m and $\pm$1% for the range measurement between 1m to 4m [URG-04LX Specifications, 2005]. In this simulated experiment, we computed the normal vectors of planar patches for each plane segment and represented its distribution with different colors corresponding to the color of the plane. From the results, we can assess the accuracy of the normal vectors computed for planar patches by varying the patch size. Three plots of normal vectors extracted from the same noisy range image for different patch sizes (2×2), (3×3) and (5×5) are shown in Fig. 5.5 (a), (b) and (c) respectively. Different color markers are used in Fig. 5.4 and Fig. 5.5 to distinguish between different plane segments and their respective normal vectors. We can observe that data points marked in red are floor points in Fig. 5.4. It can also be observed that normal vectors marked in red in Fig. 5.5 computed for the planar patches belonging to

the floor plane are inaccurate and spread randomly over the normal vector space for the patch size set at (2×2). As the patch size is increased, the normal vectors appear to be more densely populated and close to the real plane model. The improvement in accuracy can be noted with the other plane segments marked in different colors.

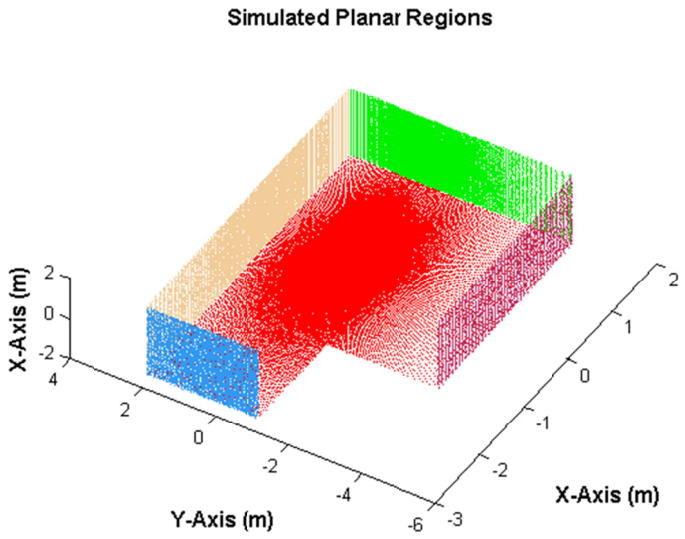


Fig. 5.4 Planar segments of the simulated environment consisting of a floor surface and four walls. Different planar segments are displayed in several colors that match the colors of the normal vectors plotted in Fig. 5.5.

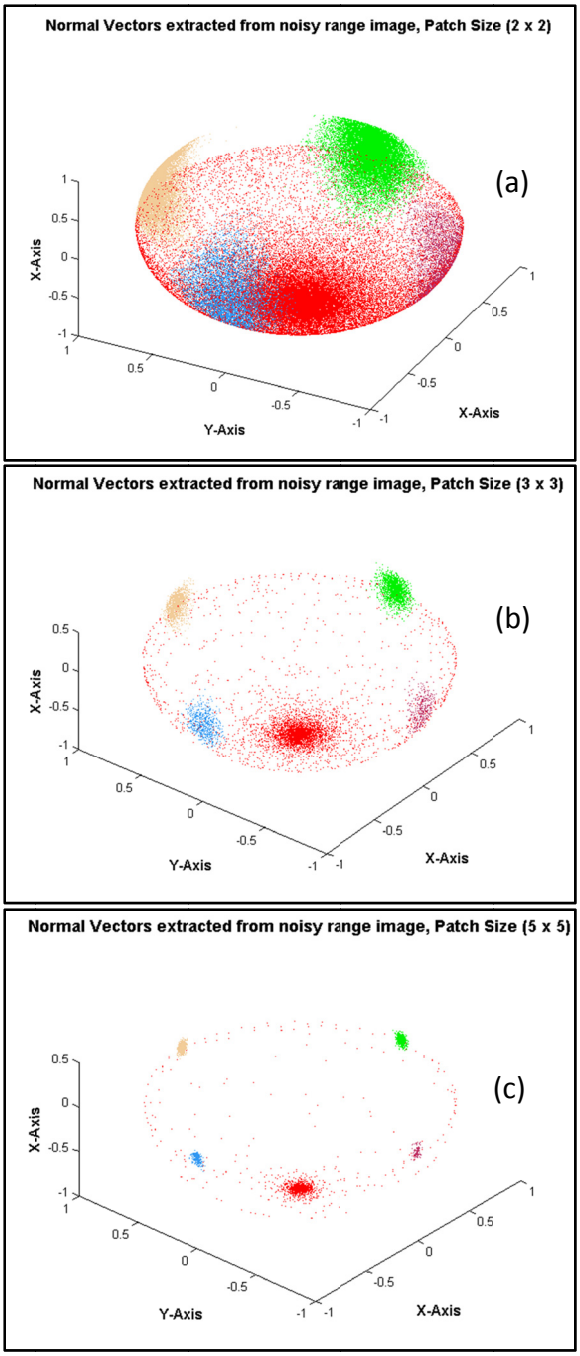


Fig. 5.5 Plot of normal vectors extracted from noisy range image with varying patch size (a) 2 × 2 (b) 3 × 3 (c) 5 × 5. The markers with five different colors in the normal vector space indicate normal vectors of patches belonging to different plane segments.

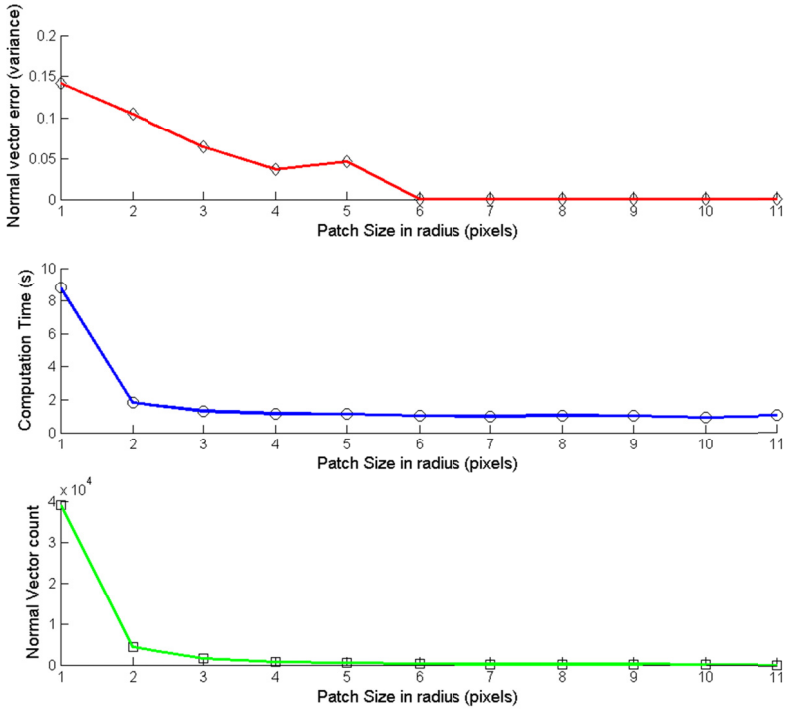


Fig. 5.6 Plot of error in computing normal vectors with varying patch size marked in red. In addition, the computation time and number of normal vectors extracted for a single planar region can be seen in blue and green respectively.

In Fig. 5.6, we can observe that the error rate of normal vectors drops to negligible amount with increase in patch size. Error rate was computed by taking the dot product of the normal vector of the patches and the actual plane model. Increasing the patch size leads to less number of patches. Further, the time required to compute the normal vectors is reduced. Although the accuracy of the normal vectors can be improved by choosing a large patch size, there is a caveat in choosing a very large patch size. The patch size is only limited by the nature of surfaces in the indoor environment, where the range images are acquired by the sensor. If the environment consists of large planar surfaces, it would be useful to choose a large patch size. However, if the indoor

environment is highly cluttered and consists of non-planar surfaces, then a small patch size is preferred which will ensure that the algorithm finds adequate planar patches.

5.5 Patch-based Plane Clustering (PPC) algorithm

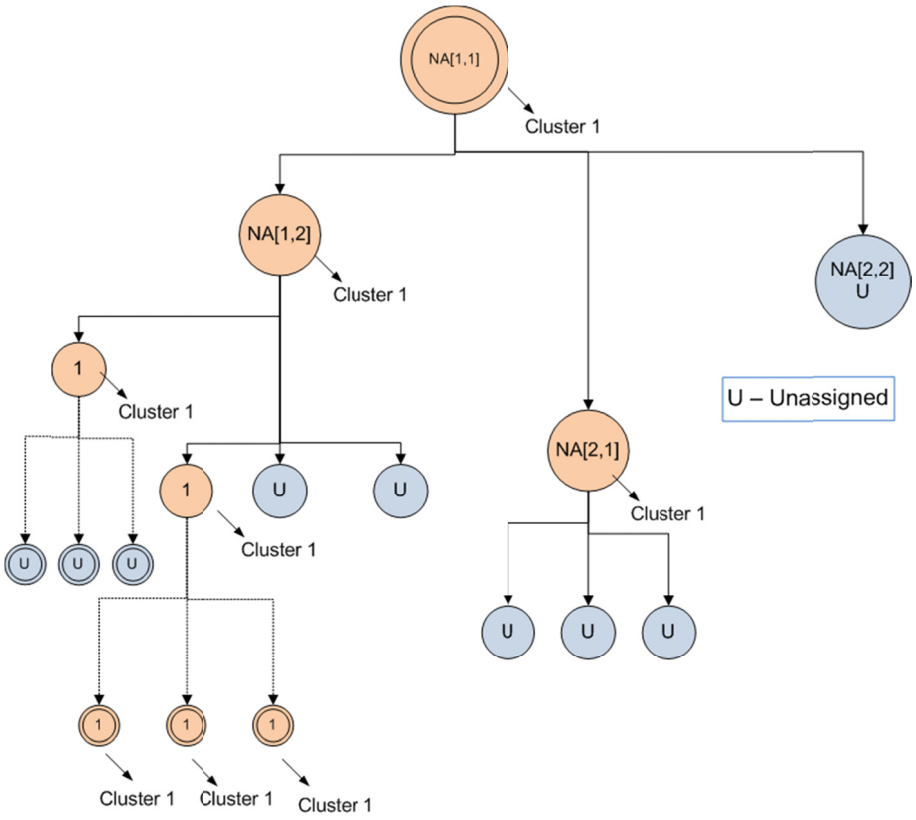


Fig. 5.7 Graph search algorithm indicates the tree structure and the assignment of the patches to a particular cluster after evaluation of the plane segmentation criteria. Orange nodes belong to cluster "1" indicated inside the circle and element in the blue nodes are still not assigned to any cluster, denoted by "U".

In this section, we propose a clustering algorithm that searches for coplanar data points of the range image in two stages. In the first stage, the algorithm starts with a seed patch and performs a breadth-wise search among all the planar patches for coplanar patches. Depending on whether the planar patches satisfy the plane segmentation criteria, the data points of the coplanar patch are added to the cluster. The algorithm

continues until all planar patches are assigned to a cluster. In the second stage, data points of the non-planar patches are individually added to the clusters after evaluating the segmentation criteria. In the first stage, the algorithm starts by assigning the first planar patch with index [1,1] as the root of the graph, labeled as cluster-1 (Fig. 5.7). The neighboring planar patches are assigned one-by-one as children to the root (parent) of the graph if and only if they meet the following plane segmentation criteria:

1. Gaussian similarity measure between the normal vectors of the parent patch and its neighboring planar patch is greater than the preset threshold (λ)[2].

$$f(\hat{n}_A, \hat{n}_B, \sigma) = \exp\left(-\frac{(\hat{n}_A - \hat{n}_B)^2}{2\sigma^2}\right) > \lambda \qquad 5.12$$

2. The orthogonal distance between the parent patch and its neighboring patch must be less than a preset threshold (δ) measured in meters. The orthogonal distance between patch $A(\hat{n}_A, p_A)$ and patch $B(\hat{n}_B, p_B)$ is given by, $dist(A,B) = (\hat{n}_A \cdot \bar{X}_B) + p_A$.

If the above two criteria are met, the neighboring patches are pushed onto a First-In-First-Out (FIFO) queue. After all the neighboring elements of the patch with index [1,1] are evaluated, the first element in the FIFO queue is popped and its neighboring patches are evaluated by coplanarity test. If the neighboring patches are coplanar to its parent patch, then they are assigned with the same cluster label and added to the FIFO queue. If they are not coplanar, the neighboring patches are unassigned. The whole operation is repeated until all the elements in the FIFO queue are popped out, which means that all the planar patches belonging to the same cluster are found. In addition, the graph search algorithm maintains continuity of the plane by comparing only neighboring

[2] The parameters σ and λ are unit-less as the Gaussian similarity metric compares normalized vectors. The range of both these parameters is [0,1]. σ is the same for all three axes of the normal vector

patches. A search for the new cluster begins when the FIFO queue is empty. The FIFO queue executes inside a double-*for* loop that searches for a new cluster row-wise. The algorithm searches for a new seed patch and skips all the planar patches that were already assigned with a cluster label. The whole process is repeated until all the planar patches are assigned with cluster labels and the algorithm moves to the next stage. In the second stage, the unassigned data points from non-planar patches are evaluated to place them in appropriate clusters. The process of adding individual data points to different clusters uses the same graph search method as described earlier. The normal vector of each data point is extracted from the local patch of radius (r) centered at that data point. This ensures that the normal vectors are accurate when dealing with noisy range images. Two clusters are merged in the process if the algorithm finds that the neighboring data points belonging to the two different clusters meet the coplanarity conditions. Data points that do not meet the segmentation criteria are put to a cluster and marked as non-planar cluster. The flow chart of the algorithm is shown in Fig. 5.8.

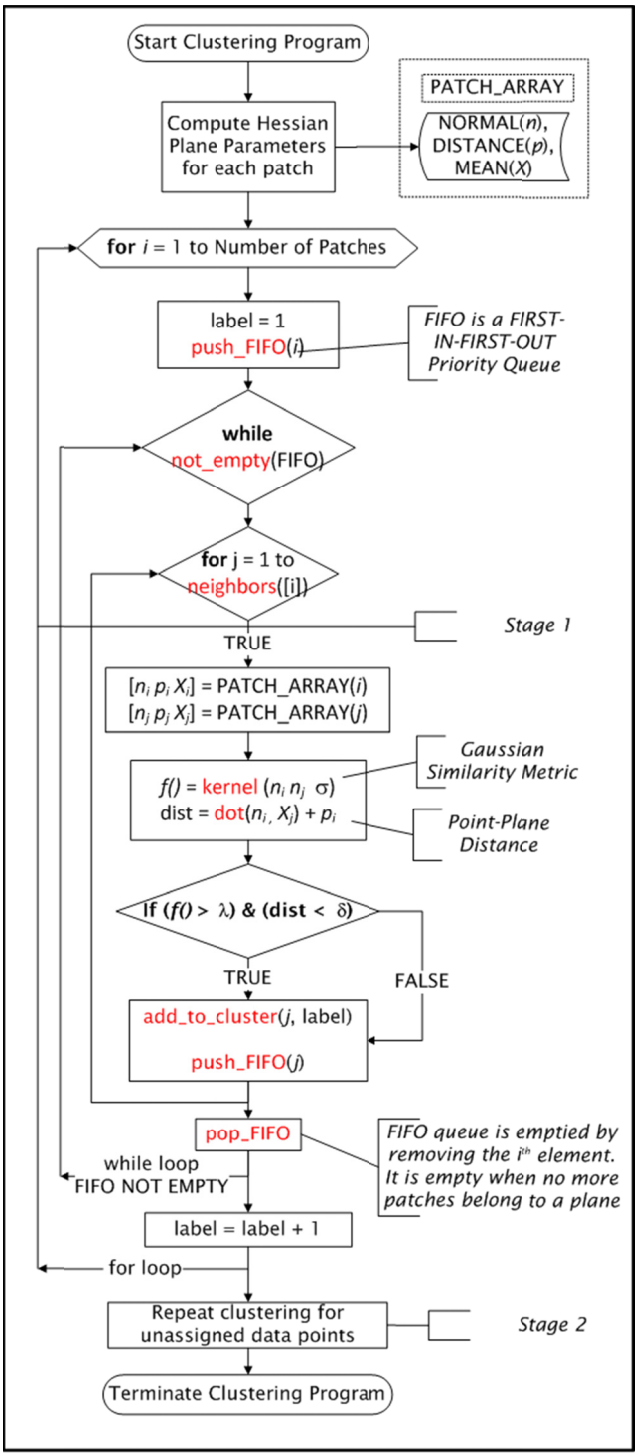


Fig. 5.8 shows a flow chart of the PPC algorithm that extracts planar clusters from range data

5.6 Discussion on computational complexity

Most plane segmentation algorithms spend bulk of its processing time in computing the plane parameters or fitting the data points to a specific plane model. Our proposed PPC algorithm exploits the redundancy of data points that lie on the same plane. The planar parameters are computed patch-wise and a distinction is made been the planar patches and non-planar patches. If we assume that the range image containing (n) data points was grouped perfectly into respective planar patches of size, $N = (k \times k)$, then the time for computing the planar attributes is reduced by a factor of (N). The graph search algorithm follows a Breadth-First Search (BFS) to add similar clusters with a computational complexity as $O((\frac{n}{N}).\log(\frac{n}{N}))$. In addition, the clustering algorithm uses a queue that maintains a list of patches that belong to the specific cluster. The queue provides a constant time access for inserting and deleting elements into the FIFO buffer. Once planar patch points are added in batches, the unassigned data points from the non-planar patches that lie on the border of two or more patches are considered one at a time. The number of unassigned data points is dependent on the nature of surfaces in the indoor environment. In few occasions, grouping of the range image into patches may result in a non-planar patch which includes data points from two or more planes. These patches are excluded in the first stage of clustering process. Later, the data points are individually added to the clusters after evaluating the segmentation criteria. If the environment is highly cluttered, the PPC algorithm behaves similar to the RG algorithm as it can find only few planar patches and the remaining data points belong to the non-planar patches.

5.7 Boundary detection of clusters

The data points in the range image are assigned to their respective clusters as described in the previous section. In this section, we describe the extraction process of the

boundary points or the convex hull of each cluster. The extracted boundary points will eventually replace the data points in the range image as the planar features. This feature extraction facilitates real-time 3D robot mapping and form a pre-cursor step to feature-based 3D registration, which use planar features as input. We use the labels of each of the clustered points in the range image (2D array) to identify the boundary points of a given cluster. As we move along the border of the cluster in a given array, the indices of the bordering label is selected as one of the boundary points. This is repeated for all clusters. This functionality is provided by a boundary detection program in image processing toolbox of MATLAB 7.6.0 to obtain the boundary points. In addition, the program is useful in identifying the boundary of holes in a given planar cluster. After extracting the boundary points, we fit them with as many 3D lines as possible if the collinear points satisfy orthogonal distance threshold ($0.01m$ was used in the experiment) and obtain the edge points that best fits the boundary of the planar region. If the data points do not satisfy the threshold, then they are retained as boundary points without fitting them to a line (e.g. data points lying on an edge of a plane segment with a curved boundary). The extracted boundary points together form an irregular polygon. We note that boundary points cannot be fit directly with 3D lines, since data points are affected by noise resulting in non-coplanar points. The boundary points are projected to fit them on a single plane along the major axis so that they are coplanar. The major axis is defined as the normal vector component that has the maximum value along the x, y or z axis. If n_x is the major axis of the normal vector of the planar cluster i.e $n_x > n_y \,\&\, n_x > n_z$, then the x-component of each boundary point is recomputed as

$$x_i = \left(-n_y.\, y_i - \, n_z.\, z_i + p\right) \Big/ n_x \qquad 5.13$$

where $i = 1 \ldots m$ boundary points of a given cluster.

