View Planning for Range Acquisition of Indoor Environments

Kok-Lim Low

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2006

ABSTRACT

KOK-LIM LOW: View Planning for Range Acquisition of Indoor Environments

(Under the direction of Anselmo Lastra)

This dissertation presents a new and efficient next-best-view algorithm for 3D reconstruction of indoor environments using active range sensing. A major challenge in range acquisition for 3D reconstruction is an efficient automated view planning algorithm to determine a sequence of scanning locations or views such that a set of acquisition constraints and requirements is satisfied and the object or environment of interest can be satisfactorily reconstructed. Due to the intractability of the view planning problem and the lack of global geometric information, a greedy approach is adopted to approximate the solution. A practical view metric is formulated to include many real-world acquisition constraints and reconstruction quality requirements. This view metric is flexible to allow trade-offs between different requirements of the reconstruction quality. A major contribution of this work is the application of a hierarchical approach to greatly accelerate the evaluation of the view metric for a large set of views. This is achieved by exploiting the various spatial coherences in the acquisition constraints and reconstruction quality requirements when evaluating the view metric. The hierarchical view evaluation algorithm is implemented in a view planning system targeted for the acquisition of indoor environments using a monostatic range scanner with 3D pose. The results show great speedups over the straightforward method used in many previous algorithms. The view planning system has also been shown to be robust for real-world application.

The dissertation also describes how the view metric can be generalized to incorporate general acquisition constraints and requirements, and how the hierarchical view evaluation algorithm can be generalized to scanners with general pose, and to scanners with bistatic sensors. A simple extension is also proposed to enable the hierarchical view evaluation

algorithm to take into account each view’s sensitivity to the potential pose errors in the physical positioning of the scanner.

A computed new view must produce a range image that can be accurately registered to the previous scans. In this work, a metric is developed to estimate the registration accuracy of the views. This metric considers the amount of overlap, the range measurement errors, and the shape complexity of the surfaces.

To my wife, Hsin-Yee,
who has never stopped believing in me.

ACKNOWLEDGEMENTS

This work would not have been possible without the help and support of many people. I am especially grateful to my advisor, Anselmo Lastra, for his guidance and encouragement throughout the many years of my graduate studies. He is a great advisor, teacher and friend. I thank him for his support, his understanding and his extraordinary patience.

I want to thank the members of my dissertation committee—Professors Guido Gerig, Dinesh Manocha, Marc Pollefeys, and Greg Welch—for their invaluable feedback and suggestions on my research and dissertation. I thank Greg Welch for the many interesting discussions and for his guidance, especially in my earlier years at UNC when I was working with him on the projector-based rendering and virtual reality projects. I also thank Henry Fuchs for suggesting the use of a scanned real-world model in one of the virtual reality projects. That was my first experience in working with range data.

John Thomas and Herman Towles gave me a lot of technical and engineering assistance on the experiments I did for this and other related work. I thank them for being always ready to lend a hand. I also thank the people at 3rdTech, especially Rich Holloway and Nick England, for their help and support with the DeltaSphere scanner. I thank Lars Nyland, the inventor of DeltaSphere, for helping me understand how the scanner works, and some of the principles and concepts of range sensing and 3D reconstruction. I thank Chad Hantak, Than Williams and Rui Wang for working together and sharing ideas on related problems. I want to thank the UNC Walkthrough Group for the beautiful Brooks' House model that I used in my experiments. I thank Andy Wilson especially, for spending time to tidy up the files of the Brooks' House model before passing them to me. I thank NSF for supporting this research work.

My friends in Chapel Hill offered me much-needed companionship and encouragement during the most stressful times in my doctoral studies. I am very grateful to Jingyu Yan, Zhu He, Catherine Tian, Ke Xu, Lily Shi, Ping Bai, Rawiwan Panjawatanakul, C. D. Poon, Adrian Ilie, Young Troung and Psyche Lee. I thank Young Truong and Ping Bai for helping me with the statistics problems I encountered in this project. I also want to thank those friends who were not with me in Chapel Hill but never stopped giving me their support and encouragement. They are Seet-Chong Lua, Cheaw-Fern Ong, James Chen, Yu-Chen Yeh, Ying-Chi Lin, Wei-Chao Chen and Irene Shang.

I thank my parents, my parents-in-law, my grandparents, Aunt Sanyi, my brothers and sister, Cousins Joanna and Wendy, and Bingyi for their love and support.

Finally, I thank my wife, Hsin-Yee, for always believing in me. Without her, I could not have done this.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Active range sensing is being used in a wide range of applications. In manufacturing inspection, manufactured parts are scanned from a set of different views, and the acquired range data provides information on the surface geometry to allow detection of possible defects on the parts. In CAD/CAM, old parts are often reverse-engineered by making range scans of them and using the range data to reconstruct digital 3D models of the parts. Other applications include object recognition in computer vision, and visual exploration and navigation in robotics.

In computer graphics, traditionally, 3D digital models of objects and environments have been modeled using CAD-like modeling tools or are procedurally generated [Foley1992]. When it comes to modeling complex real-world objects with large amounts of geometric details, these approaches often prove inadequate and impractical. This is the main reason we are beginning to see, in the computer graphics community, the increasing use of active range sensing to reconstruct 3D models of real-world objects.

To create a high-quality 3D reconstruction of an object or environment, a huge amount of range data has to be acquired, stored and processed. Even just a decade ago, this was a major challenge. Today, the situation has improved considerably, thanks to the advancement in range scanning technology, the great increase in affordable computer processing power and storage, the inventions of efficient algorithms, and the availability of powerful specialized software tools. As an indicator of the high reconstruction quality that can be achieved today, in the Digital Michelangelo Project [Levoy2000, Levoy2003], the digital model of the 23-foot tall statue of David consists of two billion polygons, and was created from 480 range scans made at a sampling resolution of 0.29 mm and depth resolution of 50 microns.

Multiple range scans from different scanning locations are almost always necessary to reconstruct a fairly complete model of a geometrically-nontrivial object or environment, since it is impossible to acquire the entire object or environment from any single scanning location. This is mainly due to occlusions, sensing surfaces at grazing angles, and the imaging limitations of the range scanner, such as its limited field of view and depth of field. They result in holes in the model. Figure 1.1 shows an example of a model made from a single simulated scan of a synthetic indoor environment. These holes must be filled in by additional scans made from different locations. The range data from the multiple scans are then merged together in post-acquisition processing to obtain a more complete 3D model of the object or environment. Figure 1.2 shows two views of a model constructed by merging range data from eight different scans.

The set of scanning locations must be chosen carefully so that a 3D digital model of the object or environment can be satisfactorily reconstructed with minimal number of scans. This task is called *view planning*, and generally, in addition to the scanning locations, it also involves the determination of the scanner's orientations and imaging parameters. A scanner's pose (position and orientation) and its associated imaging parameters are collectively called a *view*. Usually, there are many constraints and requirements that have to be considered when selecting a view. For example, the scanner must never be placed too close to any surfaces, and the surfaces to be scanned must be within the limited field of view of the scanner.

View planning is not only important to the acquisition of range data, it is also essential for the efficient acquisition of colors and surface reflectance properties of objects or environments [Lensch2003]. Both geometric information and surface reflectance properties are necessary for realistic reconstruction. However, the work described in this dissertation is concerned only with the acquisition of surface geometry using range scanners.

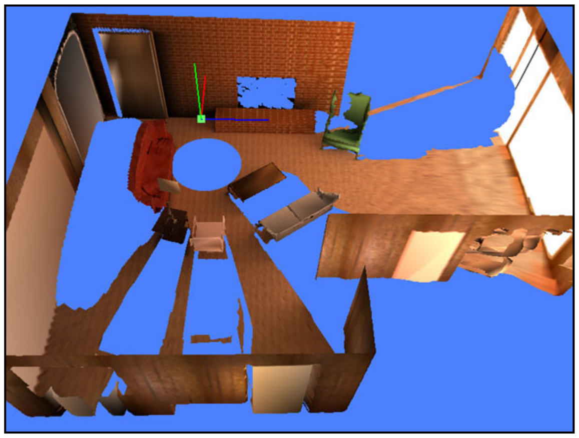


Figure 1.1: A model produced by a single simulated range scan of a synthetic indoor environment. The holes in the model are the results of occlusions, sensing at grazing angles, over-absorbent surfaces, and the limited field of view of the scanner.

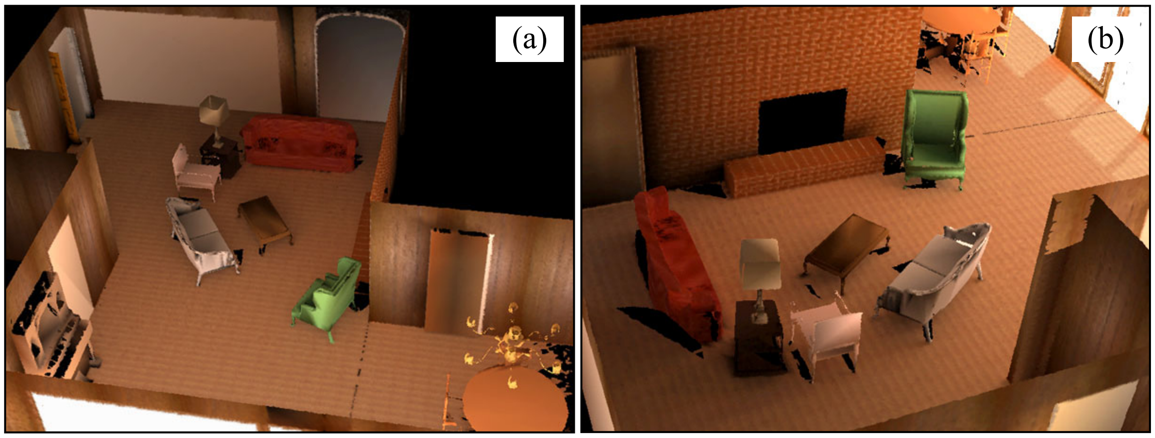


Figure 1.2: Two different views of a model reconstructed from eight range scans of the same synthetic indoor environment shown in Figure 1.1. Except for the first location, seven of the eight different scanning locations are planned by an automated view planner. The reconstructed model has significantly fewer and smaller holes than those from any single scan.

1.1 Motivation

Traditionally, view planning for range acquisition relies heavily on human decisions. Unless the object to be scanned is geometrically very simple, the set of selected views can be very inefficient, i.e. far more than the optimal, and they may not satisfy the *reconstruction quality requirements*. This can be a serious problem because by the time the deficiencies are discovered, the site or the object may no longer be readily accessible. One possible way to improve the manual view planning is to provide online visualization feedback to the human operator.

With online visualization feedback, after each new scan is made, it is immediately registered and merged with the partial model. The updated partial model is displayed interactively to the human operator, who can then observe the effect of each new scan on the partial model. Effective visualization should also provide other relevant information that can help the human operator plan the next scanning view. For example, the partial model's surfaces could be color-coded to indicate which regions have not met the quality requirement.

One of the major challenges is to have very efficient registration, merging, and rendering of the partial model, so that the human operator does not need to wait too long for the visualization feedback. Rusinkiewicz has implemented a real-time model acquisition system [Rusinkiewicz2002] that allows range scans to be made at video rate and provides real-time visualization feedback to the human operator. While it is claimed that real-time visualization feedback is effective for manual view planning to completely scan an object, the acquired scans can be excessively too many. In order to maintain reliable registration, the whole system pipeline has to run at very high rates, and that has limited the system to be feasible only for small objects.

Humans are relatively good at high-level view planning for coverage of simple objects, but even experienced operators will encounter considerable difficulty with topologically and geometrically complex shapes [Scott2003]. Online visualization feedback may be helpful, but manual view planning is generally a slow and error-prone process [Levoy2000]. When multiple constraints and requirements have to be considered, it can become overwhelming or impossible for the human operators to produce a reasonably good view plan.

Automated view planning might be able to take over the difficult view planning task from human operators. There have been some successes [Pito1996a, Reed1997, Sequeira1996, González-Baños1999, Nüchter2003], but most of them were only demonstrated to work for relatively simple objects or environments. Their problems and solutions were often over-simplified, where important practical *acquisition constraints* and *reconstruction quality requirements* were left out, and only a very small and incomplete set of views was evaluated. Nevertheless, by using automated view planning, it becomes possible to fully automate the entire model acquisition process if the scanner can be positioned by a software-controllable positioning system, and the scanner's imaging parameters can be set by software too. In a semi-automatic system, the automated view planner may have to provide a visual guide to tell the human operator where to position, how to orient, the scanner and how to set the scanner's parameters.

1.1.1 Reconstruction of Environments

Most of previous reconstructions have been on single small to mid-sized objects. With great improvements in the sampling density and depth precision of many of today's long-range scanners, together with the reduction in size and weight, reconstruction of environments have become more common. These environments include building interior, building exteriors, natural environments, and even part of a city. High-quality environment reconstruction has many important applications, such as building 3D digital archives of heritage sites, generating digital models of architectural structures for architecture and engineering studies, creating virtual worlds for virtual reality, and recording crime scenes for law enforcement purposes.

Range acquisition of real-world environments is generally more difficult than for objects. Unlike a small object, which can be brought to a laboratory for range acquisition, scanning a real-world environment requires the scanner and other equipment to be transported and set up at the location. The scanning team usually has less control over the scanning conditions in an environment than in a laboratory. For example, in an outdoor environment, there may be people or cars moving around, and weather may be unsuitable. The demand on the scanner positioning system may be high too. For example, when scanning a mid-sized interior, an ideal positioning system should be able to reach every location in the empty space if all visible surfaces are to be acquired. Real-world environments can be cluttered with many objects, which results in high visibility complexity. This makes view planning very difficult. Moreover, to fully automate the acquisition process, an obstacle-free path has to be planned when moving the scanner from one viewpoint to the next.

In this work, the goal is to automate view planning for the range acquisition of real-world indoor environments to reconstruct models for virtual reality walkthrough. Most common indoor environments, such as offices and residential dwellings, are complex enough and challenging for automated view planning. Compared to most outdoor environments, acquisition conditions inside indoor environments are generally much easier to control. For example, it is easier to keep people out of the region to be scanned, keep objects from being shifted and moved, and remove or cover up objects that cannot be scanned well (for example large and shiny objects). Moreover, the acquisition will not be affected by the change of weather. This allows the research to better focus on the view planning issue and not be concerned with other unrelated issues. Chapter 3 gives details about the requirements on the indoor environments.

