# RANGE SEGMENTATION AND REGISTRATION FOR 3D MODELING OF LARGE SCALE URBAN SCENES

by

Cecilia Chao Chen

M.S., Simon Fraser University, 2002

B.E., Huazhong University of Science and Technology, 1997

A dissertation submitted to the Graduate Faculty in Computer Science

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

The City University of New York

2007

UMI Number: 3278424

UMI Microform 3278424

Copyright 2007 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

© 2007

Cecilia Chao Chen

All rights reserved. This work may not be

reproduced in whole or in part, by photocopy

or other means, without the permission of the author.

# APPROVAL

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.

Professor Ioannis Stamos

Date — Chair of Examining Committee

Professor Ted Brown

Date — Executive Officer

Professor George Wolberg

Profeessor Zhigang Zhu

Dr. Andrew Miller

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

# Abstract

RANGE SEGMENTATION AND REGISTRATION FOR

3D MODELING OF LARGE SCALE URBAN SCENES

by

Cecilia Chao Chen

Adviser: Professor Ioannis Stamos

The goal of our research is to build an automatic 3D modeling system for large scale urban scenes. In this thesis work, various novel algorithms have been developed for segmentation and registration, focusing on important geometric features existing in urban scenes: linear features, circular features, planar regions, and arbitrary objects. The algorithms include: edge-based segmentation, region-based segmentation, linear feature-based registration, circular feature-based registration, and segment merging. Given a large number of unregistered 3D range images of an urban site as an input, our system automatically, accurately, and efficiently produces a tightly merged 3D model. The 3D models produced by this system can be used in urban planning, historical preservation, and virtual reality applications.

# Acknowledgements

First of all, I would like to express my sincere thanks to my supervisor, Dr. Ioannis Stamos. His guidance and support have been essential for the accomplishments in this thesis. I am indebted to him for his valuable ideas and discussions regarding this research, and his continuous and patient advising on academic writing. I am very grateful to Dr. George Wolberg, Dr. Zhigang Zhu, and Dr. Andrew Miller, for being members of my thesis committee, for their precious time in reviewing my thesis, and for providing insightful comments and suggestions.

My gratitude also goes to other members in our Computer Vision and Graphics Lab at Hunter College and Professor George Wolberg's lab at City College. I would like to thank Lingyun Liu and Marius Leordeanu for their priceless advice and discussions. Many thanks to Yevgeniy Pavlov for the implementation of shape descriptor, and Gene Yu for providing the ICP module.

I am thankful to my good friends, Lily Zhou, Fei Chen, Shana Hyvat, and Wen-ju Cheng, for their friendship and help over the years.

Finally, I would like to dedicate this thesis to all my family members. Their unconditional love and encouragement always accompany me wherever I go, and empower me in whatever I do.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Reconstruction of 3D digital representations of the real world is one of the most challenging and exciting problems in computer vision. With the advancements in computer graphics and imaging technologies, many coarse 3D models, previously approximated with simple geometric shapes, are currently being replaced by high resolution, high accuracy photo-realistic 3D models. Our research aims at automatically building such highly accurate 3D models for large scale urban scenes. These 3D models are useful in applications of urban planning, historical preservation, and virtual reality.

Due to the sophistication of the structure and texture of many urban scenes, the process of generating a highly accurate 3D model using state-of-the-art graphics software packages normally takes weeks or even months. In recent years, however,

the advancement of the range scanning technology has resulted in tools for assisting the building of 3D models more efficiently and more accurately. The range scanners capture dense 3D surface points and form range images, in which surface curvatures and geometric shapes are well preserved. The 3D models can then be constructed using these range images.

## 1.1 Problem Description

A typical 3D modeling system of a large scale urban scene is composed of the following phases (Fig. 1.1):

**Data Acquisition.** A scene (e.g. interior or exterior of a building) is scanned using a range scanner. One complete scan results in a capture of a set of 3D surface points, collectively known as a range image. In order to obtain the complete surface of the scene and minimize the effects of occlusion, multiple range scans from a wide variety of viewpoints need to be acquired.

**Preprocessing.** Noise removal and hole filling take place in this phase. Cleaner data leads to better performance in later phases.

**Range Segmentation.** During range segmentation, each range image is segmented into geometrically meaningful partitions. Furthermore, high level features, such as lines, curves, planes, and arbitrary geometric shapes are extracted.

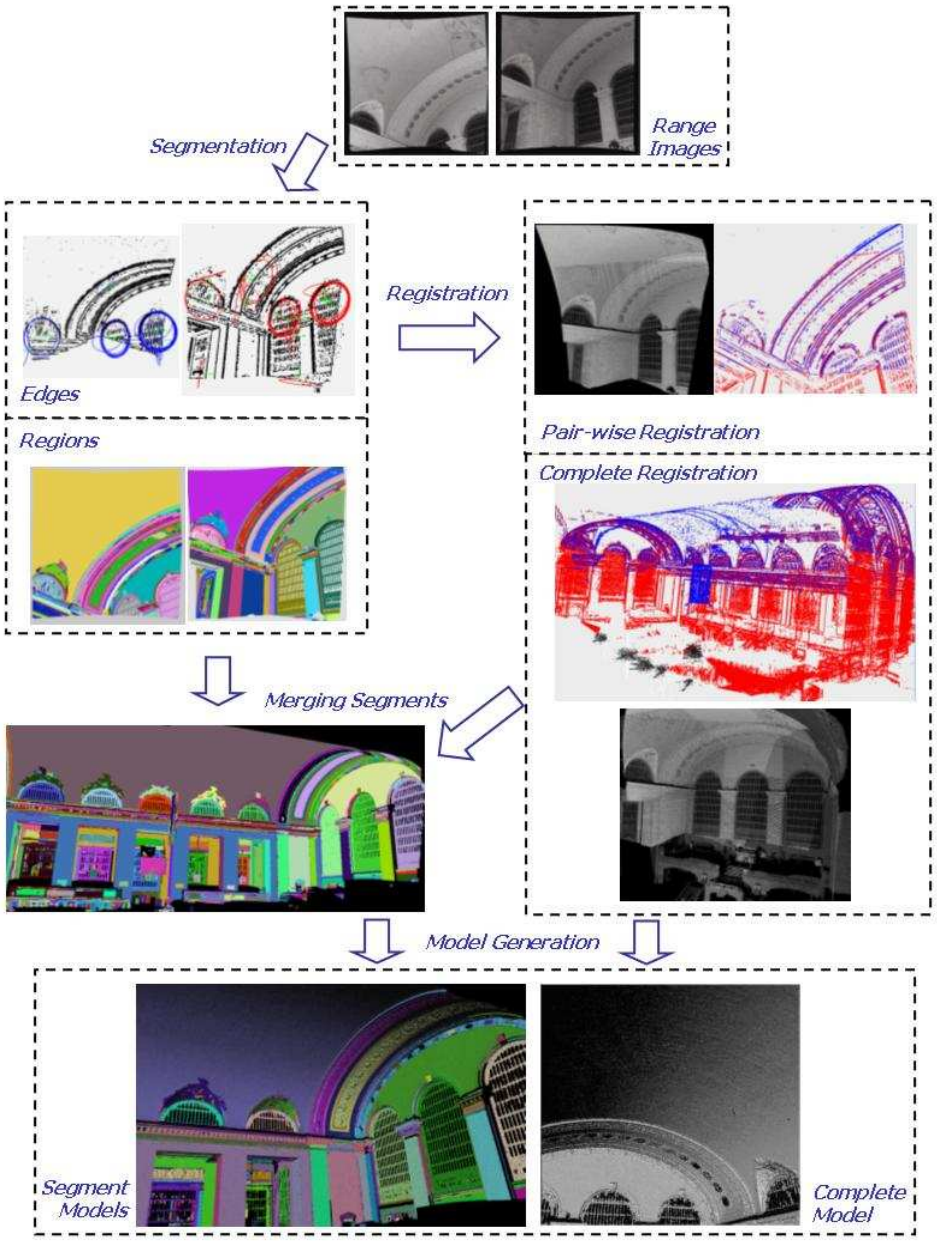



Figure 1.1: 3D modeling system of a large scale urban scene.

**Range Registration.** During range registration, the relative positions between overlapping range images are calculated. This allows placing all the range images into the common frame of reference.

**Model Generation.** Using the segmented and registered range data, 3D models can be generated either for the complete scene, or for each range segment. The generated models could be surface meshes or solid models, depending on application needs.

The precision of the results in every phase is essential for the accuracy of the final 3D model. The first phase (data acquisition) is mostly affected by scanning equipment and environment. The second phase (preprocessing) involves removing obvious data noise and filling holes. These steps are performed manually.

The phases of range segmentation and range registration are of great importance to this modeling system, but none of the previous methods have provided a satisfactory solution; more detailed discussions will be provided in Chapter 2. Our research has been concentrating on developing novel algorithms for range segmentation and range registration. The algorithms we developed are efficient, effective, and highly automated. The results of segmentation and registration facilitate the last phase, model generation.

## 1.2 Methodologies

Laser range scanning of large-scale objects is a recent technological development. That is why previous range segmentation and registration methods were only focusing on small scale objects acquired in controlled laboratory environments. These methods are not efficient and not robust enough for large scale datasets. Therefore, new algorithms that have the following characteristics need to be devised: a) they should be able to process large amounts of range data within a reasonable amount of time, and b) they should achieve satisfactory results using minimum human assistance. Our novel algorithms were designed with these requirements in mind.

### 1.2.1 Range Segmentation Algorithms

We have developed two fully automated segmentation algorithms which accurately and efficiently extract surface features useful for registration and modeling. The input to both algorithms is a range image, and the output is a set of features, that are described in the following paragraphs.

The first algorithm performs ***edge detection that results in 3D line and 3D circle extraction***. The edge detection approach adapts and revises the popular Canny 2D edge detection technique for the extraction of 3D continuous edges from a range image. Using these edges, linear and circular features are detected. These features are utilized in the registration phase.

The second algorithm performs ***region segmentation***. This algorithm follows the standard region growing approach, and segments the range image into regions of three types: planar regions, non-planar smooth regions, and non-smooth connected regions. The output of this algorithm is a set of meaningful regions with highly accurate boundaries. This information is useful for modeling and object recognition.

### 1.2.2 Range Registration Algorithms

We have developed two algorithms for fast and accurate registration based on different types of features. The input to both registration procedures is a set of range images, each with features extracted from the segmentation phase. The output is the set of transformations that transform all range images into a common coordinate system.

The first algorithm we have developed is ***a semi-automatic registration system based on linear features***. This algorithm registers structures containing abundant linear and planar features. For each overlapping range image pair, this algorithm automatically analyzes the 3D lines and outputs a transformation which matches the most number of lines. In the cases of scene symmetry, a user-friendly graphical interface assists the user to either select the correct transformation out of a list of possibilities or to perform necessary adjustments.

The second algorithm we have developed is ***an automated registration based on circular features***. This algorithm registers scenes by matching 3D circular edges

extracted from the edge segmentation procedure. This algorithm also provides a more robust and efficient method of ranking candidate transforms. In our experiments, the best one is always closest to the true transformation, and the registration error is very small.

Since both of our methods are feature-based algorithms, we post-process our registration results with point-based ICP optimization, to minimize registration error. Our experimental results show that the ICP routine is usually completed within a few iterations, which further implies the high accuracy of our segmentation and registration processes.

To extend our registration algorithms to match arbitrary shapes, we have also studied ***pattern matching based on statistical shape descriptors***. With preselected patterns, we have obtained satisfactory experimental results.

### 1.2.3 Segment Merging Algorithm

When 3D models are generated based on registered range images with segmented regions, it is necessary to merge overlapping regions from different range images, in order to generate coherent segments in the combined 3D scene. Based on each merged segment, a different mesh is generated. This approach provides us with 3D models corresponding to individual urban objects (such as facades, windows, ceilings, etc.) within a complete large scale urban scene. Our algorithm of ***merging segmented***

***regions*** takes a set of segmented range images with their registration transformations as an input, and outputs a coherent segmentation for the entire scene.

## 1.3 Contributions

The range segmentation and registration algorithms described in this thesis are the major components of the modeling pipeline of large scale urban scenes. We have applied these algorithms to both interior and exterior scenes that contain a large variety of geometric shapes, and the algorithms performed very well. Below is a list of our contributions:

(1) A semi-automatic registration system is developed. This system includes an automated linear feature registration system and a context-sensitive user interface for necessary user interactions. This work was published and presented at the 5th International Conference on 3-D Digital Imaging and Modeling in June 2005 [Chen and Stamos, 2005].

(2) A fully automated system is developed for segmenting and registering range images containing circular features. This work was published and presented at the 3rd International Symposium on 3D Data Processing, Visualization and Transmission in June 2006 [Chen and Stamos, 2006]. It was also published as a part of a journal paper in the International Journal of Computer Vision, 2007 [Stamos *et al.*, 2007].

(3) A novel segmentation algorithm is developed to extract planar, smooth non-planar, and non-smooth connected segments. In addition, registered segmented images are merged to form coherent segments. This work was published and presented at the 6th International Conference on 3-D Digital Imaging and Modeling in August 2007 [Chen and Stamos, 2007].

(4) The promising experimental results on pattern matching based on statistical shape descriptor bring us one step closer to a multi-feature registration system of high accuracy, low computational complexity, and minimum human interaction.

## 1.4 Thesis Outline

The remainder of the thesis is organized as follows: Chapter 2 discusses related work in range segmentation and range registration. Chapter 3 introduces range image acquisition and representation, together with examples of range images used in this thesis. Chapter 4 presents our two segmentation algorithms: an edge-based algorithm for extracting linear and circular features; and region-based algorithm for extracting region segments. Chapter 5 describes two registration algorithms: a line-based registration system with user interface; and an automated circular feature-based registration. Our research results for matching arbitrary shapes based on statistical features are also presented. Chapter 6 introduces our work on merging segmented range images for modeling. Chapter 7 concludes the thesis, providing some directions for future work.

# Chapter 2

# Related Work

## 2.1 3D Modeling Approaches

As computer graphics and computer vision fields progressed, the technology of 3D modeling has also gone through a few generations.

The earliest 3D models were developed for the purpose of computer animations. These are virtual models built with the aid of graphical software packages (e.g. Maya, 3ds Max, AC3D). The process of building such models is as follows: based on the design of a virtual object composed of different parts, computer engineers manually select and configure the geometric shape of each part, and then define their relative positions. Given all the information, the graphical software creates a complete 3D model of a virtual object.

With the emerging need for modeling real world objects, new approaches had to be designed, as 3D models of the real world objects cannot be built using "virtual modeling" approaches. This led to the addition of multiple operations into the modeling process, such as object measuring and shape simplification, among others. These additional steps made the reconstruction of real world objects possible, with the limitation that the approach can only be applied to objects having simple shapes and a relatively small volume. Reconstructing large scale urban structures down to fine details, using this method, is however impractical, as it would consume many human hours and intense human labor – thus the need for better approaches.

With the development of computer vision technologies, such as shape from shadow, shape from motion, and image-based rendering, easier and faster approaches became possible. Instead of first building a 3D geometric model and then rendering the object on 2D screen, these approaches analyze 2D images or videos of an object and obtain information on its 3D surface shape. The information is however limited, as it only includes the silhouettes from certain viewpoints, and the depth and general shape of certain parts. With this 3D shape information and available 2D images, views from arbitrary viewpoints may be generated by interpolation. The image-based rendering approach is fast even for large objects, and it preserves the visual effects of the object surface very well. Its drawback, however, is that, the interpolation results may not be natural and accurate, as the actual 3D model of the object is often not fully recovered.

Therefore, this approach is inapplicable if a highly accurate 3D geometric model is required.

More recently, laser range scanning has become one of the most useful technologies for assisting in 3D modeling. There have been numerous projects for reconstructing statues [Bernardini *et al.*, 2002], historical sites [Miyazaki *et al.*, 2000] [Guidi *et al.*, 2005], and urban buildings [Stamos and Allen, 2002] using ranging scanning. Compared to the previous two approaches, data acquisition takes more time, but the collected range data contains much more abundant information about the geometric shapes of the scanned object. With well designed computer software, range data analysis and 3D modeling could be highly accurate and highly efficient. Below, we briefly discuss related methodologies in range image analysis, and our algorithm design for the purpose of large scale scene modeling.

## 2.2 Range Segmentation

Range image segmentation, or range segmentation, partitions a range image into meaningful parts and extracts important geometric features from it. An edge map or a set of surface patches is the basic output of a segmentation process, depending on whether it is an edge-based or a region-based segmentation. In addition, many algorithms compute characteristic parameters for these edges and regions, in order to form higher level 2D structures (lines and curves) or 3D structures (planar, quadric

and other higher order surfaces).

Due to the similarity between range images and 2D images, most of the 2D image segmentation techniques, such as edge detection, region growing, and pixel clustering, can be applied to range image segmentation. Instead of pixel intensity in 2D images, depth and the local plane normal of each point are used for computation in range image segmentation. A comprehensive study and comparison of most segmentation algorithms prior to year 1996 can be found in [Hoover *et al.*, 1996]. Below we discuss some of them in detail, together with more recent research publications.

### 2.2.1 Edge-based Approaches

Edge-based segmentation attempts to extract discontinuities in both depth and surface orientation. In [Hoffman and Jain, 1987], Hoffman and Jain categorize edges into three types: step edges (neighboring pixels with significant depth difference), roof edges (neighboring pixels with wide difference in normal orientation), and smooth edges (gradual and smooth variations in normal orientation). Techniques of 2D image processing such as edge detection and boundary closing have been adapted to locate edge points and to form closed region boundaries.

Most of edge detection techniques use morphological methods [Krishnapuram and Gupta, 1992] or edge masks [Wami and Batchelor, 1994] [Bellon *et al.*, 1999] to initiate a set of edge points. Then, edge linking [Jiang, 2000] and thinning [Bellon *et al.*, 1999]

[Bellon and Silva, 2002] are carried out. At this point, the segmentation does not analyze surface parameters, but rather provides a continuous and closed edge map containing all feature curves in the range scan.

Most edge-based segmentation algorithms are threshold-based and sensitive to noise. Thresholds are often determined empirically, to avoid over-segmentation or under-segmentation. In [Bellon and Silva, 2002], Bellon and Silva present their segmentation algorithms with adaptive thresholds, determined locally at each point's neighborhood. This algorithm does not depend on rigid threshold values, which makes it applicable in unsupervised systems. Its limitation is that it only segments planar and polyhedral objects.

The methods of edge-based segmentation are most useful when the generated edge map is sufficient for scene understanding (e.g., the scene is composed of objects having simple geometric shapes such as cubes and spheres). When more complicated curved surfaces exist in the scene, edge-based segmentation does not provide information about these curved surfaces. In this case, the region-based segmentation algorithm, introduced in the next section, is more useful.

### 2.2.2 Region-based Approaches

As opposed to edge-based approaches, region-based methods analyze surface areas separated by edges in more details. This type of method has been playing a dominant role in applications where region representation is important for image analysis and object recognition. The region-based segmentation involves grouping pixels into surface regions based on the homogeneity or similarity of surface properties. Different algorithms have been proposed for different surface models. Our discussion begins by introducing methods analyzing simple geometric shapes, followed by those analyzing regions with more general shapes.

In [Pulli and Pietikäinen, 1993], Pulli and Pietikäinen present an algorithm that utilizes three pieces of information at every point for segmentation: the x and y components of the normal vector, and its depth value. These three values are normalized and then utilized as R/G/B intensities to form a color image. With a hierarchically connected component analysis method, this color image is used to segment the original range scan into either planar or smooth curved surfaces. The algorithm in [Han *et al.*, 1987] analyzes the surface normal at each point, and extracts the parameters for surfaces of planes, cylinders, and spheres from the range scan. Another segmentation method proposed by Jiang *et al.* [Jiang *et al.*, 1996] uses scan lines as high-level primitives for segmentation. Each scan line is fitted to a quadratic curve, and then neighboring scan lines are merged, if they satisfy certain normal constraints, forming

regions of smooth surfaces.

The above region-based segmentation methods are restricted to surfaces having simple geometric shapes such as polyhedral, spherical or cylindrical. These methods do not apply to objects having arbitrary shapes. The general approach to analyzing arbitrarily shaped objects is to fit variable-order surface models to range data subsets. Typically, an iterative process randomly chooses small initial seed regions, and groups points into connected regions by merging neighboring homogeneous regions.

Besl and Jain [Besl and Jain, 1988] provide a general solution to approximate regions with polynomial surfaces of degree less than or equal to 4. An initial partition is computed by point classification according to eight fundamental forms based on the sign of the Gaussian and mean curvatures. Then, regions are iteratively grown by containing more neighboring points of the same fundamental form, and testing on the surface fit in the order of planar, biquadratic, bicubic, and biquartic, until the final partition is achieved.

Following this work, Djebali *et al.* [Djebali *et al.*, 2002] segment surfaces based on polynomial approximations and region growing. Similarly, Bab-Hadiashar and Suter [Bab-Hadiashar and Suter, 2000] sequentially fit a subpopulation of a model with linear or quadratic equations representing planar surface and partial quadratic surface; in [Marshall *et al.*, 2001], Marshall *et al.* analyze geometric primitives such as planes, spheres, cylinders, cones and tori, and find parameters for these 3D data

set by nonlinear least-square fitting.

The least mean squares (LMS) estimate used in the above methods is robust to Gaussian perturbations, but is very sensitive to other classes of noise because all data points are taken into account equally and one outlier can completely corrupt the fit. Statistical techniques known as robust estimators have thus become very popular, because they can tolerate data from different statistical populations (e.g. neighboring surfaces, impulsive noise) while estimating the parameters of the dominant population (the actual surface). Representative robust estimators include RESidual Consensus (RESC) [Yu *et al.*, 1994] [Gotardo *et al.*, 2004], Adaptive Least K-th order squares Estimator (ALKS)[Lee *et al.*, 1998], and Minimum Unbiased Scale Estimator (MUSE)[Miller and Stewart, 1996]. These algorithms have been shown to outperform LMS, but they suffer from the large number of sampling operations, leading to high time complexity.

As we mentioned earlier, region-based segmentation algorithms extract meaningful partitions from a range image using homogeneity or similarity of surface properties as the extraction criteria. In the algorithms we have discussed thus far, similarity in surface curvature is used as extraction property. In some applications, however, a more intelligent approach to region grouping is needed. For example, if there is a chair in the scene, basing the decision on surface curvature will lead to the chair being segmented into many partitions. This is due to the fact that many of the chair's

surfaces have different orientations. An alternative segmentation strategy could be to combine a set of points spatially closer to one another as compared to other points in the scene. Such a segmentation procedure would group all the chair's points into one segment, producing a much more meaningful result.

Yu *et al.* [Yu *et al.*, 2001] propose a novel segmentation algorithm that produces such results. Their approach extends the concept of "normalized cut" [Shi and Malik, 2000] from 2D image segmentation. It is a top-down algorithm which recursively partitions a point set into two subsets using a pairwise similarity measure computed from the 3D location, normal estimation, and returned laser intensity at each point. The resulted partition gives a natural segmentation to the scene (e.g., walls, chairs, lamps, tables, sofa). The drawback is the time complexity due to the partition search in each iteration, as well as the top-down hierarchical segmentation process.

The general weakness of region-based segmentation methods is the time complexity resulting from the large amount of computation involved in function fitting for every region in every iteration. An additional drawback is that these algorithms produce closed segmented regions but do not provide accurate boundary information.

As both edge-based and region-based segmentation methods have drawbacks, some hybrid approaches have been devised [Fan *et al.*, 1987] [Yokoya and Levine, 1989] [Bhandarkar and Siebert, 1992]. These methods attempt to keep the strength of both edge-based and region-based approaches, while minimizing their weaknesses. More

specifically, the hybrid approaches carry out both edge detection and region growing, sequentially or simultaneously, providing higher efficiency and rich descriptions of both the surface primitives and region boundaries.

### 2.2.3 Discussion

Region-based algorithms provide explicit descriptions about geometric shapes of surfaces and provide closed regions, but are complex, time-consuming, and do not provide accurate boundary information.

Edge-based algorithms present border location more precisely, use simple operations, and can be efficiently implemented. However, they are inherently unstable, cannot guarantee closure of boundaries, and fail when the object has very few or no sharp edges.

Most of the reviewed segmentation methods produce edge maps and/or regions approximated with polynomial functions. The results of these methods are generally used for small scale object modeling and reconstruction. Our segmentation algorithms consider the features of large scale urban buildings, and focus on extracting linear features and planar features accurately. We have designed two segmentation algorithms, extracting either edge features or region features, respectively, for different purposes. For the purpose of extracting robust geometric features for efficient registration, we

implemented an edge-based algorithm for extracting linear features and circular features. This algorithm will be explained in Section 4.1. For the purpose of object modeling, we implemented a region-based algorithm that emphasizes highly accurate surface normal analysis and extraction of meaningful regions (similar to [Yu *et al.*, 2001]). The algorithm will be introduced in Section 4.2.

## 2.3 Range Registration

Range image registration, or range registration, involves finding the relative position and orientation between overlapping range image pairs, and then transforming them into a common coordinate system, so that a 3D model of the entire scene can be built.

Existing registration algorithms can be categorized into two types: point-based or feature-based. Point-based registration algorithms start with an approximate registration, and then iteratively refine that registration by gradually reducing the error between overlapping areas in range images. Feature-based registration methods utilize correspondences between a number of features in the range images to compute the transformation between them.

### 2.3.1 Point-based Methods

Among the existing surface matching methods, the iterative closest point (ICP) algorithm proposed by Besl and McKay [Besl and McKay, 1992] and Chen and Medioni

[Chen and Medioni, 1992] has been proven to be the most appropriate method for accurately registering a set of range images. ICP starts with two range images and an initial guess for their relative transformation, and iteratively refines it to minimize an error metric. In [Besl and McKay, 1992], this error is the mean square of distances between corresponding points, while in [Chen and Medioni, 1992], it is the mean square of distance from each point in one image to its closest plane in the other image. Each iteration has three steps: compute the closest point/plane for each point in the overlapping area; compute the transformation; apply the transformation. The algorithm terminates when the change in error falls below a preset threshold. A detailed explanation of finding the closest points and computing the transformation that minimizes their distances is stated in [Besl and McKay, 1992]. The ICP algorithm can be applied to most kinds of different representations of geometric data. It is shown to achieve fast convergence, and can handle a reasonable amount of noise.

Since ICP is a non-linear local search algorithm, it suffers from many problems commonly associated with local searches, such as slow convergence and the tendency to fall into and remain in local minima. Many variants of ICP have been proposed, affecting all phases of the algorithm from the selection and matching of points to the minimization strategy. To speed up the convergence, there are algorithms with random or uniform sampling points, or accelerated with a K-D tree [Masuda *et al.*, 1996] [Johnson and Kang, 1997]. Variations of the minimization strategy include

direct minimization via Levenberg-Marquardt algorithm [Fitzgibbon, 2001], normal shooting for computing error [Dorai *et al.*, 1997], and measurement error distribution [Okatani and Deguchi, 2002]. To improve the robustness, [Gelfand *et al.*, 2003] proposes a point selection strategy that improves geometric stability, [Johnson and Kang, 1997] combines color information into corresponding point selection, [Sharp *et al.*, 2002] combines curvature information into corresponding point selection, [Zhang, 1994] uses a coarse-to-fine strategy to select proper point pairs to handle outliers and occlusions , and [Brown and Rusinkiewicz, 2004] proposes a hierarchical ICP approach to find feature correspondences and uses a thin-plate spline to wrap around the surface for error computation.

A table that lists out many of the variants of ICP and summarizes their features can be found in [Standford ICP]. Rusinkiewicz and Levoy [Rusinkiewicz and Levoy, 2001] enumerate and classify many of these variants, and evaluate their effects on the converging speed with which the correct alignment is reached. Based on this analysis, they further propose a high-speed ICP algorithm by combining those algorithm variants which are proven to help with fast convergence.