5.8 Experimental results

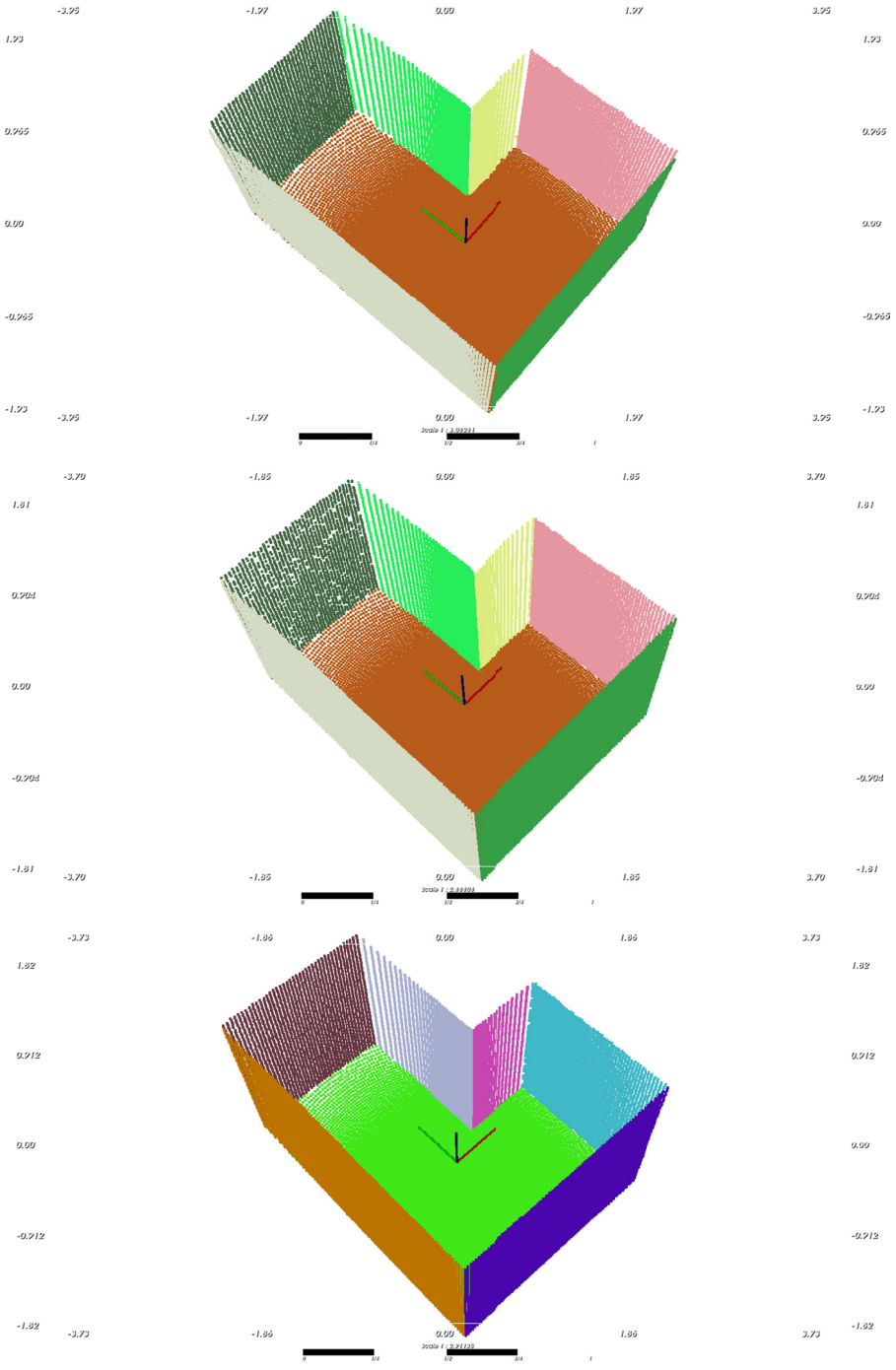
In this section, the performance of the PPC algorithm was evaluated in a number of experimental scenarios. In the first scenario, the range image was obtained through simulation to produce a complete planar environment as the ground truth and to compare the performance of the PPC algorithm with the RG and RANSAC plane segmentation algorithms at different noise levels of range measurement. Then, the PPC algorithm was evaluated in real-time using real range images acquired by a robot in a corridor consisting of largely planar surfaces. The third experiment was designed to test the robustness of the PPC algorithm in a highly cluttered environment. The range images that are acquired by the range scanners in both the simulated and real environments are stored in a 2D array in the spherical coordinate system.

5.8.1 Quantitative analysis in a simulated environment

The 3D range images are simulated in a planar environment consisting of eight planes, which includes walls, a floor and a ceiling. The range data was obtained in form of a (360×360) range image stored in a 2D array using ray tracing and ray-polygon intersection algorithm. The sensor coordinate frame is assumed to be at a certain pose in the middle of the room. The range is calculated along the ray that originates at the center of the coordinate frame and intersects a planar segment along a certain direction. The indices of the array establish connectivity of each element to its neighbors. The direction of the ray is computed using unique elevation (θ) and azimuth (φ) angles. The range data was simulated with noise added and measured as a percentage of the range distance. This noise is an approximative model of the real range image described in the laser range scanner specification sheet. We varied the noise in the range image up to 1% of the range distance and evaluated the performance of the PPC, RG and RANSAC plane segmentation algorithms at different noise levels. The three

parameters (σ, λ, δ) of the PPC algorithm were fine-tuned empirically to achieve the best clustering results for range images simulated at different noise levels. The clustering results for two Range Images (RI) with noise variance of ($\sigma_{RI}^2 = 0.5\%$) and ($\sigma_{RI}^2 = 1\%$) are shown in Fig. 5.9 and Fig. 5.10 respectively. In these figures, we included all extracted planes for visualization except the ceiling to avoid overlapping of extracted ceiling and floor data points. We can observe in Fig. 5.9 that the three algorithms in comparison show an accurate segmentation under $\sigma_{RI}^2 = 0.5\%$ noise level. However, the RG algorithm resulted in under-segmentation for one distant plane from the center under $\sigma_{RI}^2 = 1\%$ noise level. This plane was split into two individual planes, which can be observed in Fig. 5.10(b).

Segmentation results of each algorithm can be fine-tuned by a set of parameters. In PPC algorithm, the parameter (σ) in the Gaussian similarity metric governs the tolerance of noise levels in the normal vectors. A large value of σ indicates less tolerance for noise in the normal vectors of the patches. It is set to lower values when the noise is higher than ($\sigma_{RI}^2 \geq 1\%$) of the range measurement. The parameter (λ) is the cut-off threshold for the Gaussian similarity metric function to identify coplanar patches. If the (λ) is set to low values, coplanar tolerance is high. More patches can be assigned to a cluster resulting in over-segmentation. If the (λ) is set to higher values, coplanar tolerance is low and may result in under-segmentation. Both (σ, λ) are unit-less quantities and has a range of [0,1]. The parameters can be varied depending on the noise levels of the range sensor. The parameter δ (measured in meters) is a measure of the orthogonal distance between the plane models of two patches when evaluating the segmentation criteria. A description of the parameters of the RG algorithm can be found in [Poppinga, Vaskevicius et al. 2008] and the RANSAC segmentation algorithm in [Schnabel, Wahl et al. 2007]. All parameters were set empirically to achieve the highest qualitative accuracy for each of the algorithms.

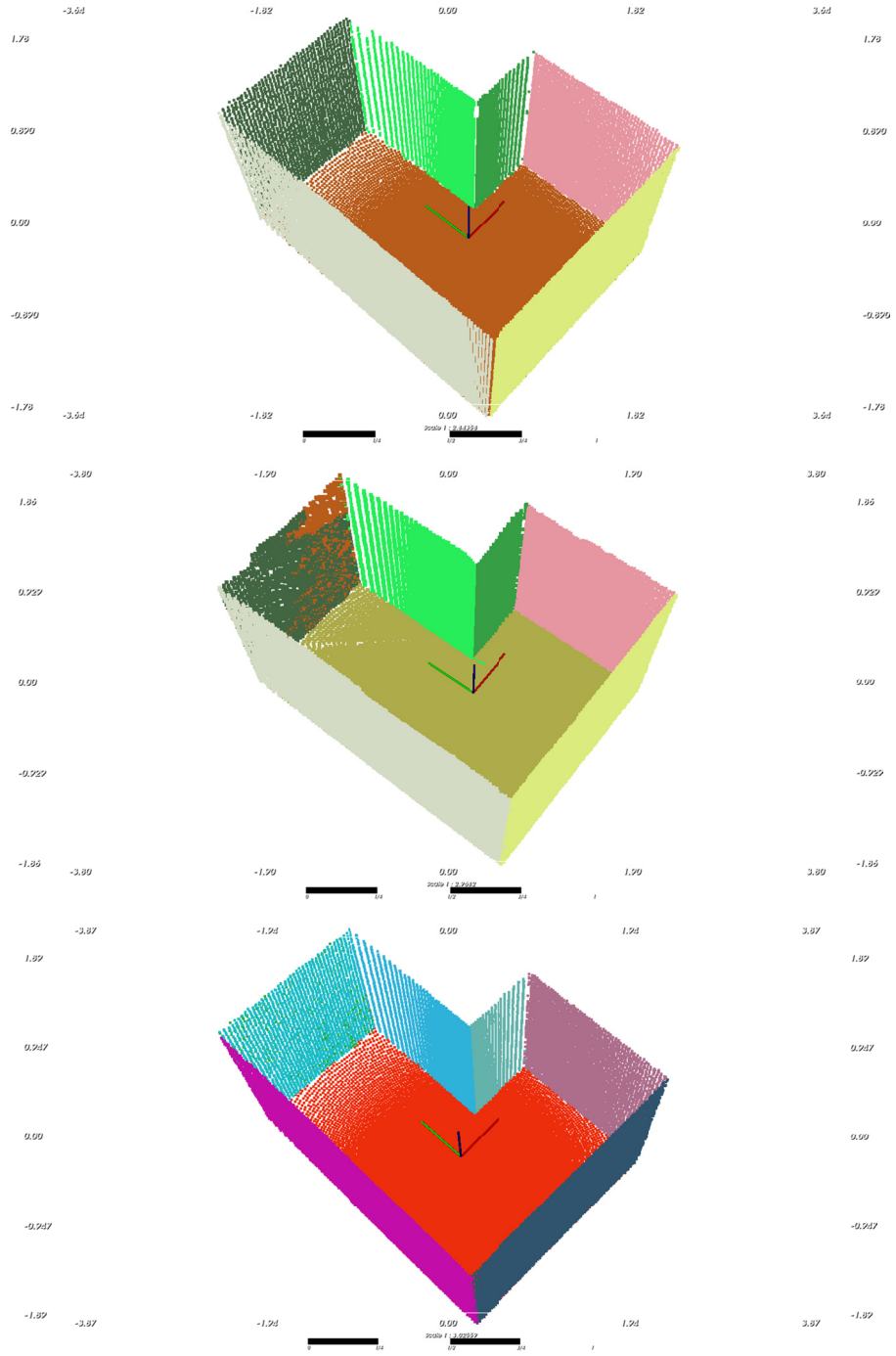


(a)

(b)

(c)

Fig. 5.9 Qualitative results of (a) PPC, (b) RG and (c) RANSAC plane segmentation algorithms for a noisy range image with noise levels of 0.5% of the range distance measured.



(a)

(b)

(c)

Fig. 5.10 Qualitative results of (a) PPC , (b) RG and (c) RANSAC plane segmentation algorithms for a noisy range image with noise levels of 1% of the range distance measured.

Table 5-1. Performance of PPC, RG and RANSAC plane segmentation algorithm

Patch-based Plane Clustering							
True Values		Noise levels - 0.0 variance		Noise levels - 0.005 variance		Noise levels - 0.01 variance	
Plane	Cluster Size	Accuracy (%)	Over-Seg (%)	Accuracy (%)	Over-Seg (%)	Accuracy (%)	Over-Seg (%)
P1	40708	99.98	1.27	99.89	1.08	99.57	0.58
P2	39432	99.86	0.30	99.49	0.21	99.73	0.07
P3	18022	99.59	0.90	99.68	0.75	99.68	1.08
P4	13815	98.15	0.41	98.23	0.29	98.41	0.32
P5	6293	98.92	0.41	99.22	0.84	99.19	2.07
P6	4721	95.36	2.73	95.53	2.50	96.00	3.20
P7	3564	94.22	1.96	94.44	4.43	94.47	2.33
P8	2537	99.33	1.69	99.68	1.89	99.01	1.06
Normal calc. time (ms)			13.857	-	15.399	-	15.524
Clustering time (ms)			35.668	-	31.598	-	33.069
Refinement time (ms)			16.096	-	27.262	-	36.833
Total time (ms)			65.621	-	74.259	-	85.426
				Parameters Used			
		Sigma (σ)	0.98		0.98		0.95
		lamda (λ)	0.98		0.95		0.955
		delta (δ)	0.012		0.015		0.028

Region growing plane segmentation							
Ground Truth		Noise levels - 0.0 variance		Noise levels - 0.005 variance		Noise levels - 0.01 variance	
Plane	Cluster Size	Accuracy (%)	Over-Seg (%)	Accuracy (%)	Over-Seg (%)	Accuracy (%)	Over-Seg (%)
P1	40708	100.00	0.36	99.98	0.27	99.92	0.30
P2	39432	99.98	0.13	99.95	2.35	99.78	2.29
P3	18022	99.98	0.47	99.93	0.32	99.77	0.53
P4	13815	100.00	0.66	99.96	0.58	99.56	0.77
P5	6293	99.52	0.37	99.49	0.70	95.47	1.03
P6	4721	96.08	0.49	99.94	0.57	99.60	1.42
P7	3564	99.97	0.45	76.04	0.67	-	-
P8	2537	99.96	11.82	99.57	3.31	99.92	3.74
Total time (ms)			19158.7	-	19710.5	-	18714.1
				Parameters Used			
		delta (δ)	0.25		0.25		0.4
		epsilon (ε)	0.1		0.1		0.1
		gamma (γ)	0.005		0.008		0.014

RANSAC plane segmentation							
Ground Truth		Noise levels - 0.0 variance		Noise levels - 0.005 variance		Noise levels - 0.01 variance	
Plane	Cluster Size	Accuracy (%)	Over-Seg (%)	Accuracy (%)	Over-Seg (%)	Accuracy (%)	Over-Seg (%)
P1	40708	100.00	0.37	99.98	0.48	99.92	0.52
P2	39432	100.00	0.40	99.98	0.46	99.90	0.41
P3	18022	99.97	0.14	99.71	0.01	99.36	0.32
P4	13815	99.98	0.49	99.67	0.32	98.67	0.14
P5	6293	99.95	1.62	99.38	0.78	99.08	0.51
P6	4721	100.00	0.17	99.81	0.32	99.70	1.78
P7	3564	99.97	0.20	99.05	0.00	95.17	0.00
P8	2537	99.92	0.20	99.68	0.04	99.37	0.00
Total time (ms)			1120.32	-	1103.62	-	1184.06
				Parameters Used			
		threshold (λ)	0.005		0.015		0.022

Table 5-1 lists the quantitative results of plane segmentation of noisy range images in a simulated environment by the three algorithms, namely PPC, RG and RANSAC plane segmentation. The table includes the actual cluster size of each planar segment (ground truth), accuracy of segmentation of each cluster in percentage, over-segmentation in percentage with respect to the actual cluster size and overall computation time for each algorithm. It can be noted that the PPC algorithm was able to extract all eight planes undergoing a small percentage of over-segmentation or under-segmentation under noisy conditions. The computation time in each case was under 0.08s. The number of respective cluster points varied slightly compared with the RG and RANSAC plane segmentation algorithms, which depends on whether the edge points were included or excluded from the plane segments. The RANSAC-based plane segmentation executed under 1.18s and was able to accurately extract all eight plane segments for both noisy range images. For the range image with 1.0% noise, the RANSAC-based plane segmentation algorithm extracted edge points as a separate plane (272 points). The RG algorithm was successful in extracting eight planes for the range image with 0.5% noise. However, one of the wall segments was under-segmented splitting them into two separate planes for the range image with 1% noise (Fig. 5.10). Further fine-tuning of parameters for RG algorithm resulted either in one plane segment bleeding into another or under-segmentation.

We can attribute the significant improvement in speed of the PPC algorithm to eliminating redundancy in data fitting. The redundancy can be seen when the Hessian plane parameters and covariance matrix of all data points for a given planar segment are recomputed whenever a new point is added to a cluster by the RG algorithm. In RANSAC plane segmentation algorithm, the data points are repeatedly tested to verify if it fits a hypothesis plane model and sometimes tested for an incorrect hypothesis plane model. We avoid the redundant approaches and instead compute Hessian plane

parameters of each patch at the beginning of the algorithm and hence incur an additional computation cost, which is seen in Table 5-1. Once the planar patches are included in their respective clusters, the unassigned points are added to their respective cluster, which incurred an additional computation cost less than 0.02s for the 360×360 range image. The overall processing time of the PPC algorithm is only less than $1/10^{th}$ of the RANSAC-based plane segmentation algorithm and much less compared to the RG algorithm. The accuracy of the PPC algorithm varies between 99% and 94% for different cluster sizes. PPC algorithm under-went a small percentage of over-segmentation ranging from 0.07% to 3.2% for the range image with 1% noise.

5.8.2 Qualitative results of PPC algorithm in structured indoor environments

We analyzed the performance of the PPC algorithm in a largely planar environment consisting of walls, ceiling, floor and doors as shown in Fig. 5.11. The algorithm was able to cluster the planar patches to a good accuracy in the indoor environment (Fig. 5.12). The Gaussian parameters ($\sigma = 0.95$, $\lambda = 0.95$) and the distance threshold ($\delta < 0.01$m) was set empirically while fine-tuning the PPC algorithm.

5.8.3 Qualitative results of PPC algorithm in a cluttered environment

In this experiment, we acquired four range images at different locations in a highly cluttered laboratory environment (Fig. 5.13) to evaluate the robustness of the clustering algorithm. The qualitative results of the PPC algorithm are presented in the form of clusters and represented by different colors as shown in Fig. 5.14. Table 5-2 presents the quantitative results of clustering four range images in the cluttered environment. We achieved a high data compression rate close to 96%. The PPC algorithm was robust to sensor noise and able to cluster the planar regions from all four range images. The number of clusters extracted in the process is dependent on the parameters, which were fine-tuned empirically to achieve highest accuracy for the PPC algorithm. The PPC

algorithm was executed in Matlab 7.6.0, 2GB RAM, 2.4 GHz Intel® processor to extract polygons from the four range images. Since we do not have the ground truth model of the real environments, it is difficult to make meaningful quantitative comparison with other algorithms such as the RG and RANSAC-based plane segmentation algorithms.

Fig. 5.11 Setup shows a ground robot acquiring a 3D range image in an office corridor.

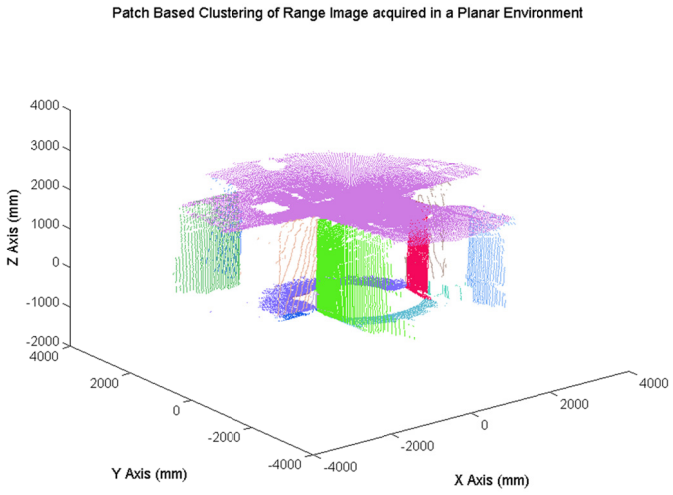


Fig. 5.12 shows clustered range image acquired in the office corridor.

Fig. 5.13 Setup showing a pioneer® robot mapping in a cluttered indoor environment

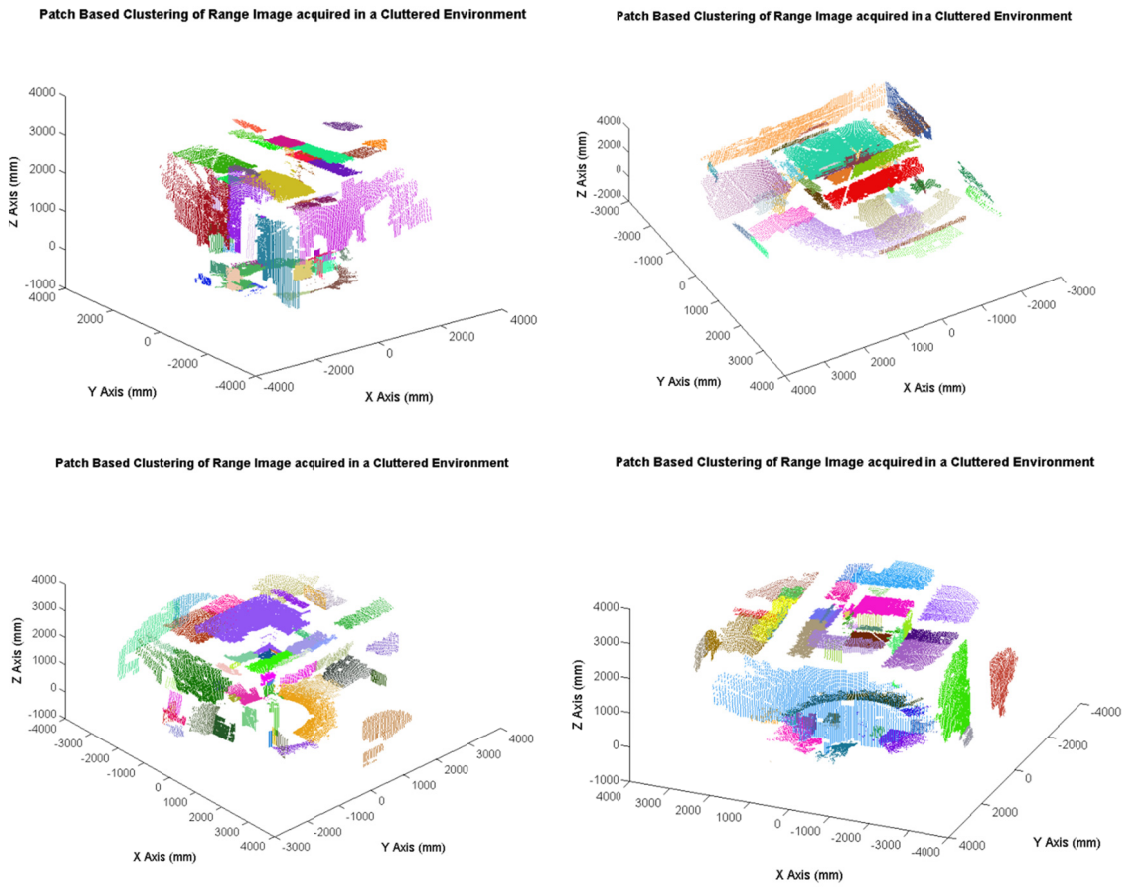


Fig. 5.14 Result of patch-based plane clustering of four range images acquired in a highly cluttered indoor environment at 4 different locations.