1.2 Automated View Planning

As mentioned before, the goal of this work is to automate view planning for the range acquisition of real-world indoor environments to reconstruct 3D digital models for virtual reality walkthrough. For 3D reconstruction, a priori knowledge of the environment's geometry is not available to the view planner. The first scan is made from a view appropriately selected by a human operator, and for the subsequent scans the planner must determine the next best views based on the information it has already collected from the previous scans. This is why the view planning problems for 3D reconstruction are often called the *next-best-view problems*.

Since global geometric information is unknown, a next-best-view problem cannot be solved globally to get an optimal solution. It is inherently a local optimization problem [Kakusho1995]. This local problem is NP-hard, and is often solved approximately using a greedy-approach approximation algorithm [Scott2003]. Using the greedy approach, the view planning solution operates in a tight iterative approach as part of the *model acquisition cycle* shown in Figure 1.3. During a typical acquisition session, the model acquisition cycle is repeated multiple times. In each cycle, after a scan is made, it is first registered (or aligned) with the partial model of the environment. Then the information of the new scan is merged into the partial model, and this updated model is used for the computation of a new view for the next scan.

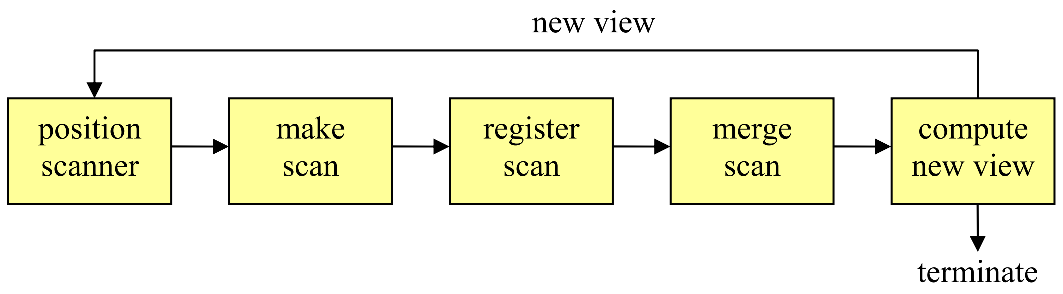


Figure 1.3: The model acquisition cycle with automated view planning.

A greedy next-best-view algorithm computes by evaluating a *view metric function* for each view, and outputting the view that maximizes the metric function as the best view for the next scan. Each view may consist of several parameters, and thus the solution space may be high-dimensional. The view metric must be designed to take into account the requirements on the reconstruction quality and the many acquisition constraints. Some of the common reconstruction quality requirements are the completeness of coverage and the sampling quality of the acquired surfaces. Examples of acquisition constraints include the visibility between the scanner and the surface to be acquired, the limited field of view and depth of field of the scanner, and the minimum clearance distance between the location of the scanner and any object in the environment.

The major challenge to a practical next-best-view solution is to develop an efficient method to evaluate the view metric function for a large set of views, using information provided by a partial model of the environment. The evaluation of each view can be computationally very expensive, since a large amount of information of the partial model may be involved, and visibility computations and constraint evaluations are expensive. This is the main difficulty that has limited many previous next-best-view algorithms to incomplete search space, simple and small objects, incomplete set of constraints, and low-quality acquisition. Some early algorithms even ignore self-occlusion of the objects [Connolly1985].

1.3 Contributions

One of the major contributions of this work is the novel application of a hierarchical approach to greatly accelerate the evaluation of the view metric function for a large set of views. This is achieved by exploiting the various spatial coherences in the acquisition constraints and reconstruction quality requirements when evaluting the view metric. For example, if an acquisition constraint is satisfied between a view *v* and a surface point *p*, then usually the same constraint is also satisfied between *p* and other views in the neighborhood of *v*, and likewise, between *v* and other surface points in the neighborhood of *p*. This two-way spatial coherence can be exploited by grouping adjacent views and surfaces in the partial model into view volumes and surface patches, respectively, and adaptively subdividing them during evaluation according to the change of spatial coherence of the constraints and requirements. The algorithm has been implemented in a view planning system and has demonstrated great speedups (one to two orders of magnitude) over the straightforward view evaluation method used in previous next-best-view algorithms.

In the experiments described in this work, the hierarchical view evaluation approach is applied to my target next-best-view problem in which the range scanner is monostatic and the views are 3D positions with fixed orientation. A monostatic range scanner has only a single viewpoint, as opposed to a scanner based on triangulation, which has two or more viewpoints. This dissertation further describes how the hierarchical approach can be generalized to scanners with more general pose, and to scanners with bistatic sensors (for example, a triangulation-based range sensor). The dissertation also describes how general acquisition constraints and requirements can be incorporated into the hierarchical approach.

The dissertation also proposes a simple extension to the hierarchical view evaluation algorithm to take into account each view's sensitivity to the potential pose errors in the physical positioning of the scanner. To the best of my knowledge, this is the first next-best-view algorithm that directly incorporates views' sensitivity to pose errors in the computation of new views.

Another contribution of this work is the design and formulation of a simple, general and practical view metric that is able to include many real-world acquisition constraints and reconstruction quality requirements. This metric is also flexible to allow trade-offs between different requirements of the reconstruction quality.

One of the most important acquisition constraints is that the computed new view must produce a range image that can be accurately registered to the partial model. The registration is essential to localize the actual pose of the scanner, and also to allow the new range image to be correctly merged to the partial model. However, this registration is not guaranteed to be successful. Factors that affect the registration accuracy between two surfaces are (1) the amount of overlap between them, (2) the shape constraint on the 3D rigid-body transformation between the two surfaces, and (3) the range measurement errors. In this work, a *registration accuracy metric* has been derived to estimate the registration error, and is used to ensure that the new scan to be acquired from the planned view can be successfully registered with the previous ones. The registration accuracy metric considers all three factors that affect the registration accuracy.

To the best of my knowledge, this work is the first to be able to exhaustively evaluate a large set of 3D views with respect to a large set of surfaces, and to include many practical acquisition constraints and reconstruction quality requirements, especially the sampling quality requirement, and the registration constraint that considers surface shape complexity and range measurement errors.

1.4 Thesis

My thesis is

> *Efficient and practical next-best-view computation for range acquisition of indoor environments can be achieved using a hierarchical approach that exploits the various spatial coherences in the acquisition constraints and reconstruction quality requirements.*

In support of my thesis, I have investigated and implemented a next-best-view planning system that takes into account the many practical real-world acquisition constraints and reconstruction quality requirements. I have implemented the hierarchical approach of view evaluation to exploit the various spatial coherences in the acquisition constraints and quality requirements. Simulations have been performed, and the results show great speedups over the straightforward view evaluation method used in many previous next-best-view algorithms. Typical speedups range from 10 to 100 times. The system has been tested on a real indoor environment using the DeltaSphere-3000 range scanner [DeltaSphere], and has been shown to be robust for real-world practical use. The results of the view planning for both the real acquisition and simulations have been very satisfactory. Figure 1.4 shows the sequence of five views computed for the acquisition of a real indoor environment, and the resulting model constructed from the five scans.

1.5 Organization

This dissertation is organized as follows:

- Chapter 2 provides brief introductions to the active range sensing technologies and the 3D reconstruction process, and reviews the previous work on view planning for 3D reconstruction and other applications.
- The first two sections of Chapter 3 introduces the general view planning problem and defines the specific view planning problem that concerns this work. The later sections present an overview of the proposed next-best-view algorithm and describe in detail the view metric and some of the major components of the solution.
- One of the major solution components is presented in Chapter 4, which describes how the hierarchical view evaluation method is used to efficiently evaluate the acquisition constraints and reconstruction quality requirements. In the later part of the chapter, the hierarchical approach is generalized for more general scanning pose, acquisition constraints and quality requirements.
- Chapter 5 addresses the issues regarding the registration of a new range scan to the partial model. This is another major component of the next-best-view solution. The first part explains how registration is performed in the view planning system, and the later section derives some conditions to pre-determine whether the range scan produced at a view can be successfully registered. The chapter uses an example to

illustrate how the registration conditions can be used and to demonstrate the effectiveness of the conditions.

- In Chapter 6, three sets of results are presented to demonstrate the use of the next-best-view planning system for the range acquisition of indoor environments. Two sets are from simulations and one set from scanning a real indoor scene.
- Chapter 7 discusses some issues not addressed in the earlier chapters, describes directions for future work, and concludes the dissertation.

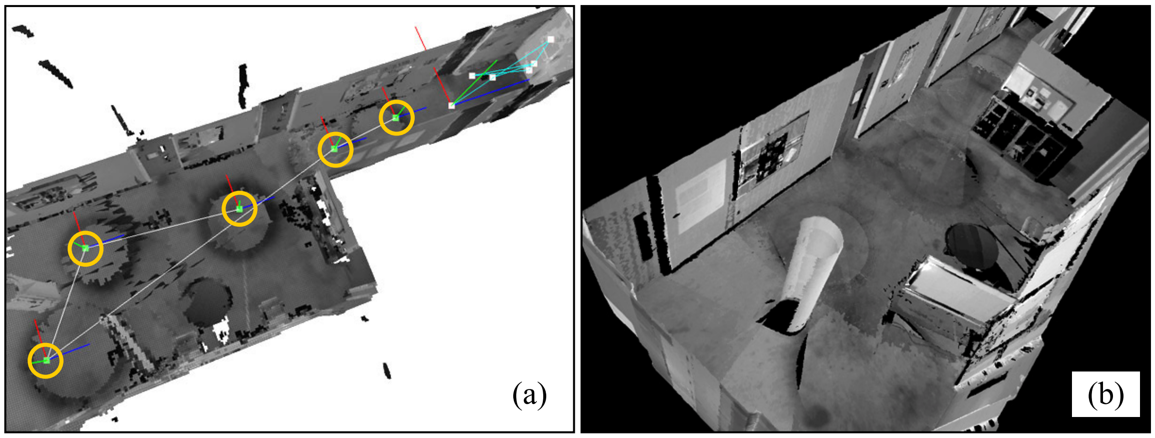


Figure 1.4: The results of the acquisition of a real indoor environment using five 3D views computed by the next-best-view planning system. (a) The chain of five scanner views from which the five scans were made. These views may be at different heights from the floor. (b) The triangle mesh model constructed by merging the five scans.

Chapter 2

Background

The work described in this dissertation assumes the use of an active range sensing device to obtain the geometric information of an environment. The next-best-view algorithm analyzes the partial geometric information to derive views for acquiring new geometric information. After the range data are sufficiently acquired, they usually need to go through a 3D reconstruction process in order to be properly integrated into a 3D digital model.

This chapter provides a brief introduction of 3D modeling from active range sensing, and reviews the previous work on view planning. View planning is used in the data acquisition stage of the 3D reconstruction process, which is the first stage of the process.

2.1 3D Modeling From Active Range Sensing

There are several ways to create a detailed 3D digital model of a real-word object or environment. Some objects have geometric shapes that can be accurately described by a set of mathematical formulae or rules, which can be readily programmed into a computer to have the 3D models generated. However, there are still many objects that cannot be conveniently modeled using this approach. These objects require explicit measurements of their surface geometry. The simpler of these objects may just require position measurements of only a small well-chosen set of surface points. The position measurements are then used on a CAD model to determine the positions of the modeling primitives. On the other hand, objects with high geometric complexity will require dense measurements of their surfaces.

Active range sensing has become a common approach to make dense, detailed geometric measurements of complex real-world objects and environments. With an adequately dense set of measurements of an object, geometric details can be accurately reconstructed on the 3D digital model. Active range sensing is described in more details in the following subsection, and the next subsection explains the 3D reconstruction process in which a 3D model is constructed from the set of range measurements.

2.1.1 Active Range Sensing

There are two classes of *non-contact measurement* techniques for acquiring 3D surface geometry of objects—*passive sensing* and *active sensing*. Passive sensing techniques attempt to imitate the way the human eyes work. Light (or other form of energy) is received by the sensing device from the environment, and no energy is emitted for the purpose of sensing. Examples of passive techniques include stereo vision and shape-from-X techniques (e.g. shape-from-shading) [Trucco1998, Forsyth2002]. These techniques can only work well under restrictive ideal conditions, and are not suitable for most general class of objects and environments. For example, in stereo vision, the stereo matching problem is ill-posed in general, and the matching process can break down if the object's surface is insufficiently textured.

Active sensing techniques overcome the problems of passive sensing by emitting structured light (or other form of energy) in the direction of the object or environment to be digitized. A detector then receives the structured light that is reflected from the object's surface. Active sensing can be classified into *triangulation* and *time-of-flight* systems.

Triangulation range sensors work on the principle of triangulating a measurement spot on the object from a pair of physically separated structured light source and light detector. Triangulation-based range sensors are capable of very precise (less than 100 micrometers [Blais2003]) depth measurement, but they usually have a very short standoff distance (about a meter). Therefore, they are more suitable for measuring small objects.

Time-of-flight range sensing is based on the measurement of time delays generated by light traveling in a medium. In a *pulsed-wave time-of-flight* system, the distance is computed from the time taken for the emitted light pulse to travel from the source to the target surface and get reflected back to the source. Very accurate timing sources are required and depth precisions (usually around centimeters) are much lower than those of triangulation-based systems [Blais2003]. Time-of-flight systems based on *continuous wave modulation* get around the measurement of short pulses by modulating the power of the laser beam. Time delay is computed from the phase difference between the emitted and received waveforms[1]. Using continuous wave modulation, sub-millimeter depth precision can be achieved over long distance (10 to 100 meters) [Blais2003]. Time-of-flight systems are best suited to distance measurement and environment modeling at medium to long ranges.

Time-of-flight range sensors are usually *monostatic* sensors, meaning that the light emitter and detector are at almost the same location. A surface point on an object can be measured by a monostatic sensor if it is visible to the single location where the emitter and detector are located. Sensors based on triangulation are *bistatic* or *multistatic* because the light emitter and detector must be at different positions. A surface point can be measured only if it is simultaneously visible to both the light emitter and detector.

Most of the active range sensors that use laser emit only spot or profile structured light. This allows only one point or a profile of points on the objects to be measured at a time. In order to obtain a two-dimensional range map, the spot or profile structured light has to sweep, or "scan" across an imaginary two-dimensional imaging plane. This is usually done using rotating mirrors, and/or a mechanical pan-tilt unit to steer the laser source. Such devices are often called *range scanners*. When a scan is complete, a *range image* is produced.