Because any ICP-type method only finds a local minimum, it requires an initial registration that is reasonably close to the true alignment, in order for the set of closest point correspondences to be correctly established. The earliest approaches for satisfying this constraint included either having a controlled scanning environment

or manually selecting a few sets of corresponding points on two range images. In both cases, an approximate transformation between any two range images could then be easily calculated. More recently, feature-based registration techniques which can automatically estimate an initial registration, have been proposed. Since the feature-based methods usually provide a coarse registration that requires further refinement, these methods are referred to as coarse registrations. The point-based registrations such as ICP algorithm, on the other hand, are called fine registrations.

### 2.3.2 Feature-based Methods

In Feature-based registration algorithms, correspondences between a number of features in the range images are first established, and then a transformation is calculated based on them, in order to register the range images. There have been many different features utilized in previous research. Roughly, they can be categorized into those using geometric features [Thirion, 1996] [Stamos and Leordeanu, 2003] and those using statistical features [Johnson, 1997] [Johnson and Hebert, 1997] [Osada *et al.*, 2001].

Previously utilized geometrical features include point features and line features. In [Thirion, 1996], a special type of feature points called *extremal points* are extracted. Extremal points are points whose relative positions are invariant to 3D rigid transforms. The matching feature is the local geometric shape around these extremal points. In [Stamos and Leordeanu, 2003], Stamos and Leordeanu develop a

line-matching approach that matches between range images of urban buildings. The features used for registration are 3D lines extracted from the segmentation of the range images. The algorithm evaluates each conjecture that two lines from two scans are matching each other, and then computes a unique transformation with these two line pairs. The transformation is further applied to the lines from two range images, and the total number of line matches is recorded as the measure of goodness of the registration. The transformation with the highest score is considered as the estimated transformation between these two pairs. The final global registration is based on the graph of pairwise registration, and the transformations with highest scores are used to convert each scan to the coordinate system of one pivot scan.

Geometrical feature matching is a straightforward and easy-to-implement approach. However, it is very sensitive to noise, and it is applicable only to images containing designated features. The approach of using statistical features, on the other hand, is a more robust one, and can be applied to arbitrary surfaces. One popular approach of statistical feature matching is the spin-image introduced by Johnson and Hebert [Johnson, 1997] [Johnson and Hebert, 1997]. A spin image is a 2D histogram of the relative positions of nearby points with regard to a certain point, i.e., a two-dimensional descriptor of the local shape of a free-form 3D surface at that point. By comparing the spin-images of two different points, we obtain a measure of local shape similarity between the surfaces surrounding those two points. This similarity

is used to find corresponding points from two scans and then estimate the transformation between two range scans. Other similar point-based features have also been proposed, such as point signature, point fingerprint, signed distance field, surface signature, and tensor of curvature.

The spin images have been widely applied to the registrations of range images, because they are invariant to rigid transformations, and do not depend on specific geometric types. However, this method has three drawbacks: the accuracy is sensitive to varying scan resolutions; the extracted point signatures have local support and the similarity threshold is specified by the user; the registration result usually requires further refinement. Recently, some improved methods were proposed to resolve the above issues. Examples include multi-resolution spin images[Brusco *et al.*, 2005] and textured spin-images[Quynh Dinh and Kropac, 2006]. In [Carmicheal and Hebert, 1998] [Broz *et al.*, 1999], spin image matching followed by ICP optimization is applied to register large 3D point sets of complex model buildings.

Recently, more robust statistical methods are utilized for range image registration. Extended Gaussian Images are utilized in [Makadia *et al.*, 2006] in order to estimate 3D rotation. Other methodologies in 3D shape analysis and object recognition can also be used in range registration. Summaries and analysis of most of these technologies can be found in surveys [Loncaric, 1998] [Tangelder and Veltkamp, 2004]. Among them, the research by Princeton Shape Retrieval and Analysis Group

has been of great interest and success. In one of their papers, [Osada *et al.*, 2001], the authors represent the shape descriptor by the probability distribution of the distances between randomly sampled point pairs. The similarity measure between the descriptor of a region and that of the model is based on the average difference between their probability distributions. This approach, although designed to match between complete 3D object models, can be adapted into our registration procedure. Details will be given in Section 5.3.

### 2.3.3 Discussion

Point-based registration methods, such as the ICP algorithm and its many variants, are used to obtain fine registrations among range images. These methods iteratively update the transform to reach an optimized registration, provided that a close enough initial registration is known. They are time consuming for large-scale range data, and easily converge to a local minima.

Feature-based methods, on the other hand, are usually called coarse registration methods. The transformations are computed based on extracted features from the range images, and the feature extraction phase often introduces more noise and inaccuracy, leading to an approximate registration that is in need of refinement. These algorithms are therefore frequently used as preprocesses of point-based registration methods. Some of the extracted features characterize the object surface in a higher

level than points, and may be combined with segmentation results and be useful for object recognition.

For the purpose of registering large scale urban building scans, our registration approach involves two steps: a feature-based coarse registration and a point-based fine registration using ICP. For the first step, we have implemented two algorithms, registering based on linear and circular features, respectively. These algorithms will be described in more detail in Section 5.1 and 5.2.

# Chapter 3

# Range Image Acquisition

## 3.1 Basic Principle of Laser Scanning

The range images used in our research are acquired by time-of-flight laser range scanners, which are suitable for capturing large scale urban scenes. A range scanner is analogous to a photo camera, and range scanning is analogous to photo-capturing (Fig. 3.1). Just like a camera, a range scanner has a cone-like field of view. Just like a photograph, the range image is indexed on a rectangular grid representing the directions of sample points. Additional similarity to photo-capturing is that range scanning only collects information about visible surface. The information being collected, however, is different between a photograph and a range image. While a photograph records the color intensity, a range image records the 3D position and laser reflectance

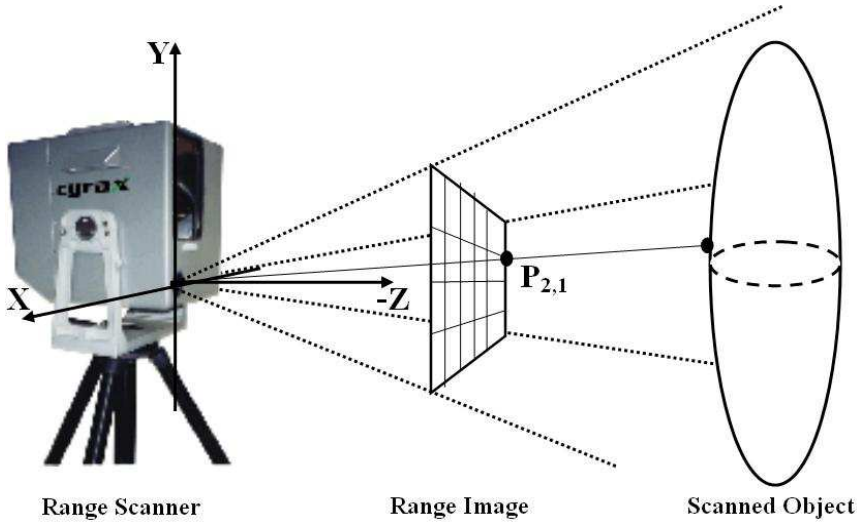



Figure 3.1: Range Scanning. The four dotted lines define the scanner's field-of-view. In the scanner's local coordinate system, $-Z$ axis points vertically out of the scanner's front face, $X$ and $Y$ axes are parallel to the two sides of its front face. For the sake of simplicity, a $5 \times 6$ range image is shown. $P_{2,1}$ is located on the $2^{nd}$ row and $1^{st}$ column of the range image grid, representing the 3D point captured by the corresponding laser beam.

intensity of each point.

When the laser scanner is positioned towards a scene or an object, it sequentially emits laser beams and receives returning beams reflected from the scanned surface. The directions of the laser beams are aligned in horizontal rows and vertical columns, with a constant horizontal angle and vertical angle between neighboring beams, respectively. Each laser beam detects a single surface point, providing two pieces of information: reflected laser intensity, and the time of flight of the laser beam. The distance from the scanner to this point is calculated using the time of flight. Then,

based on the distance and the angle of the beam, the 3D position of this point is calculated within the local coordinate system relative to the scanner (see Fig. 3.1). If, for some laser beam, there is no surface point to reflect it back, an empty point with coordinates $(0, 0, 0)^T$ is associated with this beam. All the 3D point coordinates are recorded and indexed on the 2D array defined by the laser beams. This 2D array is collectively known as a range image or a range scan.

The mathematical representation of a range scan is defined as follows: Each range scan $R$ is represented as an $N \times M$ array of 3D points, where $N$ is number of sample surface points in each row, and $M$ is the number of sample surface points in each column. Each point $\mathbf{P}_{i,j}$ $(i = 1 \ldots N, j = 1 \ldots M)$ is associated with four values $(x, y, z, l)^T$, where $(x, y, z)^T$ is its 3D coordinate in the scanner's local coordinate system, and $l$ is the laser intensity of the returned laser beam. The indices $(i, j)$ define the point $\mathbf{P}_{i,j}$'s position on the 2D range image grid. Based on the 2D range image grid, it is possible to define connectivity relationships. Every point has an 8-neighborhood around it, and the connectivity relationship on the 2D grid corresponds to the connectivity relationship among corresponding points in the 3D domain.

## 3.2 Leica HDS 2500 Laser Scanner

The range data used by our experiments are acquired by a Leica HDS 2500 laser scanner [Levoy *et al.*, 2000]. This scanner has a maximum $40^o \times 40^o$ field-of-view and

operates over distances of up to 100m. The single-point range accuracy of this scanner is $6mm$. This scanner can generate high resolution scans at a speed of approximately 1000 points per second, each scan containing up to a million sample points.

## 3.3 Acquired Data Sets

Our experimental data includes five urban buildings in New York City, shown in the following figures. During the data acquisition phase, these buildings were scanned, each taking a day or two, with the number of sample points captured in each scan ranging between 500,000 and 1 million. For each building listed below, we show one or two photographs, and the range images registered with the algorithms described in this thesis.

**Fig. 3.2 The Thomas Hunter building of Hunter College, CUNY**. It is a rectangular building with flat side walls, containing abundant linear features and planar areas.

**Fig. 3.3 The Shepard Hall building of City College, CUNY**. It is a complicated architecture that resembles a Gothic cathedral.

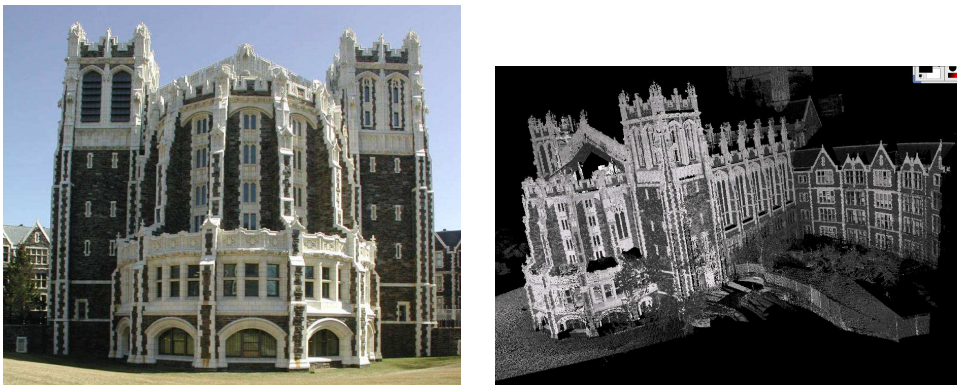
**Fig. 3.4 The Cooper Union building**. It is a building of national historic landmark. It has four rectangular facades.

**Fig. 3.5 An interior Great Hall**. It is a music hall inside the Shepard Hall building. It is a rectangular hall, with a spherical stage and many interior decorations.

**Fig. 3.6 The interior of the Grand Central Station**. It is a land-mark building in NYC. It is a rectangular hall with an arch ceiling, and contains many large circular windows.

(a) (b)

Figure 3.2: The Thomas Hunter building of Hunter College, CUNY. (a) Photograph. (b) 14 registered range images shown.



(a) (b)

Figure 3.3: The Shepard Hall of City College, CUNY. (a) Photograph. (b) 24 registered range images shown.

(a) (b)

Figure 3.4: The Cooper Union building, NYC. (a) Photograph [3]. (b) 13 out of a total of 31 registered range images shown.

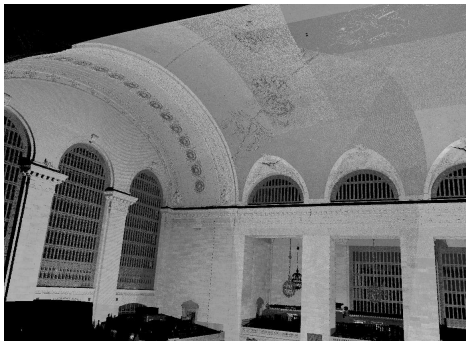
(a)

(b)

(c)

Figure 3.5: The Great Hall, a concert hall inside the Shepard Hall. (a) Photograph 1 - the stage. (b) Photograph 2 [1] - view from the stage. (c) 21 registered range images shown.

(a)



(b)

Figure 3.6: The Grand Central Station, NYC. (a) Photograph [2]. (b) 8 out of a total of 37 registered range images shown.

# Chapter 4

# Range Segmentation

In computer vision, range segmentation refers to the process of partitioning a range image into multiple regions and extracting important geometric features. Segmentation results in a simplified representation, which is more meaningful and easier to analyze.

There are two types of range segmentation: edge-based and region-based. In an edge-based segmentation, region boundaries are extracted from a range image, analyzed, and further identified as 3D geometric features such as lines, circles, or other curves. In a region-based segmentation, continuous regions are extracted, analyzed, and further identified as 3D structures such as planes, curved surfaces (smooth non-planar regions), or point clusters representing 3D objects (non-smooth regions).

In this chapter, we describe two algorithms: an edge-based segmentation algorithm

that extracts linear and circular features; and a region-based segmentation algorithm that partitions the image into planar regions, smooth non-planar regions, and non-smooth regions.

## 4.1 Edge-based Segmentation

This section introduces the algorithm and experimental results of our edge-based segmentation. The input to this algorithm is a range image, and the output is a set of continuous edges identified as either linear or circular features.

The algorithm, during its edge identification process, assumes that the 3D edges could be categorized as one of the following two types: (a) edges formed by surface normal discontinuities (roof edges), and (b) edges formed by depth discontinuities (step edges). Furthermore, the step edges are divided into the edges formed by one surface occluding another (occlusion edges), and edges formed by 3D surface boundaries (boundary edges).

In order to identify these 3D edges, our algorithm begins by decomposing the surface orientation change at each point into four variation values. Using these values, the algorithm forms four directional variation images, each encoding the surface orientation change along one range image grid direction (details in Section 4.1.1). Then, Canny edge detection is utilized to extract 2D edges from each directional variation image. Finally, the 2D edges are projected back to the 3D space to form 3D edges.

Below are the major steps of this algorithm:

**Forming 2D directional variation images.** The range image is decomposed into four 2D directional variation images, each encoding the surface orientation change along one of the four directions on the 2D grid on which the range image is organized.

**Detecting 2D edges.** 2D edges are detected on each directional variation image using modified Canny edge detection algorithm.

**Generating 3D edges.** 2D edges from the four directional variation images are combined and projected back to 3D space to form 3D edges.

**Linear and circular feature extraction.** 3D edges are identified as either linear or circular. Furthermore, function fitting is applied to these edges, estimating the parameters of the 3D features they represent or are a part of.

### 4.1.1 2D Directional Variation Images

Let us explain our algorithm using a range image of the Grand Central Station as an example (Fig. 4.1). The directional variation images are obtained from the range image as follows. The surface orientation change at each point is decomposed into variations along four grid directions. This grid is the 2D structured grid on which each range image is organized (Chapter 3). We thus obtain four values at every 3D point,

and name them directional variation values. Taking each directional variation value at every point as a grayscale intensity at every corresponding pixel on the structured 2D grid, we form four directional variation images. Below are the algorithmic details.

At each point $P$, let $B_1$ and $B_2$ be its two neighbors along one of the four grid directions (see Fig. 4.2(a)). The vector from $P$ to $B_1$ is $\mathbf{V_1}$, and from $P$ to $B_2$ is $\mathbf{V_2}$. The variation at each direction for point $P$ is computed as the angle between $\mathbf{V_1}$ and $\mathbf{V_2}$ normalized with $\pi$. This provides a value in $(0, 1]$[1] as the intensity value for $P$'s corresponding pixel, denoted as $p$, on this 2D directional variation image. Note that if $P$ is an empty point, its corresponding pixel $p$ does not have a calculated intensity value. In this case, it is set to $-0.1$. If $P$ has only one neighbor along a grid direction, the angle between $\mathbf{V_1}$ and $\mathbf{V_2}$ is also unavailable, and the intensity value is set to 0 in the corresponding directional variation image. In later sections, we will see that these values are used for detecting boundary edges.

As can be seen from Fig. 4.2, each 2D directional variation image emphasizes surface normal change along one direction. Combining the edges detected from each directional variation image provides a complete set of edge points.

[1]The angle between $\mathbf{V_1}$ and $\mathbf{V_2}$ is always larger than 0, as points $P$, $B_1$, and $B_2$ are located on different rows or columns of the grid structure produced during the scanning procedure. The largest possible value for this angle is $\pi$. Therefore, the normalized value is in the range $(0, 1]$.

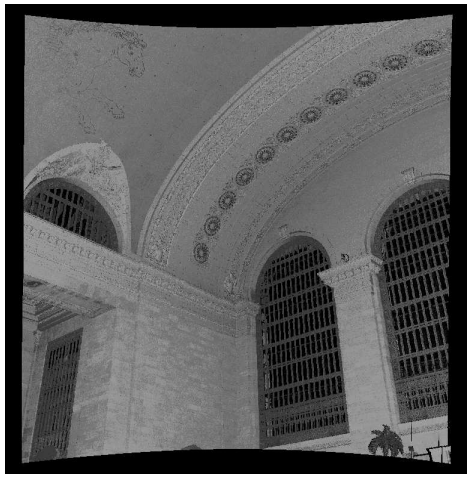


Figure 4.1: A range image of the interior of the Grand Central Station.

### 4.1.2 2D Edge Detection

2D edge detection is performed on each of the four directional variation images (examples shown in Fig. 4.2). This results in four edge maps as shown in Fig. 4.4. Below, we describe the 2D edge detection algorithm in detail.

Gaussian smoothing is first applied to each directional image for noise suppression. Then, gradients along $x$ and $y$ direction, $g_x$ and $g_y$, are computed at each pixel using Sobel operators. With $g_x$ and $g_y$ we compute the gradient magnitude $g = \sqrt{g_x^2 + g_y^2}$. Edge direction $dir$ at each pixel is determined by slope angle $angle = arctan(g_y/g_x)$: if $angle \in [0, \frac{\pi}{8}] \cup [\frac{7\pi}{8}, \pi]$, $dir$ is horizontal; if $angle \in (\frac{\pi}{8}, \frac{3\pi}{8})$, $dir$ is positive diagonal; if $angle \in [\frac{3\pi}{8}, \frac{5\pi}{8}]$, $dir$ is vertical; and if $angle \in (\frac{5\pi}{8}, \frac{7\pi}{8})$, $dir$ is negative diagonal.

Canny edge detection then performs non-maximum suppression to obtain a thin

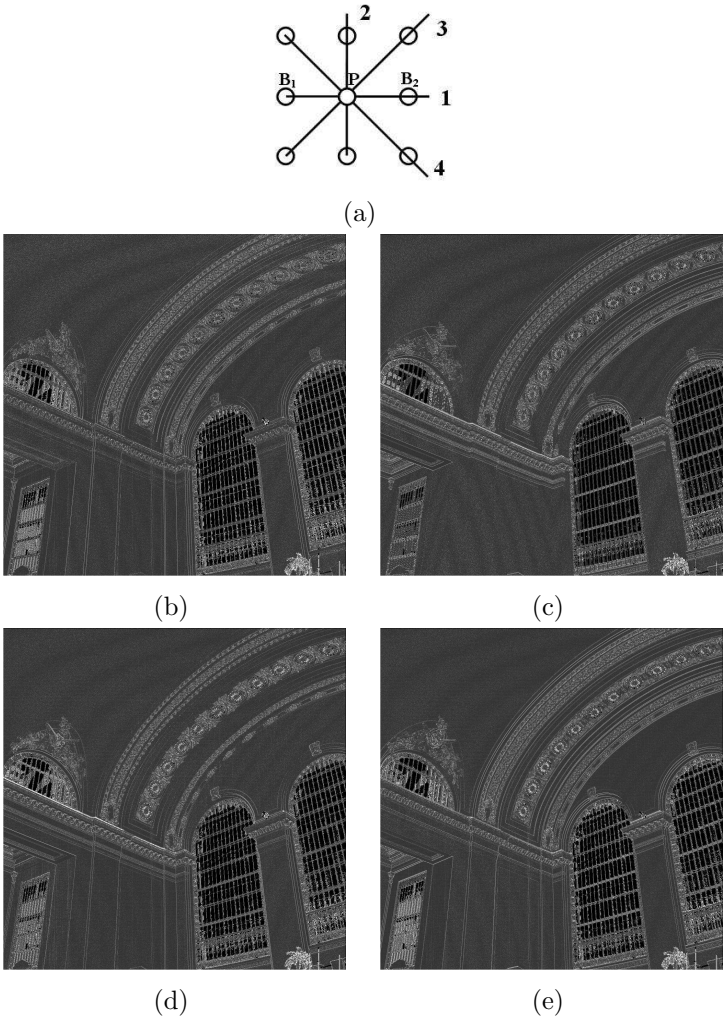



Figure 4.2: Directional variation images formed from Fig. 4.1. (a) Four grid directions: 1-Horizontal, 2-Vertical, 3-Positive Diagonal, 4-Negative Diagonal. $B_1$ and $B_2$ are $P$'s neighbors along direction 1. (b)–(e) Directional variation images of Fig. 4.1 in the above four directions respectively. Brighter points have larger variation values, and darker points have smaller variation values (see text in Section 4.1.1). The intensities are slightly scaled for best display. Note that each type of directional variation image responds strongest to edges perpendicular to the direction of computing variation.

edge, followed by hysteresis thresholding, outputing a specified amount of edge points. In our algorithm, we reverse the order of these two procedures, due to the following considerations: 1) Instead of deciding the number of edge points by ratio, we aim at finding all the points whose neighborhoods contain more significant change than expected in the high resolution range scans of large-scale urban scenes. 2) Applying thresholding in the last step causes discontinuous edges. We, however, prefer to keep edges as continuous as possible, for the purpose of accurate edge linking and circle fitting in the later phases. As such, our algorithm uses a generous threshold 0.35 (allowing the angle change of $5^o$ [2] in any direction) for edge magnitude, followed by non-maximum suppression.

At this point, our algorithm has detected all roof and step edges. However, occlusion edges need to be identified and only the foreground edges should be kept in order to reflect the true geometry of the scene (similar to the shadows in 2D images). In our algorithm, non-maximum suppression votes off edge points based on magnitude. This happens regardless of the edge point being on a foreground surface or a background surface (Fig. 4.3). We therefore find and remove all background edge points, while adding back those foreground edge points voted off by a background neighbor.

[2]We know that at each non-boundary point, the directional variation value $v \in (0, 1]$, so the difference between any two points $d_v \in (0, 1]$. If we allow an angle change of $5^o \approx 0.0873$, Sobel operator enlarges the difference to at most 4 times, which gives 0.35 for $g_x$ and $g_y$. But since only one of them could reach that maximum value, the threshold of magnitude is thus set to 0.35.

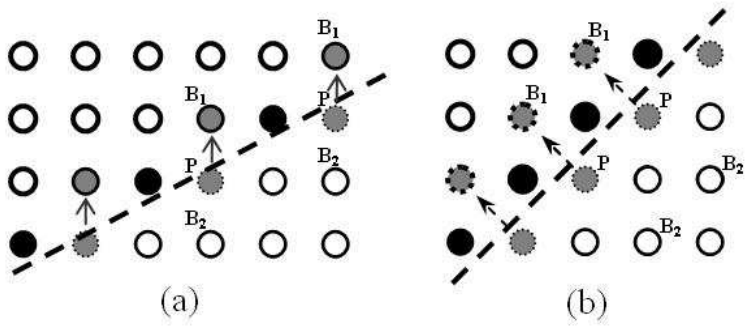



Figure 4.3: Occlusion edge and its correction. Dashed line: real occlusion edge. Black points: foreground edge points. Gray points below the edge: background edge points to be removed from edge map. Gray points above the edge: foreground points to be added as edge points. In (a), they are added into edge; in (b), they stay as non-edge since the edge is already connected and traceable.

To find these points, we map the 2D edge points back to 3D range scan and label a 3D point $P$ if its corresponding pixel $p$ is an edge point. For an edge point $P$, let $B_1$ and $B_2$ be its two neighbors perpendicular to its edge direction.[3] If $Distance(P, B_1) >> Distance(P, B_2)$ and $Depth(B_1) << Depth(P)$, then $B_1$ is a foreground point, and $P$ is a background edge point. In that case $P$ is labeled as non-edge, and $B_1$ is labeled as an edge point (Fig. 4.3(a)). Notice that in the case of Fig. 4.3(b), when $P$ already has two other neighbors on both sides of $B_1$ being edge points, it is unnecessary to add $B_1$ to form a thick edge and consequently be fitted to two lines. By symmetry, edges of other directions can be analyzed and processed similarly.

The next step involves combining four edge maps by taking the union of all edge

[3]In here as well as later in this section, "edge direction at a point" means the 2D edge direction of its corresponding pixel.

points. The combined edge map contains the indices of all 2D edge points, which are also the indices of all 3D edge points in the original range image. These 3D edge points then need to be linked to form continuous edges for feature extraction.

Before edge linking, however, we need to identify a special type of edge points. These are corner points or high curvature points, detected by a Harris corner detector. By testing whether there are more than one principle directions formed by all edge points in its local neighborhood, corner points or high curvature points are identified and removed from the set of edge points. This process is performed due to the following reasons: 1) By removing corner points, we break the connections between edges of different directions, thereby simplifying edge linking and fitting (e.g. the corner points connecting edges in Fig. 4.5(a)). 2) Clusters of high curvature points can be extracted for pattern recognition, as they sometimes show interesting patterns. Although discarded in the current process, later in Section 5.3, we will show our experiments on utilizing a high curvature point cluster, the rosette pattern in Fig. 4.5(a), for region matching on arbitrary geometric shapes.

With the remaining non-corner edge points, isolated points are deleted, and short gaps (1 or 2 pixels) are filled along the local edge direction. Then, continuous edge points are linked by tracing close neighboring points along edge directions. The edge linking utilizes the structured grid on which the range image is represented for resolving neighbors. Only long edges (30 points or more) are kept for later processing.