Table 5-2. Performance of PPC clustering algorithm in cluttered environment

Range Image	1	2	3	4
No. of Extracted Planes	52	54	67	60
No. of clustered points	91182	83497	78532	81813
No. of boundary points	6123	6014	5673	5426
Data Compression (%)	**96.81**	**95.90**	**96.14**	**96.31**

PPC algorithm parameters (σ = 0.95, λ = 0.95, δ = 0.04m)

5.9 Limitations of the PPC algorithm

The PPC algorithm showed a significant improvement in computation time without much loss of accuracy in comparison with the state-of-the-art RANSAC and RG algorithms. The analysis in section 5.4 indicates that normal vectors calculation of local patches can be accurate enough for plane segmentation, if the patch size is properly selected. It is especially true in structured indoor environments consisting of large planar surfaces, where the larger patch size leads to more computational efficiency. However, it is necessary to select a small patch size in highly cluttered environments. In the worst case scenario, if the environment is highly cluttered with no planar regions, then the PPC algorithm behaves invariably like a RG algorithm as it cannot find any planar patches to cluster. The sampling of range data into patches or breaking down the range image into grids leads to inaccurate inclusion of points around the border of the polygons where some patches may include points from two or more planes. A higher threshold for the Gaussian similarity metric eliminates the inclusion of this type of patch into neighboring planar clusters. A lower threshold leads to false positive data points being included into the cluster. Thus, an additional step was necessary to handle border points & patches. The refinement process is an additional step we incorporated in our

model where the points of border patches were tested for coplanarity condition. The test ensured that the data points along the border of each polygon were included into the correct cluster if they were found to be a false positive result for a given cluster. The computation time of this refinement process is a function of number of clusters extracted from a range image, which in turn is dependent on the environment being scanned. However, the time taken by the refinement process is negligible and the overall computational time of the PPC algorithm is only less than 1/10 of that of the RANSAC-based plane segmentation algorithms and much less compared to the RG algorithm.

CHAPTER 6

Polygon-based scan registration

We present a methodology of using polygon features extracted from overlapping range images for scan registration. In the process, we determine the relative pose from the two partially overlapping range images. The relative pose is used to transform and align one range image with respect to another in a reference frame. A two-stage approach is necessary to align the overlapping range images. The first stage deals with establishing the corresponding polygons from the two polygon sets extracted from the range images. The second stage deals with the optimization problem where the geometric distance between the corresponding polygons are minimized using a nonlinear parametric estimation approach.

6.1 Polygon correspondence

Polygon correspondence is the problem of identifying the polygon pairs, one from each polygon set extracted from two partially overlapping range images. Let P and Q be two polygon sets that have a partial overlap for establishing the correspondences and perform polygon registration. While performing experiments, we took adequate measures to ensure that the range images from the dual robots have a minimum overlap. Before we establish correspondence, the initial pose estimate $[R_{init}, t_{init}]$ from the camera pose estimation algorithm is applied to the polygon set Q.

$$Q' = R_{init}.Q + t_{init} \tag{6.1}$$

The experimental results indicate that pose estimate from the camera forms a good estimate and applying the pose transformation on the second polygon set geometrically brings it closer to their corresponding polygons. The algorithm uses all of the geometric attributes of a polygon, namely the Hessian plane parameters, its position in space (mean of boundary points) and the area of a given polygon to establish the correspondence. Polygon Correspondence (PC) algorithm (Refer to Algorithm 3) identifies the polygon in set P corresponding to polygon in the set Q' that represent the same plane in the environment. We probabilistically determine whether the two polygons reside in the same vicinity using the Radial Basis Function (RBF) Gaussian metric kernel defined by

$$kernel(n^p, n^{Q'}) = \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{\left(n_x^P - n_x^{Q'}\right)^2 + \left(n_y^P - n_y^{Q'}\right)^2 + \left(n_z^P - n_z^{Q'}\right)^2}{2\sigma^2}\right) \quad 6.2$$

where $n^p, n^{Q'}$ are the normal vectors of the polygons P and Q'.

The two polygons in set P and set Q' are corresponding, if and only if it satisfies the RBF Gaussian similarity metric and can be fine-tuned by the parameter (σ). Also, the distance between the mean position of the two corresponding polygons must meet a distance criterion (δ). (Line 9, Algorithm 3). The algorithm performs well since it is entirely dependent on geometric attributes of polygon features of two range images transformed by an initial estimate of pose. The algorithm does not include a polygon if its corresponding polygon is missing due to occlusion or noisy range data. Their correspondence is individually weighed by the Gaussian metric to see if the polygon attributes satisfy the geometric proximity threshold and are oriented in the same direction given by their normal vectors. The polygonal area (Eqn. 6.3) is computed for each polygon and filtered to a preset threshold to eliminate small sized polygons that may not lead to further improvement while registering polygon sets as discussed in section 6.2.

$$2 * A(P) = n_p.\left(\sum_{i=1}^{N-1}(V_{2i} - V_0) \times (V_{2i+1} - V_{2i-1}) + (V_{2N} - V_0) \times (V_k - V_{2N-1})\right) \qquad 6.3$$

where n_p is the normal vector of the polygon, $V_{i=1,\ldots.m}$ are vertices that define the m boundary of the polygon, $N \leq (n-1)/2$ is the greatest integer, $(k = 0)$ if n is odd or $(k = m - 1)$ if m is even, and $A(P)$ is the area of the polygon. This equation was proposed by [Gelder 1995] and is based on decomposition of quadrilaterals.

Algorithm 3 Correspondence（P, Q'）

1. N = length (P) // P and Q' contain a list of polygons. Each element
2. M = length (Q') // contains nx3 vertices, n is different for each polygon
3. **for** $i = 1\ to\ N$
4. **for** $j = 1\ to\ M$
5. $[\hat{n}_1, d_1, m_1]$ = **fit_plane** (p_i)
6. $[\hat{n}_2, d_2, m_2]$ = **fit_plane** (q_j)
7. y = **kernel** $(\hat{n}_1, \hat{n}_2, \sigma)$ // Gaussian similarity metric
8. d = **dist** $(\hat{n}_1, d_1, m_2)$
9. if $(y > \gamma)$ && $(d < \delta)$ // Corresponding polygon pair
10. index = **add_to_list** (i, j)
11. **end if**
12. end for
13. end for
14. end of program

The *fit_plane* function in (Line 5, Algorithm 3) computes the Hessian plane parameters (n, d) and the mean (m) of all vertices of polygon. The *kernel* function in (Line 7, Algorithm 3) is the Gaussian metric function given in Eqn. 6.2. Following are some caveats to the correspondence problem that we considered. A planar region in the environment may be represented by unequal number of polygons from two different

scans. Most often, a planar region is partially scanned in one of the 3D scans and hence the two polygons will have different irregular shapes. The attributes that do not change are normal vector and their position in space. A one-to-one relationship is established between two polygonal sets corresponding to a planar region based only on these two attributes.

6.2 Polygon registration algorithm

In the previous section, polygon correspondence algorithm identifies k polygon correspondences between the polygon set P and Q', which forms the input to the polygon registration algorithm. In this section, we introduce the polygon registration algorithm based on minimizing the corresponding polygon-polygon distance extracted from the two partially overlapping range images. A control-flow diagram of the PR algorithm is provided at the end of this section. A polygon is defined by a set of boundary points. Each of the boundary points are affected by noise and it is also essential to deal with irregular polygon shapes. Let polygons P be the reference set with a fixed pose. Let polygon set Q be the input set which needs to be transformed to align with polygon set P. As a first step, the planar projection of polygon vertices in set Q over their corresponding polygons in P is computed to eliminate the noise aberrations and flatten the vertices on a single plane. In addition, one-to-one point correspondence between boundary points of corresponding polygon-pairs is established. This forms the input to the polygon registration algorithm. This operation is repeated in each iteration of the LMA to reduce the effect of noise in range measurement. Since the algorithm deals with flat surfaces and minimizes the distance between the boundary points and their projection on a planar surface, the ideal distance metric would be the orthogonal distance measure.

The projection function computes the projected points of Q over P. Assume that the polygon, $p_x \in P$ corresponds to polygon, $q_y \in Q$. Let X_k be k boundary points in the

polygon, $q_y \in Q$. Let $\langle \hat{n}_p, d_p \rangle$ be the Hessian plane parameters for the given polygon, $p_x \in P$. We extract the boundary point projection (Y_k) for each of the boundary points X_k over the polygon $p_x \in P$ given by Eqn. 6.4 and 6.5.

$$dist = \hat{n}_p . X_k - d_p \tag{6.4}$$

where $dist$ is the orthogonal distance from polygon p_x to point, X_k

$$Y_k = X_k - dist . \hat{n}_p \tag{6.5}$$

We minimize the following equation to obtain the non-linear parameters $(\alpha, \beta, \gamma, t_x, t_y, t_z)$ embedded in Transformation Matrix (T).

$$minimize\ E = \sum_{k=1}^{K} \left\| T_o T_{k-1} X_{k,} Y_k \right\| \tag{6.6}$$

where T_o is the initial transformation matrix, T_{k-1} is the estimated transformation matrix during each iteration.

$T_{4\times4}$ is the transformation matrix that includes both rotation $(R_{3\times3})$ and translation $(t_{3\times1})$ given by

$$R = R_{Z,\gamma} \times R_{Y,\beta} \times R_{X,\alpha} \tag{6.7}$$

$$t = [t_x \quad t_y \quad t_z]^T \tag{6.8}$$

$$\mathrm{T} = \begin{bmatrix} \mathbf{C}(\gamma).\mathbf{C}(\beta) & \mathbf{C}(\gamma).\mathbf{S}(\beta).\mathbf{S}(\alpha) - \mathbf{S}(\gamma).\mathbf{C}(\alpha) & \mathbf{C}(\gamma).\mathbf{S}(\beta).\mathbf{C}(\alpha) + \mathbf{S}(\gamma).\mathbf{S}(\alpha) & t_x \\ \mathbf{S}(\gamma).\mathbf{C}(\beta) & \mathbf{S}(\gamma).\mathbf{S}(\beta).\mathbf{S}(\alpha) + \mathbf{C}(\gamma).\mathbf{C}(\alpha) & \mathrm{s}(\gamma).\mathbf{S}(\beta).\mathbf{C}(\alpha) - \mathbf{C}(\gamma).\mathbf{S}(\alpha) & t_y \\ -\mathbf{S}(\beta) & \mathbf{C}(\beta).\mathbf{S}(\alpha) & \mathbf{C}(\beta).\mathbf{C}(\alpha) & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.9}$$

In homogeneous coordinate system, T is expressed as above where $\boldsymbol{S}$ indicates *sine* function and $\boldsymbol{C}$ indicates *cosine* function.

The rotation matrix (R) consists of 9 DOF non-independent parameters, which are computed using the three angles $(\alpha_z, \beta_y, \gamma_x)$. Instead, there is a convenient way to reduce the rotation matrix into a 3 DOF independent axis-angle parameters, where the

rotation can be expressed by an angle ϕ about an axis vector $W = (w_x, w_y, w_z)$. The Rodriques formula (Eqn. 6.10) provides an effective way to transform R to $\langle W, \phi \rangle$ and vice-versa.

$$R_{3\times3} = I_{3\times3} + \frac{sin\|W\|}{W}.[W]_\times + \frac{1 - cos\|W\|}{\|W\|}[W]_\times{}^2 \qquad 6.10$$

where $[W]_\times$ also known as "cross product matrix" is given by the skew-symmetric matrix,

$$[W]_\times \triangleq \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix} \qquad 6.11$$

$\langle W, \phi \rangle$ is retrieved from the rotation matrix $R_{3\times3}$ as follows,

$$\phi = \|W\| = \cos^{-1}\left(\frac{trace(R) - 1}{2}\right) \qquad 6.12$$

$$W = \frac{\|W\|}{2\sin\|W\|}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \qquad 6.13$$

The Rodriques formula in Eqn. 6.10 is substituted into the rotation component (R) of the transformation matrix (Eqn. 6.9). The rotation matrix is now defined by the axis-angle (w_x, w_y, w_z, ϕ) and the translation vector $t = (t_{x,} t_y, t_z)$.

$$T = [R_{[w,\varphi]};\ -R_{[w,\varphi]} * t] \qquad 6.14$$

The transformation T is homogenized by padding $[0\ 0\ 0\ 1]$ as the last row of the matrix. The final transformation of the polygon vertices $X_i = (x_i, y_i, z_i)$ is given by the function,

$$f(X_i, x) = T * (x_i, y_i, z_i, 1)^T \qquad 6.15$$

We obtain the numerical solution of relative pose variables defined by $x = (w_x, w_y, w_z, t_{x,} t_y, t_z)$ by applying the LS minimization technique based on LMA

[Marquardt 1963]. This nonlinear minimization technique proved to be the most stable of the iterative techniques while considering the non-linearity embedded in the Euclidean space. In this approach, the LMA minimizes the LS distance between the polygon set (Y_i) and the transformed polygon set, $f(X_i, x)$. The difference function is given by,

$$r_i(x) = [Y_i - f(X_i, x)] \tag{6.16}$$

The objective function to be minimized is given by

$$\sum_{i=1}^{m} \big(r_i(x)\big)^2 = ([Y_i - f(X_i, x)])^2 \tag{6.17}$$

Jacobian of the above objective function is computed as

$$J = \left[\frac{\partial r_i}{\partial w_x} \frac{\partial r_i}{\partial w_y} \frac{\partial r_i}{\partial w_z} \frac{\partial r_i}{\partial t_x} \frac{\partial r_i}{\partial t_y} \frac{\partial r_i}{\partial t_z}\right] \tag{6.18}$$

We used Matlab® symbolic toolbox for deducing the Jacobian functions with respect to the pose parameters. The Hessian matrix of the function $f = J(x)^T J(x)$ is substituted into the Eqn. 6.19 given below, which is the Levenberg-Marquardt algorithm

$$x^{(k+1)} = x^{(k)} - (J(x)^T J(x) + \mu_k I)^{-1} J(x)^T r(x) \tag{6.19}$$

$\mu_k I$ can be seen as an approximation to the second derivatives of the Hessian function and guarantees convergence. μ_k is reduced by a factor of 10 if the objective function is converging to a global minimum or increased by a factor of 10 if the objective function is diverging and helps speed up the convergence rate. The initial rotation (R_{init}) and the translation (t_{init}) obtained by the camera pose estimation algorithm is used to transform the polygon set Q'. The rotation matrix (R) is converted into axis-angle parameters using the **Rodrigues** equations in (6.12) and (6.13). The **LMA** computes the nonlinear parameters (x), which are obtained from Eqn. 6.19 at the end of k^{th} iteration. The axis-angle parameters are converted back into Euler angles using the Rodrigues

equation (**InvRodrigues**) (Eqn. 6.10). The Euler angles $(\alpha_z, \beta_y, \gamma_x)$ and the translation vector $t = (t_x, t_y, t_z)$ are then applied to transform the polygon set Q', which is expected to align the two polygon pairs $X_i \in Q'$ and $Y_i \in P$ one step closer to the complete alignment of two range images. The algorithm checks if the percent change in minimization error is more than a preset threshold $(E < 0.001)$. If not, the polygon **projection** function computes new points to be minimized with the input set and the iterations continue until the objective is reached. The simulation and experimental results include the performance of the algorithm under noisy conditions and is provided in section 6.3.

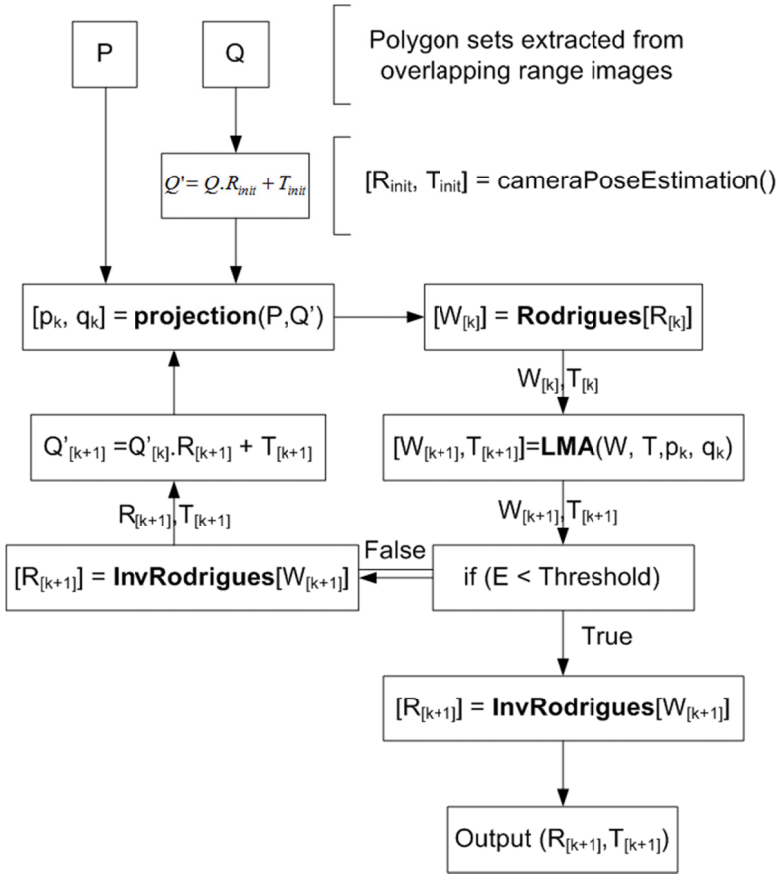


Fig. 6.1 The control flow diagram of the Polygon Registration (PR) algorithm

6.2.2 Global relaxation

In this work, our focus is on polygon-based registration and will consider exploring global relaxation techniques as future work. Here, we summarize the steps to taken to improve the accuracy of the global map. Multiple range images fused together with respect to global reference frame may lead to accumulated errors that can be attributed due to noisy range images. The registration algorithms compute the relative pose from partially overlapping range images, which may not be all that accurate when the range images consist of noisy outliers. The inconsistent pose of the robots are then transformed to a global frame. A large number of pose transformations over long distances lead to an inconsistent global map. A loop closure is improbable in such cases. A global relaxation technique is an important step as described in the flow diagram (Fig. 4.3) to perform loop closure of large maps. [Surmann, Nüchter et al. 2004] introduces an automatic global relaxation technique known as simultaneous matching to complete loop closure of global maps. [Pathak, Birk et al. 2010] provides another approach to global relaxation where only translation errors are corrected and the rotational aspect is ignored considering that an orientation sensor or accurate measure is obtained through registration. We implemented following steps to eliminate accumulating errors leading to global map construction. We begin with computing the 6 DOF pose parameters between two partially overlapping range images. One of the two range images is assumed to be part of the global reference frame and the other range image is transformed with respect to the global reference frame using the pose parameters computed in the previous step. This process is repeated until all range images are fused together to form a global map. Every new addition of the range image to the global map follows with a test for loop closure. A confidence measure is obtained using the Gaussian similarity metric, which compares the corresponding polygonal features of the last range image transformed with respect to the global reference frame and each of the

previous range images added to the global map. A successful loop-closure match will lead to computing the relative pose parameters between the two range images that close the loop. The difference in the final pose estimation between the globally transformed pose and the previous step is the net accumulated distance error. The error is distributed evenly, which is reflected in the relative pose of each of the range images fused together to form the global map. We suggest readers to refer to [Nüchter 2009; Surmann, Nüchter et al. 2004] for more on global relaxation techniques.

6.3 Experimental results

The dual-robot system consists of two Pioneer® ground robots mapping the indoor environment. The robots are equipped with a Hokuyo® range sensor, which is mounted on a panning device. The perspective camera is mounted in line with the mirror of the laser scanner at a known distance. Calibration is performed at the beginning of the experiment to determine the internal parameters and distortion parameters of the camera. The experiment was conducted in two different regions of the indoor environment, viz. 1) a structured corridor that is mostly planar and lacking any visual features, 2) a highly cluttered laboratory environment that is partially planar. In addition, we performed simulation to test the robustness of the PR algorithm with respect to noise. The algorithms are tested in MATLAB 7.6.0 and for preprocessing the sensor data. The graphical visualization software is programmed in Java3D®. All programs were implemented on Intel Core 2 Duo 2.4 GHz processor, 2GB RAM.

6.3.1 PR algorithm simulation

Two identical sets of irregular polygon boundaries were simulated in Euclidean space and Gaussian noise was added to each boundary point of the polygon. One polygon set was transformed to a known orientation and translation. Both the polygon sets were

registered using the PR algorithm. The noise levels of boundary points of the polygon were measured in decibel given by,

$$SNR = 10.log_{10}\frac{\mu}{\sigma}$$

where μ is the mean of range distance of all boundary points of the polygon set and σ is the noise variance. The PR algorithm converged to a global minimum every time irrespective of the noise levels (Fig. 6.2). The minimization error converged at a higher level with increasing noise levels, which was expected. The computational time (Table 6-2) of the PR algorithm did not increase with higher noise levels, which indicated that algorithm's execution time is not dependent on the noise levels. The estimated values of pose parameters $(\alpha, \beta, \gamma, t_x, t_y, t_z)$ with varying noise levels are shown in Fig. 6.3 and Fig. 6.4.