A range image is an *organized point cloud*. In an organized point cloud, each point's connectivity with its neighboring points is known and is usually implicit. In an unorganized point cloud, each point's relationship with its neighboring points is unknown. Organized point clouds are produced by systematically sweeping the structured light source across the imaginary two-dimensional imaging domain.

Informative surveys of the different active range sensing technologies can be found in [Besl1989], [Beraldin2000] and [Blais2003]. A short review of the different types of time-of-flight sensing techniques is presented in [Sequeira1999]. A fairly exhaustive list of optical 3D sensors manufacturers is provided in [Papadopoulos2001].

[1]Continuous wave modulation is used in the DeltaSphere-3000 3D Scene Digitizer [DeltaSphere]. This range scanner is used in experiments described in this dissertation.

2.1.2 The 3D Reconstruction Process

The 3D reconstruction process produces a 3D digital model by acquiring and integrating multiple sets of range data of an object or environment. It is assumed that each set of range data is a range image—an organized point cloud—and this section focuses on reconstruction methods for organized point clouds.

The process of 3D reconstruction of an object or environment typically consists of the following steps, as seen in Figure 2.1: (1) range acquisition, (2) range image registration, (3) merging of range images, and (4) post processing. The registration, merging and the post processing are usually very time-consuming, therefore they are normally done offline, after all the range scans have been acquired. Details of the steps are given next, and more detailed examinations of some of the 3D reconstruction steps can be found in [Forsyth2002], and in [Hilton1997a].

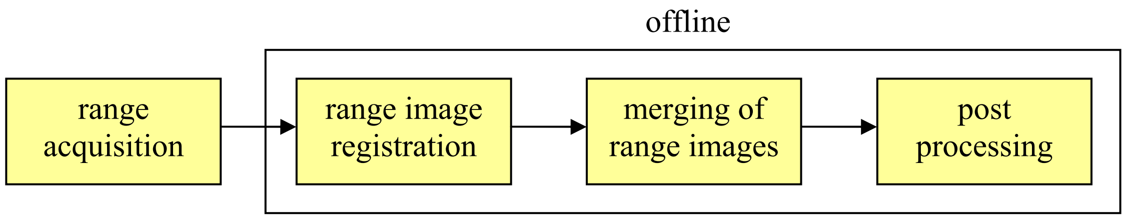


Figure 2.1: Steps in the 3D reconstruction process.

(1) **Range acquisition.** In the range acquisition step, the range scanner is positioned at a set of different poses (and/or the object is moved with respect to the scanner) to scan different parts of the object. Imaging parameters of the scanner can be set differently at each pose. For example, the sampling density and field of view can be adjusted appropriately for each pose. A range image is produced when the scanner makes a "scan" of the object from a given pose and obtains a 2D array of range points expressed in the scanner's local imaging coordinate frame. The objective of the acquisition step is to acquire range images that cover as much of the object's surface as possible. This almost always requires multiple scans to be made due to the self-occlusion by the object, and also due to the imaging limitations of the scanner, such as a limited range and a limited field of view. If the 3D model to be

reconstructed has to meet some quality requirements, then the acquisition process must attempt to satisfy them. For example, to achieve a certain minimum sampling density on all visible object surface regions, the scanner must be placed at an appropriate distance and orientation from each surface region. If the range images have to be registered after the acquisition step, then it must be ensured that every scan has enough overlapping regions with at least one or two other scans, and the overlapping regions must have sufficient geometric complexity to constrain the relative 3D rigid-body transformation between the range images. To ensure that the constraints and requirements are satisfied, proper planning of the scanner's positions and orientations (and imaging parameters) is necessary.

(2) **Range image registration.** The next step is to register the multiple range images, so that they are aligned with one another in a common coordinate frame. During the range acquisition step, if the pose of the scanner is precisely known when each scan is being made, then it is not necessary to register the range scans, because the scans just have to be appropriately transformed into a common coordinate frame. However, this is normally not the case because the positioning and pose measurement systems always have errors, and their accuracies are usually much less than the depth accuracy and sampling density of the scanner.

In order to achieve successful registration of all the range images, every range image must have sufficient overlapping regions with at least one other range image and the overlapping regions must have sufficient shape complexity. To start the registration, most registration algorithms require all the range images be already roughly aligned [Chen1992, Nishino2002]. This rough alignment may come from the approximate pose data provided by the tracking system or the positioning system of the scanner, or it may come from a human operator who manually positions the range images to roughly align them. After that, an iterative optimization method is usually used to achieve more precise alignment among all the range images [Pulli1999, Bergevin1996, Nishino2002]. Many of such registration algorithms that simultaneously align multiple range images are based on the idea of the Iterative Closest Point (ICP) algorithm [Besl1992, Chen1992, Rusinkiewicz2001]. The ICP algorithm is used for pair-wise registration of only two surfaces.

(3) **Merging of range images.** After the scans have been registered, there is redundant data at the overlapping regions. The objective of the merging step is to combine surface information in all the scans into a single non-redundant model. An ideal merging algorithm should take into consideration the physical characteristics of the scanner and the measurement quality or uncertainty of each individual range point, so as to create the most plausible model. Many of the merging methods for organized point clouds can be classified as surface mesh integration approaches [Turk1994, Boissonnat1984, Rutishauser1994, Soucy1995, Pito1996b], or volumetric approaches [Curless1996, Hilton1997, Roth1997].

The zippering method [Turk1994] proposed by Turk and Levoy is a surface mesh integration approach. Each scan is first converted to a triangle mesh. The redundant surfaces at the overlapping regions between each pair of meshes are removed by eroding their boundaries until they just meet. Then, the boundary triangles on one mesh are clipped against those on the other mesh. Next, the redundant clipped triangles are removed, and the two triangle meshes are joined together by re-triangulation of the boundary region. After all the meshes have been zippered together, each vertex in the final model is moved to a consensus position given by a weighted average of positions from the original range images.

An example of a volumetric approach is the method proposed by Curless and Levoy [Curless1996]. Each range image is first scan-converted to a discrete signed distance function represented in a 3D uniform grid. Then, one at a time, each signed distance function is weighted by its corresponding weight function and accumulated into a resulting signed distance function. The weight function represents the measurement confidence at each point of the range image. Finally, using the marching cubes algorithm [Lorensen1987], the final model is generated by extracting an isosurface from the volumetric grid of the resulting signed distance function.

The resulting 3D model from the above merging methods is a polygonal boundary representation of the 3D object.

(4) **Post Processing.** Many types of post processing operations can be done to the merged model. Some of the common ones are hole filling, simplification, surface

fittings, and surface fairing. Holes on the reconstructed model can come from a few sources. Self-occlusion by the object, together with the physical limitations in positioning the scanner, is one unavoidable cause. Another cause is the difficult reflectance properties on the object surface—both overly-absorbent and overly-reflective surface can cause “drop outs,” that is, missing samples in the range images. Another cause of holes comes from human errors. It may be due to human carelessness that some visible parts of the objects are just missed during range acquisition. Holes in the reconstructed model are difficult to patch up automatically, unless additionally knowledge about the object can be provided to the hole-filling algorithm. This knowledge can be provided interactively by a human operator [Wang2002], or the algorithm just uses some heuristic assumptions about the missing surfaces [Wang2003a, Davis2002].

If the merged model is a polygon mesh, it usually consists of an excessive number of polygons. This model can be optimized by reducing the number of polygons at low-curvature regions [Schroeder1992, Garland1997]. Another form of optimizing or approximating the merged model is to fit higher-order surface primitives to the model’s surface. A good review of some of these methods can be found in [Söderkvist1999]. The model may also be smoothed by some surface fairing methods, for example [Taubin1995], to reduce the effects caused by noise in the range images.

2.2 Previous Work on View Planning

This section reviews previous approaches to the view planning problems. View planning problems can be broadly classified into two groups: (1) *non-model-based view planning* (NMBVP) problems or next-best-view (NBV) problems, and (2) *model-based view planning* (MBVP) problems. In contrast to non-model-based view planning or next-best-view planning, model-based view planning operates with a priori knowledge of the geometric information of the objects or environments of interest. It is often employed in automatic inspection of manufactured parts where CAD models of the parts are already available before the actual manufacturing and inspection. Although the next-best-view problems are the focus of this work, their similarities with the MBVP problems become apparent when each iteration of the NBV problems is recast as a problem in scanning well-defined areas that represent missing data in the current geometry [Hall-Holt1998].

In this section, reviews of the previous work on view planning are divided into four application areas. For non-model-based view planning, the two areas are 3D reconstruction of objects and 3D reconstruction of indoor environments, whereas for model-based view planning, the two areas are object inspection and image-based modeling and rendering. The four application areas are listed in the following table.

	Application Areas
Non-model-based view planning	• 3D reconstruction of objects • 3D reconstruction of indoor environments
Model-based view planning	• Object inspection • Image-based modeling and rendering

The reviews focus on the view evaluation approaches taken by the different view planning algorithms, the dimensionality of the views, the solution space that is explored, and the different acquisition constraints and requirements that are considered by the view planning.

There are a few past surveys of the research work in view planning. Scott et al. [Scott2003] offered a thorough definition of the view planning problem in their comprehensive survey of the existing view planning approaches. The focus is on automated 3D object reconstruction and inspection by means of active, triangulation-based range sensors. The paper describes the components of a typical imaging environment, the steps in a reconstruction cycle, the assumptions, requirements and constraints, and some possible performance measures. It then surveys and compares different techniques for automated 3D object reconstruction and inspection. Work on view planning for environment reconstruction has been intentionally excluded. The authors classified the view planning problems into two top-level categories—model-based and non-model-based, and further subcategorized the different techniques according to the particular geometric representation used.

The survey by Tarabanis et al. [Tarabanis1995] is probably the most cited view planning survey paper. The paper initially considers three vision tasks—inspection, object recognition, and scene/object reconstruction, but the reviewed papers are exclusively on inspection via conventional intensity imaging. The focus is on the determination of camera and illumination poses and parameters, so that important object features can be robustly detected. The view planning methods are categorized into the generate-and-test approach, the synthesis approach, and expert systems. Many of the papers described in their paper are summarized in Section 2.2.3.

Other short reviews of papers on the next-best-view problems have been written by Hall-Holt [Hall-Holt1998] and Hilton [Hilton1997a], and they are found on the World Wide Web.

2.2.1 3D Reconstruction of Objects

Almost all existing view planning methods for 3D object reconstruction employ the straightforward greedy approach by attempting to search for the view that maximizes a view metric function based on the currently known information about the object. The greatest difference among these methods may be their view metric functions, which reflect what constraints and requirements are being considered in the view planning. Other differences include the view space considered, the sampling of the view space, the methods of evaluation of the metric functions on the sample views, and the representations of the partial models.

The paper by Connolly [Connolly1985] is probably the first paper on next-best-view planning for 3D reconstruction of objects from range images. The goal is to determine a set of “covering views” of a given object. Connolly took the greedy approach. An occupancy octree is used to represent the partial model. The range sensor is assumed to always point towards the center of the object, and the solution at each iteration is a direction for placing the sensor. The view space considered is therefore only two-dimensional.

Connolly described two algorithms to search for the next best view—one is called the planetarium algorithm, and another the normal algorithm. In the planetarium algorithm, a sphere is set up around the object. The sphere is evenly sampled along the latitudes and longitudes, and at each sample point, a hidden-line image of the octree partial model is generated from only the surface and unseen-space voxels. The area of the visible unseen-space voxels in the simulated image is determined and provides a measure of how good the sample viewpoint is, and the best of all the sample viewpoints is chosen as the sensor direction. The planetarium algorithm takes into account self-occlusion of the object.

The planetarium algorithm was deemed too slow, so the normal algorithm was presented. In the normal algorithm, the area of the octree faces that are common to both unseen-space and empty-space voxels is considered. Six separate area sums are kept since the faces can be oriented in six axis-aligned directions. One of the two directions along each of the three axes is chosen if it has the larger area sum. The final sensor direction is computed as the sum of the three chosen directions weighted by their respective area sums. This algorithm considers only local visibility and global self-occlusion is ignored.

The paper does not address the issues of errors in range images, in registration, and in positioning of the range scanner. The models of the range sensor and the positioning system are not described at all. Although the author mentioned the collision avoidance constraint in the conclusion, only the visibility/occlusion constraint is actually considered in one of his algorithms. The author also did not mention how the hidden-line images in the planetarium algorithm are generated. Nevertheless, the sampling of viewpoints on a sphere, and the assumption that the sensor is always pointing at a center point, are used by other researchers in many later next-best-view algorithms, for example [Whaite1990, Banta1995, Zha1997, García1998].

Among the many next-best-view algorithms, the method of Pito [Pito1996, Pito1996a, Pito1999] is unique in that an intermediate 4D representation is used to record information of discrete light rays that can reach surface points on the partial model of the object. This representation is similar in idea to that of light field rendering [Levoy1996].

When searching for the best view, a potentially large set of discrete views has to be evaluated before the best of them is chosen for the next acquisition. In order to increase the efficiency of the evaluation, Pito introduced a 4D scalar field, dubbed the positional space, which consists of a 2D positional space surface (PSS) and the positional space directions (PSD). The positional space is used to record the light rays that can reach points on the occlusion surfaces and the low-confidence surfaces of the partial model. The PSS, which is a surface enclosing the object, is discretized into uniform cells for representation. Each cell is attached with a set of PSD, which is a polar coordinate system, and is also discretized for representation. Therefore, each sample in the discretized positional space represents the sum of “values” of the surface points observed by a bundle of approximately equal rays.

To evaluate a view, each ray from the view is transformed into the positional space, and the corresponding "surface value" already in the positional space is retrieved. These values are accumulated for each view, and the view with the highest accumulated value is chosen as the next acquisition view.

Although the scanner's pose used in Pito's experiment has only one degree of freedom, the positional space method can be applied to pose of higher degree of freedom. The method can also be applied to wide range of scanner models, as long as each ray from the scanner can be modeled. However, many important issues about the positional space representation have not been addressed. Since the method requires a somewhat expensive overhead to compute the positional space representation, it is only advantageous when the number of views to be evaluated exceeds some value. Another issue is the space requirement of the positional space representation. A major issue that is worth investigating are the implications of the discretization resolutions of the PSS and the PSD.

Whaite and Ferrie [Whaite1990, Whaite1991] presented an approach to view planning based on the paradigm of minimizing uncertainty. They assumed that each range image can be segmented into regions. Their objective is to reconstruct the object model by fitting a superellipsoid to each region, and view planning is used to acquire range images so that the uncertainty in the fitting can be minimized.