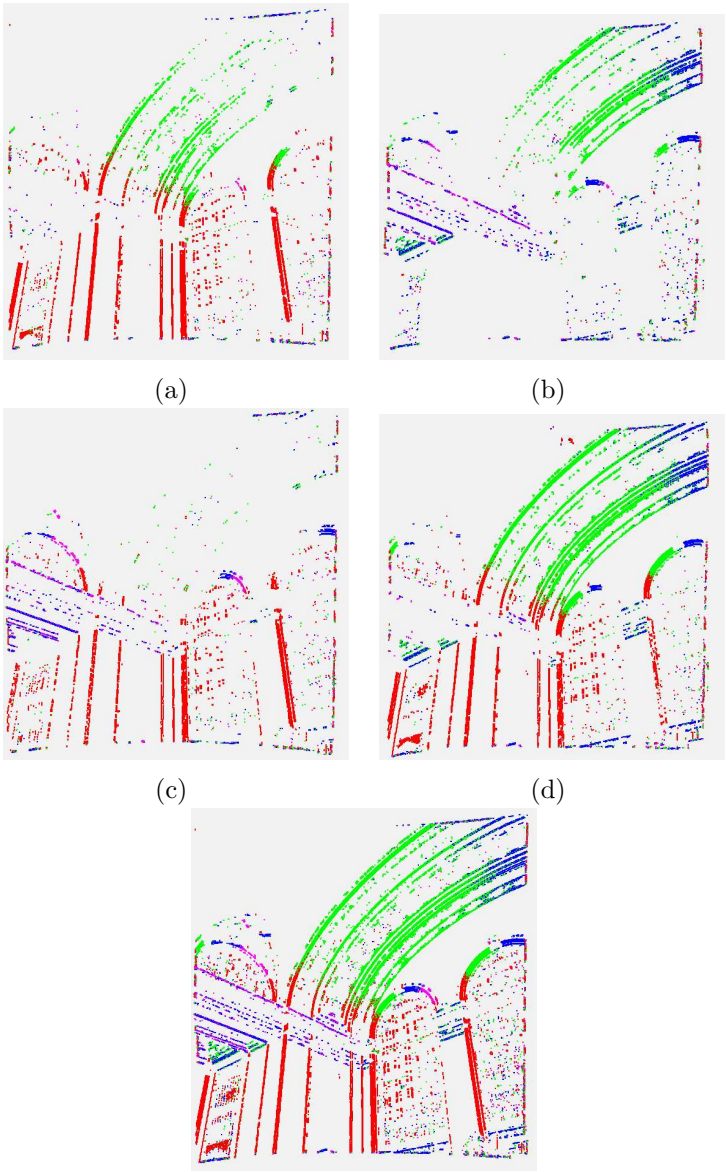
The final combined edge map is shown in Fig. 4.4(e). Fig. 4.5 shows the details in areas with corner points and a circular window.

### 4.1.3 Linear and Circular Feature Extraction

The next step involves fitting linear and circular segments to the extracted linked edges. Each linked edge describes a curve in 3D space. At the current stage, our algorithm extracts linear features and circular features, as these features occur frequently in most urban buildings, and can be identified robustly.

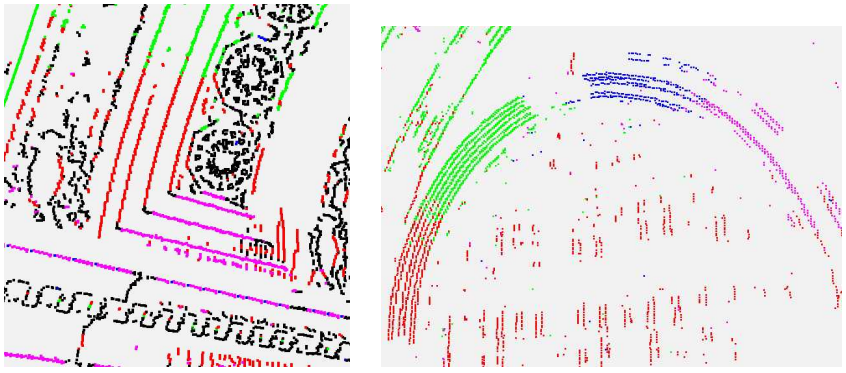
We first analyze the linearity of all the 3D edges. For each linked edge from Fig. 4.4(e), its best-fit line direction $\mathbf{V_{max}}$ and best-fit plane normal $\mathbf{V_{min}}$ are computed. A curve is considered linear if the line fitting error (average distance of all points to the fitted line) is less than a threshold. We set this threshold to be 3cm, which is approximately the average distance between neighboring range points in the range images in our experiments.

In order to identify circular curves from the remaining nonlinear curves, we first identify planar curves and then carry out circle fitting. The average perpendicular distance of all points to the fitted plane is used to discard 3D curves that are non planar (a generous threshold of 0.5m is used). For each of the remaining planar curves, all points are projected onto their fitted plane. After this process, the 3D curve becomes a set of 2D points in the 2D space $\Pi$ of the fitted plane. Circle fitting



(a) (b) (c) (d) (e)

Figure 4.4: Edge points of range images of Fig. 4.1. Note that the color at each point (red/green/yellow/magenta) indicates its edge direction (see text), hence the same point usually has the same color in the four edge images. (a)-(d) Edge points detected from Figs. 4.2(b)-(e) respectively. (e) Combined edge image from (a)-(d).



(a) (b)

Figure 4.5: Zoom-in on edge points in Fig. 4.4(e). (a) Black points are corner points (to be removed from consideration). (b) circular edges along the window frame.

is done in this space.

Taking the common approach of least square fitting[4], we compute the center $(a, b)$ and radius $r$ of the circle by finding an approximate null-vector of an $n \times 4$ design matrix, where $n$ is the number of points on the curve. Consider the circle function $(x-a)^2+(y-b)^2-r^2=0$. It can be written as $x^2+y^2-2ax-2by+a^2+b^2-r^2=0$. Let $(x_i, y_i)$ be the 2D coordinates of all points $p_i (i = 1, ..., n)$ on the curve. Then, the circle equation for all points can be expressed as as a multiplication of the $n \times 4$ matrix $M = [M_1 \;\; M_2 \;\; ... \;\; M_n]^T$ where $M_i = [x_i^2+y_i^2 \;\; -2x_i \;\; -2y_i \;\; 1]$ (for $i = 1, ..., n$), with unknown vector $[1 \;\; a \;\; b \;\; a^2+b^2-r^2]^T$. The null-vector of the design matrix,

[4]3D hough transform method is not applicable here, since the radii of existing circles are unknown and may vary within a large range.

computed by SVD, provides the solution. The circle fitting error is then computed as

$$c_{err} = \sqrt{\frac{\Sigma_i^n (distance(p_i - center) - r)^2}{n}}.$$

The ratio ($\frac{c_{err}}{r}$) must fall below a threshold (0.02) to verify that the planar 3D curve is a circular arc. Finally, the center of the fitted circle is converted back from $\Pi$ to the 3D space. We now have three parameters to represent each oriented 3D circle: 3D center point, radius, and plane normal. These parameters extracted from all 3D circles are later utilized for range registration in Section 5.2.

### 4.1.4 Segmentation Results

Fig. 4.6 shows the segmentation results of two range images of the Grand Central Station. Figs. 4.6(a)(b) are two overlapping range images, both capturing a corner of the interior hall. The line segments extracted from these two range images are shown in Figs. 4.6(c)(d), and the extracted circles are shown in Figs. 4.6(e)(f), respectively. In Figs. 4.6(e) and (f), only circles with radii between 3.0m and 5.0m are displayed. These are the ones most useful for circle-based registration on these range images. During the program run, we detected all circles with radii between 2.0m and 20.0m. Moreover, when this algorithm is applied to different data sets, these thresholds are set based on the observed scale of the range data, or based on user preference, if the

approximate sizes of useful circular features are known.

Fig. 4.7 shows a few more scans of the Grand Central Station, including its front wall ((a)(c)) and the side wall ((e)(g)), together with circles extracted from them.

## 4.2 Region-Based Segmentation

Edge-based registration places an emphasis on extracting edges, but provides little information about the regions enclosed by or neighboring to these edges. Region-based segmentation, however, identifies smooth or continuous regions, and then partitions a range image into a composition of various regions. This composition usually corresponds to meaningful facades and objects in the scanned scene, and can be further utilized for object extraction and scene understanding.

Our novel region-based segmentation categorizes regions into three types: planar, smooth non-planar, and non-smooth connected segments. In the context of large scale urban structures, planar and smooth non-planar regions often represent facades, floors, or ceilings, and non-smooth connected segments represent objects such as windows, furniture, or decorations.

Our algorithm extracts the above three types of regions accurately and efficiently. It consists of the following steps: a) Normal computation of each 3D point (Section 4.2.1), b) Extraction of planar and smooth non-planar regions (Section 4.2.2), and c) Extraction of non-smooth connected regions (Section 4.2.3).

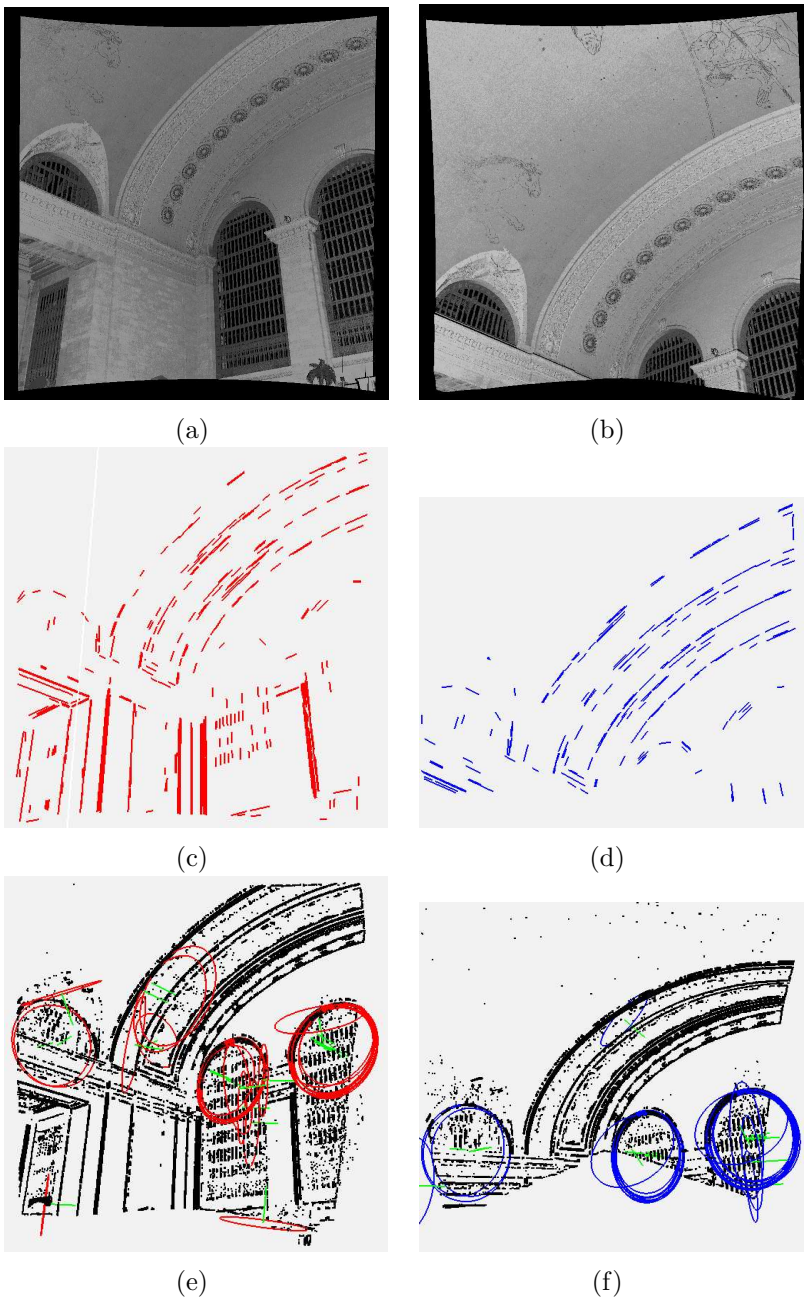



Figure 4.6: (a) An interior range image of the Grand Central Station (Fig. 4.1). (b) Another range image that overlaps (a). (c) Line segments extracted from (a). (d) Line segments extracted from (b). (e) Circles extracted from (a). (f) Circles extracted from (b). In (e) and (f), black point are the edge points, and all fitted circular arcs are represented with colored full circles, with green lines indicating their normals. Note that three circular windows are detected in both images. The images are rotated to the best angle to observe all circles. Therefore, some of the circles appear as ellipses due to the viewing direction.

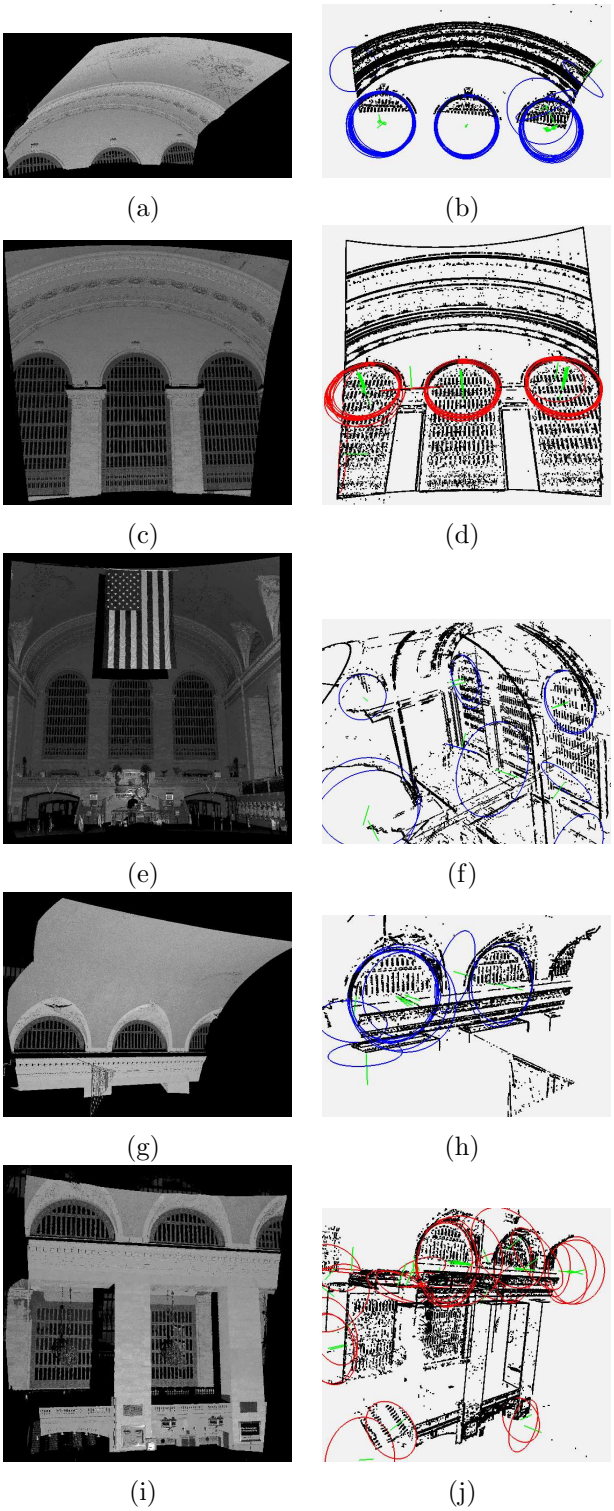



Figure 4.7: More Grand Central Station scans with extracted 3D circles.

### 4.2.1 Normal Computation

The first step of most range image segmentation algorithms is the computation of the surface normal at every 3D range point [Stamos and Allen, 2002] [Bellon and Silva, 2002] [Besl and Jain, 1988] [Gotardo *et al.*, 2004] [Chen and Stamos, 2006]. This is a critical step for precise region extraction, as surface normal is often estimated inaccurately for points near boundaries, leading to possible misclassification of these points. Our algorithm calculates local surface normals for all locally planar points with high accuracy, in order to minimize classification errors and obtain highly accurate region segmentation.

The local surface normal of a range point is usually estimated by fitting a local plane to all the 3D points in its $k \times k$ neighborhood. As mentioned above, simply fitting a local plane within a $k \times k$ neighborhood will result in inaccurate calculation of surface normals close to surface boundaries, or in non-locally planar areas. Below, we first present several types of local $k \times k$ neighborhoods, and then describe how our algorithm handles each case.

Fig. 4.8 shows several cases of point $P$ (the black dot) and its local neighborhoods (all the hollow dots). Without loss of generality, we display $5 \times 5$ neighborhoods. In each sub-figure, the arrow shows the scanning direction. Regions $S, S_1$, and $S_2$ are true surfaces, while $F$ is the fitted local plane at $P$. In all cases except (f) $S, S_1$, and $S_2$ are approximately planar. In (f), $S$ is a general surface. In Figs. 4.8 (a) and (b),

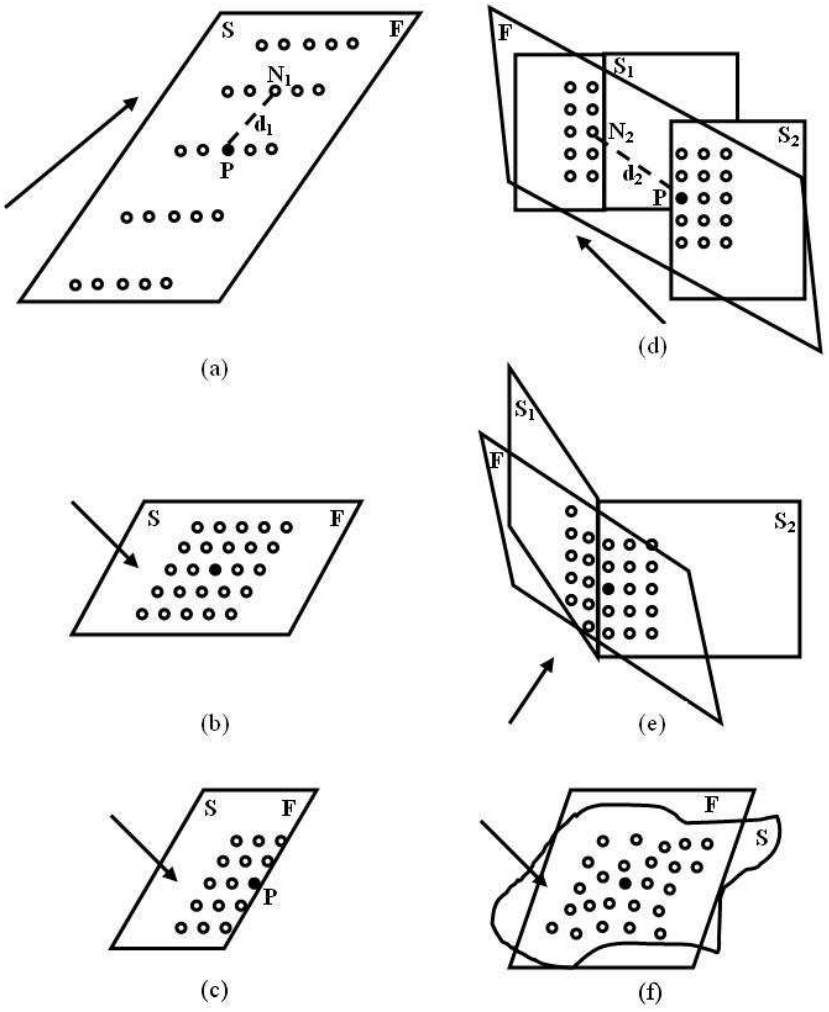



Figure 4.8: Different types of local surface.

$P$ is a locally planar point, but the angle between its local plane and the scanning direction differs in these two cases. In Fig. 4.8 (a), the angle is small. Therefore, some of $P$'s neighbors are farther away than others. In Fig. 4.8 (b), the angle is large, i.e. the viewing direction is almost vertical to $P$'s local plane. Thus, all its neighbors are almost evenly distributed. In Figs. 4.8 (c),(d), and (e), $P$ is an edge point. Fig. 4.8 (c) shows a jump edge with missing neighbors; Fig. 4.8 (d) shows a jump edge between two planes; Fig. 4.8 (e) shows a fold edge. In Fig. 4.8 (f), $P$ is a non-planar point.

Previous methods [Stamos and Allen, 2002] [Bellon and Silva, 2002] [Gotardo *et al.*, 2004] [Chen and Stamos, 2005] set a distance threshold for selecting the neighbors of a point $P$ for surface fitting. The distance threshold is either a pre-selected constant, or is calculated from the point distribution in the range image. Although these approaches correctly calculate the surface normal for most locally planar points, they may fail in the following cases:

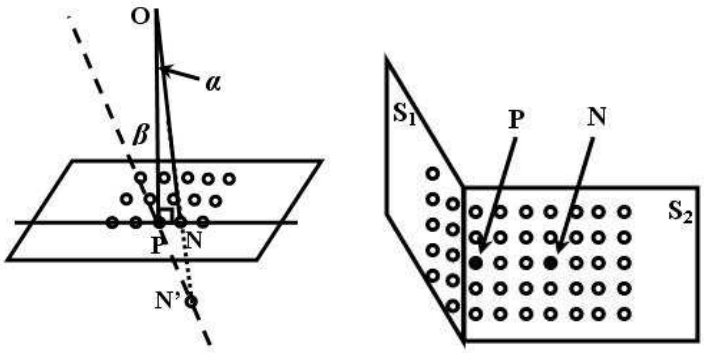
In Fig. 4.8 (a), all of $P$'s neighbors should be used to calculate surface normal, while in Fig. 4.8 (d), only those neighbors located on plane $S_2$ should be used. If both such cases exist in a range image and $d_1 > d_2$, then some surface normal computations will be inaccurate, irrespective of the selected distance threshold. Another difficult case is caused by fold edges (Fig. 4.8 (e)). Neighbors from both $S_1$ and $S_2$ are used to compute surface normal of $P$, resulting in plane $F$. Previous approaches including optimal scale selection [Unnikrishnan *et al.*, 2006] and k-nearest neighbors [Mitra and

Nguyen, 2003] are not capable of computing $P$'s true local plane, $S_2$, in this case. Methods with robust estimators [Lee *et al.*, 1998][Unnikrishnan and Hebert, 2003] are possible solutions, but they have higher time complexity as compared to our method.

In our algorithm, we carry out local plane fitting followed by refining steps, in order to achieve accurate calculation of surface normals for points in all the above cases. During this process, points are categorized into four types: Interior planar points (type 1), wide jump edge points (type 2), narrow jump edge and fold edge points (type 3), and non-planar points (type 4). Interior planar points are used to refine the surface normal calculation for edge points. Below we describe the algorithm in more detail.

(1) First, we consider all existing neighbors in the $k \times k$ neighborhood of every point $P$ and fit a local plane. We set $k$ to be 5 in our experiments. During the plane fitting, the maximum distance between any neighbor point to the fitted plane, $d_{max}$, is used to measure the goodness of fit. By setting a very small threshold $\theta_{fit}$ (in our algorithm $\theta_{fit} = 0.01m$, which is slightly larger than the maximum expected error of our range scanner 6mm), we identify planar points (Figs. 4.8 (a)(b)) and edge points (Fig. 4.8 (c)). If $d_{max} < \theta_{fit}$, then this point $P$ is considered to have a planar local neighborhood, thus labelled as type 1 point.

(2) Second, we recalculate the surface normal for some points (i.e. for points that fall into the case of Fig. 4.8 (d)). More specifically, we consider points which




(a) (b)

Figure 4.9: (a) Distance estimation between neighbors (see text). (b) Refinement of $P$'s surface normal with its neighbor $N$.

lie on relatively wide jump edges. In this step, we discard neighbors of $P$ by using a dynamic threshold that depends on its depth, $depth_P$, and the average angle $\hat{\alpha}$ between neighboring scanning beams in the range image, as explained below.

Assume point $P$'s local surface plane is perpendicular to the scanning direction (see Fig. 4.9(a)). The distance between $P$ and its neighbor $N$ is $|PN| = dist_P = depth_P * tan(\alpha) \doteq depth_P * \alpha$ (see footnote[5]). This value is the expected distance between neighboring points. Note, however, that the scanning direction may not be exactly perpendicular to surface normal. As the angle $\beta$ between surface line $PN$ and scanning direction decreases, the distance $|PN|$ becomes larger. When $\beta = 15°$, as the dashed line $PN'$ shows, the distance $PN' \approx \frac{PN}{sin(\angle PN'N)} \approx \frac{PN}{sin\beta} \approx 4 \cdot PN \doteq 4 \cdot depth_P \cdot \alpha$.

[5]We actually calculate $depth_P * \alpha$ for better performance, because $tan(\alpha) \doteq \alpha$ when $\alpha < 15°$. In our range image data, angle between neighboring scanning beams is at most 0.002°, and in most range images, $\alpha$ is small enough to satisfy $tan(\alpha) \doteq \alpha$.

In order to accommodate for cases where $\beta$ ranges between 15° and 90° (most surfaces in our scans are within this range), we calculate $4 \cdot depth_P \cdot \alpha$ at every point P and use it as a loose distance threshold for its immediate neighbors.

Since the threshold derived above is a very loose one, it can only be used to accurately compute surface normals for points near wide jump edges (Fig. 4.8 (d)). Similarly to step 1, local plane fitting is carried out with neighbors closer than the distance threshold derived above, and the goodness of fitting is measured by comparing the fitting error $d_{max}$ with $\theta_{fit}$. Well-fitted points (i.e. points with $d_{max} < \theta_{fit}$) are categorized as type 2 points (Fig. 4.8(d) with large $d_2$). The processing of other narrow jump edges is described in step 3 below.

(3) This step aims at correcting normal calculations (surface fitting) for narrow jump edges (Fig. 4.8 (d)) as well as fold edges (Fig. 4.8 (e)). The previous two steps identified all planar points that are relatively far from planar region boundaries. Points near boundaries would still have inaccurate surface normals. Let us illustrate the calculation of surface normals of these points using Fig. 4.9(b). $P$ is not identified as planar due to the large plane fitting error. However, it is at most $k$ points away from an identified planar point $N$ that is on the same plane with it because $N$'s surface normal was accurately calculated with its $k \times k$ neighborhood. As such, our method searches around a neighborhood slightly larger than $k \times k$, and locates the best neighbor point $N$ so that the distance from $P$ to $N$'s local plane is minimum.

Specifically, the algorithm considers every point $P$ that is neither of type 1 nor of type 2. $P$'s $P$'s type 1 or type 2 neighbors are selected within its $(k+2) \times (k+2)$ neighborhood, and the distances of $P$ to all their local planes are compared. Also, the distance between $P$ and the center of its own fitted local plane is considered (if $P$ has been fitted twice in steps 1 and 2, the one with smaller fitting error is selected). Among these point-to-plane distances, the smallest one is selected. If the distance of $P$ to the local plane of its neighbor $N$ is the smallest, and this distance is smaller than threshold $\theta_{fit}$, we then consider point $P$ to be belonging to $N$'s local plane. In that case, $P$'s local surface normal is set to be the same as $N$'s surface normal. The planar points identified in this step are categorized to be type 3 points (Fig. 4.8(d) with small $d_2$ and Fig. 4.8(e)).

(4) All the remaining points are categorized as type 4 points (non-planar points as in Fig. 4.8(f)), and their surface normal is calculated from either step 1 or step 2, choosing the one that produces the smaller surface fitting error.

### 4.2.2 Extraction of Planar and Smooth Non-planar Regions

After the calculation of local surface normals, a region growing procedure is carried out. This procedure involves clustering neighboring points using the following metric: two neighboring points $P$ and $N$ are a part of the same region if a) they are spatially

close, b) they have similar local surface normals, and c) they have similar laser intensity. In previous algorithms, the distance threshold is static - either pre-selected, or determined empirically. This, as we explained earlier, may cause inaccurate segmentations. In our algorithm, we replace the point-to-point distance measurement with a point-to-plane distance measurement. In particular, we determine if a point $P$'s neighbor $N$ belongs to the same local plane as $P$ by checking the distance from $N$ to $P$'s local plane instead of distance between $P$ and $N$. This is advantageous because point-to-plane distance is less sensitive to surface orientation. Therefore, it is reasonable to set a constant threshold on point-to-plane distance.