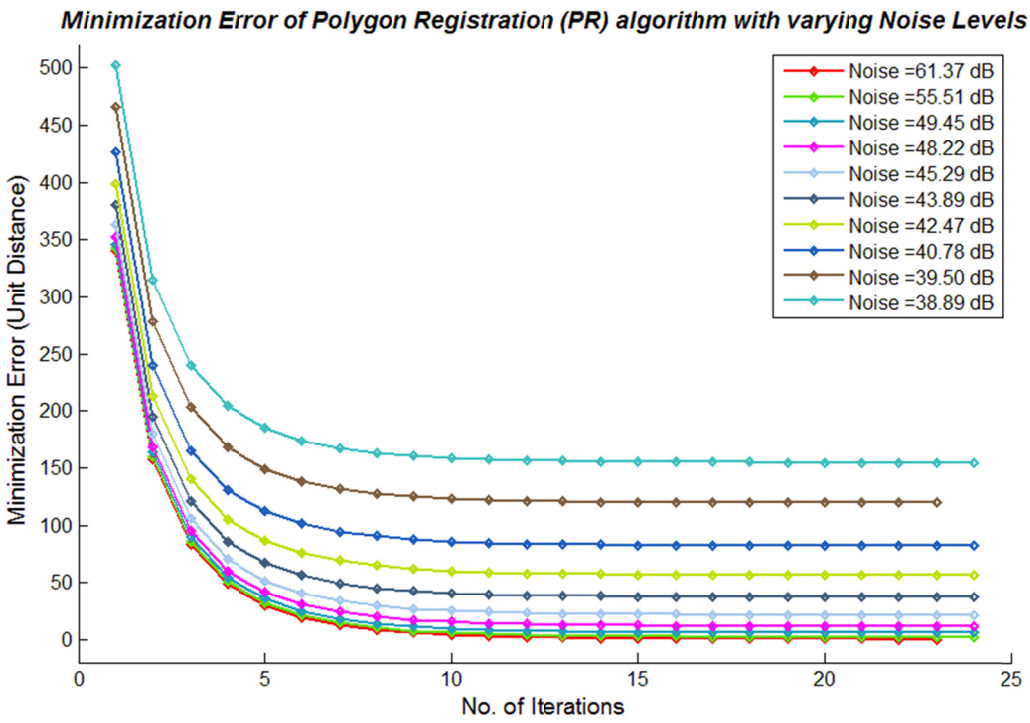


Fig. 6.2 shows minimization error resulting in registration of two sets of polygons defined by their boundary points. The variance noise levels (Gaussian) of the boundary points are increased (with increments of $\sigma = 0.02$). PR algorithm converges to a global minimum. The LS error increases with higher noise levels.

Table 6-1. Computation time of Polygon Registration algorithm with varying noise

Noise Levels (σ)	0.02	0.04	0.06	0.08	0.10	0.12	0.14	0.16	0.18	0.20
Avg. Time (s)	1.181	1.039	1.032	1.032	1.031	1.033	1.038	1.040	0.996	1.074

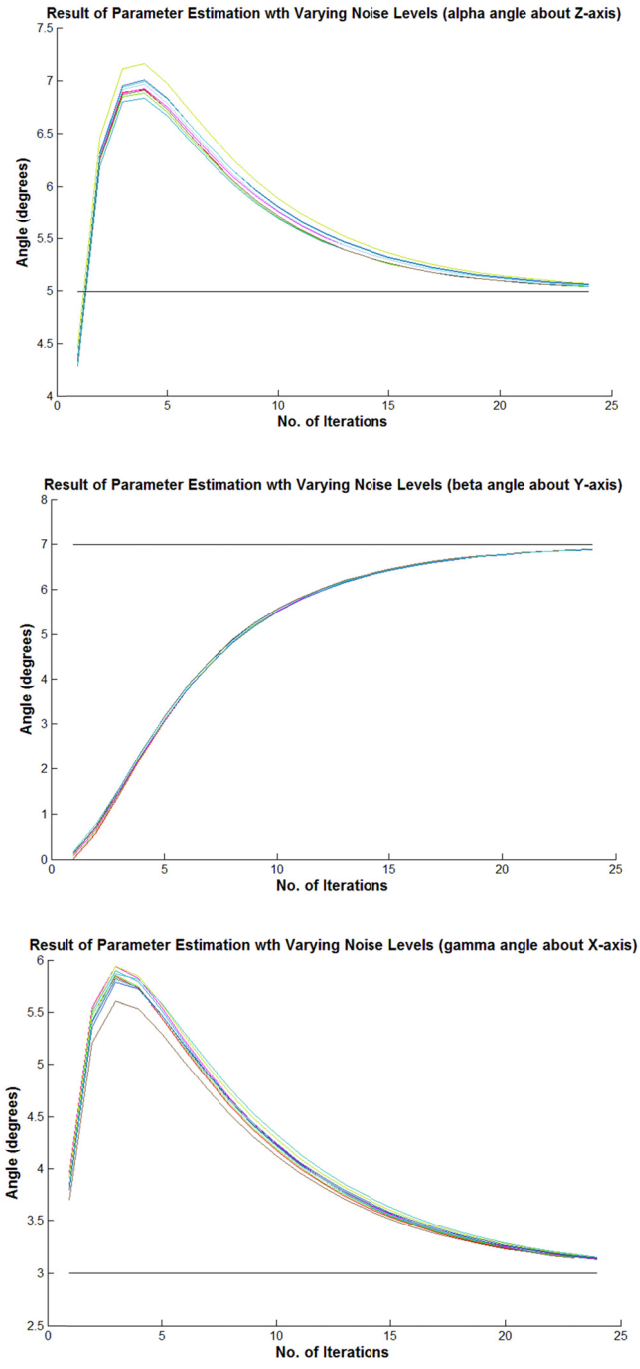


Fig. 6.3 The three rotation angles (α, β, γ) about Z, Y, X axes estimated by the PR algorithm with varying noise levels. The graph legend for this graph is the same as in Fig. 6.2. The solid line indicates the ground truth.

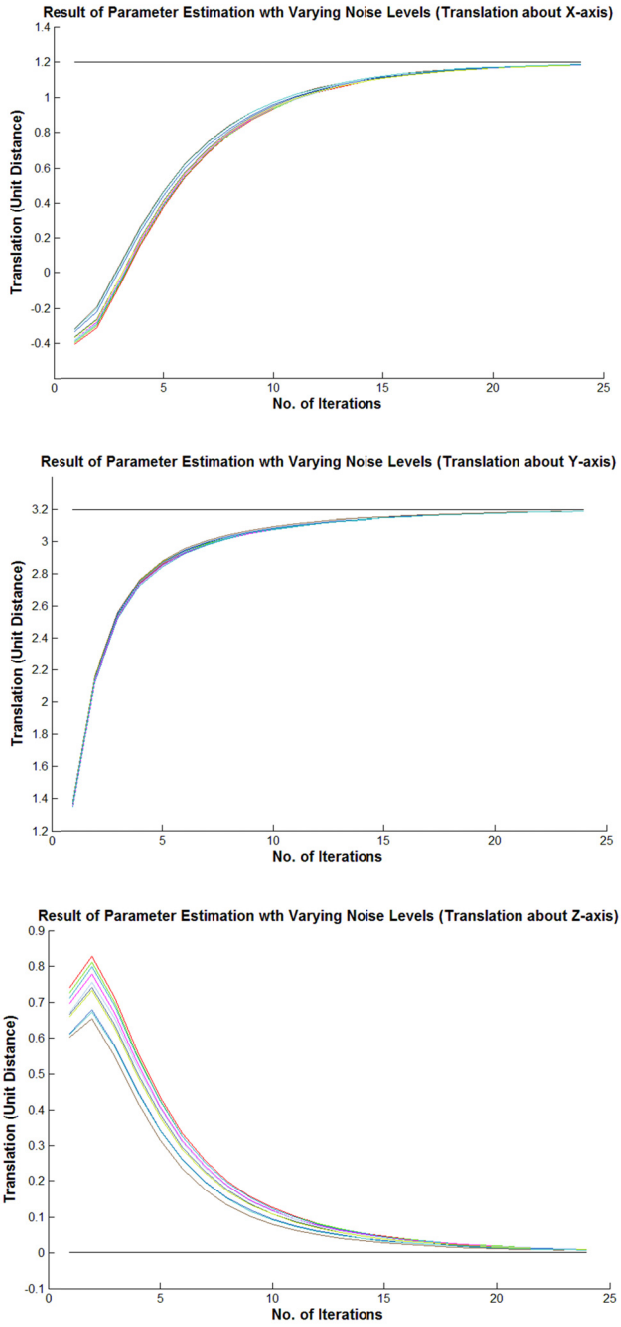


Fig. 6.4 The translation parameters (t_x, t_y, t_z) estimated by the PR algorithm with varying noise levels. The graph legend for this graph is the same as in Fig. 6.2. The solid line indicates the ground truth.

6.3.2 Experimental validation by loop closure

Two ground robots were deployed to construct a 3D map of the indoor corridor of Electrical Engineering department at the Grove School of Engineering, City College of New York. The robots acquired a total of 37 scan pairs and images from the perspective camera. The two robots were commanded to move in tandem and alternately take new positions as described in section 4.4.1. The distance between the robots was maintained such that there is at least a minimum partial overlap between the two range images. The ground robots are equipped with a 3D range scanner and a perspective camera. The robots were mounted with a cube and quadrangular visual markers were glued to all sides of the cube. The markers could be easily tracked from all directions using the camera on the other robot. The markers have different patterns on each side and the vision algorithm that includes the camera pose estimation algorithm extracts the exact orientation of the robot based on the pattern. Quadrangular markers are of $0.15m$ in width and height, which is big enough to accurately estimate the pose of the camera with respect to the markers, at a distance between $1\sim4m$. The camera images and range data were processed to acquire a complete 3D map of the indoor corridor. At first, the raw distorted image acquired by the trailing robot was undistorted using calibration parameters. The camera pose estimation algorithm then computed the camera pose (on the trailing robot) with respect to the marker coordinate frame of the lead robot. The initial pose was applied to transform the range image of the lead robot. Polygons extracted from overlapping range images after applying initial transformation are shown in Fig. 6.6. The PC algorithm then extracted the corresponding polygons. It picked up 100% true positives with a good initial pose estimate. In case of occlusions, the corresponding polygons were left out in the registration process. The corresponding polygons of a few scan pairs are shown in Fig. 6.7. Number of corresponding polygons extracted in 37 scan pairs and the computation time is listed in Table 6-4. The PR

algorithm converged to a global minimum in each case (Fig. 6.8). The initial estimate from the pose estimation algorithm aided the PR algorithm to converge to a global minimum for each scan-pair. We noted that graphical visualization of 37 point clouds transformed into their respective global poses turned out to be memory-hogging and it took a heap space of 490.71 MB whereas a polygon map took about 13.07 MB of heap space. Average computation time for each sub-process involved in the 3D mapping process is given in Table 6-2. The complete polygon map of the indoor corridor after global relaxation is shown in Fig. 6.9. A partial view of the interior and the exterior fused range image point clouds are shown in Fig. 6.10 and Fig. 6.11. The dual robots were deployed at various locations in the corridor with a loop. The robots path was $\sim 124.55m$ and resulted in an error of $3.9446m$ of accumulated distance after all range images were fused. With this experiment, the robot achieved the loop closure while constructing a global map with an error of $\sim 3.16\%$ of the total distance. A simple global relaxation technique was applied to the robot nodes where the range images were acquired. The outcome of relaxation is shown in terms of the change in robot trajectory Fig. 6.12. It shows the pre-relaxation path and post-relaxation path of the robots. A complete polygon map (exterior) of the indoor corridor after fusing multiple polygon sets can be seen in Fig. 6.13.

Table 6-2. Average execution time of all steps in 3D mapping for a typical run

Mapping process	Average execution time (sec)
Marker pixel coordinates extraction	2.8098
Camera pose estimation	0.0413
Polygon extraction	2.70
Polygon Correspondence (PC)	0.014
Polygon Registration (PR)	3.63
Total Time	9.1951

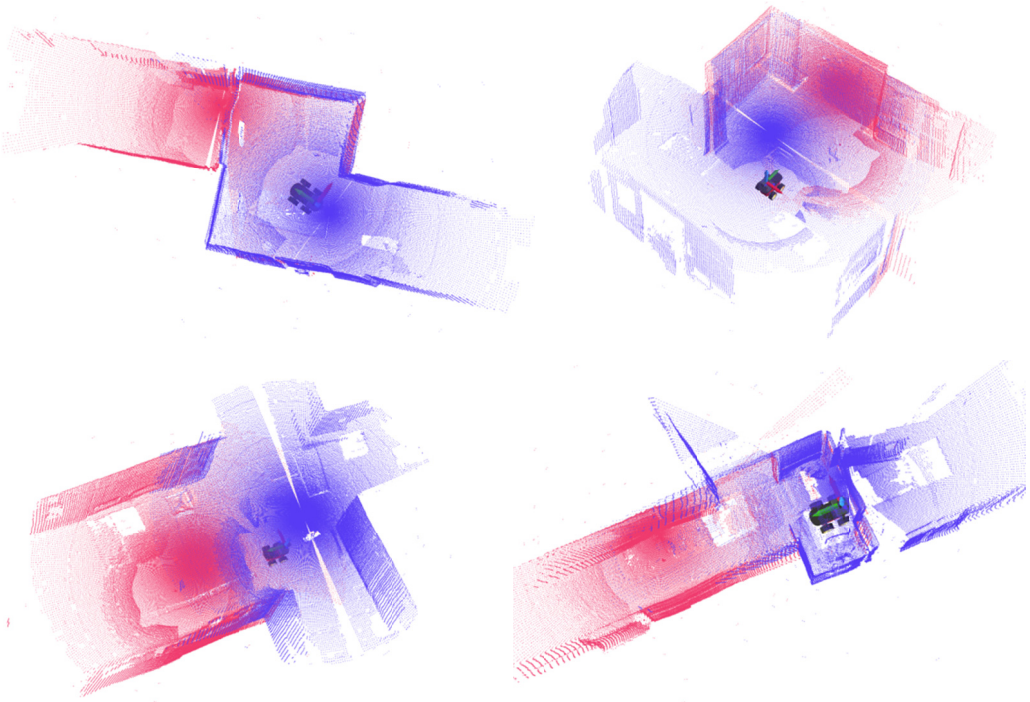


Fig. 6.5 Fused 3D point clouds after applying pose as estimated by the polygon-based 3D registration algorithm

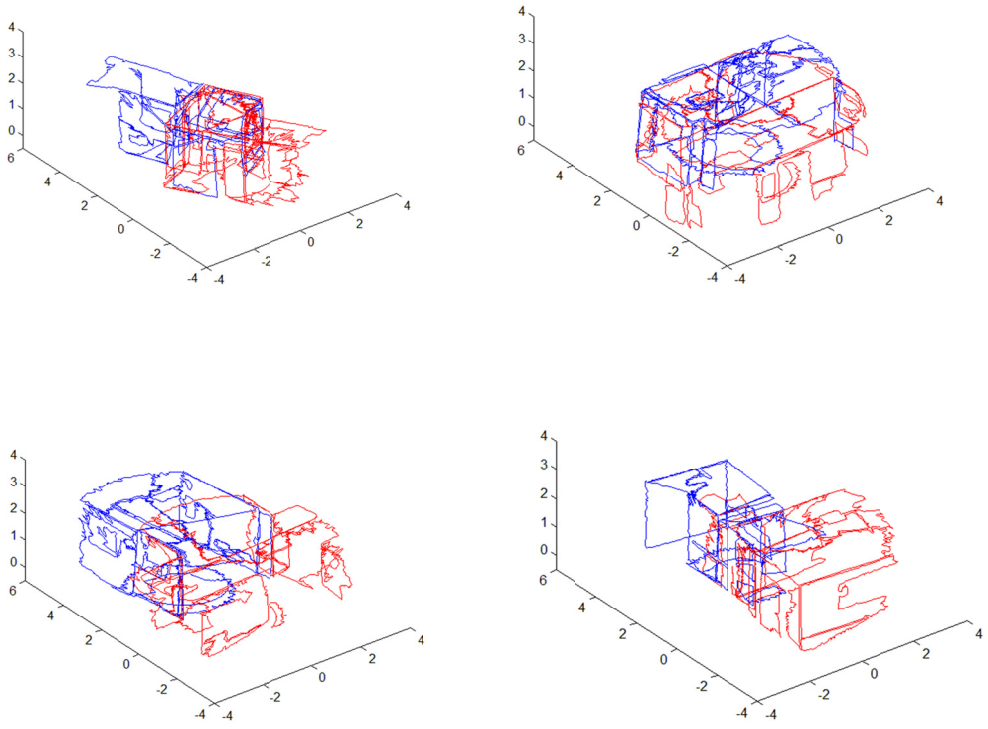


Fig. 6.6 shows overlapping polygon sets after applying the initial transformation (obtained from camera pose estimation) to the second set of polygons.

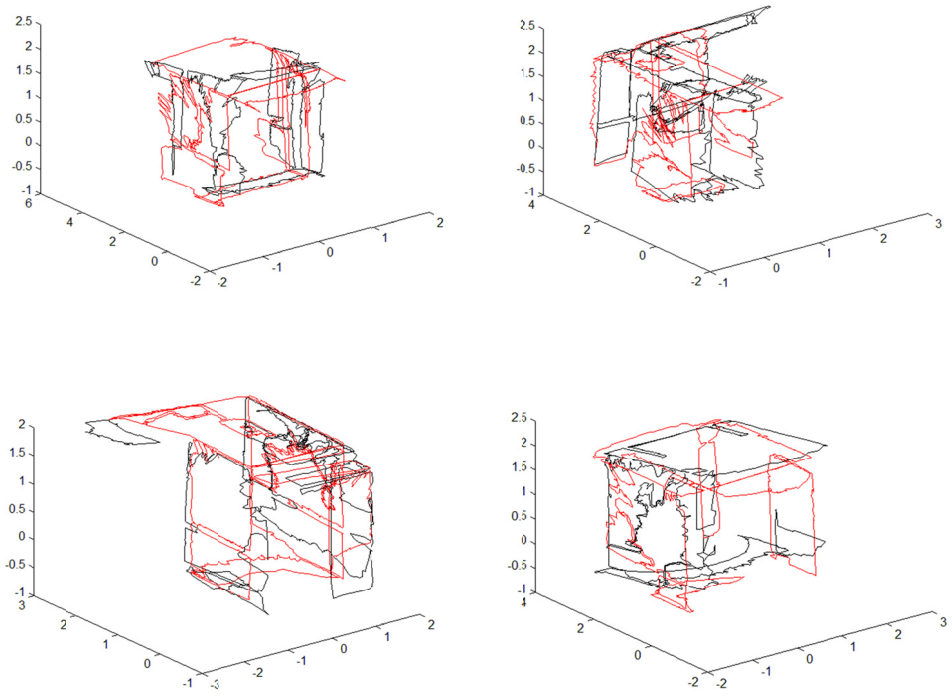


Fig. 6.7 indicates the polygons from two successive sets after establishing the correspondence.

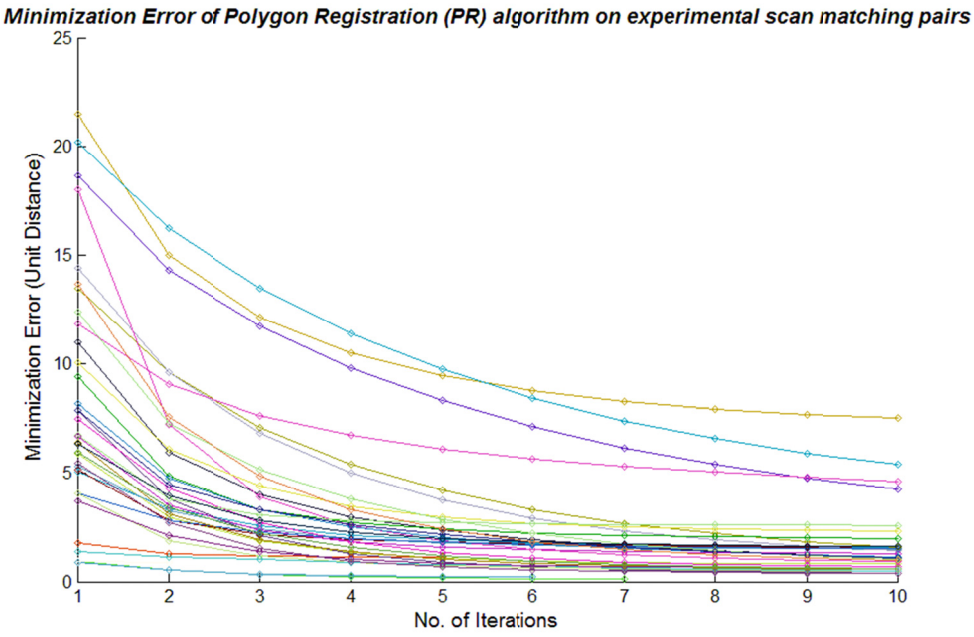


Fig. 6.8 Iterations of the PR algorithm over 37 scan matched pairs.