For each segmented region of range points, the best-fit superellipsoid is computed. Due to noise in the range data, and the fact that not every true surface can be exactly fitted with a superellipsoid, there is uncertainty about the validity of the best-fit superellipsoid. This uncertainty is expressed as a volume of region (uncertainty region) in the superellipsoid parameter space around the parameters of the best-fit superellipsoid. A confidence level is chosen to define the boundary of the uncertainty region, and all parameters values within the boundary represent acceptable superellipsoids. Since the uncertainty region can be very complex, Whaite and Ferrie proposed to approximate it with an ellipsoid, called the ellipsoid of confidence, which is derived from the second-order approximation of the local neighborhood around the parameters of the best-fit superellipsoid. The ellipsoid of confidence is defined by a covariance matrix that can be conveniently obtained from Hessian matrix produced by the Levenburg-Marquardt optimization.

The ellipsoid of confidence is then transformed, with approximation, into the 3D space to form a shell of uncertainty around the best-fit superellipsoid. From the shell of uncertainty, a surface uncertainty measure is used to assign uncertainty value to each point on the best-fit superellipsoid. An uncertainty image can be formed by rendering the superellipsoid with pixels shaded with the uncertainty values on the superellipsoid surface.

The range scanner is assumed to be positioned on a sphere, directed to the center. Each new scanner position is obtained by choosing a nearby position on the sphere that produces an uncertainty image that has the highest total uncertainty value. The uncertainty image is a z-buffered image of the uncertainty images of all the current superellipsoids in the partial model. Therefore, self-occlusion is taken care of.

Though the positioning system used in their experiment has error much larger than that of the range data, the authors did not address the problem of misalignment of range images. However, they did mention that misalignment of range images is a problem for integrating data from different views.

Reed et al. [Reed1997, Reed1998] presented a semi-automated view planning method, where an occluded surface must first be selected by the user as the target surface. A visibility volume is computed for the target surface by considering occlusions and the imaging constraints, such as the minimum standoff distance, the required minimum resolution, and the maximum inclination of the range sensor to the target surface. The target surface can be entirely seen by any point in the visibility volume without occlusion. The visibility volume is then intersected with the volume that represents all the possible placements of the range camera allowed by the robotic arm. The real sensor is then moved into the resulting volume and directed towards the target surface, but Reed did not mention any particular method to determine the final sensor position inside the resulting volume. The view planning method is based on the approach of the MVP system [Tarabanis1995a], which is described in a later section.

The imaging environment consists of a triangulation-based range camera mounted on a robotic arm, and a turntable on which the object of interest is placed. However, in his model reconstruction algorithm and view planning algorithm, the range camera is simply assumed to be a monostatic sensor. It is not clear over how many degrees of freedom the range camera can be positioned and what parameters can be set.

The partial model is represented as a solid model bounded by mesh surfaces derived from the acquired range images. After a range image is acquired, a solid is formed by sweeping the mesh in the sensing direction. Each surface on the solid is tagged as an imaged surface or an occluded surface. The new solid is then merged with the existing composite partial model via regularized set intersection, and the surface-type tags are correctly propagated to the new composite model. Registration of the range images is done by calibration of the range camera, robot arm and turntable.

2.2.2 3D Reconstruction of Indoor Environments

The next-best-view problems for 3D reconstruction of indoor environments are apparently more difficult than that for objects, probably due to the much larger volume of view space and the difficulty and ineffectiveness of simplifying the view space into more convenient space that encloses the object, such as a view cylinder or a view sphere. Nevertheless, some work on view planning for environments still oversimplify the view space to just a 2D horizontal plane.

Sanchiz and Fisher [Sanchiz1999] presented a NBV algorithm for 3D reconstruction of an unknown indoor scene. Their approach assumes the value of each view in the view space follows a smooth function, and use a combination of a hill-climbing optimization method and an exhaustive search method to search for a local optimal view at each acquisition iteration.

The view space considered is five-dimensional—3D position, pan and tilt. Each test view is evaluated by a specially designed objective function, whose value is determined by the amount of existing and unknown surfaces that can be seen from the view. Sufficient existing surfaces must be seen at a new view to ensure overlap with the newly acquired surfaces, so that they can be registered. Low-resolution ray tracing is used to estimate the amount of each surface type seen from the test view. The amount of surface quality improvement is also incorporated into the objective function.

The optimization of the objective function is done using two methods. A hill-climbing method is used for the 3D position of the range scanner. For every 3D position tested, an exhaustive search approach is used to find the pan and tilt.

The authors provided some results from experiments on synthetic 3D scenes. One drawback they discovered is that when almost the whole scene is recovered and there just remain few isolated views, the objective function is almost flat in the whole view space, and the local hill-climbing method will fail.

Sequeira et al. [Sequeira1996, Sequeira1999] took a different approach to the problem. Each new range image is segmented into polygonal planar patches, and merged with the patches in the partial model. The occlusion surfaces are represented by planar patches.

Each view planning iteration may produce more than one view. The algorithm starts by creating a view volume for each occlusion patch, in which any point can see the occlusion patch entirely. Each view volume is represented by a 3D polyhedron, and its shape and volume are determined by occlusion by other surfaces, the minimum stand-off distance of the scanner from the occlusion patch, its maximum range, minimum clearance distance from all surfaces, and the angle between the occlusion patch's normal and the direction to the scanner's position. Many of these 3D view volumes are then intersected, and the result may be zero, one, or more "global" volumes.

In each global volume, the best 3D view position is determined by optimizing an objective function, which favors view positions that can see a large projected area of the occlusion patches, have small angle between the viewing directions and the occlusion patches' normals, and are close enough to the occlusion patches for denser sampling. After a 3D position is determined in a global volume, the other acquisition parameters are determined, such as the image field and the image resolution. Overlap constraint is also considered when selecting the parameter values. In order to cover the occlusion patches visible from the 3D position, more than one scan may be needed from the same location. The authors did not provide details about how the parameter values and the number of scans are determined.

Many essential details are not included in the papers. For example, it is not clear whether the global volumes are the results of the intersection of all viewing volumes, or the results of the intersections of many different subsets of viewing volumes. The latter seems to make more sense, but it immediately leads to a combinatorial problem. Another problem with this method is that for some occlusion patches, there may exist no viewpoint that can see the patches in their entirety, and therefore no view will be computed for these patches. However, many of these patches can still be fully covered by combining the partial visibility from more than one viewpoint.

González-Baños et al. [González-Baños1999, González-Baños1998, González-Baños2001] reduced the view planning problem for 3D reconstruction of indoor scenes into a 2D problem. They separated the view planning into two distinct stages. The first stage builds a 2D map, similar to a floor plan, using a 2D next-best-view algorithm. The next stage then uses the 2D map in a 2D model-based view planning algorithm to produce a set of viewpoints to acquire 2D range images of 3D surfaces. Both stages employ a randomized strategy to generate 2D test viewpoints.

The result of the 2D map-building algorithm [González-Baños1999] is a polygon representing a cross-section through the environment at a given height. The range sensor is mounted on a mobile robot, and it acquires a 1D range image by making a 180-degree sweep in a horizontal plane at the given height above the floor. Each range image is combined into a composite partial model represented as polylines and “safe regions”. Safe regions are derived by shrinking the empty space enclosed by the polylines and occlusion edges when the maximum reliable range of the range sensor, and the allowable inclination angle of the laser to the surfaces are considered.

The next best view is computed by generating a set of random 2D positions in the shrunk safe region of the current partial model, evaluating each of them, and choosing the best. The safe region is shrunk by the radius of the robot to avoid collision. Each test position is evaluated by considering the lengths of the acquired edges and occlusion edges that it can see, and the path length from the previous scanning position. The length of the acquired edges must be above a threshold to ensure sufficient overlap for model alignment. The lengths of edges visible from a test position are approximated by casting a fixed number of equally spaced rays. The 2D map building is terminated when there is no occlusion edge in the partial model, or when there is no occlusion edge longer than a threshold.

In the second stage, the 2D map is used as the model for planning 2D viewpoints to acquire 2D range images. The objective is to minimize the number of sensing operations. Two randomized greedy algorithms were proposed [González-Baños1998], but they were not implemented. The first algorithm begins by sampling the interior of the polygonal 2D map, and computes the portion of the polygon’s perimeter “scannable” from each sample viewpoint. A point on the perimeter is scannable from a sample viewpoint if it is visible, is within the range of the scanner, and its angle of inclination to the direction of the viewpoint is smaller than a threshold. The perimeter is then decomposed into elements such that no portion of the perimeter is only partially scannable from any sample viewpoint. Each element is associated with the sample viewpoints that can scan it. Finally, a subset of the sample viewpoints is selected using the greedy algorithm for the set-covering problem.

The second algorithm tries to improve the first one by not producing sample viewpoints that cannot cover any part of the perimeter. The algorithm starts by selecting a point on the unseen portion of the perimeter and computing a region from which the point can be scanned. The region is randomly sampled, and the sample viewpoint that has the highest coverage is added to the solution. The 2D map is updated to reflect the change in the unseen portion. Next, another point on the updated unseen portion of the perimeter is selected and the above steps repeated until the unseen boundary is less than a threshold.

Nüchter et al. [Nüchter2003] used a next-best-view approach very similar to the 2D map-building algorithm of González-Baños et al. [González-Baños1999] for automated range acquisition of indoor environments. A 2D range image is acquired at each viewpoint (as opposed to a 1D range image). It is then registered and merged with the partial model. A 2D horizontal cross-section of the partial model is extracted and used for planning the next viewpoint, using a randomized strategy to generate sample viewpoints.

2.2.3 Object Inspection

For quality and manufacturing process control, manufactured parts must be inspected to identify defects and to assess their deviations from the reference models. Some of the inspection tasks are non-contact and can be automated using computer vision systems. Such an inspection system may use active range sensing to acquire geometric information of the object, but in many cases, ordinary non-stereo intensity or color images are sufficient for the required purposes. Sometimes, complete coverage of an object's surface is not necessary because views are required to observe only a selected set of important features on the object's surface. To aid the identification of the important features in the intensity or color images, many inspection systems include appropriately positioned light sources to illuminate the object, so that the features can be made to appear more prominent. Effective inspection requires proper planning of the views of the camera and the corresponding light source positions. Since a reference geometric model of the object is available beforehand to the view planner, a complete plan of the camera views and light positions can be pre-compute before the actual inspection.

Like most methods in 3D reconstruction of objects, many feature-detection view planning methods assume the camera views (and light source positions) are located on an imaginary view sphere, and the camera is always directed towards the center of the sphere. The HEAVEN system by Sakane et al. [Sakane1987, Sakane1991] computes, for each feature of interest, one camera view and three light source positions, so that the feature can be imaged unoccluded and unshadowed. During actual inspection, the acquired images are used to derive geometric information about the feature using a photometric stereo technique [Woodham1980]. The camera view and three light positions are selected from a set of discrete locations on a sphere surrounding the object, and the camera is assumed to always point towards the center of the sphere. These discrete locations are created by recursive subdivision of an icosahedron. Those locations that are occluded from the feature of interest are eliminated. Next, all combinations of four unoccluded locations for the camera and three light sources are compared according to a criterion, and the best combination is chosen. The criterion estimates how well the surface geometric information can be derived using the photometric stereo setup, and is a function of the directions from the feature to the three light sources, and the feature's surface normals to the camera and light sources.

Each camera view or light source position computed by the HEAVEN system is only two-dimensional. However, the overall computation is expensive due to the combinatoric complexity of exploring all combinations of four locations, even though the occluded locations have already been eliminated from consideration. The solution space is essentially eight dimensional.

The VIO system [Sakane1992] is very similar to the HEAVEN system, in that the camera and light source are assumed to be on a view sphere. Each solution of the system is a position of the camera, a position of a light source, and a subset of edge features to monitor on the target object. Discrete locations on the view sphere are first grouped into regions according to the set of objects edges they can see. Each region is independently evaluated with respect to the edge features on the object, and those that do not meet a specified threshold are eliminated from further consideration. Then, all possible camera-illuminator pairs are formed from the remaining locations on the sphere, and each pair is ranked based on how well and how many features can be imaged with the setup. The highest-ranked is taken to be the best choice.

To avoid the combinatorial problems of considering camera and light positions simultaneously, Yi et al. [Yi1990] implemented a camera-illuminator planning system that first computes the best camera position, independently of the light position, and then compute the best light position given the computed camera position. Again, the camera and light positions are assumed to be on a view sphere.

Tarabanis et al. took a very different approach from the above methods in their MVP sensor planning system [Tarabanis1991, Tarabanis1995a, Tarabanis1996]. Instead of sampling the search space and testing each sample point, the MVP system uses the task requirements to characterize the solution space analytically. The MVP system does not include illumination planning, but each camera view is eight-dimensional, consisting of the 5D camera pose, the distance between the back principal point and the image plane, the focal length, and the lens aperture. To compute the best 8D camera view, firstly, a polyhedron is constructed to represent the set of 3D viewpoints that can see all the features in their entirety. Other constraints, such as the feature-resolution, depth-of-field and field-of-view constraints are characterized by analytic closed-form relationships. These constraints form bounding hypersurfaces in the 8D space, and the final step is to perform a constrained optimization to find the 8D point in the bounded space that is furthest from all the bounding hypersurfaces and the polyhedron faces. This view is considered the best because it satisfies all the specified constraints and it is the most robust to sensor placement and configuration errors.

Not all automatic object inspection systems work by detecting features. Tarbox and Gottschlich [Tarbox1995] implemented a more general-purpose inspection system, called IVIS, that is more suitable for inspecting featureless objects. In contrast with the above feature-detection systems, this system uses a triangulation-based active range sensor to acquire a complete geometric model of the object during actual inspection. Thus, the view planning component of the system is required to plan a set of views to completely cover the object surface. Each view consists of a camera position and a structured light source position, which are both located on a discretely sampled view sphere. For each location of the camera, only six possible light source positions are considered, and they are all of a fixed distance (baseline distance) from the camera.

Three view planning algorithms were proposed in [Tarbox1995]. Central to the algorithms is the measurability matrix, which records whether each discrete surface point on the object is measurable from each camera-light source pair. A surface point is measurable by a camera-light source pair if it is visible from both locations and the angle of incidence of the light ray to the surface point is within a specified threshold. One of the view planning algorithms uses a greedy approach, where at each step, the view that can measure the most uncovered surface points is selected. Another algorithm takes a global optimization approach to attempt to find the least number of views to cover the entire object surface. The algorithm starts by generating a small set of randomly selected views and then perturbing the views in a simulated annealing process [Kirkpatrick1983, Aarts1985]. After the process, if the resulting views do not cover the entire object, the number of views is incremented and randomly placed on the sphere, and the simulated annealing process repeated.