At the initial state of region growing, all points are unlabelled. Starting from the upper-left corner of the range image's grid, we select the next unlabelled point as a seed to grow a new region. Each time a new region is formed, points in this new region are labelled with an integer as their region label. The first region is labelled as region 0, and the succeeding formed regions are labelled with $1, 2, 3, ..., n$. Assuming the seed for a new region is point $P$, we test whether each neighbor point, $N$, belongs to $P$'s local plane. If it does, $N$ is labelled with $P$'s region label, and $N$'s neighbors are further compared with $N$ to continue region growing. This recursive procedure continues until no more neighbor points are found to belong to this region. When comparing $N$ with $P$, three thresholds are used: a) the distance from $N$ to $P$'s local plane should be smaller than a threshold $\theta_{point2plane}$; b) the intensity difference

between $N$ and $P$ should be smaller than a threshold $\theta_{intensity}$; and c) the surface normal difference between $P$'s and $N$'s local planes should be smaller than an angle threshold $\theta_{angle}$. In our algorithm, these three thresholds are $0.015m$, 0.01, and $2^\circ$ respectively. Note that $\theta_{intensity}$ and $\theta_{angle}$ are not sensitive to surface orientation. Varying the values for these thresholds leads to coarser or finer segmentation, and the values in our algorithm are selected empirically.

This algorithm extracts locally planar regions, i.e. either planar or smooth non-planar regions (see results). We can distinguish between these two types by performing a planar fit. Planar regions will generate small fitting errors. Finally, we can associate various descriptors with these regions (e.g. area, 3D bounding box, orientation), which can be used in applications of feature matching and shape analysis.

### 4.2.3 Extraction of Non-smooth Regions

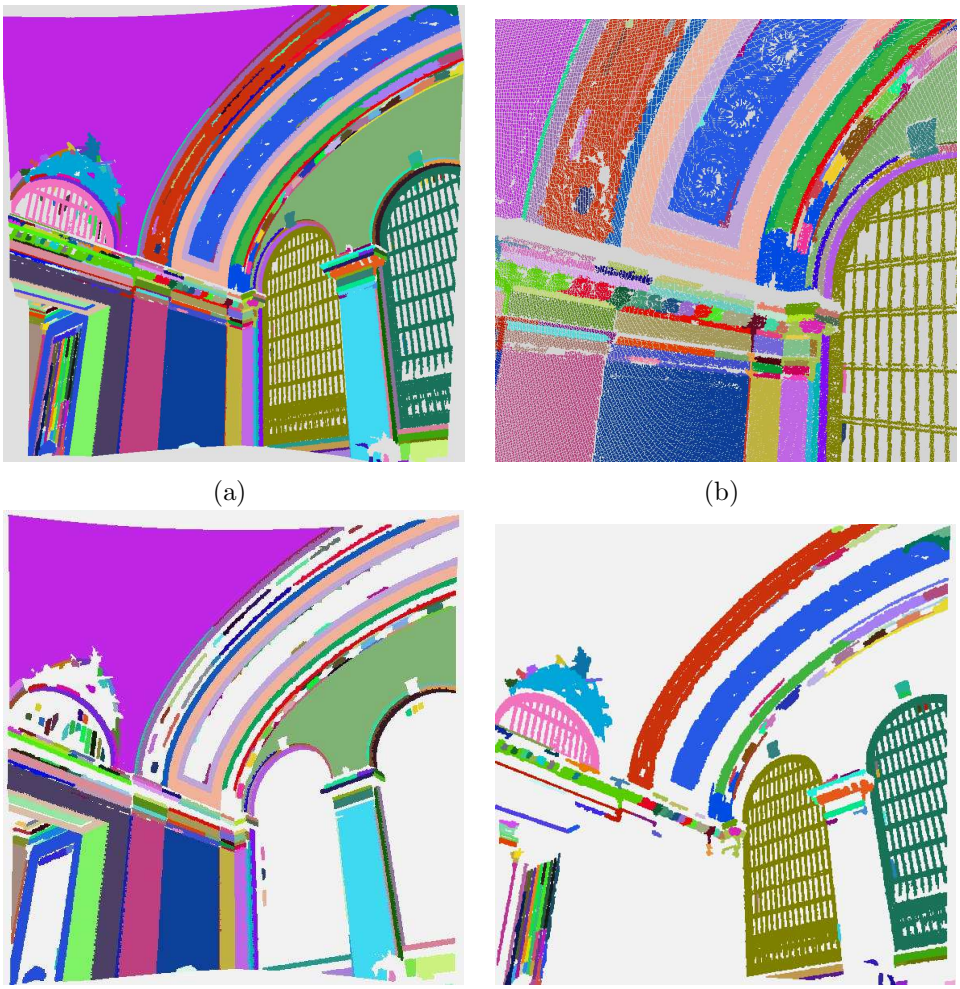
After the previous previous region growing procedure calculates major planar and smoothly varying surfaces (Fig. 4.10(c)), based on the remaining unlabelled points, a second round of point clustering is carried out to cluster non-smooth regions. In urban scenes, after large smooth surfaces are identified, the remaining non-smooth regions are usually objects, such as windows, lights, and furniture. In range images, these objects have varying surface normals, and they usually form point clusters separated from each other by smooth background regions. Therefore, the clustering criterion

of non-smooth region points is defined with higher emphasis on spatial closeness and lower emphasis on surface normal similarity. The process for clustering non-smooth regions is similar to that of growing planar regions. The difference is in the details of comparing neighboring points.

Due to the reason described above, we compare intensity, point-to-point distance, and angle between surface normals of two neighboring points with thresholds $\theta_{intensity}$, $\theta_{point2point}$, and $\theta'_{angle}$. The value for $\theta_{point2point}$ is $dist_P * r$, $r$ being an empirically calculated constant larger than 1. The threshold $\theta'_{angle}$ is a large value (in our algorithm it is 65°) to avoid growing regions with the neighbors having large angles between their surface normals. We assume that a large angle between surface normals will appear at boundaries of semantically different regions.

### 4.2.4 Segmentation Results

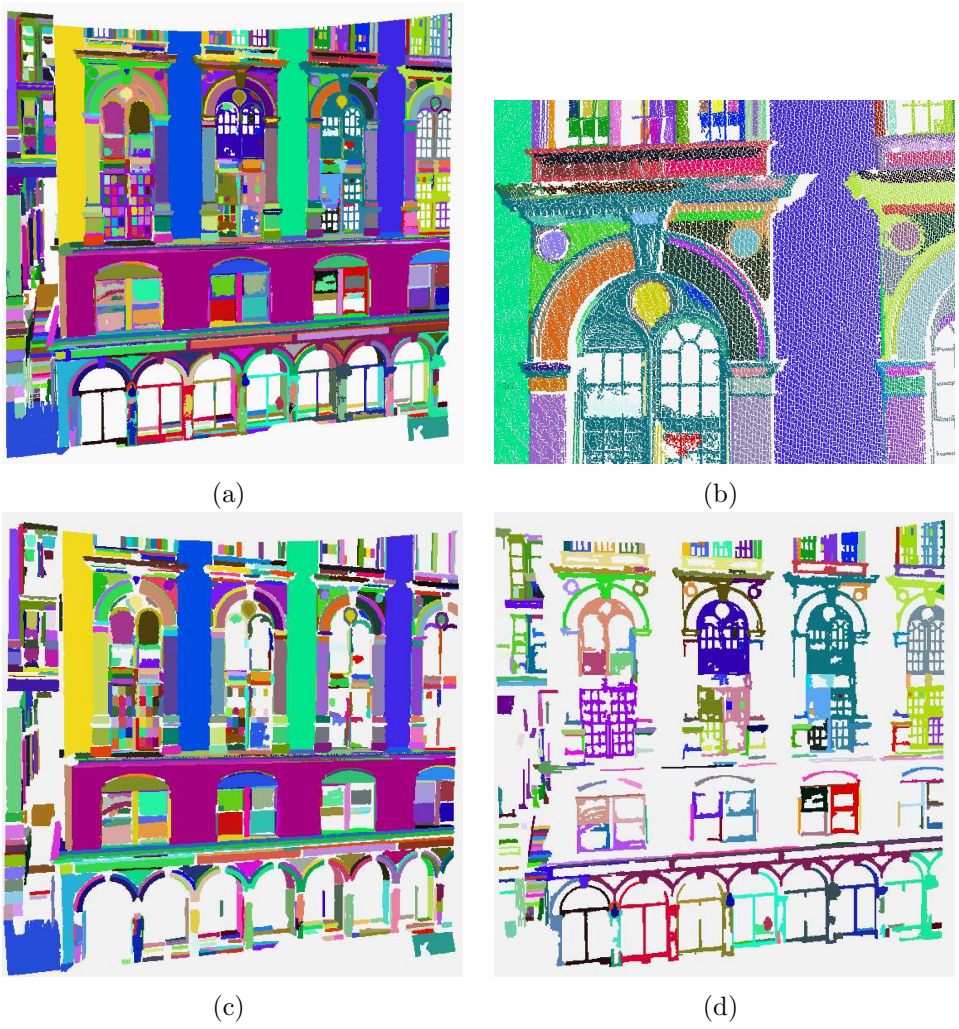
Fig. 4.10 shows the segmentation result of a range scan of the Grand Central Station (Fig. 4.1). Each segmented region is assigned to a random color computed with its label. In Fig. 4.10 (d), some regions seem to be smooth, but are clearly non-smooth when zoomed in. e.g., the long curved stripe is in fact a rosette pattern as shown in Fig. 4.10 (b). Fig. 4.11 shows the segmentation result of a range scan of the Cooper Union building. These two segmented scans, together with other segmented scans of these two buildings, will be utilized for model generation (Chapter 6).



(a) (b)

(c) (d)

Figure 4.10: Segmentation results. (a) Segmented regions as colored patches. (b) Details at region boundaries. (c) Planar and smooth regions. (d) Non-smooth regions.



(a) (b)

(c) (d)

Figure 4.11: Segmentation results. (a) Segmented regions as colored patches. (b) Details at region boundaries. (c) Planar and smooth regions. (d) Non-smooth regions.

# Chapter 5

# Range Registration

As explained in Chapter 3, a single range scan only captures surface points visible from one viewing direction. Multiple scans from different positions and viewpoints are therefore necessary to capture a more complete surface of the modeling scene. The multiple range images generated this way are defined in different local coordinate systems. Therefore, in order to combine all the range images to form a complete 3D model, their transformation into a common reference system is needed. This procedure is called range registration.

Precise registration is essential for building accurate models. As introduced in Chapter 2, previous approaches of manual registration and point-based ICP algorithm are usually very time consuming, especially for the large scale urban structures

with complex geometric features. Our algorithms are specially developed for achieving a more efficient and more intelligent registration system, utilizing these complex geometric features.

From the previous process of range segmentation, geometric features of 3D lines and 3D circular edges are automatically extracted. We are also utilizing lines and planes extracted by the method of [Stamos and Leordeanu, 2003]. In the following sections, we introduce two registration algorithms, based on the matching of linear features and circular features, respectively. We then extend our discussion to matching arbitrary shapes, and explain the algorithm for matching them with statistical shape descriptors.

## 5.1 Linear Feature-Based Registration

Planar and linear features commonly appear in many large scale urban structures. These features provide us with a possible solution to the registration problem: finding matching lines between each pair of overlapping range images to determine transformation. After all the overlapping range image pairs are registered, we can combine them by transformation composition.

Each range image may contain a large number of linear features. When matching line pairs between two range images, multiple candidate transformations might be produced. One heuristic for choosing the best transformation is to select the one

maximizing the number of matching line pairs.

Stamos and Leordeanu have implemented the above procedure, as described in [Stamos and Leordeanu, 2003]. Their algorithm, however, has two shortcomings. First, for structures containing too many lines, the exhaustive hypothesis-and-test approach in [Stamos and Leordeanu, 2003] would be impractical due to the size of the search space. Second, the heuristic for selecting the best transformation, as described above, may be incorrect when geometric symmetry exists in the scene. In this case, user assistance is needed to determine the correct registration. Our algorithm is thus designed to speed up the line matching process, achieve higher accuracy, and provide graphical user interface for necessary user assistance.

Fig. 5.1 shows the flowchart of this system. The registration system takes a large number of 3D range images as input, and produces a registered range image that contains all input images in one common coordinate system as output. The algorithm is presented in the following steps:

**Rotation Estimation.** Three major directions are extracted from each range image by clustering the linear segments and plane normals, which are extracted in the range segmentation process. A local object-based coordinate system for each range image is constructed, by computing three major orthogonal axes[1]. Then, the rotational transformation between pairs of scans can be computed quickly

[1]The main assumption of this algorithm is that the scene contains at least two major directions (i.e. at least one planar area with lines in two perpendicular directions).

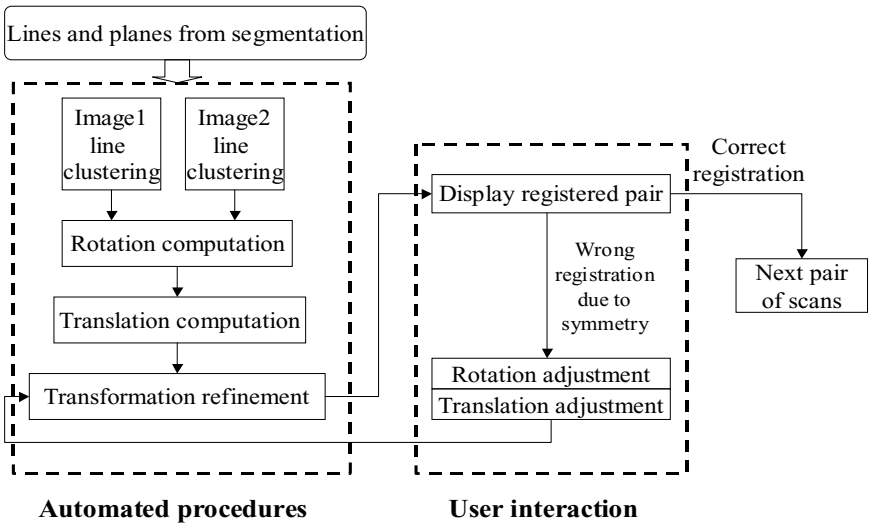



Figure 5.1: Flowchart of range to range registration with user interface.

and accurately by matching these major axes between the scans.

**Translation Estimation.** Candidate translations are estimated by matching linear segments between pairs of rotationally aligned range images. Then, these candidate translations are clustered using an unsupervised nearest-neighbor classification method. The correct translation vector should be in one of the major clusters of translations (i.e. the one appearing most frequently).

**Context-Sensitive User Interface.** The user interface is designed for displaying pair-wise registration results, and for receiving user feedback.

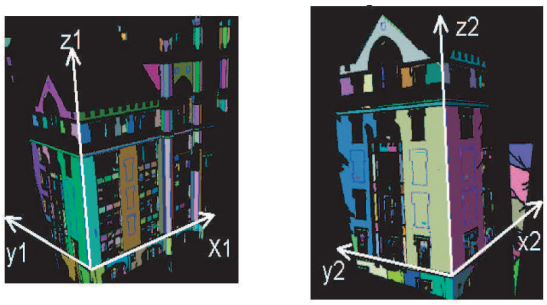



Figure 5.2: Three major scene directions extracted from two segmented range scans of the Thomas Hunter building (different colors correspond to different segmented planes). A correct match between the directions provides a unique solution for the rotational transformation between the scans.

### 5.1.1 Rotation Estimation

Man-made urban scenes are characterized by sets of linear features organized in a major vertical and a number of horizontal directions. After the segmentation phase [Stamos and Allen, 2002], the extracted 3D line directions and plane normals are clustered into three major 3D directions (Fig. 5.2). The clustering procedure groups all the line vectors into clusters (a vector becomes part of the cluster if its angle from the centroid of the cluster is smaller than an angle threshold). In most cases, this procedure extracts three major directions that are perpendicular to each other. In the cases when only two major clusters are found, we can obtain the third major direction as the cross product of the extracted two. Thus, our main assumption is that our 3D scene contains at least two major perpendicular directions. This is a reasonable assumption commonly used in urban scene settings (see [Antone and Teller, 2002]).

After obtaining three axes $\{X_1, Y_1, Z_1\}$ from the left image, and $\{X_2, Y_2, Z_2\}$ from

the right image, all possible values for the rotational matrix R that rotates $(X_1, Y_1, Z_1)$ to $(X, Y, Z)$ are computed. $(X, Y, Z)$ is any permutation of $(\pm X_2, \pm Y_2, \pm Z_2)$. There are 24 such rotation matrices that rotate the left image into the coordinate system of the right one. However, using simple heuristics, the number of candidate rotations can be significantly reduced.

Recall that the position of each 3D point recorded refers to the range scanner's inner coordinate system. Fig. 5.3 shows two local coordinate systems from two scanning positions. The laser generator/receptor is the origin of the local coordinate system. The negative Z axis points towards the 3D scene during scanning. Let us consider how the coordinate system changes from one scan to another. Without loss of generality, the right image is chosen as the pivot image, and all the point coordinates in the left image are transformed into the right image's coordinate system. If the rotation matrix is:

$$R = \begin{bmatrix} R_{00} & R_{01} & R_{02} \\ R_{10} & R_{11} & R_{12} \\ R_{20} & R_{21} & R_{22} \end{bmatrix},$$

then the unit vector $[0, 1, 0]^T$ representing $Y_1$ axis would be transformed into a unit-vector $[R_{10}, R_{11}, R_{12}]^T$ in the right image. In other words, $R_{10}$, $R_{11}$, and $R_{12}$ are the projections of $Y_1$ onto the $X$, $Y$ and $Z$ axes. During the scanning process, the Y-axis does not change dramatically between horizontally neighboring scans – with at most $45^o$ tilt-angle. Therefore, the $R_{11}$, the projection from $Y_1$ to $Y$, is usually within the

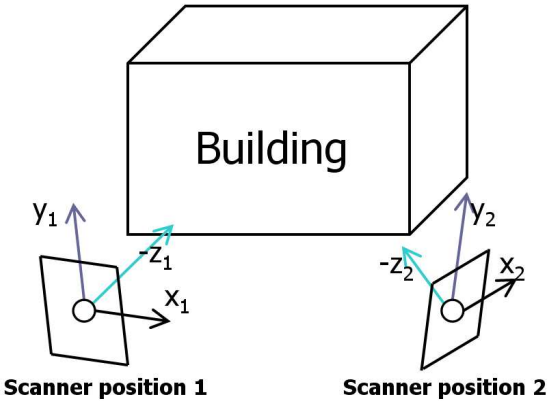



Figure 5.3: Range scanner's local coordinate systems at two different viewpoints.

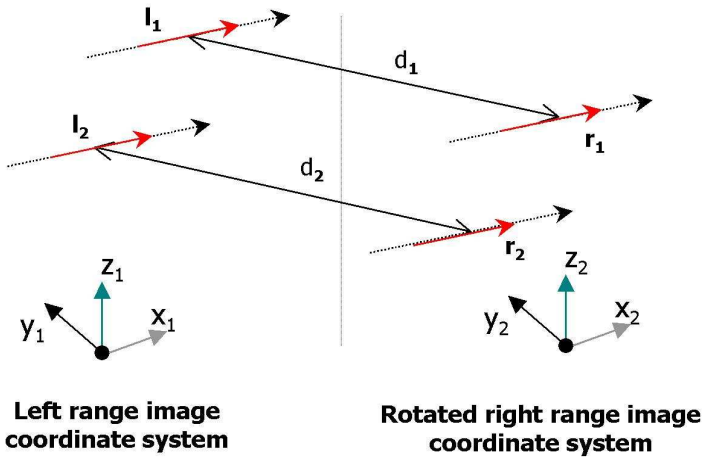
range between 0.7 ($\approx \arccos 45^o$) and 1. Similarly, the horizontally successive images are likely to be from close-by viewpoints. therefore, $R_{00}$ and $R_{22}$ are expected to have positive values, as $R_{00}$ is the projection of $X_1$ to $X$, and $R_{22}$ is the projection of $Z_1$ to $Z$. With these restrictions, the number of candidates for rotation becomes fewer than 5, and in some cases, 2 or 3. Then, with the assumption that successive images are close to each other, we order these candidate rotation matrices by the sum of the diagonal elements, $R_{00} + R_{11} + R_{22}$, and choose the one with the largest sum as the rotation matrix. The other possible rotations are kept as candidates. Note that these assumptions can be relaxed without affecting the actual outcome, since we can choose from the candidate rotations the correct one. The described heuristic is used in order to speed up our algorithm.

### 5.1.2 Translation Estimation

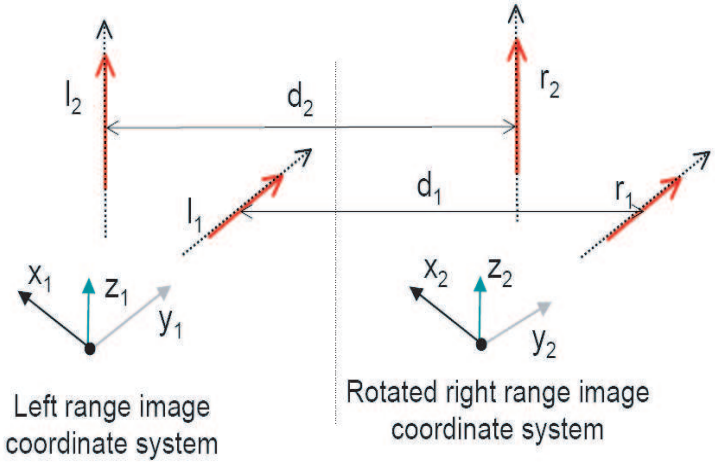
Once the rotation has been automatically computed (see Section 5.1.1), or manually selected (see Section 5.1.3), the axes of the two local coordinate systems of the two scans can be aligned. The next step is the estimation of the translational vector between the two range images. We search for matching pairs of 3D linear segments between the two scans, since two correctly matched pairs provide a unique solution for the translation. At a preprocessing stage, the distance between each pair of parallel linear segments is computed. This distance is the vector that connects the midpoints of two segments (Fig. 5.4). From every two lines in the left image ($l1, l2$) and two lines in the right image ($r1, r2$), a candidate translation is computed if and only if:

1. All four lines are parallel to each other, and the distance between l1 and r1 is equal to (within a length and angle threshold) the distance between l2 and r2 (Fig. 5.4(a)). In this case, the average of two distances is recorded as a candidate translation.

2. Lines l1, r1 are parallel to each other, and lines l2, r2 are parallel to each other, but lines l1, l2 are not parallel. In addition, the distance between l1 and r1 is equal to (within a length and angle threshold) the distance between l2 and r2 (Fig. 5.4(b)). In this case, an exact translation can be computed by the solution of an over-constraint linear system as explained in [Stamos, 2002].




(a)




(b)

Figure 5.4: Two pairs of matched lines. The vectors connecting the midpoints of the matched segments provide approximate translation between the two scans. (a) All four lines parallel to a same axis, $X$ axis in this illustration. (b) Two line pairs parallel to different axes.

The computed candidate translations are then clustered into groups of translations that are close to each other within certain thresholds of length and direction. Intuitively, the correct translation is the one that occurs most frequently. This is the one that defines the largest cluster of candidate translations. However, in order to take into account measurement noise and scene symmetry, we consider the N (N=10) largest clusters of candidate translations. The centroids of these N clusters are considered as our final candidate translations. Finally, out of these N centroids, the one that maximizes the number of line matches between the two scans is returned as the final translation vector[2].

The above automated procedure computes a transformation between any pair of images that overlap. The registered image pair is then displayed in the user interface (Section 5.1.3). The registration obtained after this stage is very accurate, but not satisfactory enough for photo-realistic 3D modeling. The inaccuracy is a result of several factors: a) The segmentation phase introduces some errors in extracted line directions and lengths, and b) The clustering methods for rotation estimation and translation estimation also introduce errors. In the clustering of 3D lines, the centroid of each cluster is selected as the representative major direction. Also, in the

[2]The number of lines that match assuming a rotational matrix and translational vector can be computed after both scans are placed on the same coordinate system. See [Stamos and Leordeanu, 2003].

clustering of candidate translations, the centroid of each cluster is selected as the representative translation. That is why, in order to minimize the registration error, the ICP algorithm needs to be applied as a post-processing step. Experimental results in Section 5.1.4 show that the automated line-based registration procedure calculates a very close transformation, and the ICP optimization is completed within minutes, with the registration error decreased to just a few millimeters.

### 5.1.3 Context-sensitive User Interface

In order to visualize the procedure of registration, as well as to allow users to correct wrong registrations due to 3D scene symmetry, a context-sensitive user interface has been developed. For each pair of overlapping scans, the system reads the segmented planar areas and linear segments. The efficient range-range registration algorithm described in the previous sections is executed first. After a few seconds, the result is displayed (Fig. 5.5). If the user is satisfied with the result, the program will proceed with the next pair of scans. If, on the other hand, there is a mistake, the system displays the following options:

(1) If the initial rotational calculation was wrong due to an erroneous match of axes, the user is presented with a set of possible orientations (Fig. 5.6). The user can select the correct orientation. The system then recalculates the translation (see Section 5.1.2), and the user is asked to verify the result.

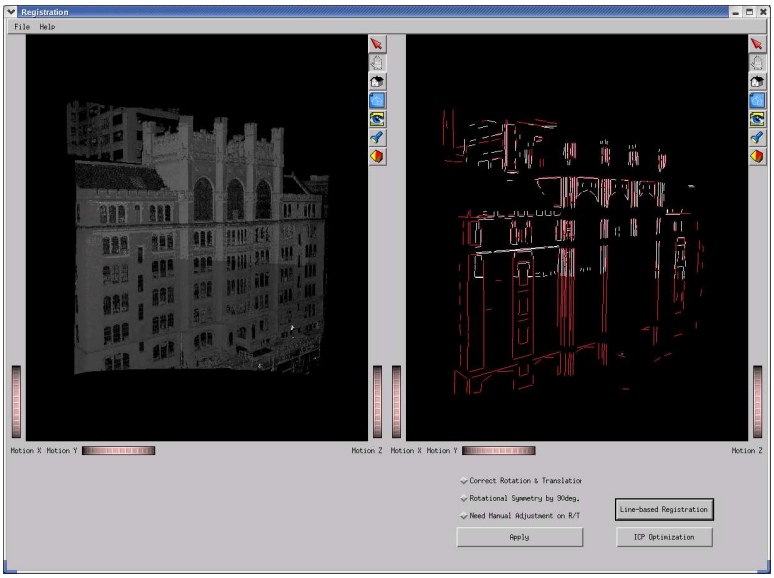



Figure 5.5: Overview of the user interface. Two automatically aligned range scans are shown. Left window: raw range scans. Right window: the same scans abstracted as linear segments (different colors are used for different scans).

(2) If the rotational calculation was correct but the result is still wrong, then the user may choose to invoke a more expensive and complementary range-range registration algorithm described in [Stamos and Leordeanu, 2003].

(3) If no automated algorithm provides a correct result, then the user needs to manually fix the resulted transformation. Note that this case can occur due to symmetry of the acquired 3D scene. Fig. 5.7 shows the screen that the user sees.

(4) After the user manually corrects the transformation, the refinement procedure that searches for matching features between all lines can be invoked.