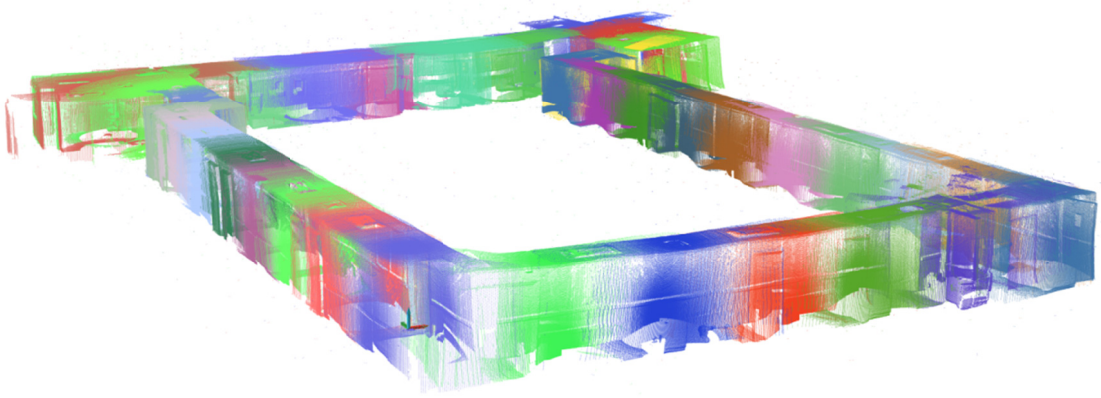


Fig. 6.9 Full exterior view of the complete 3D range image map of a 3D indoor environment acquired by two ground robots

Fig. 6.10 Partial view showing the interior regions of the complete 3D map of the indoor environment

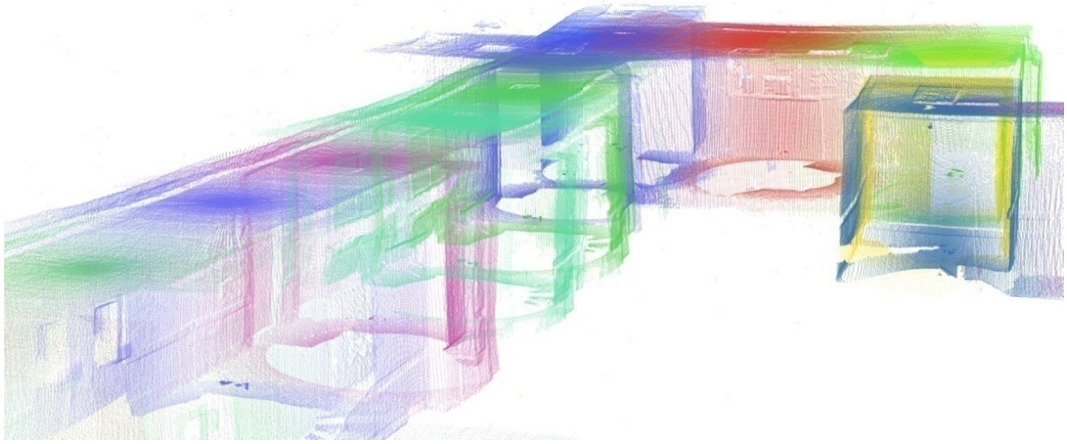


Fig. 6.11 Partial View of the exterior regions of the complete 3D map of the indoor environment

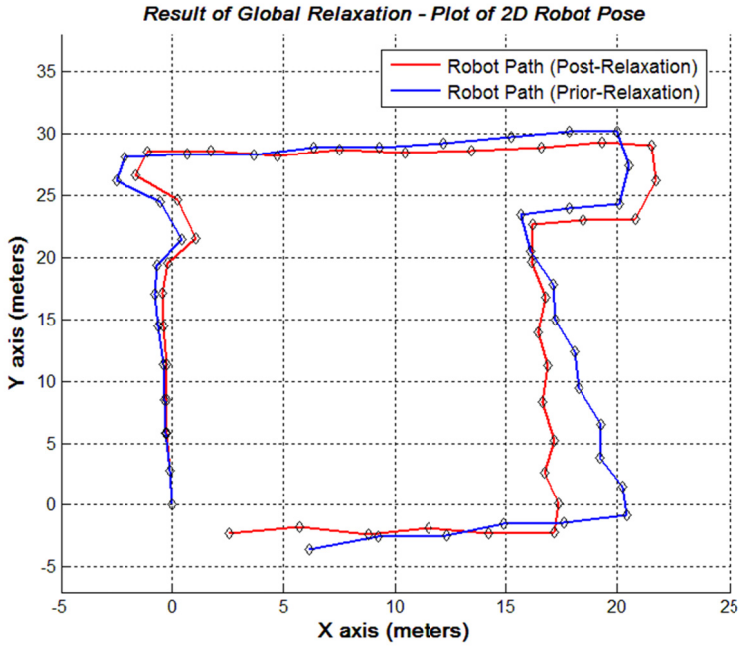


Fig. 6.12 The 2D robot poses prior to global relaxation and post- relaxation.

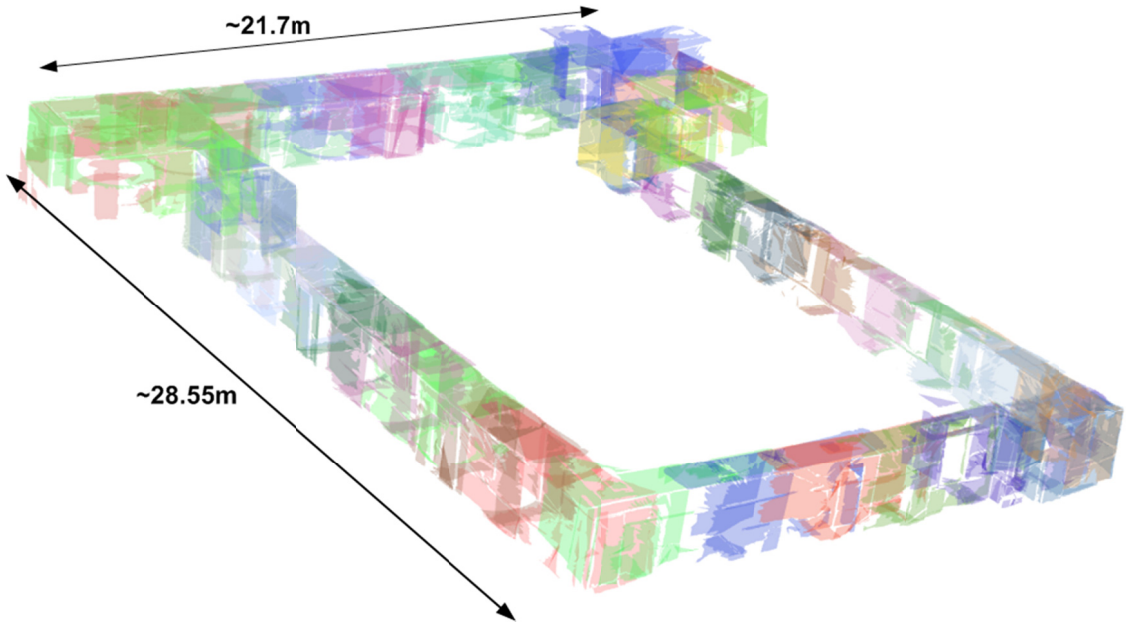


Fig. 6.13 Full exterior view of the polygon map of the corridor of the EE department at the City College of New York.

6.3.3 Comparison with the ICP algorithm

The PR algorithm differs from the ICP algorithm mainly on how the range data is perceived for establishing correspondence between two range images. However, both algorithms try to achieve LS minimization of corresponding geometric entities (points or plane) from the overlapping range images. The PR algorithm is a featured-based registration technique, which uses polygons as features. Its objective is to minimize the geometric distance between the corresponding polygons. The ICP algorithm on the other hand minimizes the geometric distance between the corresponding range data points. The first advantage of the PR algorithm over the ICP algorithm is that the correspondence between polygons can be established much more accurately in the beginning of the iteration using all the attributes of the polygon, which is described in section 6.1. The ICP algorithm establishes point-based correspondence by a close proximity measure. The accuracy of the ICP correspondence improves after several iterations, only if the algorithm converges to a minimum when the two range images are expected to be aligned. The second advantage is that the amount of range data processed by the PR algorithm is small compared to the ICP algorithm. The data compression rate of plane segmentation algorithms is usually high in largely planar environments, which is benefited by the PR algorithm. The total iteration time of both algorithms is affected by the amount of range data being processed. The ICP algorithm can minimize the range data by sub-sampling techniques, but the data is still large compared to the PR algorithm to obtain an accurate pose from the range images. We measured the performance of the two algorithms by comparing the relative pose. A position estimate of the ground truth of the robots was established. However, it was difficult to establish the ground truth of relative orientation between the robots and each scan pair was manually verified to identify a successful registration.

Table 6-3 presents a comparison of the performance of the ICP algorithm and the PR algorithm in terms of estimated relative pose from the 37 scan pairs acquired in the corridor. The scan pairs were registered using a fast variant of the ICP algorithm that makes use of the k-D trees for searching the point correspondences. The 37 scan pairs were initialized with the pose estimates that were computed by the camera pose estimation algorithm. The ICP algorithm performed well in cases when the two scan pairs had a relative translation and no rotational displacement. When the scan pairs had large rotations between them, registration process failed which could be attributed to inaccurate point correspondences at the early stage of the iteration process. On the other hand, the PC algorithm established accurate planar features and the PR algorithm successfully registered the scan pairs. A comparison of average computation time of the ICP algorithm, PC and PR algorithm for each scan pair is listed in Table 6-4. In addition, average number of polygon correspondences and the LS minimization error in registering the polygon pairs are also listed.

Table 6-3. Comparison of pose parameter estimation results of the ICP algorithm and PR algorithm

Results	ICP Algorithm (Angles)			PR Algorithm (Angles)			Relative Translation		
	α_z	β_y	γ_x	α_z	β_y	γ_x	t_x	t_y	t_z
1	-0.99	-3.74	5.25	1.10	-4.17	4.47	-0.0858	2.8250	0.0000
2	1.44	3.37	-3.18	-0.72	3.01	-2.44	-0.1043	3.0026	-0.1500
3	-1.16	-3.29	5.45	1.54	-2.79	4.97	-0.0280	2.7012	0.0000
4	-0.39	2.79	-1.60	-2.06	2.87	-1.24	0.0530	2.8229	-0.0822
5	0.84	-3.22	3.62	3.55	-2.85	3.46	-0.2139	3.1023	0.0000
6	0.00	3.53	-0.37	0.29	3.51	1.50	0.0204	2.6476	-0.0609
7	-25.09	-1.93	1.49	-22.19	-4.46	4.14	0.2524	2.4074	-0.0343
8	-17.24	0.43	-2.54	24.58	3.66	-2.52	0.2951	2.3201	-0.1299
9	-4.63	12.25	2.04	32.97	-5.57	6.23	-0.8028	3.1625	0.1000
10	-19.10	0.21	-7.02	-32.51	3.46	-2.86	-0.5830	2.6369	-0.1191
11	0.09	-4.77	1.33	-89.19	-0.93	5.02	0.4988	1.9449	0.0000
12	9.06	0.84	2.51	0.77	3.72	-0.96	-0.0776	2.8468	-0.0820
13	0.84	-4.33	1.47	0.67	-4.61	2.64	0.3802	2.9740	0.0000
14	0.45	3.83	-0.96	2.42	4.82	1.42	-0.4811	2.7715	-0.0330
15	-11.91	-6.12	2.98	-0.42	-3.82	2.42	0.3580	2.9235	-0.0386
16	2.61	4.07	-3.79	-1.42	3.82	-1.42	-0.0917	3.0382	-0.1033
17	-5.63	-3.56	4.41	1.07	-3.31	5.33	-0.2628	3.0899	-0.0112
18	-11.64	-2.32	1.54	1.53	-2.98	2.69	-0.3541	2.7846	0.0128
19	3.53	2.63	2.94	-91.50	0.55	-2.42	0.2579	2.2046	-0.0511
20	-13.54	-3.67	3.64	-16.48	-4.31	4.40	-0.1869	2.8397	0.0143
21	11.45	1.13	-1.32	-65.53	1.03	-2.50	-0.0582	3.2693	-0.1430
22	0.03	-0.16	3.89	5.52	-2.59	4.76	0.1737	2.3657	0.0438
23	-8.45	4.22	-4.08	77.53	1.18	1.65	0.1504	2.2881	-0.0321
24	-3.07	-1.65	5.44	4.53	-4.50	3.97	0.0000	2.9825	0.0519
25	2.10	4.45	-2.49	-0.50	3.22	1.05	-0.4134	2.8825	-0.0388
26	-2.52	-5.05	4.01	-1.38	-3.95	4.93	0.4729	2.8290	0.0494
27	1.19	4.51	-3.66	-1.06	2.27	-1.06	-0.3748	2.7145	-0.1099
28	-0.53	-2.59	4.64	1.29	-2.99	4.39	0.2490	2.9531	0.0312
29	2.49	5.97	-2.78	-0.46	3.51	-0.03	-0.4977	3.0755	-0.0539
30	1.85	-2.19	3.44	2.51	-3.69	3.80	0.4451	2.6953	0.0610
31	1.93	1.35	11.35	-1.90	3.13	-1.98	-0.5089	2.4923	-0.0878
32	2.50	-5.75	-1.75	-88.79	-1.46	3.28	0.1689	2.3513	0.0378
33	43.10	1.62	-0.82	-0.76	4.56	0.16	0.0000	2.9639	-0.0001
34	-1.41	-3.38	6.70	1.17	-3.22	4.34	0.3675	2.7056	0.0250
35	-2.19	5.22	-4.51	-4.15	1.90	-0.91	-0.4521	2.6879	-0.0531
36	-0.35	-4.23	4.39	2.43	-4.80	5.88	0.3502	3.0969	0.0869
37	5.51	0.51	-0.01	-3.04	3.51	-2.71	-0.4961	3.2019	-0.1003

Improper ICP Registration is noted when there is large rotational displacement between the two scans*

Outcome of scan registration of 37 scan pairs comparing rotation angles $(\alpha_z, \beta_y, \gamma_x)$ extracted by the ICP algorithm and the PR algorithm. Both algorithms were initialized with camera pose estimates. The final translation of the camera pose estimation and the PR algorithm combined is also listed (t_x, t_y, t_z).

Table 6-4. Computation time of ICP and PR algorithms, C = No. of Corresponding Planes and Minimization Error

Index	ICP Time (s)	PC Time (s)	PR Time (s)	Total Time (s)	C	Minimization Error
1	44.56	0.049	4.17	4.22	8	0.794
2	43.04	0.013	3.61	3.62	6	1.576
3	52.68	0.010	4.63	4.64	7	1.489
4	52.68	0.011	3.01	3.02	5	1.125
5	45.01	0.012	1.01	1.02	2	0.106
6	73.95	0.008	1.92	1.93	5	0.225
7	57.61	0.018	3.56	3.58	6	1.098
8	36.39	0.012	3.79	3.80	8	1.473
9	59.77	0.013	2.99	3.00	6	1.281
10	143.57	0.020	5.04	5.06	8	1.565
11	30.69	0.022	5.22	5.24	9	1.392
12	102.66	0.014	5.03	5.04	8	8.371
13	46.98	0.015	2.64	2.65	6	0.378
14	65.41	0.014	6.46	6.47	10	4.168
15	51.97	0.019	4.59	4.61	7	0.917
16	82.76	0.023	3.50	3.52	6	5.185
17	40.75	0.017	3.56	3.58	4	0.688
18	44.87	0.018	4.37	4.39	7	7.510
19	139.21	0.013	4.97	4.99	7	4.223
20	46.77	0.014	6.48	6.50	6	2.303
21	80.91	0.014	4.50	4.51	5	5.371
22	148.79	0.013	3.04	3.06	9	4.610
23	28.64	0.013	5.19	5.20	9	1.494
24	46.36	0.008	1.53	1.54	4	0.469
25	80.17	0.006	1.14	1.15	4	0.466
26	110.03	0.008	2.96	2.97	6	1.953
27	52.35	0.008	3.78	3.79	6	1.132
28	56.58	0.008	2.83	2.84	5	0.555
29	59.78	0.005	2.88	2.89	4	1.011
30	53.72	0.007	2.12	2.13	5	0.632
31	101.32	0.012	3.77	3.78	6	0.379
32	84.03	0.013	5.62	5.63	9	2.576
33	85.23	0.018	2.62	2.64	5	1.075
34	48.67	0.014	5.38	5.40	6	1.601
35	88.22	0.010	3.97	3.98	6	0.975
36	55.43	0.008	1.63	1.64	4	0.567
37	91.42	0.008	0.77	0.78	4	0.873
Avg. Time (s)	68.46	0.014	3.63	3.64		

Column (2) shows the computation time of ICP algorithm after registering point clouds sub-sampled @ 6 points. In contrast, the PR algorithm shows a sharp decrease in computation time listed in Column (5). Column (6) indicates number of corresponding polygon pairs used in the registration. Final minimization error reached by the PR algorithm between the polygon boundary points is listed in Column (7).

6.3.4 Cluttered indoor environment

The ground robots acquired the 3D range images in a highly cluttered lab environment. The initial pose estimate was given by the odometry since the relative robot pose was relatively a short distance compared to corridor map. Irregular polygons (Fig. 6.14) were extracted from the range images acquired in a highly cluttered environment consisting of few planar surfaces and largely non-planar surfaces. Large number of polygon surfaces (~25) was extracted from the two range images. The odometry provided the initial estimate between the two range images. The initial pose was applied to transform the second polygon set before establishing correspondence. The PC algorithm picked up 12 correspondences between the two polygon sets in this scenario. The PR algorithm converged to a global minimum with least squared error of 1.9073 in 8.7449 sec. The PR converged to a global minimum in 10 iterations. The fused point clouds after applying the final transformation between the two range images are shown in Fig. 6.16.

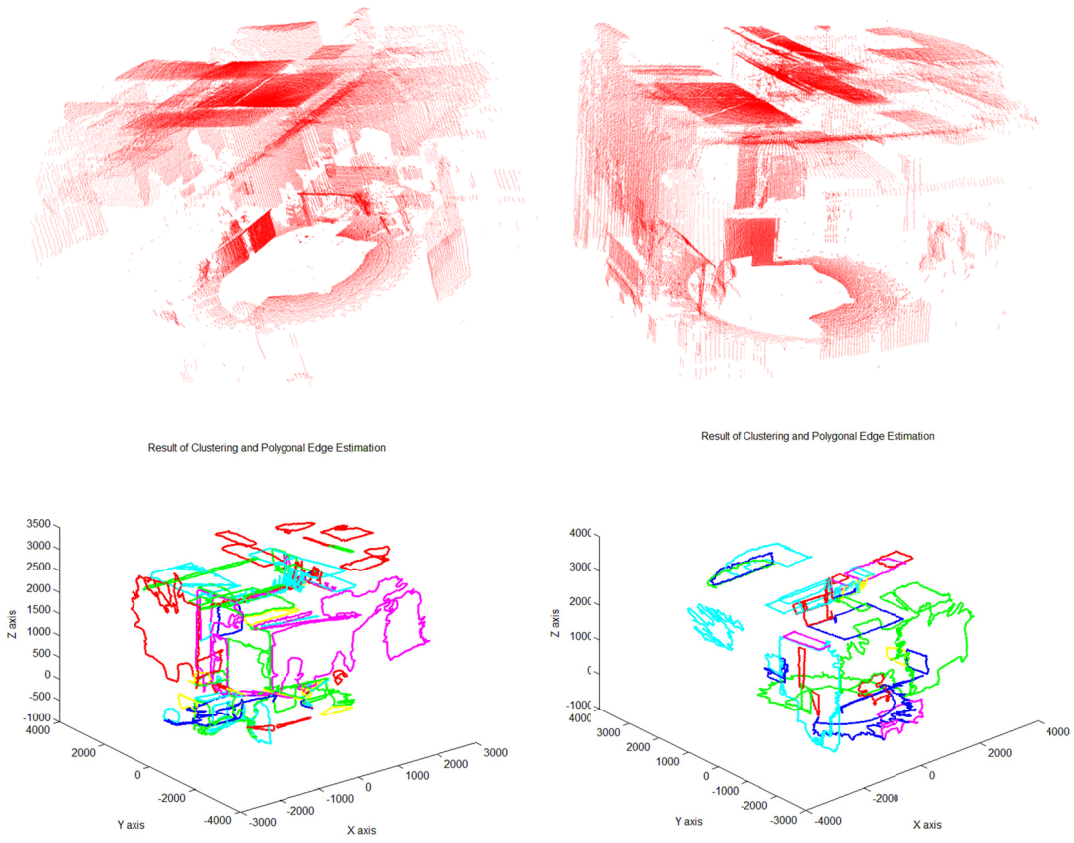


Fig. 6.14 Extracted polygons (bottom) from 3D range images (top) in cluttered environment

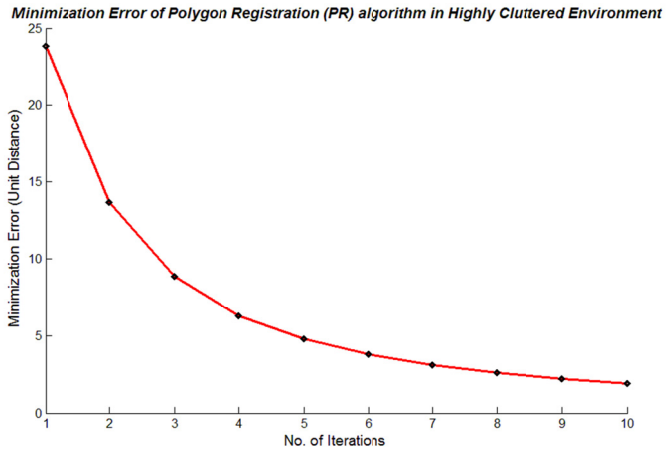


Fig. 6.15 Plot of error over 10 iterations as the two polygon sets are registered.