2.2.4 Image-Based Modeling and Rendering

In image-based modeling and rendering (IBMR), a 3D scene is modeled as a collection of reference images, rather than a set of conventional geometric primitives. Novel views of a scene can then be synthesized from the reference images using a variety of interpolation and re-projection techniques, for example [Chen1993] and [McMillan1995]. IBMR has several important advantages over traditional modeling and rendering: (i) the images can be of real world scenes, thus making the task of modeling such scenes much easier; (ii) image-based rendering algorithms are typically inexpensive and can be performed on general purpose computers; (ii) the rendering time is typically independent of the geometrical and physical complexity of the scene being rendered.

A complete IBMR solution involves both sampling and reconstruction of the plenoptic function [McMillan1995], which is a 5-dimensional function that describes all the infinite-resolution images that can be seen at all viewpoints in space (it can actually be reduced to a 4-dimensional function of all rays in space). Most of the research has been on reconstruction algorithms (efficient warping, splatting, etc.), and research in the important problem of properly sampling the plenoptic function has been less active. Proper sampling typically involves acquiring images of the scene from some appropriately selected viewpoints and view directions, and at the appropriate resolutions. The following three papers propose different techniques to attempt to solve the sampling problem for synthetic models.

Given a polygonal 3D model of a scene, a viewing volume, and a sampling quality threshold, Fleishman et al. [Fleishman1999] proposed a method to compute a small set of viewpoints in the viewing volume such that every polygon visible from the viewing volume is visible from at least one of the computed viewpoints. Each visible polygon is associated with only one viewpoint, even though it may be visible to many. The end result is an image-based model made up of a set of masked reference images, where each masked reference image from a viewpoint consists of pixels from the polygons associated to the viewpoint.

Their method starts by subdividing the polygons in the input 3D model to reduce the problem of polygons being partially visible from the viewpoints in the view volume. The next step evaluates the visibility of the polygons from each viewpoint in the view volume. Since rays that go into the interior of the view volume always intersect the volume's boundary, every surface point visible from the view volume is visible from the volume's boundary. Therefore, only viewpoints on the volume's boundary are evaluated.

The boundary is first tessellated into small patches, and then a hemispherical image of the input scene is rendered from the center of each patch. Each polygon in the scene is rendered in its unique color. In practice, each hemispherical image is rendered as multiple planar images. Each planar image is traversed and the colors of the pixels are used to determine which scene polygons are visible, and the number of pixels of each color is used to determine the sampling quality of the polygon. Each visible polygon and its sampling quality are associated with the planar camera (view), and this information is put into a database.

A greedy strategy is used to select a subset of cameras. For each camera in the database, a list of adequately sampled polygons is computed. Then all the cameras in the database are ranked according to the number of adequately sampled polygons that it adds to those already covered by previously selected cameras. A camera is selected at each iteration until all the visible polygons are adequately covered by the union of the selected cameras. For the alternative problem of selecting only *k* camera positions that cover as much of the scene as possible, Fleishman et al. proposed another camera ranking strategy. Polygons that are most likely to be seen should be favored, so for each scene polygon, the number of cameras that can see it is counted, and a camera is ranked according to the sum of the visible cameras of its associated polygons. From these selected camera positions (and parameters), the masked reference images are generated.

Stuerzlinger [Stuerzlinger1999] addressed a similar view planning problem for IBMR, but dealt with only the visibility portion of the problem, and did not consider surface sampling quality. A hierarchical visibility algorithm is used to compute the visibility between the viewing regions (viewing volumes) and the scene surfaces. A similar algorithm has been used in hierarchical radiosity [Hanrahan1991]. The basic idea is that if a surface polygon and a viewing region are partially visible to each other, then one of them is subdivided and the visibility computation is continued on the smaller viewing regions or polygons. This process repeats recursively until the "shaft" between the surface polygon and the viewing region is completely unoccluded or completely occluded, or when the potential visibility error is below a threshold. A link is created if the surface polygon and the viewing region are completely or partially visible to each other. Shaft culling [Haines1994] is used to speed the visibility determination. The result is a hierarchy of surface polygons and a hierarchy of viewing regions, with links between the polygons and viewing volumes.

The next step of the algorithm is to select a set of good viewing regions that can see all the visible surfaces. The method starts by enumerating all the finest viewing regions and all the finest subdivided polygons. The links between them are used to create a two-dimensional visibility array, which is indexed by viewing region and polygon number. Each array entry is set if the polygon and the viewing region are partially or completely visible to each other. After this, a global optimization search, using simulated annealing, is performed to select a small set of viewing regions. It works by changing the vector of solutions randomly. The objective function is the total surface area of all polygons visible from the candidate viewing regions. The method calls the optimization procedure repeatedly with increasing numbers of viewing regions. As soon as the maximum total surface area is reached, the loop terminates.

Finally, for each selected viewing region, a good viewpoint in it is found. This search is also performed using simulated annealing, and the objective is to maximize the total visible surface area. However, the computed viewpoint does not always see all the surfaces that are visible from the viewing region. Stuerzlinger argued that this is not very common, and moreover, surfaces that are missed by this viewpoint may be visible to viewpoints in other viewing regions.

In the work by Wilson and Manocha [Wilson2003], the objective is to compute image-based simplifications of a large and complex synthetic environment so that the environment can be rendered at interactive rates for walkthrough applications. In the preprocessing phase, a set of viewpoints is computed, and from each of them, the environment is sampled with a panoramic color image with range (depth) at each pixel. Since the panoramic images may overlap one another, the next step removes redundant samples in the images. For each image, the remaining samples are used to create textured polygonal meshes, called textured depth meshes, and care is taken to avoid connecting samples across depth discontinuities. Each mesh is then simplified to reduce the number of polygons, and the result is stored in a database, ready to be used for rendering during walkthrough of the environment.

The set of viewpoints used to generate the panoramic images is computed in the following way. First, the large environment is decomposed into smaller sections, and a set of viewpoints is computed for each section independently. Each section has a navigable region, and all computed viewpoints must lie within it. A 2D rectangular region is assumed in the paper.

A greedy approach is used to select the viewpoints. First, viewpoints at the four corners of the navigable region are put into the solution set. Subsequently, at each iteration, a set of candidate viewpoints is generated, and an objective function is evaluated to select the best candidate as the next viewpoint. To generate the candidate viewpoints, a 2D Voronoi diagram is created from the viewpoints already in the solution set, and the candidate viewpoints are the Voronoi vertices inside the navigable region, and the intersections of the Voronoi edges with the navigable region's boundary. The objective function is evaluated for each candidate viewpoint, and the one that can see the largest projected area of the global void surfaces (or occlusion surfaces) is chosen. The global void surfaces are the resulting skins when the surfaces and skins (occlusion surfaces) seen by each of the previous viewpoints are merged together.

Chapter 3

A Next-Best-View Solution

One of the problems in the acquisition of range images for 3D reconstruction is the determination of a sequence of scanner views such that the object or environment of interest can be satisfactorily acquired and reconstructed. Unless the object or environment is geometrically very simple, this view planning task is generally difficult for human operators to accomplish.

The goal of the work presented in this dissertation is to automate the view planning task for the range acquisition of indoor environments, where a priori knowledge of the environments' geometry is not available. View planning problems of this type, where a priori knowledge about the object's or environment's geometry is not available, are generally known as *non-model-based view planning problems* or *next-best-view problems*. There are many challenges to a practical solution to a next-best-view problem. First of all, it must overcome the intractability inherent in the problem. The next major difficulty is to be able to efficiently evaluate a large set of scanner views to determine the most appropriate ones. Each view may consist of several parameters, and thus the solution space may be high-dimensional. If the range images are to be acquired at very high density, and the view planning performed with high resolution, another difficulty of automated view planning for environments is handling and processing the potentially huge data set, which can lead to unacceptably time-consuming computation and insufficient system memory.

In this chapter, I first introduce the basic view planning problem and the general next-best-view problem, and present the computational difficulties that are inherent in them. Then, in Section 3.2, I define the specific next-best-view problem that concerns this work, describe the solution strategy that I have chosen to approach the problem, and state the objectives a solution should achieve. Section 3.3 gives an overview of my next-best-view algorithm, and gives details about how the partially acquired scene can be represented and about the view metric function used in the search for the next best views. Later chapters present detailed descriptions of the different components in the solution. One of the major components of the solution is the efficient evaluation of views with respect to the partially acquired scene, and it is deferred to the next chapter. Furthermore, in Chapter 5, another component—surface registration—is explored in great details. The main contributions of this work are presented in Chapters 4 and 5.

3.1 The Next-Best-View Problem

The objective of automated view planning for 3D reconstruction of objects or environments is to automatically determine an efficient set of scanner views such that the acquisition constraints are met and a digital model of the object or environment can be satisfactorily reconstructed. A view is defined as a pose of the scanner and the associated imaging parameters. Examples of acquisition constraints include the field of view of the scanner and the minimum distance that must be observed between the scanner and the object. The efficiency of a set of planned views is measured according to different metrics. It can be the total number of scans, the amount of acquired data, or the total scanning time. An object or environment can be satisfactorily reconstructed if all surfaces accessible to the scanner have been acquired with the required quality wherever possible. The relationship between view planning and reconstruction quality is elaborated in Section 3.2.

For 3D reconstruction, a priori knowledge of the object's or environment's geometry is not available to the view planner. The first scan is made from a view appropriately selected by a human operator, and for the subsequent scans, the planner must determine the next best views based on the information it has already collected from the previous scans. This is why the view planning problems for 3D reconstruction are also called the next-best-view problems. They are part of the more general view planning problems, which also include view planning for objects whose a priori geometric information is known, which are known as the *model-based view planning problems*.

In the most fundamental model-based view planning problem, where only visibility is considered and each viewpoint can see in all directions, the objective is to determine the minimum number of viewpoints, such that every surface point on the object or in the environment is visible to at least one of the viewpoints. This problem has been shown to be NP-hard, and is equivalent to an extended version of the well-known art-gallery problem [O'Rourke1987]. With suitable discretization of the model surfaces and the viewpoint space, the basic model-based view planning problem can be reduced to the set-covering problem [Tarbox1995, Scott2001d].

Although NP-complete, it is well known that the set-covering problem has a polynomial-time approximation algorithm whose solution is worse than the optimal by at most a logarithmic factor [Cormen1990]. The algorithm uses a greedy approach at each iteration to select the subset that covers the most uncovered elements. In the basic model-based view planning problem, this is equivalent to selecting the scanner view that can acquire the most new information. This greedy approach has been used in many other model-based view planning algorithms.

Since global geometric information is unknown, a next-best-view problem cannot be solved globally to get an optimal solution. It is inherently a local optimization problem [Kakusho1995]. The local optimization is applied to only the known geometric information, which has been acquired by the previous scans. With appropriate representation of the unknown occluded regions, for example, with occlusion surfaces (which are artificial surfaces added to connect surfaces on opposite sides of depth discontinuities caused by occlusions; see Figure 3.9(a)), an optimal solution is the smallest set of viewpoints such that all points on the occlusion surfaces are visible from at least one viewpoint. A model-based view planning algorithm can be applied to get the locally optimal solution. Then, the set of new viewpoints is used to acquire new surfaces to update the partial model, and the local optimization is repeated on the new partial model.

The local problem remains NP-hard, and thus the greedy approximation algorithm is often used in the local optimization. When the greedy approach is used, one would want to keep the partial model as up-to-date as possible by incorporating the latest scan, and ensure that the next computed viewpoint is based on this most up-to-date information. This naturally leads to a tight iterative approach. It is tight in the sense that at the end of each iteration, only a single new viewpoint is produced, and after the scan is made, it is immediately incorporated for the computation of the next viewpoint. This tight iterative greedy approach is used in almost all existing next-best-view algorithms.

However, each iteration of the greedy approach is still very computationally expensive due to the potentially huge search space and expensive visibility computations. This apparent computation difficulty has limited many next-best-view algorithms to incomplete search space, simple and small objects, and low-quality acquisition. Some early algorithms even ignore self-occlusion of the objects [Connolly1985].

When only visibility is considered and each viewpoint can see in all directions, then the size of the search space is proportional to the size of the *viewpoint space partition* (VSP) and the *aspect graph* (dual of the VSP) of the object [Plantiga1990]. Each aspect, or a region in the VSP, of a polyhedral object is a maximal set of viewpoints that can see the same set of polygons, forming images of the same topological appearance, i.e. the structures of the polygons' boundary line segments and intersection points in the images are the same. For a non-convex polyhedral object, the number of aspects is $O(n^9)$, where n is the number of vertices [Plantiga1990]. At present, the best time bound for computing the aspect graph is $O(n^9 \log n)$.

With the VSP of the partial model, at each iteration of the greedy algorithm, the objective is to select a viewpoint in an aspect that can see the largest total area of occlusion surfaces[2]. Although the number of vertices, m, of the occlusion surfaces can be much less than n, the number of aspects is generally larger than $O(m^9)$. This is because occlusion surfaces can be occluded by true surfaces, adding complexity to the VSP. The size of such a VSP is $O(m^3n^6)$.

The above discussion assumes 3D viewpoint locations, unlimited field of view, and direct visibility. In practice, besides the 3D location, other parameters of the scanner have to be determined in the solution, such as the scanner's orientation, the adjustable field of view, and the scanning resolution. These parameters add additional dimensionality to the already huge search space.

[2]All viewpoints in an aspect see the same set of occlusion surfaces but do not necessary see the same amount of surface area of the occlusion surfaces. Therefore, additional optimization must be performed within the aspect to find the viewpoint that sees the largest area of occlusion surfaces. Since, from within the aspect, the occlusion surfaces are in the same topological appearance, the optimization function should be smooth, and a numerical optimization method may be used to find the best viewpoint.

In addition, besides the simple direct visibility requirement, usually, multiple other requirements and constraints need to be taken into account. For example, a surface is considered satisfactorily acquired only if it is entirely visible to the scanner and its sampling density is higher than a threshold. These additional requirements and constraints may further increase the complexity of the search. In the next section, I specify the requirements and constraints specific to the next-best-view problem addressed in this work.

3.2 Problem Specifications

This section defines the specific view planning problem addressed in this work. The main objective of view planning is to produce a sequence of scanner views such that the set of range images acquired at the views can satisfy the requirements on the quality of the reconstruction. The relationship between view planning and reconstruction quality is discussed first. Then, the section presents the practical approach adopted by this work, and explains the objectives of the solution. Finally, it states the other assumptions, requirements and constraints that must be observed by the solution.