We call this user-interface context-sensitive because the user can translate or rotate the 3D scans only among the major axes that form the object's local coordinate

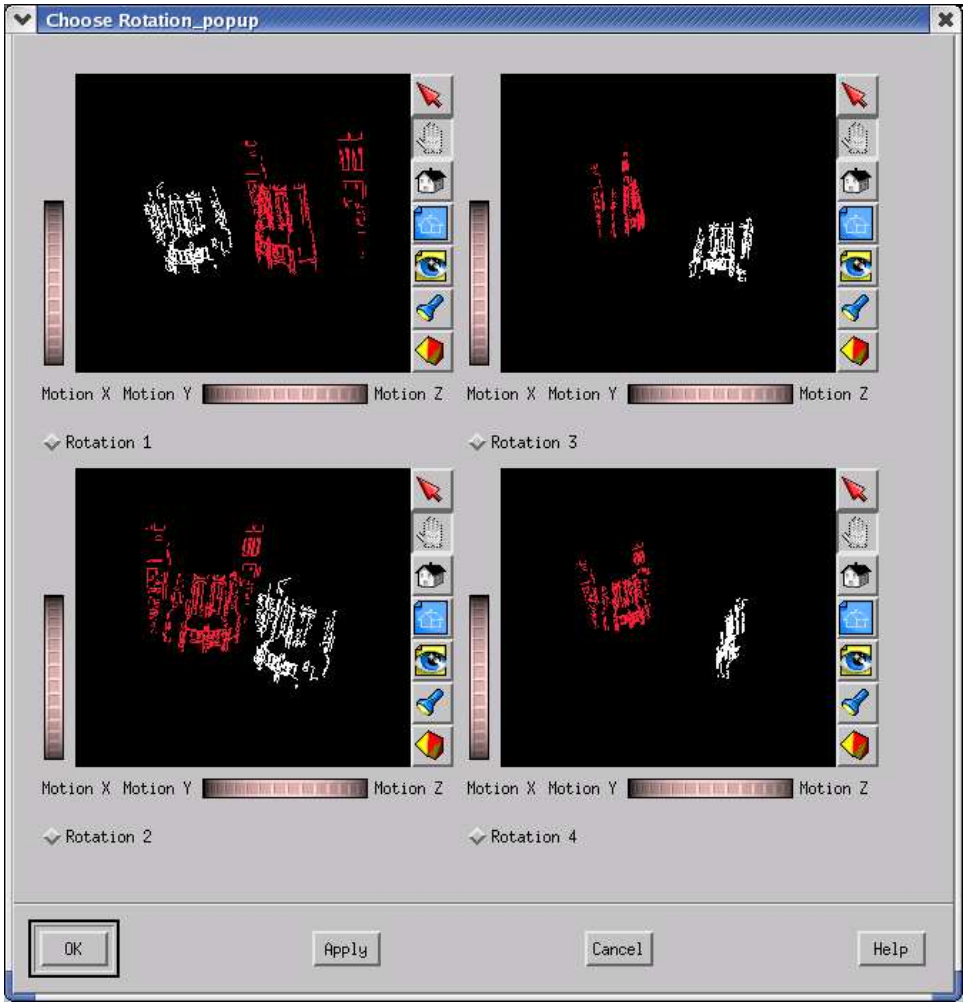



Figure 5.6: A set of possible orientations between the two coordinate systems is presented to the user to choose from. In this example, the rotation in the upper left corner corresponds to the correct result.

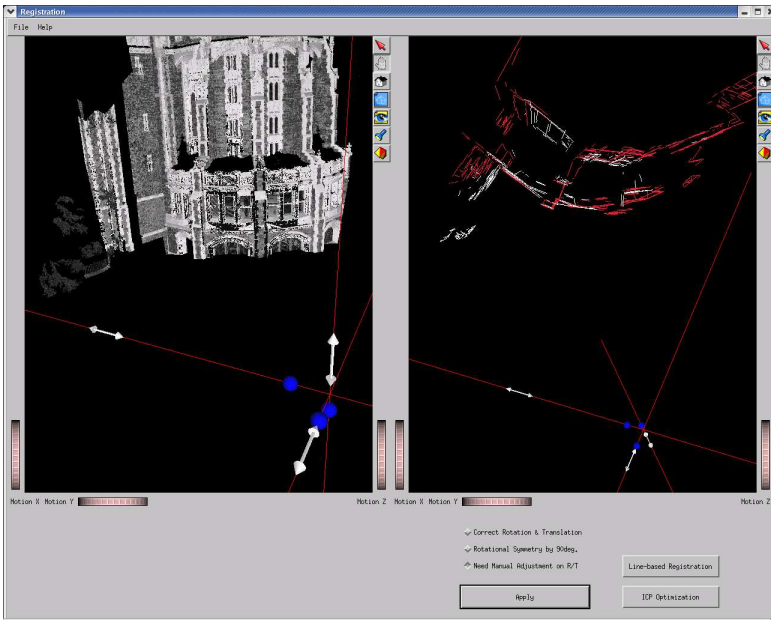



Figure 5.7: The user can manually translate or rotate one scan with respect to the other. This task is made much simpler due to the fact that the user can translate along or rotate about the major orientations of the 3D scene.

system, as shown in Fig. 5.7. The three axes of the right image are displayed as red lines. Along each axis, there is a translation dragger, and a rotation ball. By dragging each dragger, the translation in one direction is adjusted independently, and thus overlapping lines and points can be easily adjusted to the best accuracy; the rotation ball is used to adjust rotation around each axis: by dragging the ball along that axis, its translation is transformed into a rotation around that axis by the corresponding angle, which is applied to the left image. By alternatively adjusting the rotation and translation, the manual registration becomes a lot easier and more accurate than other methods of alignment, such as picking three corresponding points from both images, or translating the two scans along axes that are not related to the

geometry of the scene.

### 5.1.4 Experimental Results

We tested the semi-automatic registration system on both exterior scenes and interior scenes. The exterior scenes are three urban structures of different styles. The Thomas Hunter building (Hunter College of CUNY) is a rectangular building with flat side walls. The Shepard Hall building (City College of CUNY) has a more complicated architecture that resembles a Gothic cathedral. The Cooper Union building has four rectangular facades, but the angles between facades are not all straight angles. The interior scenes contain the Great Hall, a music hall inside the Shepard Hall building, and the Grand Central Station, a land mark building in New York City. The number of points in any of these scans is between 500,000 and 1 million, with an accuracy of 6mm per point. As a criterion of registration performance, we record the number of matching line pairs as computed in Section 5.1.2 (Fig. 5.8), and we calculate the average distance between matching planes.

With the Thomas Hunter building data, we registered 14 range images by applying 15 pair-wise registrations. Among these pairs, 13 pairs were correctly registered with the automated routine, followed by ICP optimization, and 2 pairs required the user to adjust the translation and rotation before a correct registration was obtained. The time for each automated registration is displayed in Table 5.1 (on the average 20

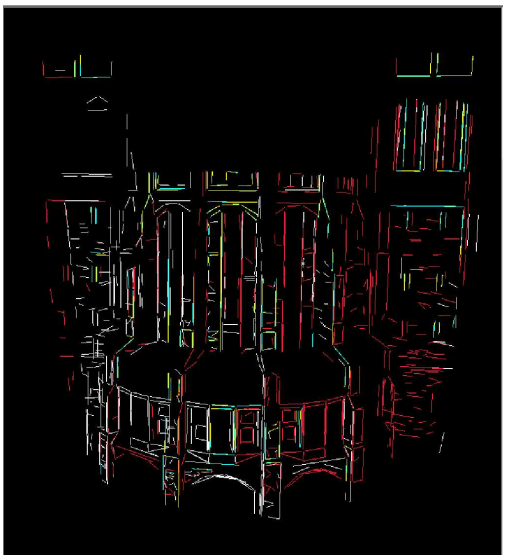


Figure 5.8: Matching lines between two scans of the Shepard Hall. White/red lines are border lines, and yellow/blue lines are the matching lines from two images respectively.

seconds per pair - 2GHz Xeon Processor - 2GB RAM). Table 5.1 also shows the average distance between matched planes[3] of registered pairs of scans, as well as how much the ICP optimization further improved the accuracy of registration. The average error over all pairs of scans decreases from 21.17 mm (before ICP) to 1.77 mm (after ICP). The final registered lines and point images are shown in Figs. 5.10(a) and (b).

Table 5.2 shows the pair-wise registration time and error measurements for the Shepard Hall (15 pairs shown). Since this building has more delicate geometric features, the segmentation produces a large amount of short line segments in various directions. Nevertheless, the experimental results show that the algorithm is quite

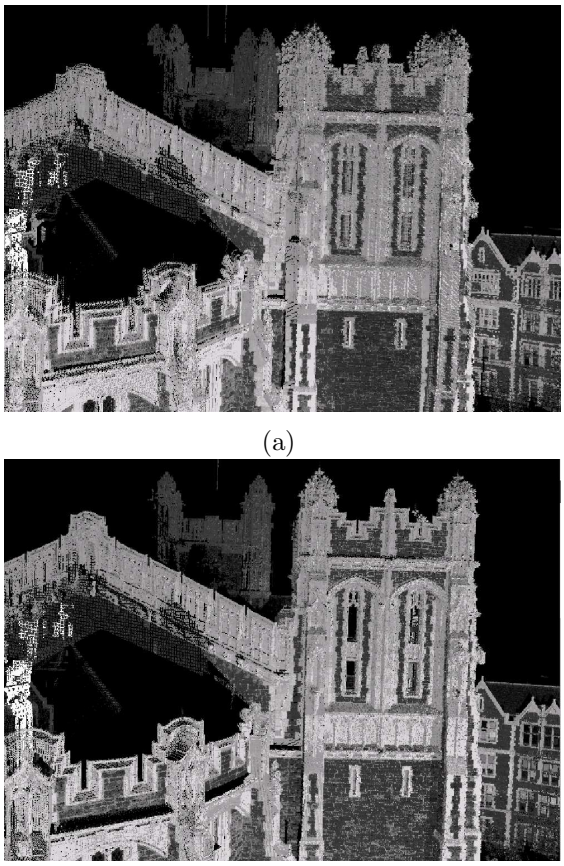
[3]Each extracted 3D line lies on the border of a segmented planar region. Therefore, matched lines between scans dictate matched planar regions.

robust: among the 24 pairs of scans, 9 pairs were automatically registered, 8 pairs needed manual translational adjustment due to scene symmetry, and 7 pairs required a careful user adjustment on rotation. Because of this, the total time of the registration is about an hour (this includes user interaction), although the automated registration of each pair takes less than one minute. When the rotation needs to be manually adjusted, the resulted registration sometimes has quite visible registration errors, as shown in Fig. 5.9(a). In this case, ICP optimization greatly improves registration accuracy (Fig. 5.9(b)). The final registered lines and point images of the Shepard Hall are shown in Figs. 5.10(c) and 5.10(d). The average error over all pairs of scans decreases from 51.72 mm (before ICP) to 3.23 mm (after ICP).

For the interior scene of the Great Hall, we registered 21 range images (Figs. 5.10(e) and (f)). Out of 44 pairs, the automated procedure produced 12 correct results, whereas 18 results needed translational adjustment due to scene symmetry and 13 needed manual adjustment of translation and rotation. The average error over all pairs of scans improved from 17.59 mm (before ICP) to 7.26 mm (after ICP) (Table 5.3). Note that in most cases the number of matching line pairs increases after the ICP optimization (this is expected when the scans are brought closer to each other). In some cases, however, the number of matching lines slightly decreases. This occurs when certain pairs of boundary lines are inaccurately extracted during the segmentation phase, and are identified as matching during the automated registration

phase. Then, as point-based ICP algorithm is applied, these boundary lines are moved away from each other, and are consequently identified as non-matching. Nontheless, the registration quality is clearly improved by the ICP optimization, as seen from the decrease in the average plane error.

In Fig. 5.11, the registered lines and range scans of two other buildings are shown. The registered 20 scans of the Grand Central Station are shown in Fig. 5.11(a) and (b), and the registered 31 scans of the Cooper Union building are shown in Fig. 5.11(c) and (d). Table 5.4 and 5.5 show the registration time and error measurements for some pair-wise registrations of these two buildings. Similar to the registration results of the previous three buildings, these two buildings are also registered efficiently and accurately.



(a)

(b)

Figure 5.9: Close up view of pairwise registration of the Shepard Hall. (a) With automated registration before ICP optimization. Range scans do not align perfectly. (b) After ICP optimization. Result has been significantly improved.

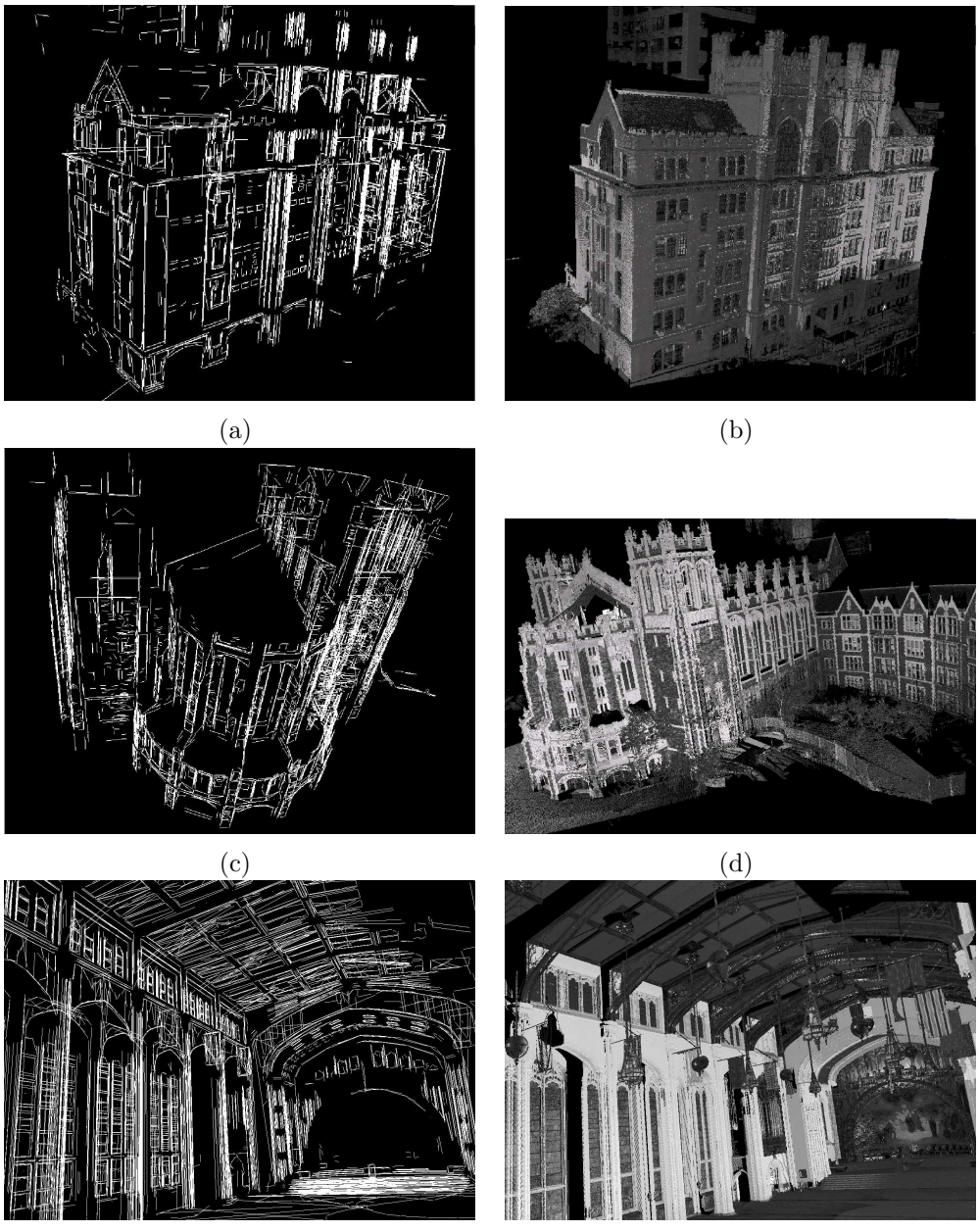
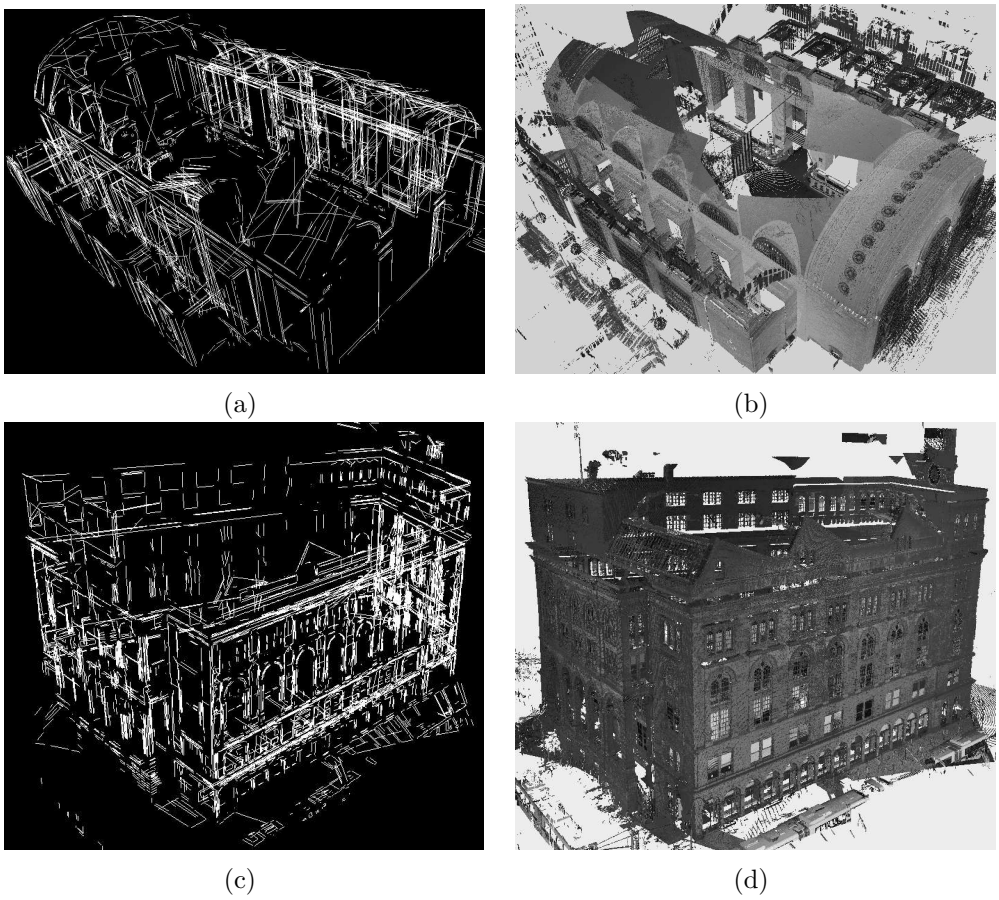


Figure 5.10: Registration results. (a) & (b) The Thomas Hunter building (14 scans, registered in 5 minutes). (c) & (d) The Shepard Hall building (24 scans, registered in 1 hour). (e) & (f) The Great Hall (21 scans, registered in 1.5 hours). Registered lines and range images shown. The lines are extracted from the range segmentation module. The range images correspond to the source scans. The gray values correspond to the returned laser intensity.



(a) (b)

(c) (d)

Figure 5.11: Registration results. (a) & (b) The Grand Central Station (20 scans, registered in 1 hour and 18 minutes). (c) & (d) The Cooper Union building (31 scans, registered in 1 hour and 10 minutes). Registered lines and range images shown. In (b), the viewpoint is specifically selected in order to display the unregistered ceiling area. The range scans of this area are registered and added with the circular feature based registration described later in Section 5.2.

| Pair | Line Pairs | $t$ | Before ICP | | After ICP | |
|---|---|---|---|---|---|---|
| | | | $N$ | $P\_err$ | $N$ | $P\_err$ |
| 1 | 322x229 | 19 | 10 | 33.95 | 26 | 1.80 |
| 2 | 322x275 | 19 | 19 | 5.87 | 17 | 1.68 |
| 3 | 243x205 | 2 | 7 | 54.70 | 11 | 1.72 |
| 4 | 205x292 | 6 | 6 | 5.15 | 7 | 0.97 |
| 5 | 292x279 | 38 | 12 | 15.12 | 36 | 1.87 |
| 6 | 279x275 | 20 | 21 | 7.72 | 20 | 0.91 |
| 7 | 275x304 | 31 | 50 | 14.09 | 32 | 1.03 |
| 8 | 304x180 | 23 | 22 | 22.51 | 22 | 2.98 |
| 9 | 195x180 | 32 | 19 | 3.85 | 33 | 1.02 |
| 10 | 195x249 | 28 | 12 | 15.74 | 27 | 2.04 |
| 11 | 180x249 | 4 | 6 | 50.74 | 18 | 1.60 |
| 12 | 129x249 | 31 | 13 | 5.66 | 31 | 2.50 |
| 13 | 249x137 | 19 | 6 | 24.79 | 26 | 3.16 |
| 14 | 129x137 | 29 | 7 | 19.32 | 37 | 2.11 |
| 15 | 137x332 | 9 | 7 | 38.36 | 9 | 1.23 |

Table 5.1: Experimental results of the Thomas Hunter building registration. $t$: time of automated registration (before ICP optimization) in seconds ; $N$: number of matching lines between the two scans; $P\_err$: average distance between matching segmented planar regions (in mm).

| Pair | Line Pairs | $t$ | Before ICP | | After ICP | |
|---|---|---|---|---|---|---|
| | | | $N$ | $P\_err$ | $N$ | $P\_err$ |
| 1 | 625x211 | 21 | 3 | 52.64 | 8 | 11.94 |
| 2 | 546x539 | 43 | 34 | 78.05 | 88 | 1.80 |
| 3 | 546x638 | 56 | 8 | 42.60 | 9 | 3.20 |
| 4 | 546x211 | 31 | 3 | 97.26 | 42 | 2.64 |
| 5 | 539x638 | 45 | 27 | 85.71 | 31 | 3.51 |
| 6 | 638x642 | 62 | 113 | 4.78 | 112 | 1.95 |
| 7 | 638x360 | 17 | 30 | 57.39 | 28 | 2.42 |
| 8 | 642x360 | 28 | 17 | 9.49 | 16 | 2.81 |
| 9 | 708x237 | 8 | 8 | 16.93 | 8 | 3.79 |
| 10 | 734x334 | 14 | 12 | 83.59 | 8 | 0.52 |
| 11 | 334x149 | 6 | 4 | 47.02 | 18 | 1.71 |
| 12 | 149x176 | 3 | 7 | 51.48 | 37 | 1.18 |
| 13 | 649x501 | 33 | 23 | 21.33 | 21 | 3.28 |
| 14 | 501x203 | 10 | 24 | 9.59 | 24 | 5.05 |
| 15 | 203x281 | 4 | 8 | 117.90 | 11 | 2.63 |

Table 5.2: Experimental results of the Shepard Hall building registration.

| Pair | Line Pairs | $t$ | Before ICP | | After ICP | |
|---|---|---|---|---|---|---|
| | | | $N$ | $P\_err$ | $N$ | $P\_err$ |
| 1 | 787x645 | 36 | 147 | 9.71 | 138 | 1.61 |
| 2 | 654x787 | 21 | 41 | 16.34 | 25 | 2.63 |
| 3 | 654x638 | 24 | 252 | 13.31 | 124 | 3.28 |
| 4 | 356x351 | 13 | 84 | 8.12 | 68 | 1.74 |
| 5 | 174x283 | 2 | 42 | 13.90 | 36 | 5.62 |
| 6 | 585x557 | 28 | 56 | 26.33 | 137 | 14.73 |
| 7 | 656x606 | 45 | 249 | 10.24 | 138 | 11.65 |
| 8 | 656x654 | 41 | 257 | 11.03 | 160 | 19.62 |
| 9 | 656x481 | 4 | 13 | 31.14 | 19 | 3.72 |
| 10 | 654x585 | 16 | 7 | 40.12 | 11 | 1.19 |
| 11 | 654x910 | 33 | 121 | 6.11 | 118 | 0.99 |
| 12 | 910x864 | 44 | 268 | 14.53 | 128 | 2.00 |
| 13 | 647x787 | 43 | 84 | 6.34 | 89 | 1.86 |
| 14 | 647x356 | 13 | 5 | 49.04 | 17 | 37.61 |
| 15 | 647x619 | 8 | 51 | 7.61 | 36 | 0.63 |

Table 5.3: Experimental results of the Great Hall building registration.

| Pair | Line Pairs | $t$ | Before ICP | | After ICP | |
|---|---|---|---|---|---|---|
| | | | $N$ | $P\_err$ | $N$ | $P\_err$ |
| 1 | 318x214 | 1 | 5 | 14.42 | 8 | 6.14 |
| 2 | 318x261 | 10 | 19 | 71.81 | 21 | 5.01 |
| 3 | 261x239 | 10 | 13 | 21.14 | 13 | 5.82 |
| 4 | 239x236 | 10 | 22 | 8.96 | 22 | 5.07 |
| 5 | 241x230 | 9 | 3 | 34.67 | 2 | 1.63 |
| 6 | 241x230 | 10 | 3 | 9.04 | 3 | 1.11 |
| 7 | 230x175 | 11 | 1 | 13.26 | 2 | 1.83 |
| 8 | 175x175 | 10 | 16 | 5.66 | 17 | 2.82 |
| 9 | 175x247 | 9 | 2 | 44.13 | 4 | 2.32 |
| 10 | 262x175 | 10 | 2 | 29.86 | 4 | 2.97 |
| 11 | 247x262 | 9 | 5 | 100.76 | 10 | 9.01 |
| 12 | 262x275 | 10 | 16 | 55.75 | 18 | 3.46 |
| 13 | 275x254 | 10 | 11 | 14.24 | 11 | 2.11 |
| 14 | 194x262 | 11 | 2 | 49.89 | 8 | 1.81 |
| 15 | 198x239 | 12 | 10 | 76.63 | 19 | 8.75 |
| 16 | 253x230 | 12 | 2 | 37.41 | 2 | 1.10 |

Table 5.4: Experimental results of the Grand Central Station registration.

| Pair | Line Pairs | $t$ | Before ICP | | After ICP | |
|---|---|---|---|---|---|---|
| | | | $N$ | $P\_err$ | $N$ | $P\_err$ |
| 1 | 394x767 | 54 | 33 | 7.48 | 37 | 1.22 |
| 2 | 343x767 | 60 | 31 | 7.79 | 43 | 0.98 |
| 3 | 320x443 | 40 | 15 | 7.67 | 14 | 1.12 |
| 4 | 767x443 | 47 | 68 | 9.86 | 42 | 1.20 |
| 5 | 837x400 | 28 | 35 | 15.17 | 36 | 1.65 |
| 6 | 837x443 | 29 | 40 | 9.06 | 16 | 2.48 |
| 7 | 189x195 | 6 | 17 | 4.92 | 20 | 1.44 |
| 8 | 280x196 | 15 | 19 | 58.10 | 33 | 1.19 |
| 9 | 189x280 | 9 | 17 | 1.92 | 17 | 1.11 |
| 10 | 208x97 | 1 | 3 | 43.89 | 8 | 0.96 |
| 11 | 208x304 | 10 | 5 | 2.47 | 10 | 0.980 |
| 12 | 145x304 | 7 | 10 | 20.59 | 12 | 14.86 |
| 13 | 194x266 | 6 | 7 | 11.64 | 6 | 2.55 |

Table 5.5: Experimental results of the Cooper Union building registration.

## 5.2 Circular Feature-Based Registration

Although many man-made urban scenes contain abundant linear features, some scenes, or some parts of a scene, may contain very few linear features. For example, the Figs. 4.6(a)(b) in Section 4.1 (displayed again in Figs. 5.12(a)(b), for reader's convenience) are overlapping, but contain large area of curved ceiling and do not have robust, matching linear features. Registration of these range images thus requires the utilization of non-linear features, in this example, circular features.

In Section 4.1, we presented our segmentation algorithm which extracts 3D oriented circular features from each range image. In this section, we introduce our algorithm of utilizing these circular features for registering range images. This algorithm expands the capabilities of line-based registration algorithms to recognize more

complicated geometric shapes in 3D scenes.

Similar to the linear feature matching procedure explained in Section 5.1, this algorithm begins by matching 3D circle pairs and calculating candidate transformations between a pair of range images. The next step, identifying the best transformation, is however more accurate and more robust, as compared to the previous approach. More specifically, surface consistency is used for verifying possible candidate transformations, and average point distance is used for selecting the best transformation.

### 5.2.1 Matching Circular Features

With the oriented 3D circles extracted from range images as in Section 4.1, possible matchings between them can be hypothesized, in order to compute transformations between overlapping range images. Using the segmentation result of two range images (Figs. 5.12(a)(b)) as an example, we describe our algorithm of range registration by matching the extracted circular features.