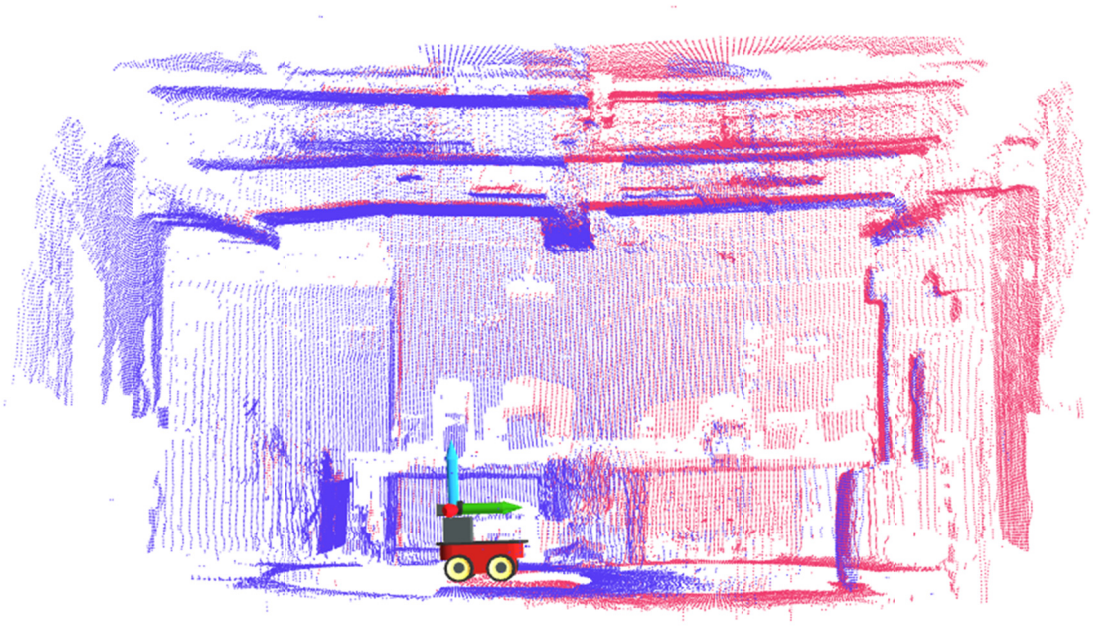


Fig. 6.16 Aligned point clouds in a high cluttered indoor environment with multitude of planes available for polygon matching.

6.4 Limitations of the PR algorithm

The polygon-based registration and mapping approach presented here is limited to a partially planar environment with at least a few corresponding planar surfaces between overlapping scans. In a more natural environment where non-planar surfaces are abundant, we will have to consider more complex surfaces and could represent them with piece-wise mathematical functions such as Splines. Piece-wise curve fitting and registration between higher order polynomial surfaces would be worth exploring as a research topic in 3D mapping.

Both polygon correspondence and registration algorithms rely on a good initial estimate between the overlapping scans for better performance, which is the same requirement for all current scan matching approaches. Specifically, the limitation stems

from the assumption that corresponding polygons are computed based on their geometric attributes. Hence, some corresponding polygons are not picked for the registration process. The algorithm benefits from minimal data input to compute the pose estimate and extremely fast. Polygon extraction and clustering is a challenging problem prior to creating polygon-based maps and depends on fine-tuning parameters that relate to the geometric attributes of a plane. These parameters have to be manually adjusted to obtain accurate polygon maps. We have addressed these issues while performing experiments and the algorithms still proved to be beneficial both in terms of speed of execution and accuracy.

In this chapter, we introduced a framework for obtaining a complete 3D map of a structured indoor environment using dual robots. This framework is a viable approach to mapping indoor environment, which is devoid of visual features and has slippery surfaces where odometry is unreliable. The robots compute an initial estimate of the relative pose between their respective range scanners using a camera pose estimation algorithm. Further, the relative pose is refined to a higher accuracy using a fast and robust polygon registration algorithm. As a first step, the 3D range scans are processed to extract polygons acquired in a structured environment. The second step involves polygon registration using nonlinear parameter estimation. We tested the algorithm using dual-robot systems equipped with multimodal sensors.

CHAPTER 7

3D Mapping with RGB-D sensors

In this chapter, we estimate the pose of robots in real-time equipped with an advanced sensor known as the RGB-D camera, where 'RGB' stands for Red-Blue-Green indicating that the camera is capable of acquiring color information in its field of view of the surrounding environment and 'D' stands for depth or range measurements. The RGB-D camera outputs large quantity of data at 30Hz, which is pre-processed to remove outliers. We used Kinect® RGB-D camera (Fig. 7.1) designed by Microsoft Corp. for data acquisition. A one-to-one pixel correspondence between the color and range data is pre-calibrated and is available as part of the factory settings. Our aim was to estimate the pose between successive frames of the RGB-D camera as robot carrying it transits through indoor environments.

Fig. 7.1 Kinect® RGB-D sensor

Kinect® sensor is capable of acquiring a maximum of 640 × 480 pixel color image and their corresponding depth values (1.2m – 3.5m) @ 30 frames per second. It has a

horizontal field of view of 57° and a vertical field of view of 43°. An RGB-D scan in the current context is defined as a pair of images of equal size: a 2D RGB image I_{RGB} and a 2D depth image I_D. The images are pre-registered, meaning that a pixel $[u, v]$ in the RGB image and the corresponding pixel in the depth image both refer to the same point $p = [p_x, p_y, p_z]^T$, measured in camera coordinates. The depth of a point (value of the pixel in the depth image) is equal to p_z.

Given the depth image and the camera parameters, we can calculate the 3D projection of each pixel to obtain a 3D point cloud. We assume that the correct color-to-depth image registration and 3D projection are handled by the device driver, and are outside the scope of this work. We note that we can treat an RGB-D scan in two different ways: as a 3D point cloud, where each point has additional color information, or as two 2D images. We can go back and forth between the two representations at any time. This redundancy is useful because we can exploit the structure of the data based on our needs. For example, when determining the nearest neighbor to a point in the 3D cloud, a reasonable (and fast) assumption is to look through its 8 neighbors in the 2D image. It also enables us to leverage algorithms from both image processing libraries such as OpenCV [Bradski 2000] and geometric processing libraries such as Point Cloud Library (PCL) in a ROS framework [Quigley, Gerkey et al. 2009]. Given two successive point clouds of the same static scene observed from different viewpoints, one can find the change in pose of the camera by obtaining the rigid transformation that best maps one point cloud onto the other. A point-based registration technique can be used to determine the pose from two point clouds such as the Iterative Closest Point (ICP) algorithm [Besl and McKay 1992]. Since a typical RGB-D scan at VGA resolution has hundreds of thousands of points, performing scan registration of a pair of overlapping 3D point clouds is computationally expensive, and does not provide a real-time solution. Thus the first step of the pose estimation system is to extract features from the RGB-D scan so that the overlapping scan pairs can be registered on a subset of the 3D data.

Since we execute the registration algorithm in Euclidean space, the ideal features would be pose-invariant between the scans. We describe three feature detection algorithms: a brightness edge detector which operates on the RGB image, a depth edge detector which operates on the depth image, and an edge detector which operates on the normal vectors extracted from the 3D point cloud. The second step of the system is to align the sparse data against a model dataset using the ICP algorithm. The model consists of features from previous RGB-D scans.

The state-of-the-art RGB-D sensors instantaneously acquire both color images (RGB) and depth of the surrounding environment. The most popular in this category are the Kinect® sensor from Microsoft and SwissRanger® SR-3000 3D camera. Kinect camera is much cheaper than its counterpart and designed for gaming applications. SR-3000 is an industrial grade 3D camera and designed for research applications. The Kinect camera acquires range and color images at VGA resolution and the depth accuracy falls as the range distance measured. SwissRanger cameras have good depth accuracy at its full range of 12-15 meters but with a smaller field-of-view, although the color images acquired are not truly RGB data. Most recent efforts to build dense point cloud maps that integrate both range and color data in the map include [Henry, Krainin et al. 2010; Rusu and Cousins 2011]. The 6D pose of the robot is estimated while constructing the map. We approach the 6D pose estimation in real-time by registration of successive frames from RGB-D camera. The RGB-D camera is data intensive and would be difficult to achieve real-time registration using raw point clouds. We extract edge features from the depth and color images and process edge features to fuse successive frames. Edge extraction from the data points is a vital step and minimizes the number of data points to be processed during 3D scan registration. Most recent edge extraction techniques from range images include [Coleman, Scotney et al. 2010; Rusu, Blodow et al. 2011; Steder, Rusu et al. 2011; Ye and Hegde 2009]. [Tomono 2009] extract edge features from stereo images to achieve 3D registration in indoor environment in real-time. A

major disadvantage of using just perspective cameras for registration is that the accuracy reduces with bad lighting conditions. Our approach to pose estimation is adaptive and relies on either brightness or depth information or both and reliably extracts edge features. Hence, if either one of feature extraction techniques fail, the registration algorithm will still be able to accurately fuse successive frames.

7.1 Brightness feature extraction

We note that while the RGB image is completely filled, parts of the depth image is incomplete with holes. This will occur any time the sensor is not able to determine the distance to the given point. In this context, we will refer to such gaps in the depth image as shadow pixels. The first source of shadow pixels is due to the fact that the optical axis of the infrared imager is not co-linear with optical axis of the RGB image sensor. Since the depth image is registered to the RGB image, there are points in the RGB image that are occluded from the point of view of the depth sensor. Another source of shadow pixels is when the depth camera is observing a highly reflective surface. Finally, we discard any ranges greater than 4 meters due to their poor accuracy and replace them with shadow pixels. We use the NaN (Not-a-Number) notation for shadow pixels in the depth data:

$$I_D(u, v) = NaN \tag{7.1}$$

The brightness feature extraction technique operates on the RGB image data to generate a set of features with minimal CPU utilization. The RGB image provided by the sensor is first converted into an intensity image I_I.

$$I_I = \frac{1}{3}(I_{RGB}[R] + I_{RGB}[G] + I_{RGB}[B]) \tag{7.2}$$

The Laplacian of the intensity image I_{LI} is computed, which locates the edges of intensity regions [Gonzalez and Woods 2008].

$$I_{LI} = \nabla^2 I_I = \frac{\partial^2 I_I}{\partial u^2} + \frac{\partial^2 I_I}{\partial v^2} \quad 7.3$$

An intensity threshold t_I is then applied to the output of the Laplacian to produce a binary intensity image I_{BI}.

$$I_{BI}(u,v) = \begin{cases} 1 \; if \; I_{LI}(u,v) \geq t_I \\ 0 \; if \; I_{LI}(u,v) < t_I \end{cases} \quad 7.4$$

The basic morphological operators are described in A.3. A 'close' morphological operator with a structuring element S is applied to the binary image to connect the features. An "erode" is then performed on the closed image I_{CL} with the same structuring element to refine the features and remove singular points. The final set of candidate brightness features is represented by the binary image I_{CB}.

$$I_{CL} = (I_{BI} \oplus S) \ominus S \quad 7.5$$

$$I_{CB} = I_{CL} \ominus S \quad 7.6$$

A set of brightness features F_B are extracted, which contain the foreground pixels in the candidate brightness image. Before adding a feature $q_{(u,v)}$ to F_B, we check that the corresponding pixel in the depth image is not a shadow. The resulting outcome of a brightness extraction algorithm on a RGB-D image acquired in an indoor environment can be seen in Fig. 7.2.

Fig. 7.2 Outcome of brightness edge feature extraction algorithm on a single RGB-D frame acquired by the Kinect® sensor in an indoor environment.

7.2 Depth feature extraction

Depth features are extracted directly from the depth image I_D, and occur at the object boundaries, from the point of view of the depth camera. Let us call areas in the depth image with low values as **foreground** (they are close to the camera), and areas with higher values as **background** (they are far away from the camera). As we noted earlier, there are also shadow areas, containing NaN values. There are two types of transitions that interest us: transitions from foreground to background and from foreground to shadow. The former always corresponds to a real-world object boundary. However, the latter can either be an object boundary or a false positive. A simple solution to the problem is to ignore foreground to-shadow transitions. However, this will fail to detect a large portion of the good edges in the scene, since a thin line of shadow pixels often appears right between the transition from foreground to background. Here, we propose a smoothing filter which removes small areas of shadow from the image.

The filter is based on a modified dilate morphological operation. We define the depth dilation function f on a pixel $I_D(u, v)$. For a given pixel (u, v) and its neighbors N,

the function returns the maximum value of all the non-shadow neighbors. The neighbors N are specified by a structuring element S. If all the neighbors are shadows, the function returns NaN.

$$f(I_D(u,v),N) = \max_{u',v' \in N} (I_D(u',v') \neq NaN) \quad 7.7$$

Further, the above function is applied to all shadow elements in the depth image I_D, to obtain a filtered depth image I_{FD}.

$$I_{FD}(u,v) = \begin{cases} I_D(u,v) & if\ \ I_D(u,v) \neq NaN \\ f(I_D(u,v),N) & if\ \ I_D(u,v) = NaN \end{cases} \quad 7.8$$

The result is that the shadow pixels which have at least one non-shadow neighbor are transformed into background pixels. Next, the Laplacian of the filtered depth image is computed to obtain I_{LD}, which locates the depth discontinuities.

$$I_{LD} = \nabla^2 I_{FD} = \frac{\partial^2 I_{FD}}{\partial u^2} + \frac{\partial^2 I_{FD}}{\partial v^2} \quad 7.9$$

Finally, we build a set of depth features F_D, containing all pixels $q_{(u,v)}$ in I_{LD} higher than a certain depth threshold t_D.

$$F_B = \bigcup_{u,v} q_{(u,v)} \ : \ (I_{LD}(u,v) \geq t_D \quad 7.10$$

The result of the depth edge extraction algorithm on a single RGB-D scan can be seen in Fig. 7.3.

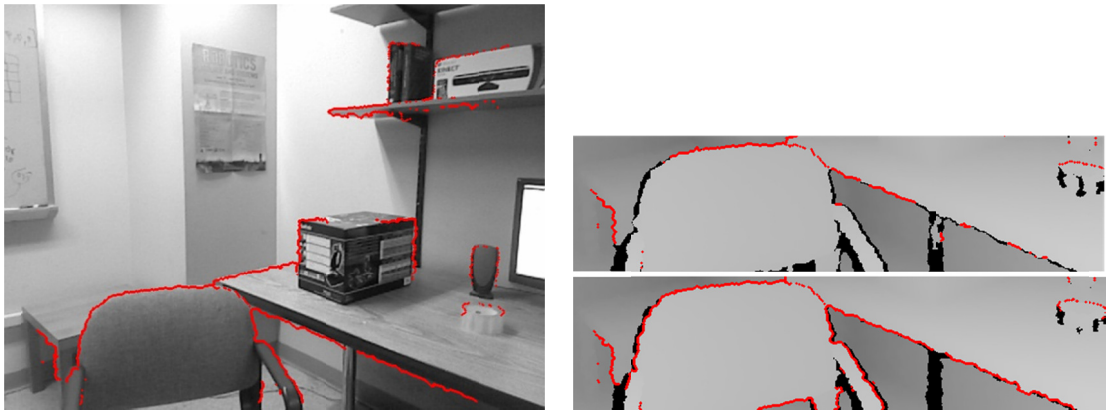


Fig. 7.3 (Left) Result of a depth feature extraction algorithm. (Right-Top) Result of depth edge detector without shadow smoothing. (Right-Bottom) Result of depth edge detector with shadow smoothing. Depth features are highlighted in red. Dark areas correspond to objects farther from the camera. Black areas correspond to shadow pixels. The morphological structuring element is of size 5×5.

7.3 Normal vector-based edge feature extraction

The Kinect sensor and the 3D range scanners output range information as 3D coordinate points (spherical) and stored in the form of a 2D array similar to bitmap (raster) images. Current generation range sensors provide an effective way to access the neighboring points of each coordinate point, which is embedded in the indices of the 2D array. The 2D array structure eliminates the need for a complex data structure such as the kD-tree for searching nearest neighbors. This is useful in extracting important attributes such as the normal vector and curvature of the local region. There is a distinct advantage in extracting and using the normal vectors for edge extraction. The normal vectors are scale invariant and form a good description of direction of the local area patch around a given point. The normal vectors of range points undergo a directional transition between two surfaces. The aim of this section is to identify those edge points with varying directions that lie on the boundary of different regions containing distinct set of normal vectors. The normal vectors computed from noisy range sensors need to be

processed prior to edge extraction of edge points. Especially, the Kinect RGB-Depth sensor has varying degree of noise that are inconsistent over the distance measured and the noise removal technique needs to be adaptive in nature. First, we discuss the normal extraction from local patches using the Principal Component Analysis (PCA) of points present in each patch. Second, we provide two approaches to edge extraction using the normal vectors. The first approach computes the scalar dot-product of neighboring normal vectors and identifies edge features in the range image by thresholding of scalar output. The second approach aims to eliminate noise by morphology based on local statistical information of the normal vectors and computes the edge features.

7.3.1 Normal Vector Extraction by PCA

The normal vector at a given point is computed by analyzing a set of points within a known radius r (measured in pixels) and includes all points within a small window of size $(k \times k)$, where $k = (2.r - 1)$. We assume that the given point and its neighboring points all lie on a plane. Any neighboring point that is located at a 3D distance greater than a certain threshold from a given point is excluded from normal vector computation. The PCA technique is the most popular technique for computing the normal vectors, given a set of points $X_i, i = 1 \ldots k^2$. Normal vectors are computed as one of the solutions to the objective function, which minimizes the sum of least squared orthogonal distance between the points and the fitted plane to the points.

$$E(A, \hat{n}) = \min \quad \sum_{i=1}^{k^2} \left(\hat{n}(X_i - \bar{X})\right)^2 \qquad 7.11$$

where $\bar{X}$ is any point on the plane. Initially, $\bar{X}$ is chosen as the mean of all points given by $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i, i = 1..n$. The Hessian plane parameters for a given surface patch is a tuple $(\bar{X}, \hat{n})$.

7.3.2 Normal-based edge extraction via dot-product

Edge features extracted from normal vectors are a perfect complement to the edge features extracted from 2D depth and color images. Especially, the data points detected by the RGB-D sensors from features such as corners and wall edges do not vary in color or brightness and the data points are close to each other. The only distinguishing factor between two or more surfaces is the orientation of surface normal local to each of the data points. Hence, a feasible approach is to analyze the geometric attribute and identify the edge points based on orientation of local points. The difficulty lies in dealing with normal vectors that are affected by non-Gaussian noise. The RGB-D depth camera is fairly noisy while measuring the range and the noise increases with the range distance measured. This section deals with a simple and effective way to extract edge points based on comparing the orientation of local points. This is viable when the range distances considered are within a limited range and not susceptible to noise. A more robust algorithm is presented in the next section to extract edge points based on analysis of normal vectors extracted from a 3D range image that effectively deals with noise. The dot-product method for edge detection has two steps: generating the normal vectors, and detecting edges by normal vector comparison. First, the data points (640×480) from Kinect® sensor are down-sampled to either of the two sizes (QVGA - 320×240) and (QQVGA - 160×120). Note that this is the only image down-sampling that the system performs on the data. At each down-sampled data point, the normal vectors are computed using PCA and stored in a 2D grid. A window size is initially chosen for analyzing the normal vectors of the neighboring points. The dot product of the normal vectors of a given point and each of its neighbors within the window size is computed. For experiments, we chose a (3×3) window and computed a total of 8 normal vectors unless the neighboring data point was out of range, given a distance threshold (d).

$$E(\cdot) = \frac{1}{k^2}\sum_{i=1}^{k^2} \hat{n} \cdot \hat{n}_i \mid \forall i, \ \ \|X - X_i\| < d \qquad 7.12$$

The average of scalar dot product $E(\cdot)$ was computed and classified into an edge point or not based on a threshold set manually. If the dot product of the neighboring normal vectors exceeds a fixed threshold t_N, the index of the normal vectors is added to a feature set F_N. This method works extremely well for less noisy data. There is still a need for a robust technique to handle noisy data points. The appendix A.3 lists a technique to remove noise by the statistical approach and edge extraction by morphology.

7.3.3 Edge extraction by mode-histogram morphology

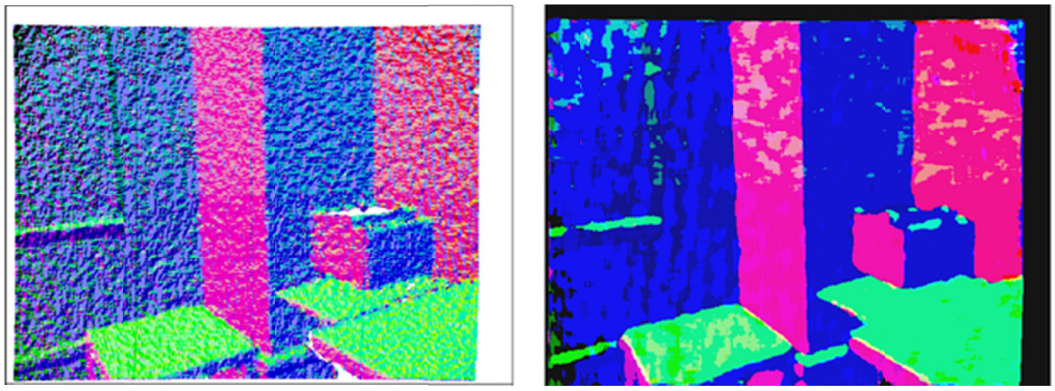
The data points from the range image have non-Gaussian noise and numerous statistical outliers. One way to analyze and remove noisy outliers is to analyze the local area information. After the normal vectors are computed from the data points, the noise is propagated into normal vectors. It is easy to create a specific number of bins for normal vectors since it is scale invariant. For range data, the distances could vary to any length and hence it will not be possible to create specific number of bins. We provide a statistical method to analyze the normal vectors and eliminate noise.