3.2.1 View Planning and Reconstruction Quality

The goal of 3D reconstruction is to produce a 3D model of the real-world object or environment (both will be referred to as object in this section, unless otherwise distinguished) that is suited for the specific application. Different applications have different quality requirements on the reconstructed models. For example, a 3D digital model of a sculpture for museum archival purposes usually requires much higher resolution and fidelity than a model used in a virtual building walkthrough. The range acquisition process can affect the reconstruction quality in the following three ways:

(1) **Completeness.** The more surface area of the object that is acquired by the range scanner, the more complete the reconstructed model can be. The amount of surface area that can be acquired is limited by many factors, such as the total number of scans that can be afforded, self-occlusions by the object, and the scanner's limited

field of view and depth of field. When the object does not fit completely inside the scanner's field of view and/or depth of field, multiple scans have to be made to cover the full size of the object. From a single scanner position, usually, some surfaces are occluded by other surfaces that are closer to the scanner. To measure these occluded surfaces successfully, multiple scans from different scanner positions are required. However, since it may be impossible to position the scanner at any location and orientation, because of physical obstruction and the limitation of the positioning system, not all occluded surfaces can be measured.

Each scan takes time to make and requires other resources, such as storage and manual labor, and these place a limit on the total number of scans that can be afforded. Consequently, when planning views for the limited number of scans, it is natural to want to maximize the amount of surface area that can be measured.

(2) **Surface sampling quality.** To achieve the desired reconstruction quality, the surfaces of the object must be measured to meet a sampling quality requirement. This sampling quality requirement ensures that small surface features are captured in the range images and can be successfully reproduced in the reconstructed model up to the required resolution. The sampling quality at a surface point consists of the surface sampling density in its neighborhood and the range measurement error. The surface sampling density is expressed as the number of samples per unit area, and is a function of the scanner's scanning resolution, the surface point's distance from the sensor, and its orientation with respect to the sensor.

A sample's range measurement error is the result of systematic error and random error. The systematic error is often caused by miscalibration of the scanner, while the random error is caused by measurement uncertainty. Range measurement uncertainty can be affected by many factors, such as the surface point's distance from the sensor [Beraldin2000], the angle of incidence of the light beam on the surface point, and the surface reflectance properties. The error is unknown, and is usually specified as a distribution function or just a standard deviation.

Therefore, to satisfy the sampling quality requirement at a surface point, the scanner must not only be able to measure it without occlusion, but must also be at the right distance and direction from the surface point. Effective view planning must take this into account to maximize the surface area that reaches the required sampling quality. Surface sampling quality of a surface region can be enhanced by combining multiple scans of the same area during merging of the range data, where overlapping data sets can increase the sampling density and "smooth out" the random errors.

(3) **Surface registration accuracy.** Before the range images can be merged into one model, they must first be aligned together into a common coordinate system [Pulli1999, Bergevin1996, Nishino2002]. In order to be able to be registered to each other, every range image must have overlap with at least one other range image. Moreover, the overlap regions must have enough samples and enough geometric constraint on the 3D rigid-body transformation so that range images can be registered accurately (Chapter 5 has more details on this). Large registration errors may result in obvious topological errors in the reconstructed model, while small registration errors may cause small surface features in the overlap regions to be "washed out". To prevent registration failure during model reconstruction, scanner views must be carefully chosen during range acquisition. One strategy is to ensure that when a new view is chosen, it should be able to register with the combined surface of the previous scans. The requirement for accurate surface registration may further increase the number of scans in the acquisition stage, because some surface regions have to be acquired multiple times to ensure overlap between range images.

3.2.2 Solution Objectives

I have adopted the greedy approach for the computation of the next best view in this work. The view planning solution operates in a tight iterative approach as part of the *model acquisition cycle* shown in Figure 3.1. At the end of each cycle, only a single new viewpoint is produced, and after the scan is made, it is immediately incorporated into the partial model for the computation of the next viewpoint.

During a typical acquisition session, the model acquisition cycle is repeated multiple times. In each cycle, after a scan is made, it is first registered (or aligned) with the partial model of the object or the environment. Then the information of the new scan is merged into the partial model, and this updated model is used for the computation of a new view for the next scan.

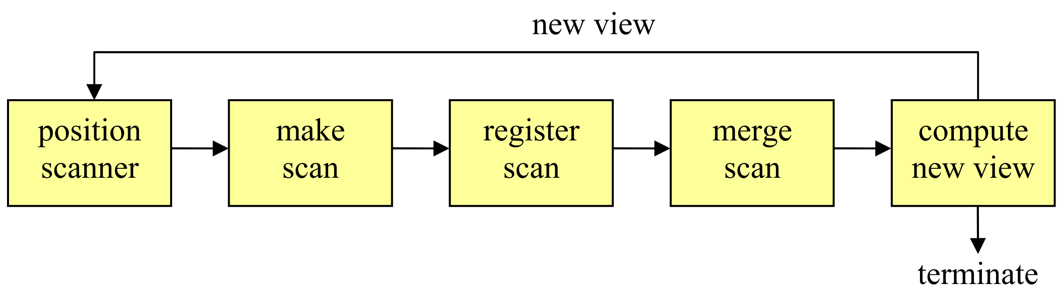


Figure 3.1: The model acquisition cycle with automated view planning.

A greedy next-best-view algorithm computes by evaluating a view metric function for each view, and outputting the view that maximizes the metric function as the best view for the next scan. Since we are striving for reconstruction quality, the view metric must take into account the three reconstruction quality requirements described in Section 3.2.1. In the simple case in which there is only the completeness (visibility) requirement, the view metric is simply the total amount of new surface area that can be seen by the view. However, when the three reconstruction quality requirements have to be considered simultaneously, the metric is no longer well-defined, because these requirements are in contention with each other. For example, trying to maximize the amount of surface area that reaches the required sampling quality may reduce the amount of new surface area that can be acquired. The three quality requirements must be combined into a scalar value, and trade-offs must be made between them. The formulation of a practical view metric is presented in Section 3.3.

Sometimes, the total number of scans to be made is not specified beforehand, and the range acquisition may be terminated preemptively by the human operator. Therefore, it is advantageous to maximize the amount of acquired information of the object or environment as early as possible, which is what a greedy algorithm does.

The next major challenge to a practical solution is to develop an efficient method to evaluate the view metric function. Even though the solution space in my specific problem is only three-dimensional, the metric function will still have to be evaluated for a large set of views. The evaluation of each view can be very computationally expensive, since the partial model can be very complex and of high resolution, and there are other requirements and constraints that need to be handled. Chapter 4 presents a novel method to evaluate a large set of views efficiently.

Besides the reconstruction quality requirements, there are additional important requirements and constraints that need to be observed when performing the view planning computation, and they are described in the following sections.

3.2.3 Indoor Environments

The target application of the view planning is to acquire range images of real-world indoor environments to reconstruct models for virtual reality walkthrough. Most of the previous reconstructions have been on single small to mid-sized objects. Range acquisition of real-world environments is generally more difficult than for objects. Real-world indoor environments can be cluttered with many objects, which results in high visibility complexity. This makes automated view planning more important and challenging. Moreover, an object has a bounding volume, and the scanner is usually positioned outside the volume, whereas, in an indoor environment, the scanner has to be placed inside. Therefore, viewpoints must avoid collision with objects inside the space, and to fully automate the acquisition process, an obstacle-free path has to be planned when moving the scanner from one viewpoint to the next. For an object, there is nothing like the discovery of a completely new room in an indoor environment. The scanner must be placed appropriately (e.g. near the door) to acquire significant new information and yet to ensure registration.

For view planning, the indoor environment is assumed to be enclosed, so that there is a well-defined finite set of surfaces to be acquired and a bounded volume of space where the scanner can be located. This requirement is not strictly necessary because whenever surfaces outside the region of interest are acquired, we can have the human operator mark out the regions not of interest, so that they will not be included in the subsequent view planning computation. Assuming an enclosed environment simplifies the system.

All the surfaces in the enclosed environment are assumed to be within the range of the range scanner wherever it is positioned in the environment. When a surface is out of the range of the range scanner, it will appear as a region of drop-outs (no measurement) in the range image, and thus the volume between the sensor location and the surface cannot be identified as empty. Figure 3.2 shows an example scan in which a large part of the environment is beyond the range of the scanner. Only a small volume can be identified to be empty, and the next viewpoint planned by the view planner can only be inside the small volume. Most likely, the next viewpoint still cannot resolve the status of most of the unknown volume. This leads to an excessively large set of viewpoints.

It is also assumed that there is no extended surface region in the environment that is overly absorbent or overly specular-reflective of the light beam emitted from the range sensor. An overly-absorbent surface does not reflect sufficient light energy back to the sensor detector, and will appear as a region of drop-outs in the range image, similar to a surface that is beyond the range of the scanner. This may result in an unnecessarily large set of viewpoints. An overly-specular-reflective surface may deflect the light beam to other surfaces, and this can cause drop-outs and outliers in the range image.

The floor in the indoor environment is assumed to be flat and level. This requirement is necessary because of the restricted way the range scanner can be positioned. The scanner can only be positioned on flat and empty regions of the floor, and must not be on top of any object. Requiring a flat and level floor should make the detection of low-lying obstacles easier. If the scanner can be placed anywhere in the empty space, then this requirement is not necessary.

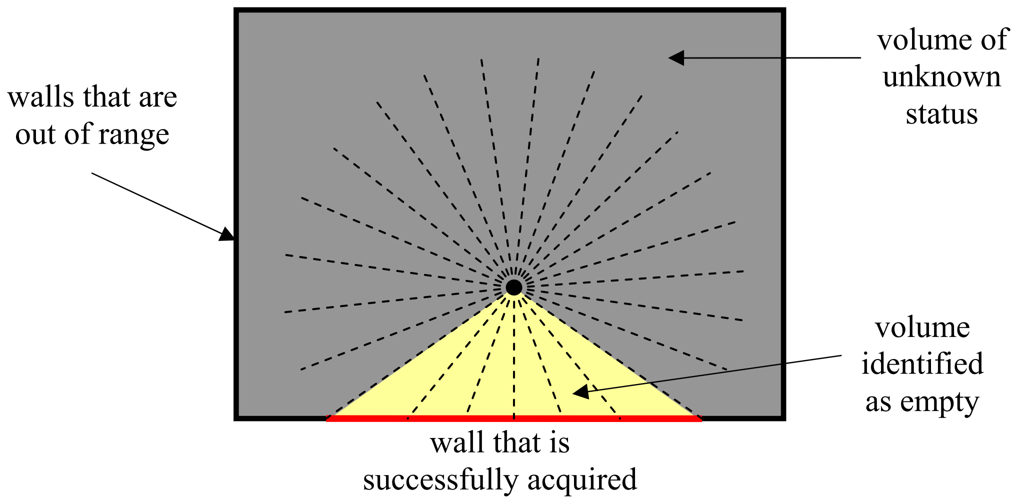

walls that are out of range
volume of unknown status
volume identified as empty
wall that is successfully acquired

Figure 3.2: An example scan in which most of the surfaces are beyond the range of the scanner. Only a small volume can be identified to be empty, and the rest is of unknown status.

3.2.4 Range Scanner and Range Images

This work assumes that the range sensor is *monostatic*. Monostatic refers to the fact that the transmitter and receiver of light (or other forms of energy) are located at the same location. This excludes the class of range sensors based on triangulation, where the transmitters and receivers need to be at different locations. Many range sensors based on the time-delay (time-of-flight) principle are monostatic. The consequence of this requirement is that the visibility of a surface point depends only on a single line of sight from the sensor location.

The next assumption is that all the range samples in a range image are measured from the same sensor location, as if there is a single center of projection or viewpoint for the entire range image. This requires the sensor to be swept in a rotational manner about the sensor location, as opposed to a translational sweep which produces a range image that has multiple centers of projection.

The monostatic and single-center-of-projection assumptions should simplify the visibility evaluations during the view planning computation, but more importantly, the "visibility coherence" around the single viewpoint can be better exploited for more efficient visibility evaluations using the hierarchical approach described in Chapter 4.

It is also assumed that the range scanner has a 360° horizontal field of view, and it is always oriented such that its vertical axis is parallel to the vertical of the environment. Together, the two assumptions simplify each scanner view to just a 3D location, instead of a 6D pose. To allow acquisition of the floors and ceilings of indoor environments, it is required that the vertical field of view of the scanner should span from below the horizon to above it. Many mid-range and long-range scanners on the market today have 360° horizontal field of views and limited vertical field of views, for example, the DeltaSphere-3000 3D Scene Digitizer [DeltaSphere], the MENSI GS200 3D Laser Scanner [MENSI], and several models by RIEGL [RIEGL].

The DeltaSphere-3000 3D Scene Digitizer [DeltaSphere] is one of the range scanners that fulfill the above scanner requirements. It is used in the view planning experiments described in this dissertation. Figure 3.3 shows a DeltaSphere-3000 mounted on a tripod. The scanner scans the environment by rotating itself step-by-step about the tripod, and in each step, a rotating mirror sweeps a laser beam in a vertical plane. The range sensor is monostatic, and the center of projection of the range image is located at the center of the rotating mirror. The field of view of the scanner is adjustable. When placed in the normal upright pose, as shown in Figure 3.3, the scanner can have a maximum horizontal field of view of 360°, and a maximum vertical field of view spanning from 55° below the horizon to 90° above the horizon. The depth of field is also adjustable, but the maximum range extends from about 1 foot to 50 feet from the sensor. At this maximum depth of field, the range measurement precision is about 0.3 inch standard deviation. The scanner can scan with a maximum scanning resolution of 20 samples per degree in both the vertical and horizontal directions. The scanning time is dependent only on the horizontal field of view and the scanning resolution. A scan with 360° horizontal field of view and 10 samples per degree resolution takes about 8 minutes to complete.