Recall that from the segmentation procedure in Section 4.1, each extracted oriented 3D circle is represented by three parameters: 3D center point, radius, and plane normal. Similarity of radius, orientation, and relative position between matching circles is then utilized for feature matching. Selecting only one circle from each range image is not sufficient for computing a candidate transformation. This is because one pair of plane normals (i.e. one vector from each image) is insufficient for computing

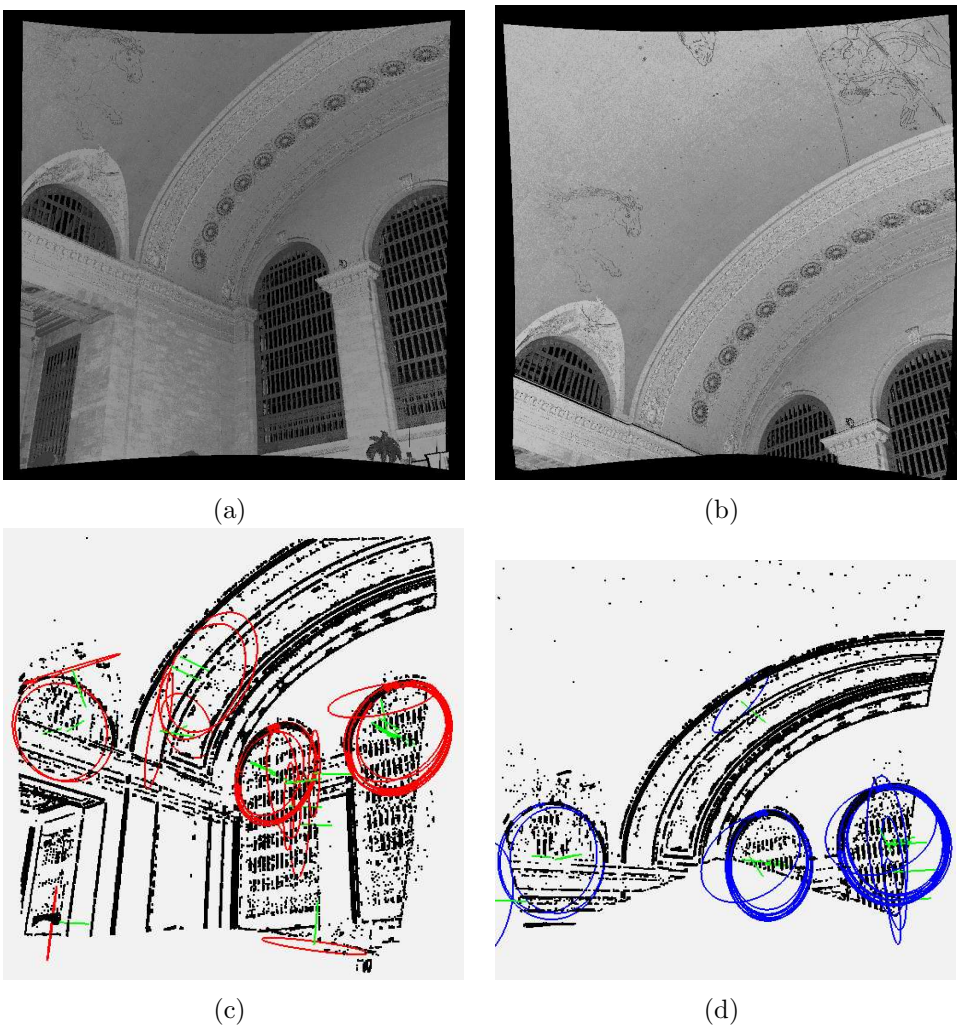


Figure 5.12: (a) An interior range scan of the Grand Central Station. (b) Another range image that overlaps (a). (c) Circles extracted from (a). (d) Circles extracted from (b). All edge points are in black, and all fitted circular arcs are represented with colored full circles, with green lines indicating their normals.

a unique rotation. Therefore, a pair of circles needs to be selected from each image, allowing for computing a unique candidate transformation.

In particular, consider a pair of circles $(C_1, C_2)$ from scan $R_1$ and another pair of circles $(C_1', C_2')$ from scan $R_2$. The pairs would be considered as matching iff

1. Circles $C_1, C_1'$ have equal radii within a threshold (maximum difference of 0.1m);

2. Circles $C_2, C_2'$ have equal radii within a threshold (maximum difference of 0.1m);

3. The distance between the centers of $C_1, C_1'$ equals the distance between the centers of $C_2, C_2'$ within a threshold (maximum difference of 0.2m);

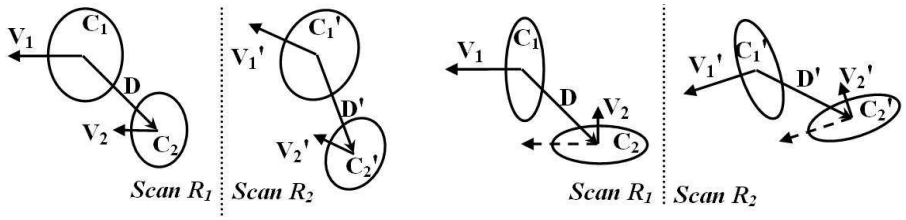
4. The angle between the normals of $C_1, C_1'$ equals the angle between the normals of $C_2, C_2'$ within a threshold (maximum difference of $10^o$).

If a pair of circles $(C_1, C_2)$ from scan $R_1$ and another pair of circles $(C_1', C_2')$ from scan $R_2$ could be considered a valid match according to the previous definitions, a transformation (rotation $R$ followed by a translation $T$) can be computed by converting the correspondence of a pair of oriented circles to a pair of 3D oriented lines, followed by a closed form formula computation of the transformation [Stamos and Leordeanu, 2003]. This approach leads to a robust transformation computation, since it is based on relative position and orientation of the features rather than exact position and orientation of each feature. In particular, two cases are considered:

**Case 1:** Circles $(C_1, C_2)$ have parallel normals $\mathbf{V_1}$ and $\mathbf{V_2}$ (the same is true for the normals $\mathbf{V'_1}$ and $\mathbf{V'_2}$ of circles $(C'_1, C'_2)$) (Fig. 5.13(a)). Let us consider the oriented line $\mathbf{D}$ that connects the centers of $(C_1, C_2)$ and the oriented line $\mathbf{D'}$ that connects the centers of $(C'_1, C'_2)$. If $\mathbf{D}$ is not parallel to $\mathbf{V_1}$ (that means that $\mathbf{D'}$ is not parallel to $\mathbf{V'_1}$), the match of the oriented line $\mathbf{D}$ with $\mathbf{D'}$ and $\mathbf{V_1}$ with $\mathbf{V'_1}$ can provide a reliable transformation (closed form formula [Stamos and Leordeanu, 2003]). Otherwise ($\mathbf{D}$ is parallel to $\mathbf{V_1}$), a reliable transformation can not be computed.

**Case 2:** Circles $(C_1, C_2)$ do not have parallel normals $\mathbf{V_1}$ and $\mathbf{V_2}$ (the same is true for the normals $\mathbf{V'_1}$ and $\mathbf{V'_2}$ of circles $(C'_1, C'_2)$) (Fig. 5.13(b)). Then, the two pairs of oriented lines $(\mathbf{V_1}, \mathbf{V_2})$ and $(\mathbf{V'_1}, \mathbf{V'_2})$ are used for the computation of a reliable transformation (closed form formula [Stamos and Leordeanu, 2003]).

From each valid matching circle pair, a candidate transformation is computed as described above. Each transformation is verified for correctness as follows. Based on the fact that overlapping images are captured from nearby positions, we discard all rotation matrices with diagonal elements smaller than 0.7 (allowing $45^o$ tilting of the range scanner about each of its $x/y/z$ axes). Note that this step reduces the number of possible transformations and thus speeds up the algorithm, but is otherwise not necessary.




(a) (b)

Figure 5.13: Two cases of matching circle pairs. The dashed line separates scan $R_1$ from $R_2$. The radii and relative position of the two circles from $R_1$ must be similar to those from $R_2$. (a) Case 1: Two circles have parallel normals. $V_1, D$ and $V_1', D'$are used to compute transformation. (b) Case 2: Two circle normals are not parallel. $V_1, V_2$ and $V_1', V_2'$are used to compute transformation.

### 5.2.2 Transformation Verification

In this step, we test whether the transformation causes surface inconsistency. Then, from all verified transformations, the one achieving the smallest average distance between overlapping range points is chosen as the best.

Surface inconsistency between two range images $R_1$ and $R_2$ is decided as follows (method described in [Huber and Hebert, 2003]). After image $R_1$ is transformed to the coordinate system of the image $R_2$, consider a ray cast from $R_2$'s center of projection toward the overlapping area of the two images. Any of the following two situations indicates a violation of surface consistency: (1) along the ray there is a point from $R_1$ but no point from $R_2$ (Free Space Violation); (2) there is a point from $R_1$ significantly in front of a point from $R_2$ on the ray (Occupied Space Violation). We follow the suggestion in [Huber and Hebert, 2003] and implement surface inconsistency detection

with two z-buffers, one for each scan.

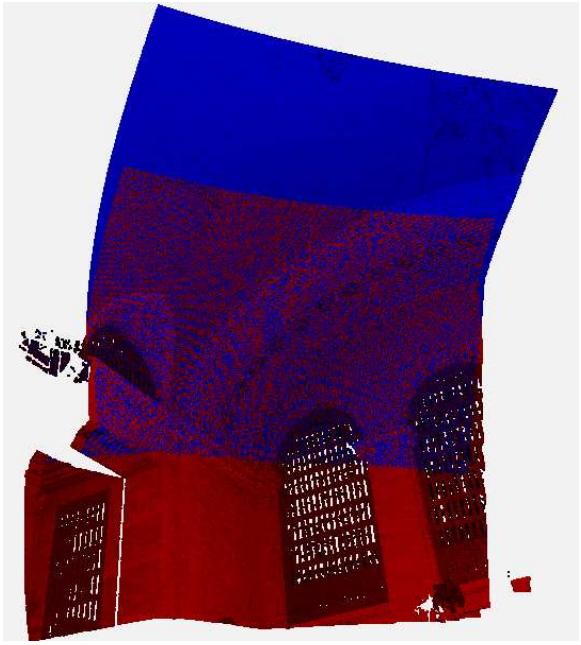
The coordinates of both z-buffers are based on the 2D grid of scan $R_2$ ($999 \times 999$ points), but sampled into $250 \times 250$ bins. One point falls into one bin if the ray from $R_2$'s center of projection toward the point passes through that bin. For each bin in $R_2$'s z-buffer, among all points that fall into it, the smallest distance from the center of projection is recorded. $R_1$'s z-buffer is filled in the same manner based on the 2D grid of $R_2$, but with points from the transformed $R_1$. By comparing these two z-buffers, we are able to detect cases of FSV and OSV. Only at those bins where distance values from two images differ no more than a distance threshold (2.0m), i.e. overlapping area, we record the difference, and take the average of all such differences to evaluate the goodness of the transformation. We choose the threshold 2.0m, because this value is a loose upper bound for point distance. Setting the threshold too low would limit the comparability of two transformations, since only those few points already very close are used for distance computation. Therefore, the difference between good transformations and bad ones is not obvious. In our experiment, we also set a loose threshold (10%) on overlapping area to filter out invalid transformations. Such a transformation may achieve very small point distance due to small overlapping area, but in fact does not imply a correct registration. During range data acquisition, the overlapping area between successive range scans is always large enough ($> 10\%$) to guarantee the possibility and the quality of an automated registration.

### 5.2.3 Experimental Results

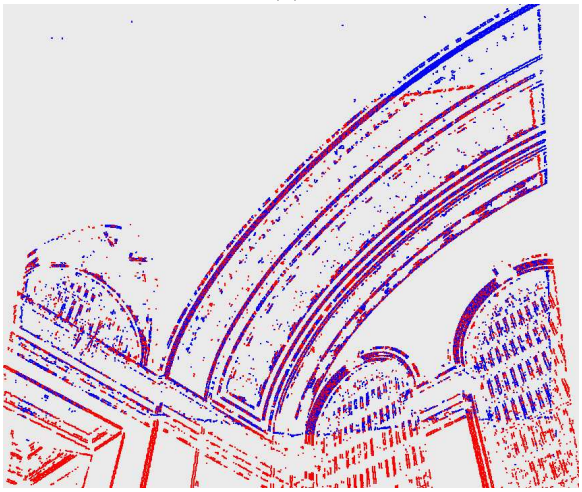
This automated registration method is used for the complete registration of the Grand Central Station scans. The best transformation of the two corner scans of Fig. 5.12 provides a registration error (average point distance in the 55.7% overlapping area) of 0.95cm. Within a few iterations of ICP [Gelfand *et al.*, 2003], an optimal transformation with a registration error of 0.90cm is obtained (Fig. 5.14).

More scans of the Grand Central Station are registered with the same technique, particularly those connecting the rectangular hall and the arching ceiling (not circular though), as shown in Fig. 4.7. Note that the lower parts of the walls (Fig. 4.7(c)(g)) contain lines and planes, and are therefore registered with the linear feature-based technique as described in Section 5.1 (see the registration results shown in Fig. 5.11(a)(b)). The upper regions with very few linear features (Fig. 5.11(a)(e)), are registered with their lower neighboring scans (Fig. 4.7(c)(g)), respectively, by matching overlapping circular windows. In Fig. 5.15, the registered scene and edge points from 4 scans are shown.

More registration results are shown in Fig. 5.16 and Table 5.4. Among all 37 scans of the Grand Central Station being registered, 20 are the lower parts registered with linear features, 13 are the upper parts registered with their lower neighbor scans based on overlapping circular windows, and 3 are manually registered, as they are curved ceiling patches without any distinguishing geometric shape information. In



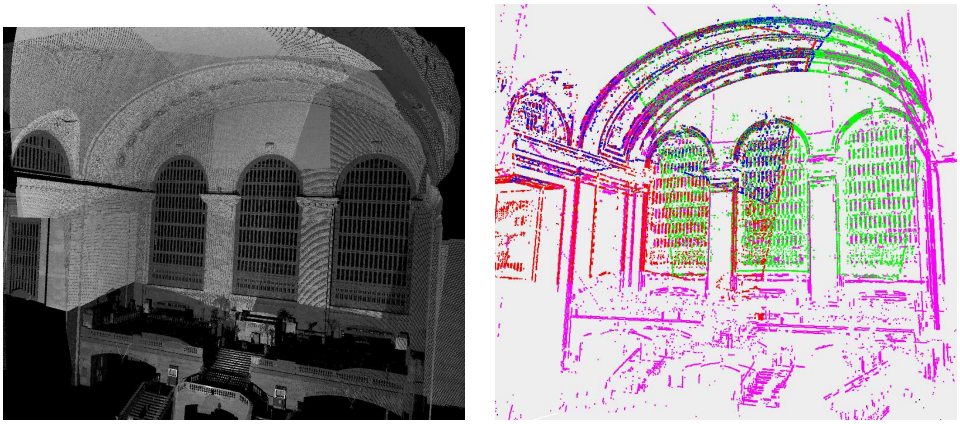
(a)



(b)

Figure 5.14: Registered images of Figs. 5.12(a) and (b). They are colored to highlight overlapping area. (a) All image points. (b) Edge points at overlapping area.

Fig. 5.16(a), registered edge points from 23 scans are visualized. The other 14 scans are not displayed to avoid occlusion: those scans compose a side wall closer to our viewpoint, and symmetric to the wall being displayed. Fig. 5.16(b) shows a part of the registered scene, and Fig. 5.16(c) shows a 3D surface model constructed from the range points in Fig. 5.16(b). In Table 5.4, we report the performance of 13 registrations based on circular features. When registering the last two pairs, it took a long time due to a large number of valid transforms from the precisely extracted circles around the window frame (as in Figs. 4.7(b)(d)). To avoid unnecessary computations, we set the program to terminate when the average distance falls below 10cm (an approximate upper bound for the distance between two neighboring points). The values in columns *RT*, *D_mat*, and *Time* are therefore recorded up to the point when an accurate enough result is reached.

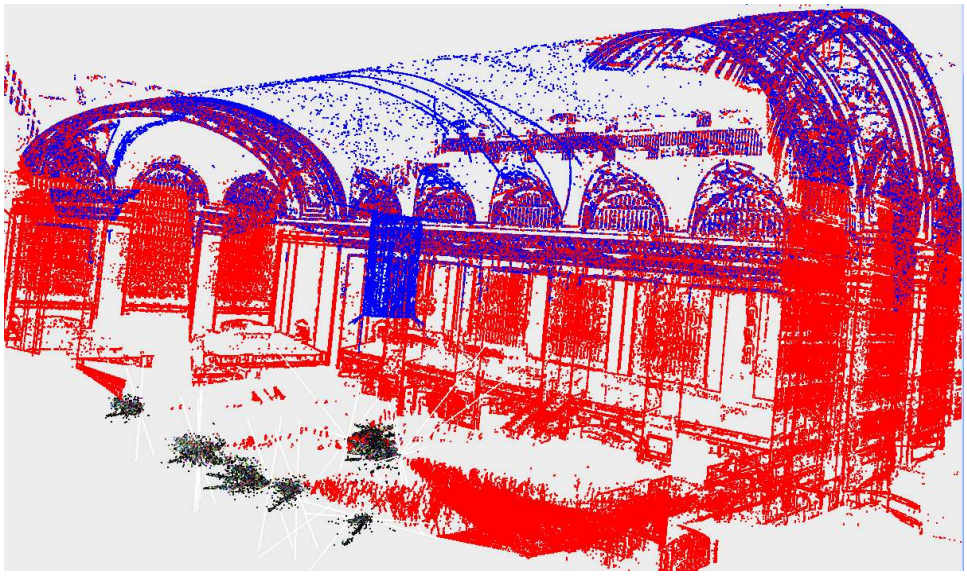


(a) (b)

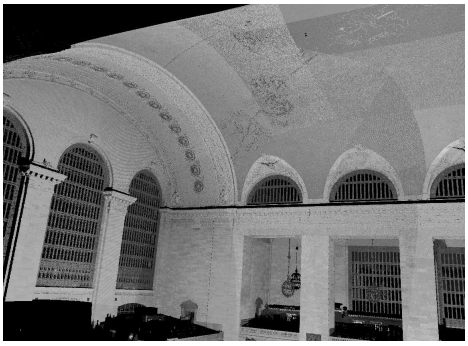
Figure 5.15: Registration results of the Grand Central Station. (a) 4 (out of 37) automatically registered scans. (b) Edge points of (a). Four colors represent edge points from four scans.

| Circles | RT | D_mat | D_opt | Overlap | Time |
|---|---|---|---|---|---|
| 26x24 | 544 | 0.95cm | 0.90cm | 55.7% | 6min |
| 24x39 | 980 | 1.11cm | 1.00cm | 29.1% | 9min |
| 21x39 | 15 | 3.42cm | 1.28cm | 16.1% | 1min |
| 24x20 | 748 | 2.13cm | 0.77cm | 38.4% | 7min |
| 13x30 | 126 | 2.01cm | 0.84cm | 28.9% | 2min |
| 21x26 | 534 | 1.68cm | 0.90cm | 35.5% | 6min |
| 23x11 | 29 | 4.26cm | 0.87cm | 28.7% | 1min |
| 14x31 | 18 | 2.65cm | 0.93cm | 27.8% | 2min |
| 31x13 | 58 | 2.34cm | 0.98cm | 23.0% | 2min |
| 37x26 | 67 | 3.83cm | 0.87cm | 37.2% | 2min |
| 23x35 | 310 | 1.20cm | 0.84cm | 26.7% | 7min |
| 49x41 | 3054 | 2.81cm | 1.02cm | 38.7% | 58min |
| 50x38 | 931 | 1.83cm | 0.92cm | 44.6% | 10min |

Table 5.6: Experimental results of the Grand Central Station registration. Meanings of columns: Number of circles from two images to register; Number of candidate transformations; Average point distance from best transformation; Average point distance after ICP optimization; Overlapping area of two scans; Execution time, including circle extraction and matching (on a Linux-based 2GHz Xeon-Processor with 2GB of RAM).



(a)



(b)

(c)

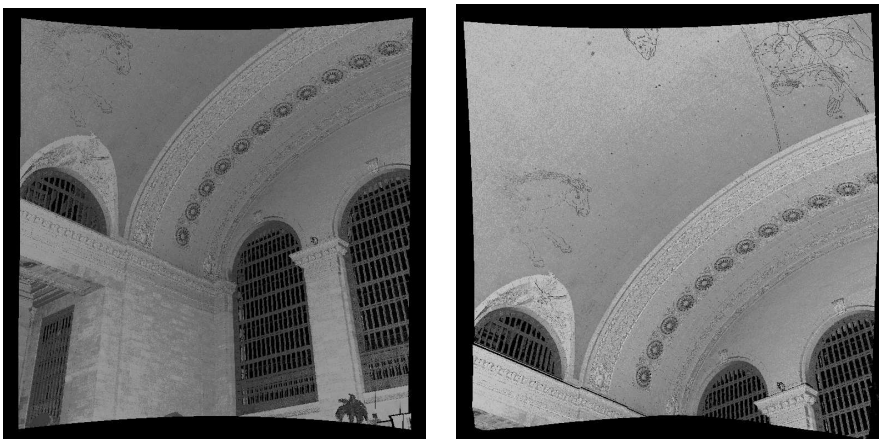
Figure 5.16: (a) Registered edge points of 23 scans of the Grand Central Station. Different colors differentiate scans from upper parts (blue) and lower parts (red). (b) 8 out of 37 automatically registered scans are shown for clarity. (c) 3D model generated with Ball Pivoting Algorithm. The smooth ceiling implies that the registration is tight and seamless.

## 5.3 3D Pattern Analysis with Shape Descriptor

As an extension of matching linear and circular features, we also designed and implemented an algorithm for matching arbitrary geometric shapes. An ability to do this becomes necessary and useful when some portions of the scene contain very few straight lines or circles, but have other outstanding 3D patterns which can be used for recognition and matching. Previous registration algorithms have shown that finding matching feature pairs is an essential step. Three pairs of matching features can provide two pairs of matching vectors, which are sufficient for calculating a transformation. If there are more than three matching feature pairs, we can then calculate multiple candidate transformations, and select the best one, using the verification method in Section 5.2.1.

The experimental data we worked on are the two scans shown in Figs. 5.12(a) and (b). For reader's convenience, the figures are displayed again in Figs. 5.17(a) and (b). The overlapping area of the range images contains part of the curved ceiling, where we manually select regions of interest for shape analysis and pattern matching. More specifically, we select two regions from one range image, and locate them in another range image.

One pattern is a group of points forming a rosette (Fig. 5.18(c)). This pattern is observed to be repeating along the curved ceiling. The pattern is selected by manually specifying a center point $P$ (the center of the middle rosette) and a radius $r$



(a) (b)

Figure 5.17: Two overlapping range images of the Grand Central Station.

(by selecting another point, $Q$, at the same rosette's circular boundary, and computing the distance between $P$ and $Q$). All the points within $r$ distance to $P$ are then selected from the range image, forming this pattern.

Another pattern, pegasus head, is extracted from the darker points in the ceiling part of the range images, corresponding to the paintings on the ceiling. Since these points show different intensities from the surrounding points in the laser reflectance image, we apply 2D edge detection on the reflectance image to obtain edge points on the 2D range image grid. We then select the corresponding 3D points from the original range image. These selected laser reflectance edge points are shown in Fig. 5.18(b). From these points, we select a point $P$ near to the pegasus' eye, and select another point $Q$ from its nose tip. Then, just as in the rosette case, we calculate a radius $r$ as the distance between $P$ and $Q$, and select all the points within distance $r$ to point

$P$, forming a pattern of pegasus head (Fig. 5.18(d)).

The rosette pattern and the pegasus head pattern are then the features utilized to match between the two range images in our experiment. First, a shape descriptor is computed for each pattern selected from a source range image (Fig. 5.17(a)), and second, we search in a target range image (Fig. 5.17(b)) to locate a region with similar shape descriptor.

The descriptor we used is a probability distribution of the distances between randomly sampled point pairs, as suggested in [Osada *et al.*, 2001]. In more details, we randomly select two points, and record the distance between these two points. We repeat this process 500,000 times, resulting in 500,000 distance values. Then, a histogram with bin size $b = 0.0125m$ (user specified based on each pattern size: in our experiment, both are approximately 1.5m) is generated with these distance values. This histogram is thus the shape descriptor for this pattern, as it encodes the shape and surface point distribution into a discrete statistical representation. Figs. 5.18(e)(f) show the shape descriptor for the rosette pattern and the pegasus head pattern described above.

Once the shape descriptor for the pattern is generated, the next step is to locate a region in the target image whose shape descriptor is closest to that of the pattern. The similarity measure is based on the average difference between the probability distributions. The algorithm goes through every point in the target image, selects

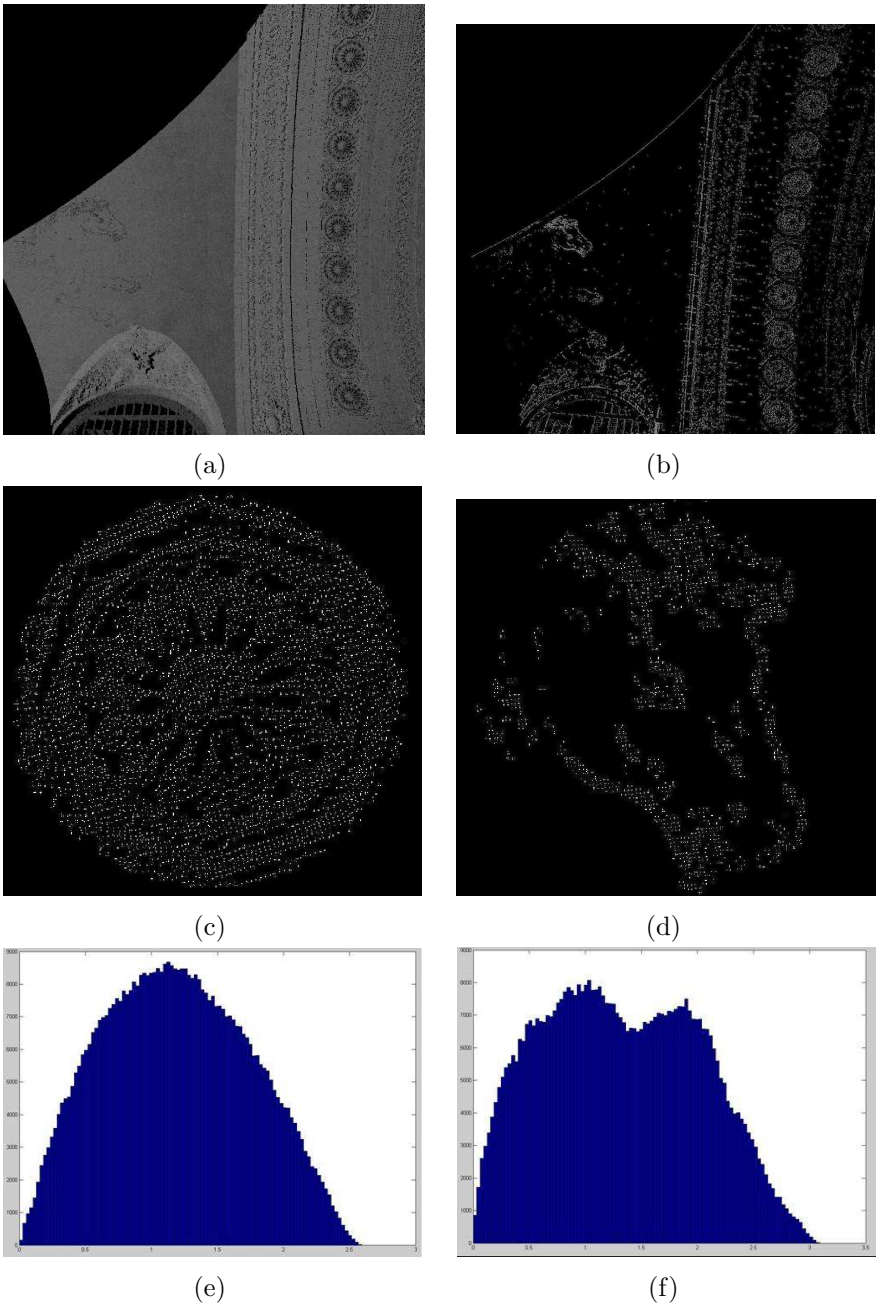


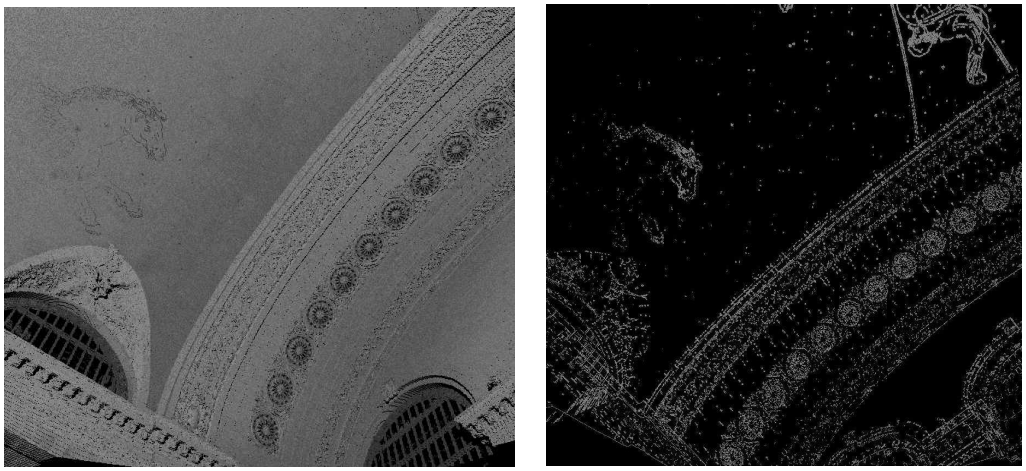
Figure 5.18: Selecting patterns from Fig. 5.17(a). (a) The ceiling part of Fig.5.17(a). (b) Edge map of (a). (c) Rosette pattern (selected from the original range image, (a)). (d) Pegasus head pattern (selected from the edge map, (b)). (e) Shape descriptor of (c). (f) Shape descriptor of (d).

all neighboring points which are within distance $r$, and generates a shape descriptor, based on all these neighboring points. Then, this target shape descriptor is compared with the pattern's shape descriptor, and the similarity measure is calculated by obtaining the average distance at each corresponding bin. The region with the highest similarity is returned as the match. Due to the repetition of the rosette pattern, we kept the ten regions with highest similarity during the calculation.