Local Mode Analysis: At a given location in the 2D grid of normal vectors, we locally analyze the mode of the normal vectors in a given patch size. The mode of a local patch is an indication of certain normal vector occurs more than others and pointing in a specific direction. On a given edge point, the mode changes from one direction indicated by a set of normal vectors to a different one and still retain the sharp edge features. The outliers are removed since the frequency of outliers is always less than the set of points in the patch points to a specific direction. The histogram is constructed by constructing 1D bins for each component of the normal vector ranging from $(-1, 1)$. The normal vector components are placed in specific bins and the mid value of the

normal vector with the highest mode is picked to be replaced by the current node in the 2D normal vector grid. A suitable number of bins are chosen such that each bin is filled appropriately and the mode turns out be accurate. Number of elements that are to be filled in the histogram is $n = k^2$. Let the quantization level in the bin be (l). Number of bins in the histogram is given by $M = n/l$. The mode of the histogram is given by,

$$bin[1 \dots M] = 0$$
$$bin[j] = bin[j] + \begin{cases} 1, & jl \leq n \in N < (j+1)l \\ 0, & otherwise \end{cases} \quad \forall j \in M \qquad 7.13$$
$$mode(b) = \max_j [bin(j)] \quad \forall j \in M$$

An image plot of normal vectors extracted using PCA, quantized in RGB space is shown in Fig. 7.4(a). The resulting outcome of transformation after applying mode-histogram morphology on the 2D image that represents the noisy normal vector space is shown in Fig. 7.4(b).



(a) (b)

Fig. 7.4 (a) 2D raster image shows noisy normal vectors extracted from a snapshot of Kinect® sensor data. Normal vectors are quantized to display as an RGB image. (b) Normal vectors after a mode histogram morphology transformation. Most statistical outliers eliminated while retaining the edge properties of corners and wall edges.

Fig. 7.5 Edge features extracted using mode-histogram morphology on local normal vectors computed from a RGB-D frame. The red markings are overlaid on RGB frame acquired by the RGB-D Kinect® sensor.

7.4 6D pose estimation

The feature sets F_I, F_D, F_N are combined together, and any duplicate features detected by more than one feature detector are removed. We create a sparse point cloud C^* from the input cloud C, where C^* contains the 3D edge points as detected by the three feature detectors (based on image attributes namely intensity, depth and normal vector). The cloud C^* is then aligned against a point cloud model using ICP. If the alignment is successful, the aligned cloud C^* is added to the model itself.

The new features, if correctly aligned, will to a large degree overlap with the old features, resulting in large redundancy in the data. To handle large number of RGB-D frames in real time, we down-sample the range image with a voxel grid filter, which is available as part of the Point Cloud Library (PCL) [Rusu and Cousins 2011]. The voxel grid filter eliminates more than one point that is part of the same grid. Even with voxel down-sampling, the model grows linearly as new areas are explored. To ensure that the size of the model remains constant, we periodically apply an aggressive statistical outlier removal filter [Rusu and Cousins 2011]. The filter removes points which are statistically far away from their nearest neighbors. Repetitive application of this filter results in eroding the data, so that features have only a limited lifespan in the model before they are discarded. The ICP algorithm was employed to determine the 6DoF pose from the two successive RGB-D frames, which minimizes the Euclidean distance between the corresponding edge feature set determined earlier. We performed this experiment by running the pose estimation system on a desktop Dual Quad Core Intel Xeon 2.0GHz CPU, using a Kinect camera operating at 30fps in VGA resolution mode. The achieved update rate was 10Hz, with a mean update time of 97 ms and a standard deviation of 10 ms. The depth and brightness feature extractor took approximately 10-15 ms per iteration each, and the dot-product normal edge detector took 25-35 ms. Fig. 7.6 shows the full 3D map obtained by rotating the servo 307 (its full range). Maintaining and

visualizing a full 3D scene decreases the update frequency of the pose estimation; nevertheless, we still achieve map building in real-time.

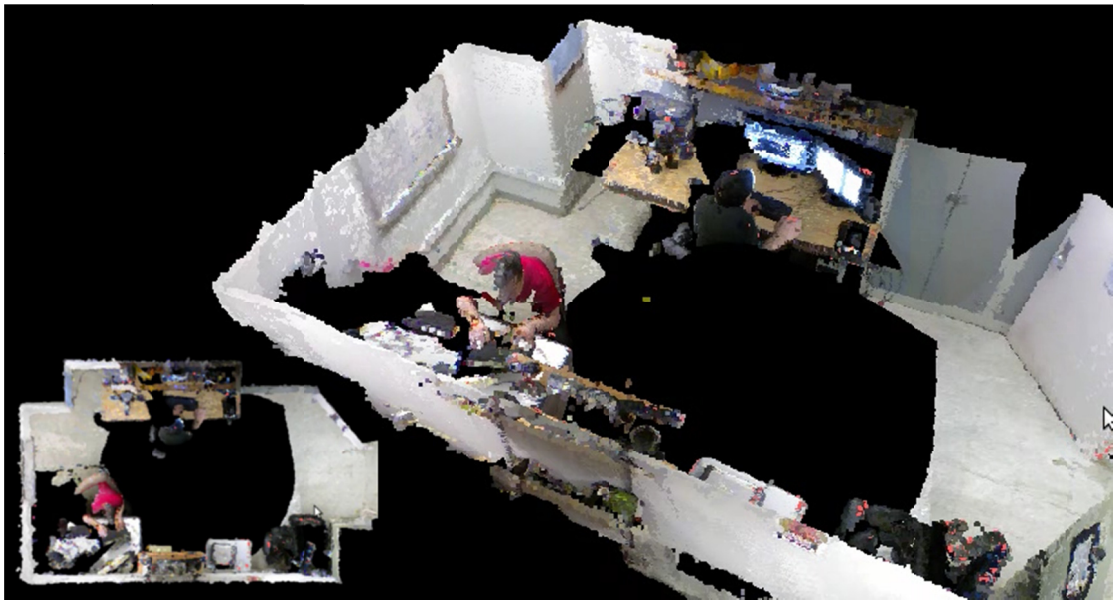


Fig. 7.6 3D map of an office built by rotating the RGB-D camera - side view and orthogonal top view.

7.5 Limitations of 6D pose estimation

The features extracted from the RGB-D frames are brightness features, depth features (border points) from range information and the local normal vectors. The correspondences between the features in successive overlapping frames are established by finding the closest points and test the proximity to each other using the Euclidean metric. If the feature correspondences turn out to be inaccurate, then there are possibilities that the ICP algorithm can diverge from the global minimum during the iterative LS minimization process.

CHAPTER 8

Conclusion and future work

An important research area in robotics community is to build quality 3D-depth maps of indoor environments by robots that function autonomously. In addition to ground mobile robots, there is a new generation of robots including wall-climbing robots, Quadrotor autonomous aerial robots, indoor and outdoor UAVs, which needs a map to navigate through the environment. Such robots need complete 3D spatial information to navigate through the environment. Our approach directly addressed this problem by constructing 3D depth maps with multiple heterogeneous robots equipped with multi-modal sensors such as 3D range scanner and perspective camera. We developed a number of algorithms that led to building 3D maps in indoor environment. The most challenging problem in map building is to efficiently register two partially overlapping range images acquired by the robots.

8.1 Conclusion

We conclude by listing major contributions, which are organized into several chapters:

In chapter 2, we discussed several camera-based pose estimation algorithms that are useful in determining the pose between multiple robots. We surveyed many camera pose estimation algorithms that can provide a good initial estimate of pose between any two robots. A closed-form solution to the P3P problem is useful in determining the camera pose in real-time, however it suffers from the multi-solution phenomenon. We developed an algorithm to identify the valid solution of pose from the P3P algorithm. We compared our algorithm with state-of-the-art iterative camera pose estimation

algorithms and evaluated their performance in terms of execution speed and pose accuracy. The P3P-Particle Filter algorithm ran fast in terms of speed of execution as it is a closed-form solution compared to the iterative algorithms without much loss of accuracy.

In chapter 3, we discussed in detail, a novel particle filter-based algorithm that identifies the valid solution from a set of solutions determined by the Grunert's method to the P3P algorithm. We also discuss the application of the P3P algorithm along with particle filter algorithm to determine the pose between multiple heterogeneous robots. This pose is used as an initial estimate to fuse overlapping range images acquired by the multiple robots. We presented experimental results to fuse the range images acquired by multiple heterogeneous robots when they are static. We proved experimentally that it is possible to initialize scan registration algorithms with camera pose estimation algorithms in a multi-robot scenario. We further extended the idea of vision-aided scan registration with dynamic multi-robot systems to construct complete 3D maps of indoor environments.

We discussed in chapter 4, the outline of our methodology to register overlapping range images acquired by multiple robots. In addition, we presented the literature review of current state-of-the-art registration algorithms. We also introduced the scenario and system architecture of the multi-robot setup used to construct 3D range maps in indoor environments.

In chapter 5, we discuss a novel patch-based plane clustering algorithm to segment the range images into polygons. The range images were extracted from indoor environments. We compared this algorithm with two state-of-the-art segmentation algorithms namely, region-growing segmentation algorithm and the RANSAC segmentation algorithm. After performance evaluation, experimental results indicated that our algorithm outperformed in terms of speed of execution without loss of accuracy in segmentation. We used this algorithm to register overlapping range images

with a polygon-based scan registration algorithm discussed in Chapter 6. We also presented experimental results of 3D map construction with a dual-ground robot setup (dynamic case) in indoor environments. The polygon-based scan registration algorithm executed in real-time and always converged to a global minimum with a good initial estimate from vision. The polygon-based maps are convenient for both map representation and visualization in structured indoor environments. This aspect was noticeable in the java visualization toolkit, which took only ~13MB of heap space to visualize the fused 3D polygon map as opposed to ~490MB of heap space to visualize fused 3D range maps.

In chapter 7, we presented experimental results to fuse multiple RGB-D frames acquired by the Kinect® sensor while estimating the pose of the sensor in motion. Kinect® sensors are extremely noisy and needed a superior noise removal algorithm. We have presented a mode-histogram morphology algorithm to remove noise from normal vectors extracted locally from range data. This normal vector information along with brightness and depth were used to extract borders and edges of objects from range and image data acquired by the Kinect sesnor. The edge points were fed into a registration algorithm to compute 6D pose between successive RGB-D frames. We successfully removed inherent noise in Kinect range data and showed that it is possible to estimate 6D pose to fuse successive RGB-D frames in real-time.

8.2 Future work

- Feature extraction is an early step in the map construction. It is a useful step for both map representation and registration of overlapping range images. Robust features can be extracted from the range images using normal vectors along with other attributes that improve the efficiency of the data correspondence problem. A hybrid approach to use several features from both range and image data can be sought in a highly symmetric environment devoid of visual features to improve registration.

- Efficient closed-form solutions to registration can be explored to fuse partially overlapping range images. Iterative algorithms can be further improved by modifying the cost function, which is to minimize the least squared distance between the corresponding features. Other metrics can replace the Euclidean distance metric often used in registration.
- Registration of overlapping range images with noisy sensors lead to error accumulation over long distances. Global relaxation is a technique that computes this error and distributes the error among fused point clouds. This operation can be performed online or offline and is dependent on features being processed. There is large room for improvement in global relaxation techniques involving 3D range maps in terms of robustness and efficiency.
- There are numerous commercial sensors in the market today that output both range and image information such as Kinect® RGB-D camera. They are cheap and accurate to short distances. The registration algorithms need to take noise into consideration while estimating the pose from RGB-D frames. One can explore probabilistic algorithms such as EM, extended Kalman filter and particle filter algorithms to estimate 6D pose in real-time. The biggest challenge will be how to eliminate redundancy and effective use of feature sets extracted from the sensor. A pragmatic approach would be to represent a likelihood estimate of robot pose to account for noise in the sensors.
- It would be useful to study path-planning and determining the next best view point for each robot as it navigates through indoor and outdoor environments.
- Indoor environments differ from outdoor environments where the objects are less structured. The map construction algorithms need to take into consideration that the features extracted in each case may not work for a different scenario. Hence, considerable effort has to be spent in designing an algorithm that is robust to changing environments.

Appendix

A.1 Relative Translation

This section provides the steps to compute the relative position (x, y, z) of the camera center with respect to the ground robot frame.

Input to the algorithm:

- World points $P_i = [x_i, y_i, z_i, i = 1,2,3]$ with respect to $(X_1 Y_1 Z_1)$ coordinate frame. (Refer to Fig. 2.1)
- Magnitude of the vectors $\|OP_1\| = s_1$, $\|OP_2\| = s_2$, $\|OP_3\| = s_3$

From definition of vectors, we obtain

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = s_i^2, \qquad i = \{1,2,3\} \tag{A.1}$$

Eqn. 12 can be expressed as

$$x^2 + y^2 + z^2 + A_i x + B_i y + C_i z = D_i, \qquad i = \{1,2,3\} \tag{A.2}$$

A_i, B_i, C_i and D_i for $i = \{1,2,3\}$ are known coefficients.

A.2 can be further expressed as linear equations. With three linear equations and three unknowns, we can solve for $P = (x, y, z)$.

A.2 Relative Orientation

In this subsection, we compute the rotation matrix between the camera coordinate frame and the ground robot frame given by $R_{3\times3}$ matrix.

Input parameters:

the origin of the camera $P = (x, y, z)$ with respect to ground robot frame from previous sub-section

3D points $P_i = [x_i, y_i, z_i, i = 1,2,3]$ with respect to ground robot frame.

2D Image points $q_i = [u_i, v_i]^T, i = \{1,2,3\}$
focal length $(\boldsymbol{f})$ of the camera
Internal Camera Parameters
scale factors (s_x, s_y)
2D Principal point (O_x, O_y)
Distance between the control points (s_1, s_2, s_3).

Note:

$$s_i = d_i + d_i', \qquad i = \{1,2,3\} \tag{A.3}$$

Output: $R_{3\times3}$ matrix

By vector definition,

$$\overrightarrow{OP_i} = \vec{P_i} - \vec{O}, \qquad i = \{1,2,3\} \tag{A.4}$$

Unit vectors are given by,

$$\widehat{OP_i} = \frac{\overrightarrow{OP_i}}{\left\|\overrightarrow{OP_i}\right\|}, \qquad i = \{1,2,3\} \tag{A.5}$$

Distance between the camera center and the image points is given by,

$$d_i' = \left\|\overrightarrow{OP_i}\right\| = \left\|(u_i.s_x), (v_i.s_y), f\right\|, \qquad i = \{1,2,3\} \tag{A.6}$$

$$\left\|\overrightarrow{P_iP_i'}\right\| = d_i = s_i - d_i', \qquad i = \{1,2,3\} \tag{A.7}$$

The distance between 3D image points and 3D world points with respect to the ground robot frame is given by,

$$\overrightarrow{P_i'P_i} = \widehat{OP_i}.d_i, \qquad i = \{1,2,3\} \tag{A.8}$$

By vector definition, the three points on the image plane represented in ground robot frame is given by,

$$P_i' = P_i - \overrightarrow{P_i'P_i}, \qquad i = \{1,2,3\} \tag{A.9}$$

We then compute the equation of the plane (normal form or Hessian) using the above three points given by,

$$\vec{n}.p = k,\ p = (x_i, y_i, z_i) \text{ is a point on the plane} \quad \text{A.10}$$

We consider the normal to the above plane as the z axis of the camera frame, $\vec{z}_c = \vec{n}$ and forms the third line of the rotation matrix.

$$\vec{z}_c = (r_{31}, r_{32}, r_{33}) = \frac{\vec{n}}{\|\vec{n}\|} \quad \text{A.11}$$

$$k = P_i' \times \vec{n} \quad \text{A.12}$$

We now compute the principal point with respect to the ground robot frame. The principal point lies on the intersection between the ***ray*** starting at the origin (Principal point) of the image plane, which points in the z-direction of the image plane and the image ***plane***.

$$P = \frac{k - O.\vec{n}}{\vec{n}.\vec{n}} \quad \text{A.13}$$

The x-axis vector of the camera frame is given by joining the principal point (P) and one of the control points (P_i), with respect to the ground robot frame.

$$(r_{11}, r_{12}, r_{13}) = \hat{x}_c = \frac{\overrightarrow{PP_i}}{\left\|\overrightarrow{PP_i}\right\|} \quad \text{A.14}$$

Finally, the y axis of the camera frame (and forms the first row of the rotation matrix) is given by the cross product of two vectors given by,

$$(r_{21}, r_{22}, r_{23}) = \hat{y}_c = \hat{z}_c \times \hat{x}_c \quad \text{A.15}$$

However, there is a small angle of rotation γ (between the x-axis of the camera frame and the vector $\overrightarrow{PP_i}$, which needs to be corrected. The angle γ is obtained in 2D image plane by computing the vectors using image pixels. X-axis vector is given by $\hat{x} = [1,0]$. Let $P = (x_0, y_0)$ and $P_i = (x_i, y_i)$, then $\widehat{PP_i} = \|(x_i - x_0), (y_i - y_0)\|$.

$$\gamma = arccos\left(\hat{x}, \widehat{PP_i}\right) \quad \text{A.16}$$

$$R_{(z,\gamma)} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{A.17}$$

Final orientation matrix is given by,

$$R = R_{(z,\gamma)} \times \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad \text{A.18}$$

A.3 Mathematical Morphology

Mathematical morphology [Haralick 1988] is a comprehensive and useful technique to process raster images acquired by the perspective camera. It finds applications in noise filtering, boundary and skeletal extraction of objects from images and many more. There are two primitive operations used to morph an image namely *dilation* and *erosion*. A composition of basic operations will decompose the shapes into meaningful shapes and segregate them from noisy or irrelevant information. During the process, the geometric attributes of the shapes embedded in the data are preserved. Morphology is described using set theory where a set of points (A) are partitioned into two subsets. One subset of points are selected to be transformed morphologically (keypoint) and its complement subset is not selected for modification (neighbors). Ideally, the dilation operation expands the image shapes and brightness factor increases, while the erosion operation exhibits contraction of image shapes and minimizes the brightness factor. A structuring element (B) is a mask and translated over the image and applied to a specific operation over each element in the image.

$$A_Z = \{c \in Z^N \mid c = a + z, \qquad a \in A\} \quad \text{A.19}$$

Our aim is to retain the edge properties in the normal vector grid while removing the noise. The dilation operation uses the maximum value of the neighbors to replace the value of the key-point and erosion does the same except that it uses the minimum value of the neighbors. These two operations are mathematically described below:

$$A \oplus B = \bigcup_{\vec{b} \in B} T_{\vec{b}}(A) \quad \text{A.20}$$

$$A \ominus B = \bigcup_{\vec{b} \in B} T_{-\vec{b}}(A) \quad \text{A.21}$$

We introduce another operation where the key-point in the set is replaced with the mode of the histogram associated with all elements obtained by placing a given structuring element (B) over the image (A). This is denoted by $A \oslash B$ given by,

$$A \oslash B = \bigcup_{\vec{b} \in B} T_{mode(B)}(A) \quad \text{A.22}$$

This operation filters noise and retains the geometric shape of the objects in an image. Intuitively, the outcome of this operation is that any statistical outlier element in a given patch is removed since the frequency of the outlier is observed to be less than the surrounding neighbors. The operation performs equally well on a 2D grid of normal vectors since they are scale invariant and can be quantized at known levels. Further, the edge feature extraction using morphology is described by the equation below:

$$Edge(A) = A - \{A \ominus B\} \quad \text{A.23}$$

BIBLIOGRAPHY

Abidi, M. A. and T. Chandra (1995). "A new efficient and direct solution for pose estimation using quadrangular targets: algorithm and evaluation." IEEE Transactions on Pattern Analysis and Machine Intelligence **17**(5) 534-538.

Asai, T., M. Kanbara and N. Yokoya (2005). 3D Modeling of Outdoor Environments by Integrating Omnidirectional Range and Color Images. International Conference on 3-D Digital Imaging and Modeling 447 - 454.

Awwad, T. M., Q. Zhu, Z. Du and Y. Zhang (2010). "An improved segmentation approach for planar surfaces from unstructured 3D point clouds." The Photogrammetric Record **25**(129) 5-23.

Bengtsson, O. and A.-J. Baerveldt (2003). "Robot localization based on scan-matching-estimating the covariance matrix for the IDC algorithm." Robotics and Autonomous Systems **44**(1) 29-40.

Bernardini, F., J. Mittleman, H. Rushmeier, C. Silva and G. Taubin (1999). "The Ball-Pivoting Algorithm for Surface Reconstruction." IEEE Transactions on Visualization and Computer Graphics **5** 349-359.