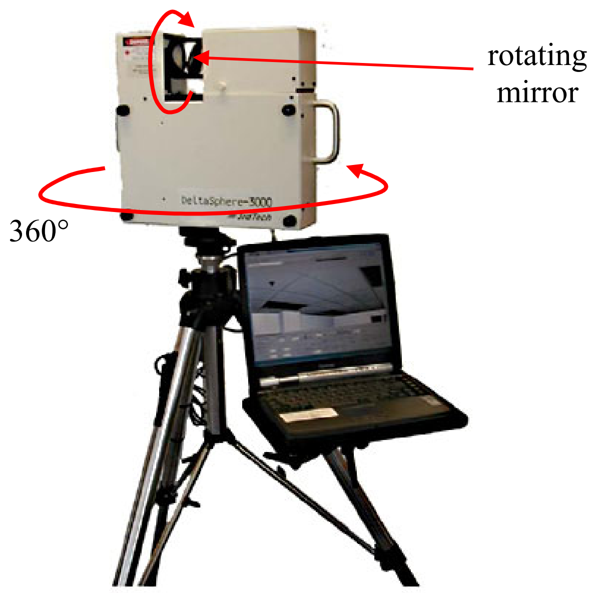


Figure 3.3: A DeltaSphere-3000 3D Scene Digitizer mounted on a tripod. Photograph courtesy of DeltaSphere, Inc.

The local 3D coordinate frame of the range image has its origin located at the center of the rotating mirror, and all range measurements are expressed as distances from the origin. Each sample 3D point in the range image is expressed in polar coordinates (r, θ, ϕ) in the scanner's local coordinate frame, where r is the distance from the origin to the 3D point, θ is the azimuth angle, and ϕ is the elevation angle (see Figure 3.4). The polar coordinates can be easily converted to Cartesian coordinates.

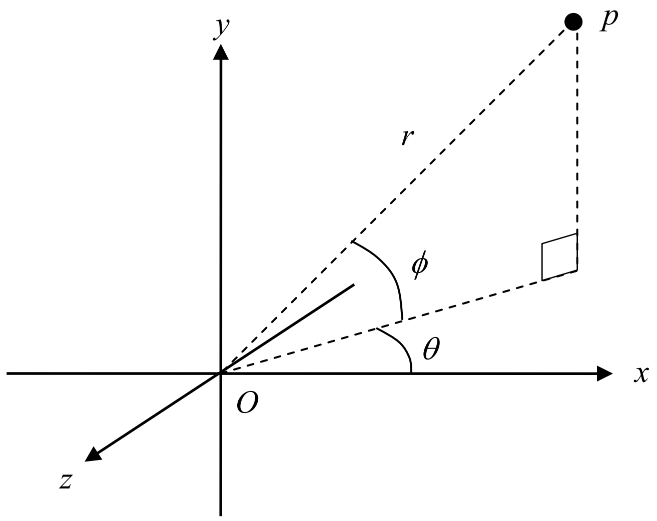


Figure 3.4: The local 3D coordinate frame of the DeltaSphere-3000 range scanner. The origin is located at the viewpoint and its y-axis is pointing up. Each sample 3D point in the range image is expressed in polar coordinates (r, θ, ϕ), where r is the distance from the origin to the 3D point, θ is the azimuth angle, and ϕ is the elevation angle.

An example range image produced by the DeltaSphere-3000 is shown in Figure 3.5. The horizontal field of view of the scan is 360° ($\theta = -180° \ldots 180°$), and the vertical field of view is 125° ($\phi = -55° \ldots 70°$). The scanning resolution is 10 samples per degree in both the vertical and horizontal directions. In the figure, the intensity at each pixel is proportional to the range value at the corresponding sample. The brown-colored pixels correspond to drop-outs, which are samples that cannot be measured successfully by the sensor. The DeltaSphere-3000 also records the reflected laser intensity detected at each sample point, and Figure 3.6 shows the image of the reflected laser intensities corresponding to the range image in Figure 3.5. The sample points in the range image are converted into Cartesian coordinates, and the 3D points are used to create a triangle-mesh model shown in Figure 3.7.

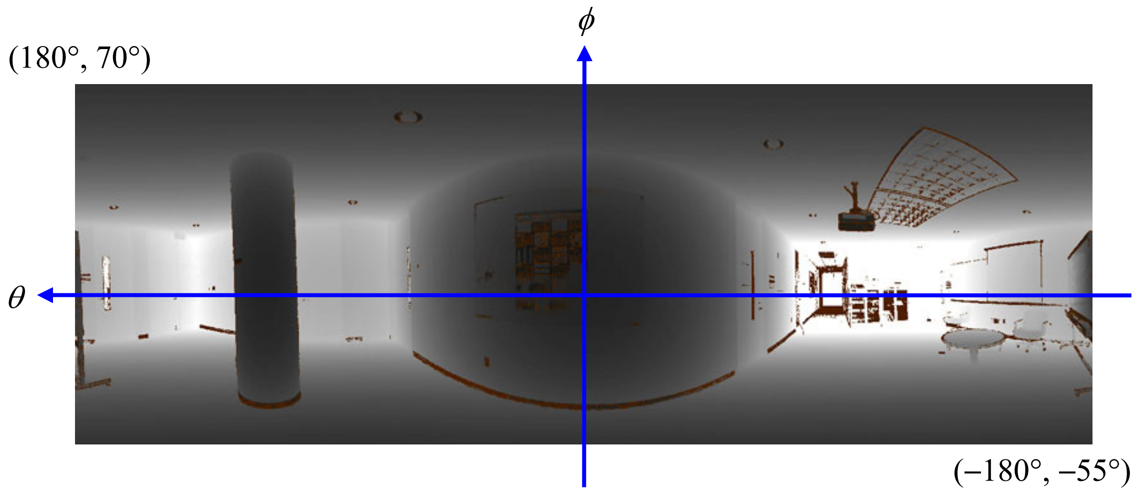


Figure 3.5: A panoramic range image produced by the DeltaSphere-3000 scanner. The intensity at each pixel is proportional to the range value at the corresponding sample. The brown-colored pixels correspond to drop-outs, which are samples that cannot be measured by the sensor.

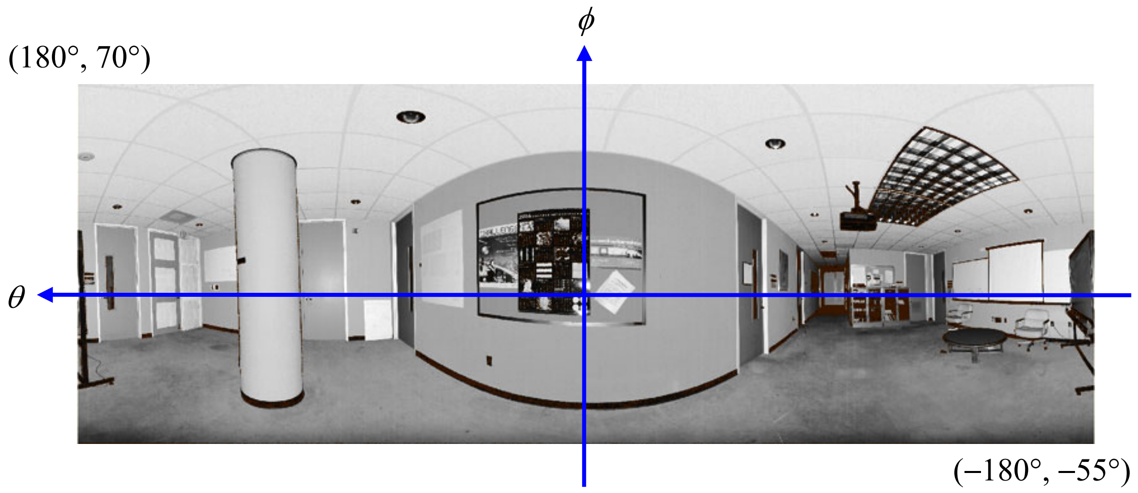


Figure 3.6: The image of the reflected laser intensities corresponding to the range image in Figure 3.5.

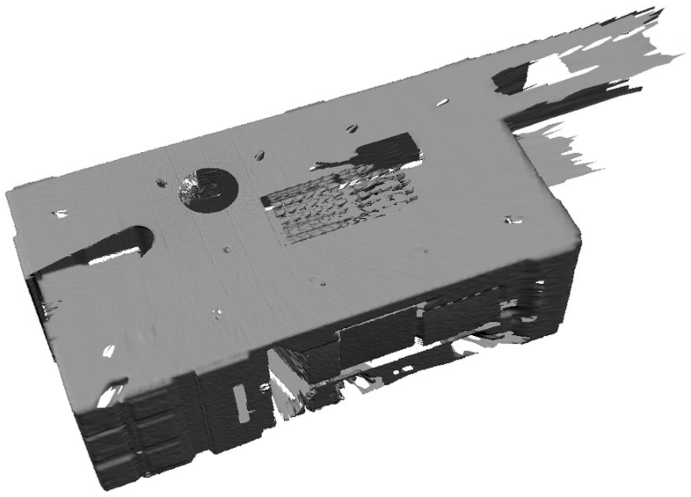


Figure 3.7: A 3D triangle-mesh model created from the range image in Figure 3.5.

3.2.5 Scanner Views

A scanner view consists of a pose of the scanner with respect to a reference coordinate frame and a set of values for the associated adjustable imaging parameters. The most general pose is a 3D rigid-body transformation composed of a 3D rotation and a 3D translation. This is also often referred to as a 6D pose. In many cases, the scanner is restricted to only a subset of the six dimensions. For example, a scanner mounted on a mobile robot, with a fixed position and orientation with respect to the robot, may have poses, with respect to an indoor environment, that are made up of a 2D translation in the horizontal plane and a 1D rotation about the vertical axis. A range scanner may have a set of imaging parameters that are adjustable, such as the field of view, the depth of field, and the scanning resolution. These parameters increase the dimensionality of a view. If any of the adjustable parameters is fixed at a constant value for the entire acquisition process, then it should not be included in the views, and should instead be considered an acquisition constraint of the scanner.

For this work, each scanner view to be computed is a 3D translation with respect to a coordinate frame in the environment. It is assumed that the horizontal field of view is always 360°, and the other scanner's parameters' values are user-specified, and will remain constant throughout the entire acquisition session. The scanner is assumed to be always in the upright orientation. It (more correctly, the scanner's local coordinate frame) can still be rotated about the vertical axis, but the 360° horizontal field of view makes it irrelevant. Therefore, what remains in a view is a 3D position of the scanner. The other parameters, namely, the vertical field of view, the depth of field, and the scanning resolution, will be treated as acquisition constraints in the view planning computation.

3.2.6 Acquisition Constraints

Limitations and characteristics of the range sensing device often become constraints that must be observed when planning a view of the scanner. These acquisition constraints can be grouped into the following three types:

(1) **Positioning constraints.** These constraints determine whether the scanner can be physically positioned at a location in the 3D empty space. For example, the scanner's position may be constrained by the physical construction of the scanner and the positioning device, and needs some space around it to operate, and thus the viewpoint must be kept at least a minimum clearance distance away from any object in the environment. The scanner may be placed only between a minimum and a maximum height above the floor, but must never be placed above any object. A view that satisfies the positioning constraints is a *feasible view*.

(2) **Sensing constraints.** These constraints determine whether a surface point in the scene can be measured from a view. A surface point cannot be measured by the scanner if there is no clear line of sight from the scanner's viewpoint to the point. If the scanner has a limited field of view and a limited depth of field, then all surface points not in the field of view and depth of field cannot be measured. Moreover, when the light beam from the range sensor is at a grazing angle to a surface point, the range measurement usually becomes unreliable. Measurements of this type of points are usually discarded when they are detected in the range image. Detection is done by estimating a surface normal at the range sample and checking whether the angle of incidence of the light beam to the sample is larger than a threshold angle. This is referred to as the *angle-of-incidence constraint*, and given a scanner position, all surface points not satisfying this constraint are considered not measured.

(3) **Registration constraint.** When the scanner is positioned at a pose planned by the view planner, it is assumed that there will be positioning error. Therefore, after a new scan is made by the scanner, it has to be aligned or registered to the previous scans accurately so that the new information can be merged properly with the old one (see Figure 3.1), and the scanner can be re-localized. However, registration of the new scan to the previous ones is not guaranteed to be successful. It is affected by several factors, and depends a lot on the planned view from which the scan is made. Therefore, when planning a new view to make the next scan, the view planning algorithm must take the factors into consideration, to ensure that the new range scan acquired from the planned view can be successfully registered with the previous ones. This issue is explored in greater detail in Chapter 5.

The difference between the registration constraint described here and the registration accuracy requirement for reconstruction quality is that the registration constraint is for the registration accuracy of a new scan to the partial model during the acquisition process, whereas the latter is for the registration in post-acquisition reconstruction. The required accuracies for both purposes are generally not the same, but if the registration constraint is satisfied during acquisition, the same registration accuracy can also be achieved during the post-acquisition reconstruction. In this dissertation, I assume that the required accuracies are the same for both purposes, and will focus on the registration constraint instead of the registration accuracy requirement for reconstruction quality.

3.3 Solution

In this section, I present a solution to the view planning problem defined in the previous section. The algorithm involves a greedy optimization to search for the best view, according to a view metric, for the next scan. Before I describe the formulation of the view metric, I first describe a strategy to represent the unknown regions in the partial model. This representation should give clues about how the missing surfaces could be acquired. Finally, the section presents an overview of the next-best-view algorithm and lists all the major components that made up the solution.

3.3.1 Partial Models

A partial model of the environment should not only record the surfaces that have been acquired, but also give clues to how the unknown surfaces can be acquired. To achieve the sampling quality requirement, it should also keep track of the sampling quality of the surfaces. It should also provide some means for surface registration to be performed between it and the new scan, and allow information in the new scan to be merged. Since the computed view can only be positioned at a location which has been determined to be empty, the partial model must keep track of known empty space in the environment. All the above are necessary to support the generation of feasible views (views that satisfy the positioning constraints), and to evaluate them using the view metric.

In this section, I first describe how a partial model may be constructed from a single scan, and later describe how partial models can be merged to produce a more complete partial model.

3.3.1.1 Partial Model from a Single Scan

When a surface region has been acquired by the scanner, the view planning system can be sure that the volume of space between the scanner and the surface region is empty. However, occlusions, drop-outs and the limited field of view of the scanner result in unknown volumes of space whose status is yet to be determined. A simulated example is shown in Figure 3.8. In the example, a light-blue box is placed in a room (the room’s ceiling is not shown), and the scanner is set up pointing towards the box (in the direction of the red axis). On the box, there is a dark circular area right in front of the scanner. Figure 3.8(b) shows the surfaces successfully acquired by the scanner. The dark circular area is too absorbent of the light from the scanner, so it appears as a region of drop-outs in the range image. The occlusion by the front face of the light-blue box not only causes part of the wall to be occluded, but the status of the volume of space between the box face and the occluded region of the wall remains unknown. As for the circular area of drop-outs, the conical volume of space between it and the scanner's viewpoint has unknown status too. Finally, the limited field of view of the scanner results in all 3D space outside the field of view to be unknown in status. In the example, the simulated scanner's field of view is limited both vertically and horizontally, which is a more common scenario than the 360° horizontal field of view used in the specific view planning problem.