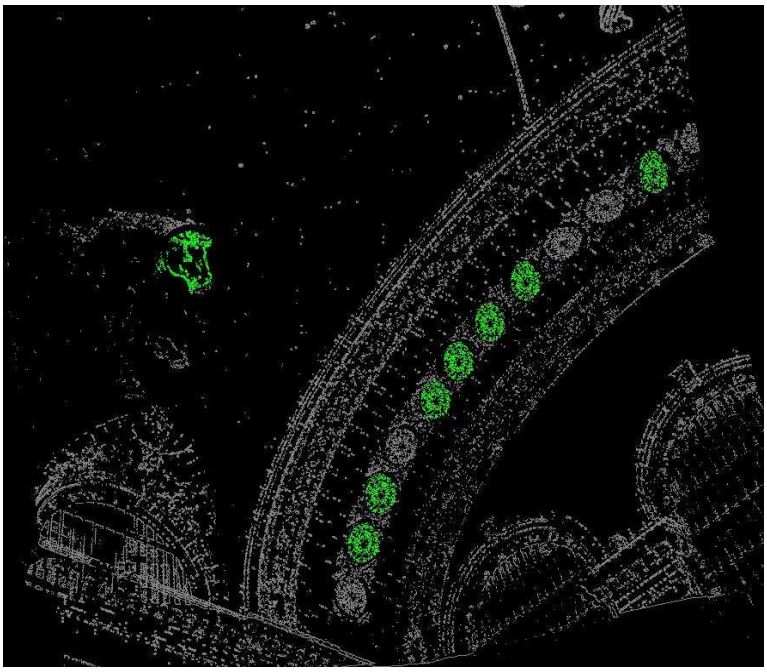
Our program correctly located seven rosettes. To observe the accuracy of the matching algorithm, we also replaced these matching regions with the original pattern points, marked with green color(Fig. 5.19(c)). The pegasus head is also successfully located and replaced in the second range image(after being processed with edge detection), as shown in Fig. 5.19(c).

The experimental results show that feature matching based on shape descriptor is quite accurate and is easy to implement. In order to incorporate this algorithm into the automated registration system, feature extraction needs to be automated.

A drawback of feature matching based on shape descriptor is the time complexity, due to the number of sampling that need to be performed, and the size of the space that needs to be searched. These complexities, however, are unavoidable if we want to maximize the probability of performing accurate matching. In order to speed up the computation, efficient algorithms would need to be designed for pruning improbable regions, thereby decreasing the search space.



(a) (b)



(c)

Figure 5.19: Pattern matching and replacing. (a) The ceiling part of Fig.5.17(b). (b) Edge map of (a), on which pegasus head pattern is searched. (c) The matching regions are replaced with two patterns, respectively.

# Chapter 6

# Segment Merging for 3D Modeling

In Chapter 4, the segmentation algorithms which partitions each range image into meaningful segments was presented. Chapter 5 described the process of registering range images of the Grand Central Station to form a complete scene. As the next step in the modeling pipeline, 3D graphical models are generated based on the registered range images. Depending on the application, we can either generate a model of the entire scene, or a model per meaningful segment in the scene. An example of a complete scene model is shown in Fig. 5.16(d), where a single surface mesh is formed based on all the registered range points.

In applications such as object recognition, however, graphical models for meaningful segments of the scene (e.g., facades, windows, ceilings, or objects) are needed.

In a modeling framework, higher order processes can then manipulate, alter, or replace individual segments. In an object recognition framework, these segments can be invaluable for detecting and recognizing different elements of urban scenes.

In order to create a 3D model for each meaningful segment in the registered scene, we utilize the segments extracted by the algorithm described in Section 4.2. The segmented planar regions, smooth non-planar regions, and non-smooth connected regions in each range image represent different meaningful regions or objects. When the range images are registered into a common reference system, all their segmented regions are transformed accordingly. As a result of this registration, some segments, originally belonging to each individual range image, now overlap and form a larger region. These segments need to be identified and merged. This merging procedure is a reconstruction of the surface composition in the original scene, as many surfaces were only partially captured by different range images, due to the field-of-view limitation of each scanning process. The segment merging procedure, therefore, results in coherent segments that correspond to urban objects of the complete large scale urban scene. From these segments, we generate a different mesh for each object using Ball Pivoting Algorithm [Bernardini *et al.*, 1999].

Our input to the segment merging procedure is a collection of registered range images, each segmented with the algorithm in Section 4.2. Our output is a number of segments that describe urban entities (e.g. facades, windows, ceilings, or architectural

details). In the following sections, we begin with restating the problem of merging segments (Section 6.1). We then describe the procedure for combining overlapping regions between two range images (Section 6.2). We further extend this procedure and explain the merging of overlapping regions between more than two range images (Section 6.3). Finally, the experimental results are presented in Section 6.4.

## 6.1 Problem of Merging Segments

In the segmentation algorithm discussed in Section 4.2, every range image is partitioned into sequentially labelled segmented regions, determined by the order of region extraction. In the previous figures of segmentation results (Fig. 4.10 and Fig. 4.11), each segmented range image is displayed with all points colored based on their region labels: points in the same region have the same color, while points from different regions have different colors.

Using the range image registration systems described in Chapter 5, partially overlapping range images are registered to form a combined 3-D scene. Consider that there are overlapping regions captured in multiple range images, and in every range image their region labels are assigned in the segmentation procedure of each range image individually. This causes different region labels for the same region, which is then displayed with different colors in the combined view (see Fig. 6.2). The mixed colored points lead to confusion when observing regions existing in multiple scans, and

results in false boundaries for large regions that spread across a few range images. It is thus necessary to detect and unify overlapping regions. By detecting and unifying overlapping regions, the overlapping segments are identified and merged, resulting in coherent segments for the complete scene.

Let us first introduce some notations and data structures in order to describe the algorithm we designed for performing this task:

1) There are $n$ segmented range images to be merged, denoted as $I_i$ ($i = 1..n$). As mentioned earlier, we have computed the transformation matrices that register these images under a common coordinate system of a pivot image. The pivot image is one of the $n$ range images, denoted as $I_p$ ($1 \leq p \leq n$). Let us further denote $M_i^p$ as the matrix to transform image $I_i$ to the coordinate system of image $I_p$. With matrix composition, we can also select another range image $I_q$ as the pivot image, and calculate the transformation from all other images to $I_q$, denoted as $M_i^q$ ($1 \leq q \leq n$, $i = 1..n$).

2) Each range scan $I_i$ is associated with a Z-buffer $Z_i$. $Z_i$ is a 2D array defined based on the 2D rectangular area covered by the scanning laser beams, separating the space of scanning direction into bins (Fig. 6.1(a)). The dimension of the array is the same as the scan's resolution, ensuring that no more than a few neighboring points fall into the same bin. In each cell, a set of three values is recorded: depth, region label, and surface normal. These values are of the point that falls into this cell and is

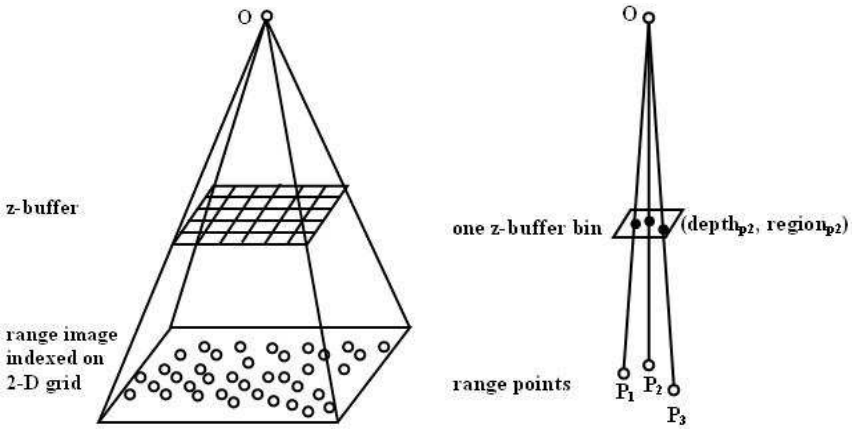



Figure 6.1: Generating z-buffer. (a) The range of each cell of the z-buffer is determined by the pivot scan. (b) $O$ is the origin. $P_1$, $P_2$ and $P_3$ are range points that fall into one bin. Since $P_2$ is closest to $O$, $P_2$'s information is used to fill this bin.

closer to the origin (Fig. 6.1(b)). When filling the z-buffer, every point in the range image finds its corresponding bin, and if the bin is not filled or the current depth value stored in the bin is larger than that of the new point, then the value set in that bin is updated with its depth, region label and surface normal.

3) We also generate a transformed z-buffer $Z_i^p$ for each range image $I_i$ with respect to a pivot scan $I_p$. Similar to $Z_i$, $Z_i^p$ is a 2D array where each bin is populated with point depth, region label and surface normal. However, there are two differences between $Z_i^p$ and $Z_i$. First, the bins of $Z_i^p$ are based on the 2D scanning area of $I_p$; second, all the points in scan $I_i$ are transformed to the coordinate system of $I_p$, and the depth and surface normal of these transformed points (together with their original region label) are used to fill into $Z_i^p$. If one point falls out of the range of $Z_i^p$, the

point is simply not considered; if multiple points fall into one bin, the point with the smallest depth is selected and its information is filled in that bin. Note that $Z_i^i$ is equivalent to $Z_i$ defined in item (2) above. From this point forward, we will refer to both as $Z_i^i$.

In the following section we first describe how the overlapping regions between two range images are combined, and then explain the merging procedure for more than two range images.

## 6.2 Merging Two Range Images

Assume that the two range images to be merged are $I_1$ and $I_2$. $I_1$ is the pivot scan, and $I_2$ is transformed to $I_1$'s coordinate system with transformation matrix $M_2^1$. Based on the 2D grid of $I_1$, we generate $I_1$'s z-buffer $Z_1^1$, as well as $I_2$'s transformed z-buffer $Z_2^1$. Since the range scanning captures the first surface point it reaches, each depth value in $Z_2^1$'s z-buffer should be equal to[1] or larger than the depth value in $Z_1^1$'s corresponding bin [Chen and Stamos, 2006][Huber and Hebert, 2003]. If the depth and surface normal in the corresponding bins of two z-buffers are equal, we conclude that these points from two images are overlapping. The corresponding regions that these two points belong to have therefore the possibility of being one same region. If a pair of

[1]In the context of this chapter, by *equal* we mean that two values differ by no more than 8mm, which is a value slightly larger than scanning noise 6mm.

regions are both smooth regions or both non-smooth regions (this information was obtained in segmentation process), and they overlap in sufficient number of z-buffer bins, they are considered the same region and should be merged and labelled with the same region label. In short, the methodology consists of first identifying and then merging all the region pairs that overlap in significant number of bins. However, not all the overlapping regions should be merged, as minor segmentation error might result in small overlaps near region boundaries. With these considerations, our algorithm of merging regions is as follows:

1. Relabel image $I_2$ so that its region labelling starts after the largest region label in $I_1$. This way, each region from the two images has a unique label. Then, after generating $Z_1^1$ and $Z_2^1$, go through each bin, and compare the values in $Z_1^1$ and $Z_2^1$ to find out all the bins that have same depth and surface normal values. Overlapping region pairs are then identified from these bins, and their overlap size is counted as the number of bins that contain it.

2. Only retain region pairs with overlaps satisfying either one of the following two conditions:

a) Condition 1: The number of overlapping bins reaches a certain proportion (0.01%) of point numbers of both range images; b) Condition 2: The overlapping bin count reaches a certain proportion (5%) of the number of bins occupied by the smaller region in the pair. This way, region pairs with a large overlap will satisfy

condition 1, while possibly small regions, but with proportionally large overlap, will satisfy condition 2.

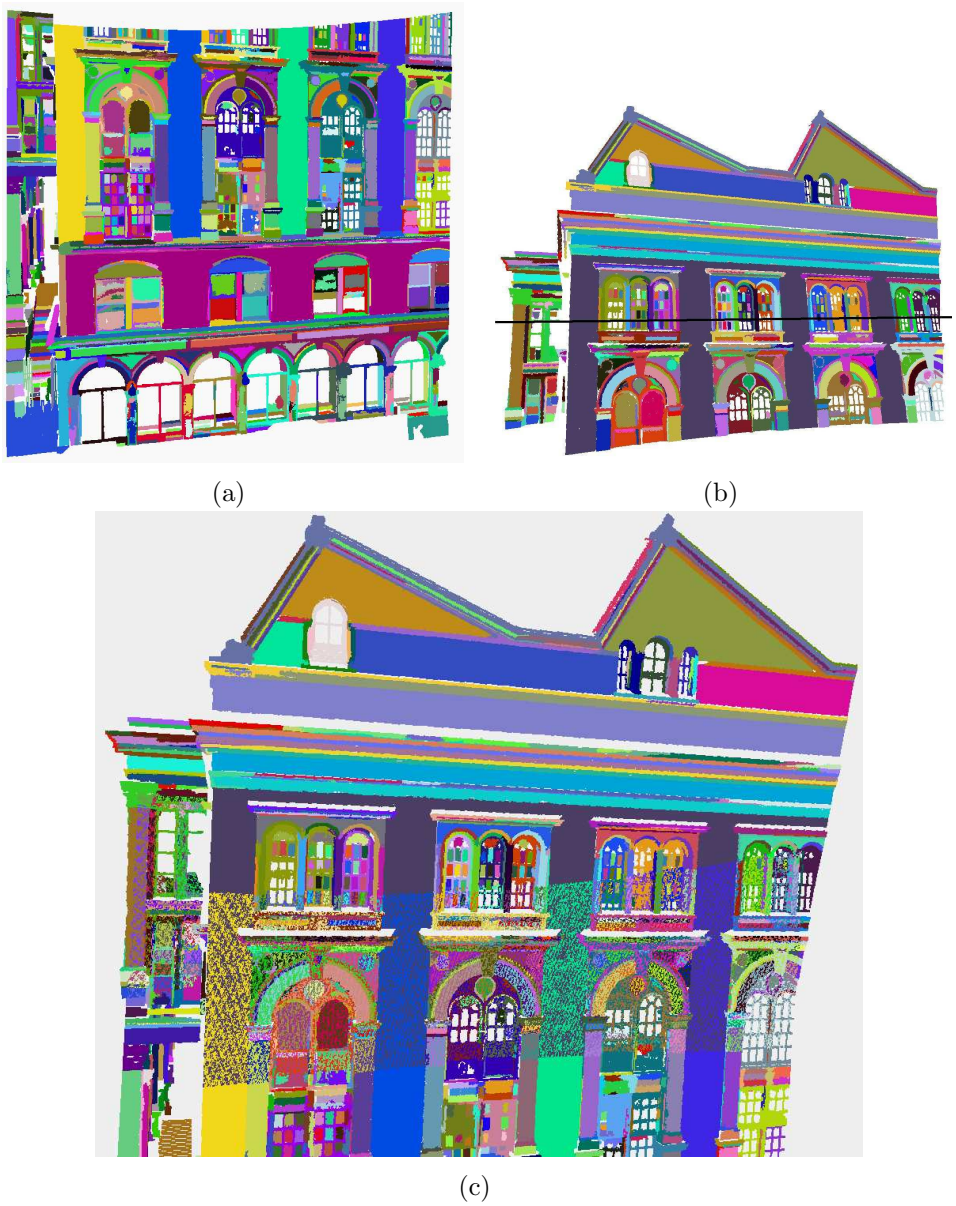
3. Form overlapping region sets based on all overlapping pairs satisfying above conditions. Assign a new label (the largest existing label plus 1) to each overlapping region set, resulting in overlapping regions being labelled the same.

Fig. 6.2 shows two segmented range images that overlap. Note that the main overlapping area contains a few windows and a large facade. Fig. 6.3 shows the merging result. The facade becomes a merged region, and the overlapping windows are also well merged.

## 6.3 Merging Multiple Range Images

Merging multiple range images consists of multiple processes of merging two range images, but with added sophistication. The increased complexity results from the fact that multiple range images might overlap at the same area, and their labels need to be changed altogether. Our algorithm considers one range image as pivot at a time, and computes overlaps within this pivot scan's z-buffer area. After all range images have been used as pivots, overlapping region labels are summarized and new labels are generated for regions to be merged.

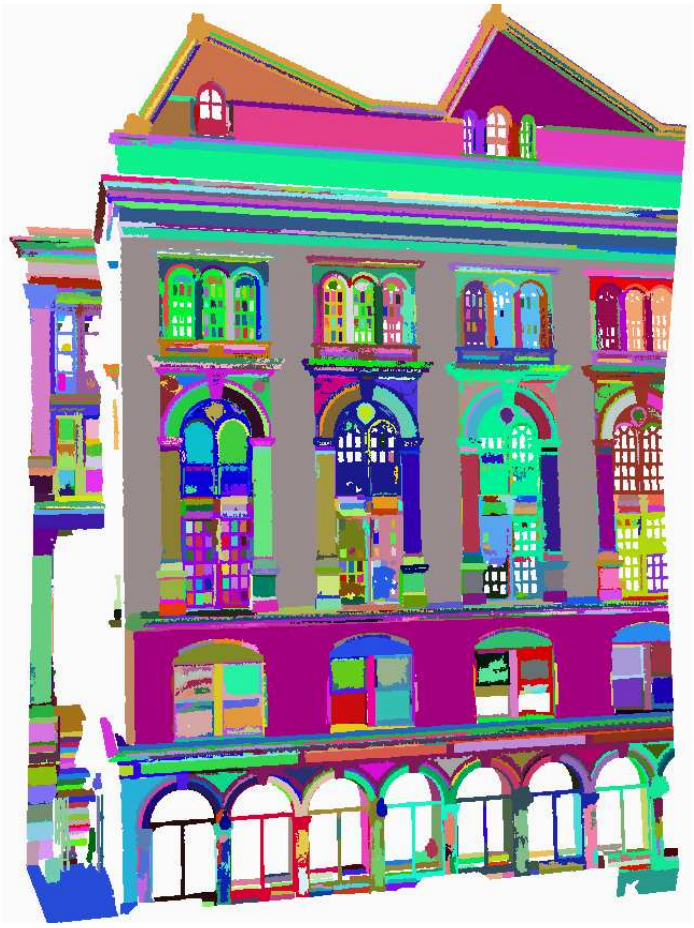
First, region labels from all range images are relabelled, so that all labels are unique. Then, the goal is to detect overlapping region sets from all range images



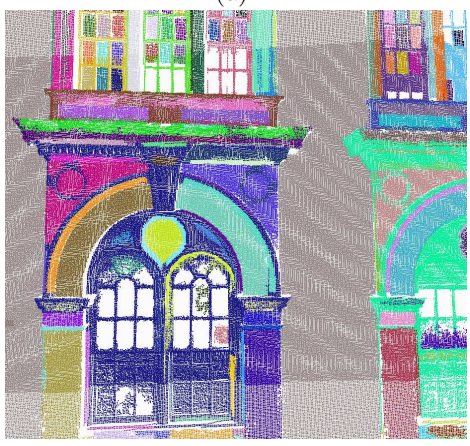
(a) (b)

(c)

Figure 6.2: (a)(b) Two overlapping segmented range images of the Cooper Union building. The area below the black line in (b) is the overlapping area. The main facade in (b) is segmented to four separated regions in (a). (c) Registered (a)(b) without merging segments; the overlapping area is displayed with mixed colors.



(a)



(b)

Figure 6.3: (a) Merging results of Figs. 6.2(a)(b); (b) Details at overlapping area.

(similar to the overlapping pairs in the context of merging two range images), and assign each set a new label. In order to generate overlapping region sets, we first identify all overlapping region pairs, and then combine all overlapping pairs to form overlapping sets.

From the range image list $I_1, I_2, ...I_n$, $I_1$ is first selected as the pivot scan, generating $n$ z-buffers $Z_i^1$ $(i = 1..n)$. By comparing corresponding bins in $Z_1^1$ with each of the transformed z-buffers $Z_t^1$ $(t = 2..n)$, the overlapping bin are counted for all region pairs that overlap within the boundary of $I_1$. Note that in any region pair in this step, e.g. $(R_1, R_2)$, one region $R_1$ is always from the pivot scan $I_1$, and the other region $R_2$ is from one of the other range images $I_t$ $(t = 2..n)$.

Then, the next image $I_2$ is considered as the pivot scan. Overlapping counts for region pairs are accumulated similarly. After every image has been used as the pivot, all the possible overlapping region pairs have been counted with their overlapping size respectively. Note that for any region pair $(R_1, R_2)$, its overlapping size is counted twice: once when $R_1$'s image is the pivot, and once when $R_2$'s image is the pivot. This is a redundant computation. Therefore, in our algorithm, for any pivot scan $I_t$, we only compare $Z_t^t$ with transformed z-buffers $Z_{t+1}^t, ..., Z_n^t$.

Similar to the procedure of merging two range images, region pairs with small overlaps are discarded. All the remaining overlapping region pairs are then combined to form overlapping region sets. With each set re-labelled to a new label, we thus

update the region labels for all the points in all range images. Note that during the merging procedure, every two scans are compared exactly once. Therefore, the order of merging, which determines the later order of pivot selection and z-buffer comparison, is arbitrarily determined. A different order will not change the segment merging result.

## 6.4 Experimental Results

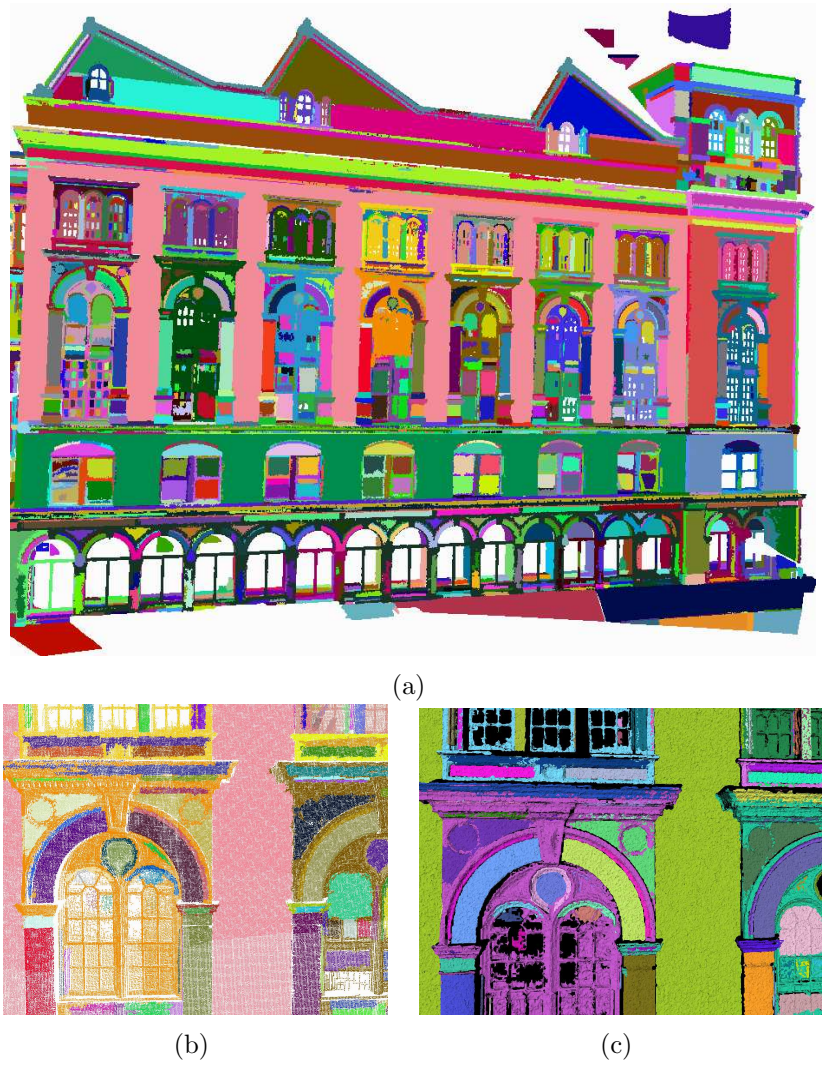
The segment merging process is applied to the registered range scans of all the five building data sets. For each building, our experiments segment each scan and combine all the segmented scans, generating 360 degree interior and exterior views respectively. For some buildings, surface meshes are generated based on merged segments using Ball Pivoting Algorithm [Bernardini *et al.*, 1999]. These merged segments and surface meshes are displayed in Figs. 6.4 through 6.9. For clarity of display, in some figures, only a part of the combined set is shown.

Fig. 6.4 shows the merged segments of 8 exterior scans of the Cooper Union building, and zoom-in details of an area where four images overlap. The execution time for segmenting each scan is 2 minutes, and for merging is 4 minutes (2GHz Xeon Processor, 2GB RAM). The merging obtains 1760 smooth regions and 382 non-smooth regions, and the average surface fitting error for smooth regions is 3mm. Fig. 6.4(b)(c) shows details of merged scan points and generated surface meshes, where the density

difference across the image boundaries is diminished.

Fig. 6.5 shows more results of the Cooper Union building, a set of 13 scans containing two major facades, and a set of 17 scans containing two other major facades. Fig. 6.6 shows the merged segments of the Thomas Hunter building (14 scans). Fig. 6.7 and Fig. 6.8 show the merged segments and surface meshes of the Shepard Hall building (20 scans) and the Great Hall (27 scans), respectively. In these experiments, every scan is segmented within 2 minutes, and every merging procedure takes approximately 10 to 30 minutes to complete.