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

Biber, P. and W. Straber (2003). The Normal Distributions Transform: A New Approach to Laser Scan Matching. IEEE International Conference on Intelligent Robots and Systems, Las Vegas, Nevada 2743- 2748.

Biosca, J. M. and J. L. Lerma (2008). "Unsupervised robust planar segmentation of terrestrial laser scanner point clouds based on fuzzy clustering methods " ISPRS Journal of Photogrammetry and Remote Sensing **63**(1) 84-98.

Borrmann, D., J. Elseberg, K. Lingemann, A. Nüchter and J. Hertzberg (2008). "Globally consistent 3D mapping with scan matching." Robotics and Autonomous Systems **56**(2) 130-142.

Bradski, G. (2000). The openCV library. Dr. Dobb's Journal of Software Development.

Carmichael, O. and M. Herbert (1998). Unconstrained registration of large 3D point sets for complex model building. IEEE/RSJ International Conference on Intelligent Robots and Systems, Kyongju, Korea 360-367.

Chang, W.-Y. and C.-S. Chen (2004). Pose Estimation for Mulitple Camera Systems. 17th International Conference on Pattern Recognition, Cambridge, UK 262-265.

Chen, C. C. and I. Stamos (2007). Range Image Segmentation for Modeling and Object Detection in Urban Scenes. International Conference on 3-D Digital Imaging and Modeling. Montréal, Canada**:** 185-192.

Chen, Y. and G. Medioni (1991). Object Modeling by Registration of Multiple Range Images. IEEE International Conference on Robotics and Automation, Sacramento, California 2724-2729.

Cignoni, P., C. Montani and R. Scopigno (1998). "DeWall: a fast divide & conquer Delaunay triangulation algorithm in E^d " Computer-Aided Design **30**(5) 333-341.

Coleman, S. A., B. W. Scotney and S. Suganthan (2010). "Edge Detecting for Range Data using Laplacian Operators." IEEE Transactions on Image Processing

Davison, A. J., I. D. Reid, N. D. Molton and O. Stasse (2007). "MonoSLAM: Real-Time Single camera SLAM." IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(6) 1052-1067.

Dyranovski, I., W. Morris and J. Xiao (2010). Multi-Volume Occupancy Grids: An Efficient Probabilistic 3D Mapping Model for Micro-Aerial Vehicles. IEEE/RSJ International Conference on Intelligent Robots and Systems, taipei, Taiwan 1553-1559.

Eade, E. and T. Drummond (2006). Scalable Monocular SLAM. IEEE Computer Society Conference on Computer Vision and Pattern Recognition 469 - 476.

Edlinger, T., E. V. Puttkamer and R. Trieb (1991). Accurate position Estimation for an Autonomous Mobile Robot Fusing Encoder Values and Laser Range Data. IARP 91, 2nd Workshop on Sensor Fusion and Environmental Modelling. Oxford, UK.

Eliazar, A. I. and R. Parr (2004). Learning Probabilistic Motion Models for Mobile Robots. 21st International Conference on Machine Learning, Canada 32.

Ellekilde, L.-P., S. Huang, J. V. Miró and G. Dissanayake (2007). "Dense 3D Map Construction for Indoor Search and Rescue." Journal of Field Robotics **24**(-) 71-89.

Elliott, M., W. Morris and J. Xiao (2006). City-Climber, a New Generation of Wall-climbing Robots. Video Proceedings of 2006 IEEE International Conference on Robotics and Automation. Orlando, USA.

Elliott, M., J. Xiao, W. Morris and A. Calle (2007). City-Climbers at Work. IEEE International Conference on Robotics and Automation (Video Proc.). Roma, Italy**:** 2764-2765.

Feng, Y., Z. Zhu and J. Xiao (2006). Heterogeneous Multi-Robot Localization in Unknown 3D space. IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China 4533 - 4538.

Feng, Y., Z. Zhu and J. Xiao (2007). Self-localization of a Heterogeneous Multi-Robot Team in constrained 3D Space. IEEE/RSJ International Conference on Intelligent Robots and Systems 1343 - 1350.

Fischler, M. A. and R. C. Bolles (1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography." Graphics and Image Processing **24**(6) 381-395.

Fox, D., W. Burgard, F. Dellaert and S. Thrun (1999). Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. Sixteenth National Conference on Artificial Intelligence (AAAI'99) 343-349.

Fox, D., W. Burgard and S. Thrun (1999). "Markov Localization for Mobile Robots in Dynamic Environments." Journal of Artificial Intelligence Research **11** 391-427.

Gao, X.-S., X.-R. Hou, J. Tang and H.-F. Cheng (2003). Complete Solution Classification for the Perspective-Three-Point Problem. IEEE Transactions on Pattern Recognition and Machine Intelligence 930-943.

Garland, M. and P. S. Heckbert (1997). Surface Simplification using Quadric Error Metrics. SIGGRAPH '97 : Conference on Computer Graphics and Interactive Techniques 209-216.

Gelder, A. V. (1995). Efficient Computation of Polygon Area and Polyhedral Volume. Graphic Gems V**:** 35-41.

Gonzalez, R. C. and R. E. Woods (2008). Digital Image Processing. Upper Saddle River, NJ, Pearson Prentice Hall.

Grzonka, S., G. Grisetti and W. Burgard (2009). Towards a Navigation System for Autonomous Indoor Flying. IEEE International Conference on Robotics and Automation, Kobe, Japan.

Hähnel, D. and W. Burgard (2002). Probabilistic Matching for 3D Scan Registration. Proceedings of the VDI-Conference Robotik.

Hähnel, D., W. Burgard and S. Thrun (2003). "Learning compact 3D models of indoor and outdoor environments with a mobile robot " Robotics and Autonomous Systems **44**(1) 15-27.

Haralick, R. (1988). Mathematical morphology and computer vision. Twenty-second asilomar conference on signals, systems and somputers 468-479.

Haralick, R. M., C.-N. Lee, K. Ottenberg and M. Nolle (1991). Analysis and Solutions of the Three Point Perspective Pose Estimation Problem. IEEE Computer Society Conference on In Computer Vision and Pattern Recognition 592-598.

Hartley, R. and A. Zisserman (2004). Multiple View Geometry in Computer Vision. Cambridge, UK, Cambridge University Press.

Heikkilä, J. and O. Silvén (1997). A four-step Camera Calibration Procedure with Implicit Image Correction. IEEE International Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico.

Henry, P., M. Krainin, E. Herbst, X. Ren and D. Fox (2010). RGB-D Camera: Using Depth Cameras for Dense 3D Modeling of Indoor Environments. Robotics Science and Systems Workshop, Zaragoza, Spain.

Hofmann, A. D., H.-G. Maas and A. Streilein (2003). Derivation of roof types by cluster analysis in parameter spaces of airborne laserscanner point clouds. IAPRS International Archives of Photogrammetry and Remote Sensing and Spatial Information Sciences, Dresden, Germany 112-117.

Hoover, A., G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. Bowyer, D. W. Eggert, A. Flitzgibbon and R. B. Fisher (2002). "An experimental comparison of range image segmentation algorithms." IEEE Transactions on Pattern Analysis and Machine Intelligence **18**(7) 673-689.

Hoppe, H. (1996). Progressive Meshes. SIGGRAPH '96: Conference on Computer Graphics and Interactive Techniques 99-108.

Horn, B. K. P. (1987). "Closed-form solution of absolute orientation using unit Quarternions." Journal of the Optical Society of America **4**(4) 629-642.

Howard, A. (2006). "Multi-robot Simultaneous Localization and Mapping using Particle Filters." The International Journal of Robotics Research **25**(12) 1243-1256

Jensfelt, P., D. Kragic, J. Folkesson and M. Björkman (2006). A Framework for vision based bearing only 3D SLAM. IEEE International Conference on Robotics and Automation, Orlando, Florida 1944 - 1950

Johnson, A. E. (1997). Spin-Images: A representation for 3-D surface matching. Robotics Institute. Pittsburg, PA, Carnegie Mellon University. **PhD Thesis:** 308.

Kasvand, T. (1988). Extraction of Edges in 3D Range Images to Subpixel Accuracy. International Conference on Pattern Recognition, Rome, Italy.

Kato, H. and M. Billinghurst (1999). Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System. International Workshop on Augmented Reality San Francisco, USA.

Kaushik, R., Y. Feng, W. Morris, J. Xiao and Z. Zhu (2008). 3D Map Construction using Multiple Heterogeneous Robots. 10th International Conference on Control, Automation, Robotics and Vision, Hanoi, Vietnam 1230 - 1235.

Kaushik, R., S. L. Joseph, W. Morris and J. Xiao (2010). Fast Planar Clustering and Polygonal Extraction from Noisy Range Images Acquired in Indoor Environments. IEEE International Conference on Mechatronics and Automation Xian, China 483-488.

Kaushik, R., S. L. Joseph, W. Morris and J. Xiao (2011). "Polygon-based 3D scan registration with dual-robots in structured indoor environments." International Journal of Robotics and Automation **(accepted for pub.)**.

Kaushik, R., J. Xiao, S. L. Joseph and W. Morris (2010). Polygon-based laser scan registration by heterogeneous robots. IEEE International Conference on Robotics and Biomimetics, Tianjin, China 1618-1623.

Kaushik, R., J. Xiao, W. Morris and Z. Zhu (2009). 3D Laser Scan Registration of Dual-Robot System Using Vision. 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. St. Louis, USA**:** 4148-4153.

Kurogi, S., D. Wakeyama, H. Koya, S. Okada, S. Inoue and T. Nishida (2008). Application of CAN2 to Plane Extraction from 3D Range Images. IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hongkong, China 2327-2332.

Leonard, J. J. and H. F. Durrant-Whyte (1991). Simultaneous Map Building and Localization for an Autonomous Mobile Robot. IEEE/RSJ International Workshop on Intelligent Robots and Systems. Osaka, Japan**:** 1442-1447.

Liu, Y., R. Emery, D. Chakrabarti, W. Burgard and S. Thrun (2001). Using EM to Learn 3D Models of Indoor Environments with Mobile Robots. 18th International Conference on Machine Learning 329-336.

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

Lu, C.-P., G. D. Hager and E. Mjolsness (2000). "Fast and Globally Convergent Pose Estimation from Video Images." IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(6) 610-622.

Lu, F. and E. Milios (1997). "Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans." Journal of Intelligent and Robotic Systems **18**(3) 249-275.

Magnusson, M., H. Andreasson, A. Nüchter and A. J. Lilienthal (2009). Appearance-Based Loop Detection from 3D Laser Data Using the Normal Distributions Transform. IEEE International Conference on Robotics and Automation, Kobe, Japan 23-28.

Marquardt, D. (1963). "An Algorithm for Least Squares Estimation of Nonlinear Parameters." SIAM Journal of Applied Mathematics **11**(-) 431-441.

Montemerlo, M., S. Thrun, D. Koller and B. Wegbreit (2002). FastSLAM: A Factored Solution to the Simultaneous Mapping and Localization AAAI National Conference on Artificial Intelligence, Edmonton, Canada 593-598.

Mori, T. and M. Hino (2009). Distance Measuring Apparatus. U. S. Patent. **US 7,630,062 B2**.

Morris, W., I. Dryanovski and J. Xiao (2010). 3D Indoor Mapping for Micro-UAVs using Hybrid Range Finders and Multi-Volume Occupancy Grids. RGB-D Workshop at Robotics; Robotics Science and Systems. Zaragoza, Spain.

Newman, P., D. Cole and K. Ho (2006). Outdoor SLAM using Visual Appearance and Laser Ranging. International Conference on Robotics and Automation, Florida 1180-1187.

Nüchter, A. (2009). **3D Robotic Mapping** The Simultaneous Localization and Mapping Problem in Six Degrees of Freedom. Berlin, Springer-Verlag.

Nüchter, A., K. Lingemann, J. Hertzberg and H. Surmann (2005). Heuristic-Based Laser Scan Matching for outdoor 6D SLAM. Advances in Artificial Intelligence. 28th Annual German Conference on AI, Germany 304-319.

Pan, V. Y. and Z. Q. Chen (1999). The complexity of the Matrix Eigenproblem. STOC '99 Proceedings of the thirty-first annual ACM symposium on Theory of computing

Pathak, K., A. Birk, N. Vaskevicius, M. Pfingsthorn, S. Schwertfeger and J. Poppinga (2010). "Online 3D SLAM by registration of large planar surface segments and closed form pose-graph relaxation." Journal of Field Robotics **27**(1) 52-84.

Pathak, K., N. Vaskevicius, J. Poppinga, M. Pfingsthorn, S. Schwertfeger and A. Birk (2009). Fast 3D Mapping by Matching Planes Extracted from Range Sensor Point-Clouds. IEEE/RSJ International Conference on Intelligent Robots and Systems. St. Louis, MO, USA**:** 1150-1155.

Paz, L. M., P. piniés, J. D. Tardós and J. Neira (2008). "Large Scale 6-DOF SLAM with Stereo-in-Hand." IEEE Transactions on Robotics **24**(5) 946-957.

Poppinga, J., N. Vaskevicius, A. Birk and K. Pathak (2008). Fast Plane Detection and Polygonalization in Noisy 3D Range Images. IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France 3378-3383.

Quan, L. and Z. Lan (1998). Linear N>=4 Point Pose Determination. Sixth International Conference on Computer Vision 778-783.

Quan, L. and Z. Lan (1999). "Linear n-point camera pose determination." IEEE Transactions on Pattern Analysis and Machine Intelligence **21** 774-780.

Quigley, M., B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler and A. Ng (2009). ROS: an open-source Robot Operating System. International Conference on Robotics and Automation, Kobe, Japan.

Rofer, T. (2002). Using Histogram Correlation to Create Consistent Laser Scan Maps. IEEE/RSJ International Conference on Intelligent Robots and Systems, Lausanne, Switzerland 625- 630

Rusinkiewicz, S. and M. Levoy (2001). Efficient Variants of the ICP Algorithm. 3rd International Conference on 3D Digital Imaging and Modeling 145-152.

Rusu, R. B., N. Blodow and M. Beetz (2011). Fast Point Feature Histograms (FPFH) for 3D Registration. IEEE International Conference on Robotics and Automation, Kobe, Japan 3212-3217.

Rusu, R. B. and S. Cousins (2011). 3D is here: Point Cloud Library (PCL). IEEE International Conference on Robotics and Automation, Shanghai, China.

Scheibe, K., M. Scheele and R. Klette (2004). Data Fusion and Visualization of Panoramic Images and Laser Scans. International Society of Photogrammetry and Remote Sensing. Dresden, Germany.

Schnabel, R., R. Wahl and R. Klein (2007). "Efficient RANSAC for Point-Cloud Shape Detection." Computer Graphics Forum **26**(2) 214-226.

Schweighofer, G. and A. Pinz (2006). "Robot Pose Estimation from a Planar Target." IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(12) 2024-2030.

Se, S., D. Lowe and J. Little (2000). Vision based mobile robot localization and mapping using scale invariant features. IEEE International Conference on Robotics and Automation**:** 2051-2058.

Sequeira, V., K. Ng, E. Wolfart, J. G. M. Goncalves and D. Hogg (1999). "Automated reconstruction of 3D models from real environments." ISPRS Journal of Photogrammetry and Remote Sensing **54**(1) 1-22.

Smith, R., M. Self and P. Cheeseman (1990). "Estimating uncertain spatial relationships in robotics." Autonomous Robot Vehicles 167-193.

Specht, A. R., A. D. Sappa and M. Devy (2005). "Edge registration versus triangular mesh registration, a comparative study." Signal Processing: Image Communication **20**(9-10) 853-868.

Spletzer, J., A. K. Das, R. Fierro, C. J. Taylor, V. Kumar and J. P. Ostrowski (2001). Cooperative localization and control for multi-robot manipulation. IEEE/RSJ International Conference on Intelligent Robots and Systems 631-636

Stamos, I. and P. K. Allen (2000). 3-D Model Construction Using Range and Image Data. IEEE International Conference on Computer Vision and pattern Recognition, SC, USA 531-536.

Steder, B., R. B. Rusu, K. Konolige and W. Burgard (2011). Point Feature Extraction on 3D Range Scans Taking into Account Object Boundaries. IEEE International Conference on Robotics and Automation, Beijing, China.

Strobl, K., W. Sepp, S. Fuchs, C. Paredes and K. Arbter. (2006). "Camera Calibration Toolbox for Matlab." Available (online) http://www.vision.caltech.edu/bouguetj/calib_doc/index.html.

Surmann, H., A. Nüchter and J. Hertzberg (2003). "An Autonomous mobile robot with a 3D laser range finder for 3D exploration and digitization of indoor environments." Robotics and Autonomous Systems **45**(3-4) 181-198.

Surmann, H., A. Nüchter, K. Lingemann and J. Hertzberg (2004). 6D SLAM - Preliminary Report on Closing The Loop in Six Dimensions. 5th IFAC Symposium on Intelligent Autonomous Vehicles, Lisabon.

Takeuchi, E. and T. Tsubouchi (2006). A 3-D Scan Matching using Improved 3-D Normal Distributions Transform for Mobile Robotic Mapping. IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China 3068-3073.

Tan, C. C. (1989). Landmark Tracking and Camera Calibration. Dept. of Electrical and Computer Engineering. Knoxville, The University of Tennessee. **M. S.**

Tarsha-Kurdi, F., T. Landes and P. Grussenmeyer (2007). "Hough Transform and Extended RANSAC algorithms for Automatic Detection of 3D Building Roof Planes from LIDAR Data." Science and Technology **36**(1) 407-412.

Tarsha-Kurdi, F., T. Landes and P. Grussenmeyer (2008). "Extended RANSAC Algorithm for Automatic Detection of Building Roof Planes from LIDAR Data." Photogrammetric Journal of Finland.

Thrun, S. (2001). "A Probabilistic online mapping algorithm for teams of mobile robots." The International Journal of Robotics Research **20**(5) 335-363.

Thrun, S., W. Burgard and D. Fox (2005). Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). Cambridge, MA, The MIT Press.

Thrun, S., Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani and H. Durrant-Whyte (2004). "Simultaneous Localization and Mapping with Sparse Extended Information Filters." The International Journal of Robotics Research **23**(7-8) 693-716.

Thrun, S., C. Martin, Y. Liu, D. Hähnel, R. Emery-Montemerlo, D. Chakrabarti and W. Burgard (2004). "A Real Time Expectation-Maximization Algorithm for Acquiring Multiplanar Maps of Indoor Environments with Mobile Robots." IEEE Transactions on Robotics **20**(3) 433-442.

Tomono, M. (2009). Robust 3D SLAM with a stereo camera based on an edge-point ICP algorithm. IEEE International Conference on Robotics and Automation, Kobe, Japan 4306 - 4311

Tomono, M. (2009). Robust 3D SLAM with a stereo camera based on an edge-point ICP algorithm. IEEE International Conference on Robotics and Automation, Kobe, Japan 4306-4311.

Triebel, R., W. Burgard and F. Dellaert (2005). Using Hierarchical EM to Extract Planes from 3D Range Scans. IEEE International Conference on Robotics and Automation, Barcelona, Spain 4437-4442.

Turk, G. and M. Levoy (1994). Zippered Polygon Meshes from Range Images. 21st International Conference on Computer Graphics and Interactive Techniques 311-318.

Wang, T., Y. Wang and C. Yao (2006). Some Discussion on the Conditions of the Unique Solution of the P3P problem. International Conference on Mechatronics and Automation, Luoyang, China 205-210.

Weib, G., C. Wetzler and E. V. Puttkamer (1994). Keeping Track of Position and Orientation of Moving Indoor Systems by Correlation of Range-Finder Scans. IEEE International Conference on Intelligent Robots and Systems 595-601.

Weingarten, J., G. Gruener and R. Siegwart (2003). A Fast and Robust 3D Feature Extraction Algorithm for Structured Environment Reconstruction. International Conference on Advanced Robotics, Coimbra, Portugal.

Weingarten, J. and R. Siegwart (2005). EKF-based 3D SLAM for Structured Environment Reconstruction. IEEE/RSJ International Conference on Intelligent Robots and Systems 3834-3839.

Weingarten, J. and R. Siegwart (2006). 3D SLAM using Planar Segments. IEEE/RSJ International Conference on Intelligent Robots and Systems 3062-3067.

Xiao, J., A. Sadegh, M. Elliot, A. Calle, A. Prasad and H. M. Chiu (2005). Design of Mobile Robots with Wall Climbing Capability. IEEE/ASME International Conference on Advanced Intelligent Mechatronics 438 - 443.

Ye, C. and G. M. Hegde (2009). Robust Edge Extraction for Swissranger SR-3000 Range Images. International Conference on Robotics and Automation, Kobe, Japan 2437-2442.

Zhang, Z. (1998). A Flexible New Technique for Camera Calibration, Microsoft Research.

Ravi Kaushik received PhD degree on October 1st, 2011 in Computer Science from The Graduate Center, The City University of New York (CUNY), NY, USA. He pursued research (2006-2011) in 3D Mapping at the CCNY Robotics Lab at The City College of New York, CUNY. He received B. E. degree (2003) in Instrumentation Technology from M. S. Ramaiah Institute of Technology, Bangalore, India. He received M. E. in Electrical Engineering awarded by The City College of New York in 2006. His past research interest included locomotion of Biped and Quadrupedal robots (2004-2006) and was partly affiliated with Dept. of Computer Science, Brooklyn College, CUNY during the same period. His current research interests include 3D mapping with multi-robot systems, range and vision sensor technologies, 3D geometric modeling and embedded systems.