Fig. 6.9 shows the merged segments from 15 interior scans of the Grand Central Station and generated surface meshes. The execution time for segmenting each image is 2 minutes, and for merging is 10 minutes. The merged segmentation contains 1393 smooth regions and 787 non-smooth regions, and the average surface fitting error for smooth regions is 4mm.



(a)

(b) (c)

Figure 6.4: (a) Merging results of 8 scans of the Cooper Union building. (b) Enlarged (a) at overlapping area. (c) Segment meshes (rendered with random colors).

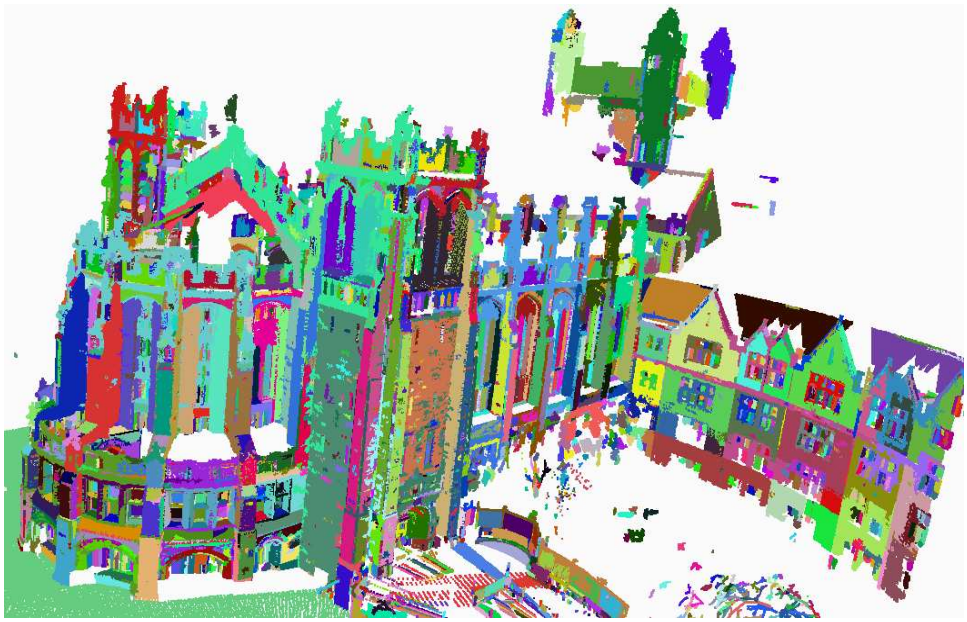
(a)

(b)

Figure 6.5: Merging results of more scans of the Cooper Union building. (a) 13 scans. (b) 17 scans.

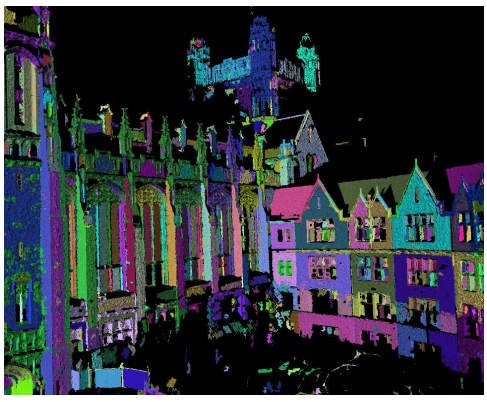
(a)

Figure 6.6: Merging results of the Thomas Hunter building.



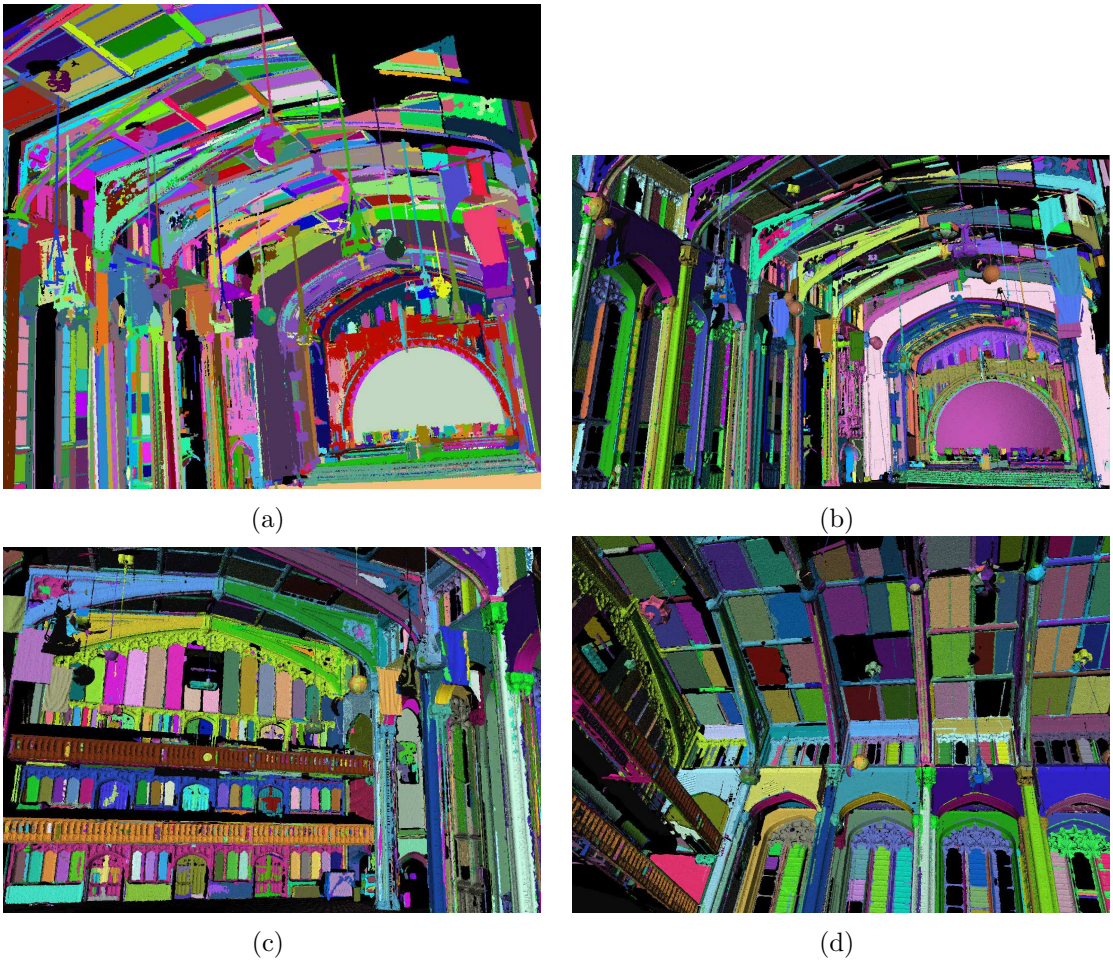
(a)

(b)



(c)

Figure 6.7: (a) Merging results of the Shepard Hall. (b)(c) Segment meshes.

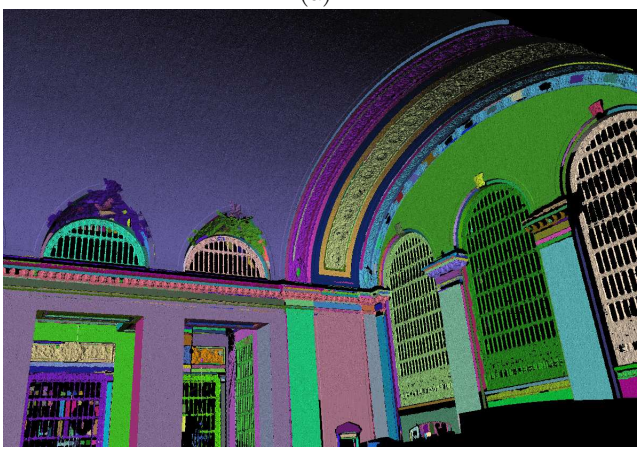


(a) (b)

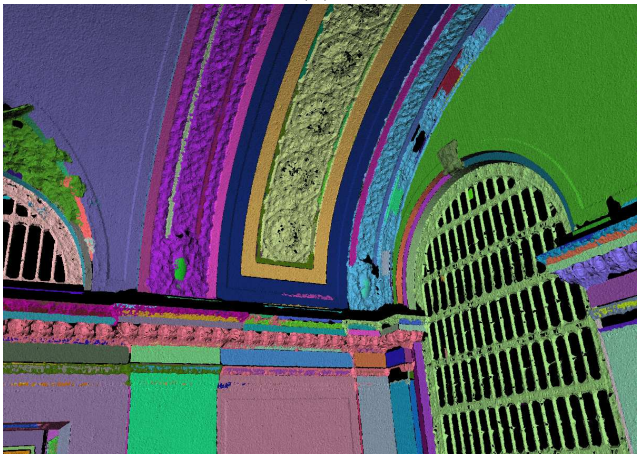
(c) (d)

Figure 6.8: (a) Merging results of the Great Hall. (b)(c)(d) Segment meshes.

(a)



(b)



(c)

Figure 6.9: (a) Merging results of 15 scans of the Grand Central Station. (b) Segment meshes. (c) Enlarged (b) at overlapping area.

# Chapter 7

# Conclusions and Future Directions

Highly accurate 3D models of large scale urban scenes have a multitude of applications in a variety of fields such as historical preservation, virtual reality, motion pictures, and gaming. Therefore, an ability to build such models accurately and efficiently is one of the main goals of computer vision research.

Continuously advancing technology of range imaging currently allows us to capture high resolution range images of urban scenes, providing abundant raw data for building highly accurate 3D models. And while the data is abundant, its size necessitates the design and implementation of efficient algorithms for analyzing and processing it, so that such models can be built within a reasonable amount of time.

In this thesis, we introduced a typical 3D modeling system, which has the following main phases: **data acquisition, range segmentation, range registration, and**

**model generation**. We then described the purpose and methodologies of each phase. Finally, we presented our novel algorithms for segmentation, registration, and segment merging, as major steps leading to modeling. The following is a summary of the contributions we have made to the field of 3D Computer Vision, and the extensions of our work that we envision.

## 7.1 Conclusions

**Range Segmentation** The two algorithms we designed place an emphasis on different types of features. Edge-based segmentation extracts precise 3D linear and circular edges, in order to provide high order edge-based features feasible for matching. The results of this method are used to assist feature-based registration. Region-based segmentation partitions the range image into meaningful planar and non-planar regions. The result of this method is used for modeling. The segmented regions are also potentially useful in the future for building semantic structures for the scene, and applicable in object recognition and scene understanding.

**Range Registration** We introduced a linear feature-based algorithm and a circular feature-based algorithm, both producing high quality registration results. Linear features and circular features occur frequently in most urban buildings, and can

be robustly identified, as compared to other geometric shapes. Therefore, we predict wide applicability of our algorithms to urban structure modeling.

**Model Generation** For generating surface models of the registered range data, we first merged corresponding segments from different range images, and then built a mesh for each meaningful segment using the Ball Pivoting Algorithm [Bernardini *et al.*, 1999]. The results are a set of coherent segments, corresponding to meaningful objects in a complete large scale urban scene.

The algorithms summarized above cover all the key technical phases of the 3D modeling system for large scale urban buildings. These algorithms have been successfully applied to modeling many landmark buildings in New York City, and efficiently generated highly accurate 3D models. These models have also been utilized in another on-going project of our lab: Generating photo-realistic 3D models [Liu and Stamos, 2005] [Liu *et al.*, 2006]. The research work done, as described in this thesis, has been presented at international conferences, 3DIM and 3DPVT, and received much attention and recognition. We believe that we have built an arsenal of methods, which can be utilized for the automatic 3D modeling of large-scale urban scenes.

## 7.2 Future Extensions and Applications

Our current achievement is the first step leading to the building of a fully automated and versatile 3D modeling system. In this system, as we envision it, different levels of geometric features will be extracted from segmentation, and utilized for intelligent registration. The registered images and their segments will then be used for building 3D models. These models, in turn, will be utilized in other computer vision tasks. To reach this high goal, more work will need to be done and the following issues resolved:

**Analysis of More Geometric Features** It would be useful to include more geometric shapes into range image analysis, e.g., ellipses, spheres, cylinders and cones. Segmentation and registration algorithms based on multiple geometric features and other statistical descriptors need to be developed. With these, the applicability of range image analysis can be expanded to buildings containing various types of geometric structures, thereby producing more meaningful segmentations and more accurate registrations for a larger variety of natural scenes.

**Computer Vision on Urban Structures** The segment models from registered scenes enable us to represent large scale urban structures in a more descriptive way. As an extension of this work, we envision a computer vision system, where we can

explore the construction of the semantic structures from the extracted meaningful regions, aiming at automatic intelligent understanding of urban scenes. There are two directions we can move towards. The first one involves the analysis of the spatial relationship of extracted planar and non-planar regions, aiming at higher level scene understanding. The other one involves feature extraction from each individual non-planar region, for the purpose of 3D shape recognition and object matching.

**Improving Algorithm Robustness** On technical details, it would be of great value to improve the estimation of thresholds in our algorithms. Although they were reasonably estimated based on our range scan data, it would be a great advancement if they could be derived from any given range data automatically, through a preprocess of range image data analysis, obtaining basic information such as average point distance, minimum depth, maximum depth, and surface curvature variation. As an alternative solution, a feasible range and sample values for each threshold can be determined based on the preprocess, and then tested and refined. The programs with different thresholds are then run on different computers in parallel. The generated results are estimated with error metric, or judged by user, so that the best or most useful thresholds can be decided. As a common problem in many computer aided tasks, different threshold values can dramatically affect the performance of these algorithms, but automatically

selecting the optimal ones is still a non-trivial problem.

**Integrated System With GUI** It would be invaluable to extend the graphical user interface to contain more functionality. Although our goal is to automate the registration and modeling system, it is still preferable to make intermediate results accessible to users via a user friendly interface. This interface would provide an integrated system for user to view, process, adjust range images and segmented regions with flexible configurations and operations.

# Thesis Related Publications

[Chen and Stamos, 2005] C. Chen and I. Stamos. Semi-automatic Range to Range Registration: a Feature-based Method. *The 5th International Conference on 3-D Digital Imaging and Modeling*, June 2005.

[Chen and Stamos, 2006] C. Chen and I. Stamos. Range Image Registration Based on Circular Features. *Third International Symposium on 3D Data Processing, Visualization and Transmission*, June 2006.

[Chen and Stamos, 2007] C. Chen and I. Stamos. Range Image Segmentation for Modeling and Object Detection in Urban Scenes. *The 6th International Conference on 3-D Digital Imaging and Modeling*, August 2007.

[Stamos *et al.*, 2007] I. Stamos, C. Chen, L. Liu, G. Wolberg, G. Yu and S. Zokai. Integrating Automated Range Registration with Multiview Geometry for the Photorealistic Modeling of Large-Scale Scenes. *The Internation Journal of Computer Vision*, 2007.

# Bibliography


[1] A photograph of the Great Hall.
*http://www.nycago.org OrgansNYC html CityCollegeNY.html.*

[2] A photograph of the Grand Central Station.
*http://commons.wikimedia.org/wiki/*
*Image:Grand_Central_Station_Main_Concourse_Rectilinear_projection_Jan_2006.jpg.*

[3] A photograph of the Cooper Union building.
*http://www.gvccc.com/tours.east.html.*

[Antone and Teller, 2002] M. E. Antone and S. Teller. Scalable Extrinsic Calibration of Omni-Directional Image Networks. *Inernational Journal of Computer Vision*, Sept./Oct. 2002.

[Bab-Hadiashar and Suter, 2000] A. Bab-Hadiashar and D. Suter. Simultaneous Model Recovery and Segmentation for Range Image Analysis. *4th Asian Conference on Computer Vision*, 2000.

[Bellon *et al.*, 1999] O. R. P. Bellon, A. I. Direne and L. Silva. Edge Detection to Guide Range Image Segmentation by Clustering Techniques. *Proceedings of 1999 International Conference on Image Processing*, October 1999.

[Bellon and Silva, 2002] O. R. P. Bellon and L. Silva. New Improvements to Range Image Segmentation by Edge Detection. *IEEE Signal Processing Letters*, February 2002.

[Bernardini *et al.*, 1999] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin. The Ball-Pivoting Algorithm for Surface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, Oct.–Dec. 1999.

[Bernardini *et al.*, 2002] F. Bernardini, I. Martin, J. Mittleman, H. Rushmeier, G. Taubin. Building a Digital Model of Michelangelo's Florentine Pieta. *IEEE Computer Graphics and Applications*, January 2002.

[Besl and Jain, 1988] P. J. Besl and R. C. Jain. Segmentation Through Variable-Order Surface Fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, March 1988.

[Besl and McKay, 1992] P. J. Besl and N. D. McKay. A Method for Registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, February 1992.

[Bhandarkar and Siebert, 1992] S. M. Bhandarkar and A. Siebert. Integrating Edge and Surface Information for Range Image Segmentation. *Pattern Recognition*, 1992.

[Brown and Rusinkiewicz, 2004] B. J. Brown and S. Rusinkiewicz. Non-Rigid Range-Scan Alignment Using Thin-Plate Splines. *2nd International Symposium on 3D Data Processing, Visualization, and Transmission*, September 2004.

[Broz *et al.*, 1999] V. Broz, O. Carmichael, S. Thayer, M. Hebert and J. Osborn. ARTISAN: An Integrated Scene Mapping and Object Recognition System. *American Nuclear Society 8th Intl. Topical Meeting on Robotics and Remote Systems, American Nuclear Society*, April 1999.

[Brusco *et al.*, 2005] N. Brusco, M. Andreetto, A. Giorgi and G. M. Cortelazzo. 3D Registration by Textured Spin-Images. *International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, June 2005.

[Carmicheal and Hebert, 1998] O. Carmicheal and M. Hebert. Unconstrained Registration of Large 3D Point Sets for Complex Model Building. *Proceedings 1998 IEEE/RSJ International Conference On Intelligent Robotic Systems*, October 1998.

[Chen and Medioni, 1992] Y. Chen and G. Medioni. Object Modeling by Registration of Multiple Range Images. *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, April 1991.

[Djebali *et al.*, 2002] M. Djebali, M. Melkemi and N. Sapidis. Range-Image Segmentation and Model Reconstruction Based on a Fit-and-Merge Strategy. *Journal of Computing and Information Science in Engineering* , December 2002.

[Dorai *et al.*, 1997] C. Dorai, J. Weng and A. K. Jain. Optimal Registration of Object Views Using Range Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, October 1997.

[Fan *et al.*, 1987] T. Fan, G. Medioni and R. Nevatia. Segmented Descriptions of 3-D Surfaces. *IEEE Journal of Robotics and Automation*, Vol.RA-3, No.6, December 1987.

[Fitzgibbon, 2001] A. W. Fitzgibbon. Robust Registration of 2D and 3D Point Sets. *12th British Machine Vision Conference (BMVC)*, September 2001.

[Gelfand *et al.*, 2003] N. Gelfand, L. Ikemoto, S. Rusinkiewicza and M. Levoy. Geometrically Stable Sampling for the ICP Algorithm. *The 4th International Conference on 3-D Digital Imaging and Modeling*, October 2003.

[Gotardo *et al.*, 2004] P. F. U. Gotardo, O. R. P. Bellon, K. L. Boyer and L. Silva. Range Image Segmentation Into Planar and Quadric Surfaces Using an Improved Robust Estimator and Genetic Algorithm. *IEEE Transactions on Systems, Man, and Cybernetics — Part B: Cybernetics*, December 2004.

[Guidi *et al.*, 2005] G. Guidi, B. Frischer, A. Spinetti. 3D Digitization of a Large Model of Imperial Rome. *The 5th International Conference on 3-D Digital Imaging and Modeling*, June 2005.

[Han *et al.*, 1987] J. Han, R. A. Volz and T. N. Mudge. Range Image Segmentation and Surface Parameter Extraction for 3-D Object Recognition of Industrial Parts. *IEEE International Conference on Robotics Automation*, 1987.

[Hoffman and Jain, 1987] R. Hoffman and A. Jain. Segmentation and Classification of Range Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1987.

[Hoover *et al.*, 1996] A. Hoover, R. B. Fisher, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. Bowyer, D. W. Eggert and A. Fitzgibbon. An Experimental Comparison of Range Image Segmentation Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 1996.

[Huber and Hebert, 2003] D. F. Huber and M. Hebert. Fully Automatic Registration of Multiple 3D Data Sets. *Image and Vision Computing*, July, 2003.

[Huber, 2001] D. F. Huber. Automatic 3D Modeling Using Range Images Obtained from Unknown Viewpoints. *Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling*, May 2001.

[Jiang *et al.*, 1996] X. Y. Jiang, U. Meier and H. Bunke. Fast Range Image Segmentation Using High-Level Segmentation Primitives. *3rd IEEE Workshop on Applications of Computer Vision*, December 1996.

[Jiang, 2000] X. Jiang. An Adaptive Contour Closure Algorithm and Its Experimental Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, November 2000.

[Johnson, 1997] A. E. Johnson. Spin-Images: A Representation for 3-D Surface Matching. *Ph.D. thesis, Carnegie Mellon University*, August 1997.

[Johnson and Hebert, 1997] A. E. Johnson and M. Hebert. Recognizing Objects by Matching Oreiented Points. *IEEE Conference on Computer Vision and Pattern Recognition*, June 1997.

[Johnson and Kang, 1997] A. E. Johnson and S. B. Kang. Registration and Integration of Textured 3-D Data. *International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, May 1997.

[Krishnapuram and Gupta, 1992] R. Krishnapuram and S. Gupta. Edge Detection in Range Images through Morphological Residue Analysis *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 1992.

[Lee *et al.*, 1998] K. Lee, P. Meer and R. Park. Robust Adaptive Segmentation of Range Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, February 1998.

[Leica] Leica Geosystems.
*http://hds.leica-geosystems.com/*.

[Levoy *et al.*, 2000] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, D. Fulk The Digital Michelangelo Project: 3D Scanning of Large Statues. *Proceedings of ACM SIGGRAPH 2000*, July 2000.

[Liu and Stamos, 2005] L. Liu and I. Stamos. Automatic 3D to 2D Registration for the Photorealistic Rendering of Urban Scenes. *EEE International Conference of Computer Vision and Pattern Recognition*, June 2005.

[Liu *et al.*, 2006] L. Liu, I. Stamos, G. Yu,G. Wolberg, S. Zokai. Multiview Geometry for Texture Mapping 2D Images Onto 3D Range Data. *EEE International Conference of Computer Vision and Pattern Recognition*, June 2006.

[Loncaric, 1998] S. Loncaric. A Survey of Shape Analysis Techniques. *Pattern Recognition*, 1998.

[Makadia *et al.*, 2006] A. Makadia, A. Patterson IV, and K. Daniilidis. Fully Automatic Registration of 3D Point Clouds. *IEEE Conference on Computer Vision and Pattern Recognition*, June 2006.

[Marshall *et al.*, 2001] D. Marshall, G. Lukacs and R. Martin. Robust Segmentation of Primitives from Range Data in the Presence of Geometric Degeneracy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, March 2001.

[Masuda *et al.*, 1996] T. Masuda, K. Sakaue and N. Yokoya. Registration and Integration of Multiple Range Images for 3-D Model Construction. *Proceedings of the 1996 IEEE International Conference on Pattern Recognition*, 1996.

[Miller and Stewart, 1996] J. V. Miller and C. V. Stewart. MUSE: Robust Surface Fitting using Unibiased Scale Estimates. *IEEE International Conference on Computer Vision and Pattern Recognition*, June 1996.

[Mitra and Nguyen, 2003] N. Mitra and A. Nguyen. Estimating surface normals in noisy point cloud data. *Proc. of the 9th Annual Symposium on Computational Geometry*, 2003.

[Miyazaki *et al.*, 2000] D. Miyazaki, T. Ooishi, T. Nishikawa, R. Sagawa, K. Nishino, T. Tomomatsu, Y. Takase, K. Ikeuchi. The Great Buddha Project: Modeling Cultural Heritage through Observation. *Proc. of the Sixth International Conference on Virtual Systems and MultiMedia*, January 2000.

[Okatani and Deguchi, 2002] I. S. Okatani and K. Deguchi. A Method for Fine Registration of Multiple View Range Images Considering the Measurement Error Properties. *Computer Vision and Image Understanding*, July 2002.

[Osada *et al.*, 2001] R. Osada, T. Funkhouser, B. Chazelle and D. Dobkin. Matching 3D Models with Shape Distributions. *Shape Modeling International*, May 2001.

[Pulli and Pietikäinen, 1993] K. Pulli and M. Pietikäinen. Range Image Segmentation Based on Decomposition of Surface Normals. *Proceedings of the Scandinavian Conference on Image Analysis*, 1993.

[Quynh Dinh and Kropac, 2006] H. Quynh Dinh and Steven Kropac. Multi-Resolution Spin-Images. *IEEE Conference on Computer Vision and Pattern Recognition*, June 2006.

[Rusinkiewicz and Levoy, 2001] S. Rusinkiewicz and M. Levoy. Efficient Variants of the ICP Algorithm. *The 3rd International Conference on 3-D Digital Imaging and Modeling*, June 2001.

[Sharp *et al.*, 2002] G. C. Sharp, S. W. Lee and D. K. Wehe. ICP Registration Using Invariant Features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, January 2002.

[Shi and Malik, 2000] J. Shi and J. Malik. Normalized Cuts and Image Semgentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.22, No.8, August 2000.

[Stamos, 2002] I. Stamos. Geometry and Texture Recovery of Scenes of Large Scale. *PhD thesis, Columbia University*, 2001.

[Stamos and Allen, 2002] I. Stamos and P. K. Allen. Geometry and Texture Recovery of Scenes of Large Scale. *Journal of Computer Vision and Image Understanding*, November 2002.

[Stamos and Leordeanu, 2003] I. Stamos and M. Leordeanu. Automated Feature-Based Range Registration of Urban Scenes of Large Scale. *IEEE International Conference of Computer Vision and Pattern Recognition*, June 2003.

[Standford ICP] Comparison of ICP algorithms. *http://graphics.stanford.edu/s̃mr/ICP/comparison*.

[Tangelder and Veltkamp, 2004] J. Tangelder and R. Veltkamp. A Survey of Content Based 3D Shape Retrieval Methods. *Proceedings of the Shape Modeling International*, 2004.

[Thirion, 1996] J. Thirion. New Feature Points Based on Geometric Invariants for 3-D Image Registration. *International Journal of Computer Vision*, 1996.

[Unnikrishnan and Hebert, 2003] Ranjith Unnikrishnan and Martial Hebert. Robust Extraction of Multiple Structures from Non-uniformly Sampled Data. *Proc. of the 2003 IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, October 2003.

[Unnikrishnan *et al.*, 2006] R. Unnikrishnan, J. Lalonde, N. Vandapel, and M. Hebert. Scale Selection for the Analysis of Point-Sampled Curves. *3rd Int'l Symposium on 3D Processing, Visualization and Transmission*, June 2006.

[Wami and Batchelor, 1994] M. A. Wami and B. G. Batchelor. Edge-Region-Based Segmentation of Range Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, March 1994.

[Yokoya and Levine, 1989] N. Yokoya and M. D. Levine. Range Image Segmentation Based on Differential Geometry: A Hybrid Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, June 1989.

[Yu *et al.*, 1994] X. Yu, T. D. Bui and A. Krzyak. Robust Estimation for Range Image Segmentation and Reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, May 1994.

[Yu *et al.*, 2001] Y. Yu, A. Ferencz and J. Malik. Extracting Objects from Range and Radiance Images. *IEEE Transactions on Visualization and Computer Graphics*, Oct.–Dec. 2001.

[Zhang, 1994] Z. Zhang. Iterative Point Matching for Registration of Free-form Curves and Surfaces. *International Journal of Computer Vision*, 1994.