To differentiate the volumes that are known to be empty from those that are of unknown status, *false surfaces* are added to the surface model in Figure 3.8(b) to bound the known empty space, so as to keep the model "water-tight". Three types of false surfaces can be added: (1) *occlusion surfaces* for occlusions, (2) *hole-boundary surfaces* for holes caused by drop-outs, and (3) *image-boundary surfaces* for range image boundaries caused by limited field of view of the scanner. These false surfaces are shown in Figure 3.9.

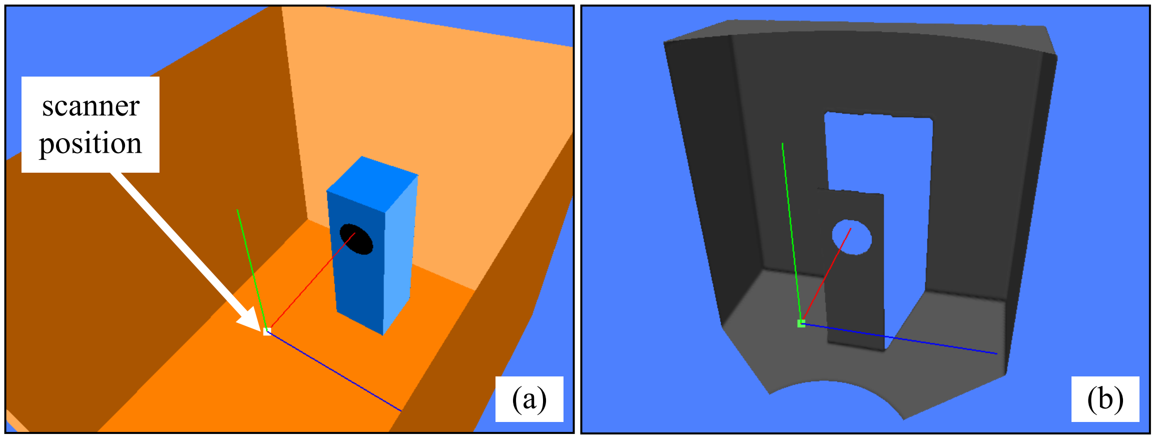


Figure 3.8: (a) The scanner is pointing towards the light-blue box (in the direction of the red axis). On the box, there is a dark circular area right in front of the scanner. (b) The surfaces acquired by the scanner. The dark circular area is too absorbent of the light from the scanner, so it appears as a region of drop-outs in the range image.

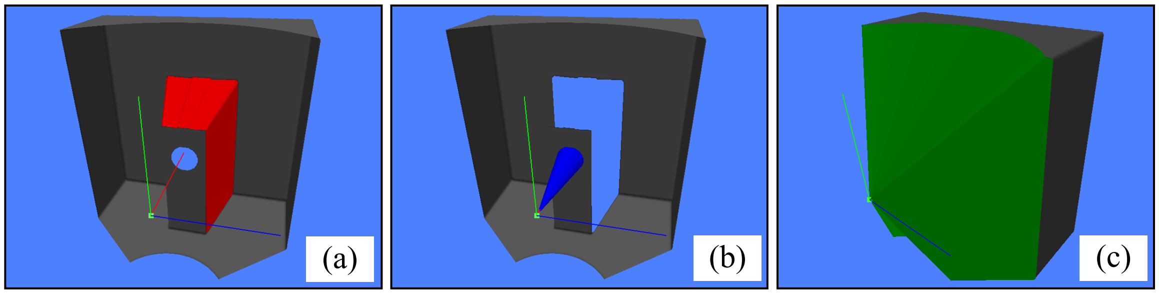


Figure 3.9: Three different types of false surfaces. (a) The occlusion surfaces are shown in red. (b) The hole-boundary surfaces are shown in blue. (c) The image-boundary surfaces are shown in green.

Occlusion surfaces are false surfaces that connect surfaces across depth boundaries. When a range image is converted to a triangle mesh by connecting neighboring range sample points to form triangles, the occlusion surfaces will be those very elongated triangles that extend across large depth differences. There are many heuristics to detect occlusion surfaces in a triangle mesh created from a range image. For example, if a triangle's longest edge is beyond a threshold length, it is considered a triangle of the occlusion surfaces. Another method, which is scale-independent, is to use the angle between the triangle's normal vector and the sensor's line of sight to the center of the triangle. If the angle is greater than a threshold angle, the triangle is considered a part of the occlusion surfaces. The volume bounded by the occlusion surfaces and behind the occluder is considered on the outside of the known empty space, and it is unknown in status, thus the next scanner view should not be located in that volume of space.

Drop-outs in the range image result in holes in the triangle mesh. Hole-boundary surface triangles are added to connect the boundaries of the holes to the scanner's viewpoint, as shown in Figure 3.9(b). In the example, the space bounded inside the cone is considered on the outside of the known empty space. Similarly, image-boundary surface triangles are added to connect samples on the range image boundaries to the scanner's viewpoint. The space outside the scanner's field of view is considered on the outside of the known empty space.

With the *true surfaces* acquired by the scanner and the added false surfaces, the known empty space is properly bounded. What is created is a solid model [Hoffmann1989] of the known empty space.

The false surfaces provide clues to how the unknown volumes can be resolved by subsequent scans. One strategy is to place the next view in the known empty space where it can see as much area of the false surfaces as possible. This allows the next scan to "penetrate" as much of the false surfaces as possible, in the hope of resolving the unknown volumes behind them. This strategy is incorporated in the view metric described in Section 3.3.2.

3.3.1.2 Merging Partial Models

Two partial models are merged by computing the union of their known empty space and the union of their surfaces, with the additional rules that (1) the empty space and the true surfaces have precedence over all false surfaces, and (2) the precedence of one false surface over another is arbitrary. The first rule means that if a false surface point in one partial model coincides with a point in the empty space of the other model, the result is empty space; or, if a false surface point in one partial model coincides with a true surface point in the other model, the result is the true surface point. Before the two partial models can be merged, they must be registered or aligned to each other.

The result of the merging is another partial model that can later be merged with other partial models of the same environment. This is how information from many scans can be accumulated to get a more complete partial model of the environment.

A 2D analogy of the merging process is shown in Figure 3.10. The 2D partial model in Figure 3.10(a) is a 2D version of the one shown in Figure 3.9. Figure 3.10(b) is a partial model created from a scan made from another viewpoint. The result of the merging is a partial model that has more known empty space and more true surfaces, and the volumes of unknown status have become smaller. It is always true that the unknown volumes will never become larger.

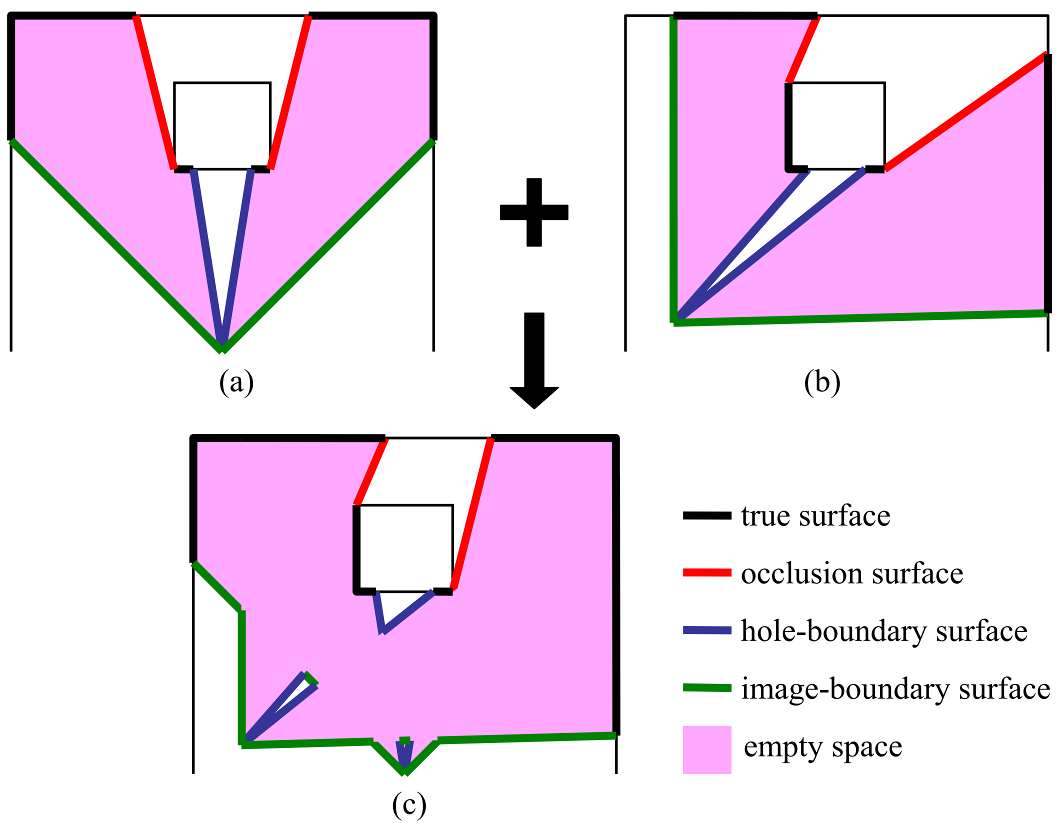


Figure 3.10: Merging of two partial models. The magenta regions are known empty space. The thick black lines are true surfaces, the red lines are occlusions surfaces, the blue lines are hole-boundary surfaces, and the green lines are image-boundary surfaces.

To support the evaluation of the view metric function, some attributes, for example the sampling quality, are associated with each surface point in the partial model. These attributes have to be properly combined during merging, and the details are described in Section 3.4. The data structure used to represent the partial models, and the implementation of the construction and merging of the partial models are also described in that section.

3.3.2 View Metric

In each iteration of the model acquisition cycle (Figure 3.1), after the latest scan has been merged into the cumulative partial scene model, a greedy approach is followed to search for the best view for the next scan. During the search, a view metric function is evaluated for each candidate view, and the view with the highest metric value is selected as the best view.

The view metric should lead to a new view that exhibits the maximum improvement on the reconstruction quality. However, as we have learnt, the three reconstruction quality requirements (completeness, surface sampling quality, and surface registration accuracy) are in conflicting competition with each other, and therefore they cannot be all maximized simultaneously. Trade-offs among the three requirements must be made in the optimization metric according to the task at hand and the relative importance of each requirement.

For this work, I use the view metric shown in Equation (3.1), where $h(v)$ is the metric value for a view *v*:

$$h(v) = f(v) \cdot r(v) \cdot \int_{p \in S} w(p) \cdot c(v,p) \cdot \max(0,\ \min(s(v,p),\ D) - q(p))\ dp \tag{3.1}$$

where

- *S* is the set of all surface points in the current partial scene model; it includes all true and false surfaces;
- $f(v)$ is 1 if view *v* is a feasible view, i.e. all the positioning constraints (see Section 3.2.6) are satisfied at view *v*, otherwise $f(v)$ is 0;
- $r(v)$ is 1 if the registration constraint is satisfied at view *v*, otherwise $r(v)$ is 0;
- $c(v,p)$ is 1 if all the sensing constraints between view *v* and the surface point *p* are satisfied, otherwise $c(v,p)$ is 0; the visibility between view *v* and *p* is one of the sensing constraints (see Section 3.2.6);
- $s(v,p)$ is the sampling density at surface point *p* if it is scanned from view *v*; this is referred to as the *new scan sampling density*;
- *D* is the sampling density requirement for all surfaces;
- $q(p)$ is the maximum sampling density at which surface point *p* has been scanned previously; this is referred to as the *recorded sampling density*; if *p* is on a false surface, then $q(p)$ is 0;
- $w(p)$ is the weight or importance value assigned to the surface type of surface point *p*; the four different surface types are the true surfaces, the occlusion surfaces, the hole-boundary surfaces, and the image-boundary surfaces.

The function max(a, b) returns the larger of a and b, and the function min(a, b) returns the smaller of a and b.

Basically, $\max(0,\ \min(s(v,p),\ D) - q(p))$ is the improvement to the recorded sampling density at surface point p if a new scan is made at view v, and given that all the acquisition constraints are satisfied. However, there is no improvement if the recorded sampling density $q(p)$ has already reached or exceeded the requirement D, even if the new scan sampling density $s(v,p)$ from view v is higher than the requirement D. The improvement in sampling density is summed up over all surface points in the partial scene model, including surface points on false surfaces. The metric value $h(v)$ is the total gain in the number of surface samples at view v, given that the acquisition constraints are satisfied.

In the metric, the false surfaces are treated just like the true surfaces, only that their recorded sampling density is 0. The weight $w(p)$ is used to trade off between the completeness and the surface sampling quality requirements. To favor completeness, one can assign a large weight value to the occlusion surfaces and a small value to the true surfaces, and vice versa.

The registration constraint is represented by a binary function $r(v)$ in the metric, which indicates whether the registration accuracy can or cannot be achieved for view v. The estimation of the registration accuracy is presented in Chapter 5.

In the metric, range measurement errors are not included as part of the surface sampling quality. The range measurement errors are not independent of the sampling density on a surface region. When the sampling density is high, the surface region is near to the scanner and/or it is facing directly at the scanner. These are the cases when range measurement errors tend to be small. Similarly, when the sampling density is low, the range measurement errors will tend to be larger. Therefore, the sampling density alone may be a good measurement of the relative surface sampling quality.

In order to evaluate the view metric, the recorded sampling density $q(p)$ must be available for each surface point on the partial model. The recorded sampling density is an attribute associated with the surface point. Another attribute that is necessary for the evaluation of the metric is the surface normal at the surface point.

In the later sections of this chapter and in the next two chapters, I describe the details of how the metric function can be evaluated efficiently.

3.3.3 Algorithm Overview

The strategy to evaluate $h(v)$ for all the views is to evaluate Equation (3.1) in pieces, from the least to the most expensive to compute. This evaluation strategy results in the execution flow of the major steps shown in Figure 3.11, which shows the model acquisition cycle with the view planning stage expanded. The main input to the next-best-view algorithm is the cumulative partial scene model which has just been updated by merging the latest scan. The result of the algorithm is a new view for making the next scan, or a decision to terminate the acquisition process. The purpose, and the execution order, of each step of the algorithm is briefly explained here, and the details, including implementation, are presented in the later sections of this chapter and in the next two chapters.

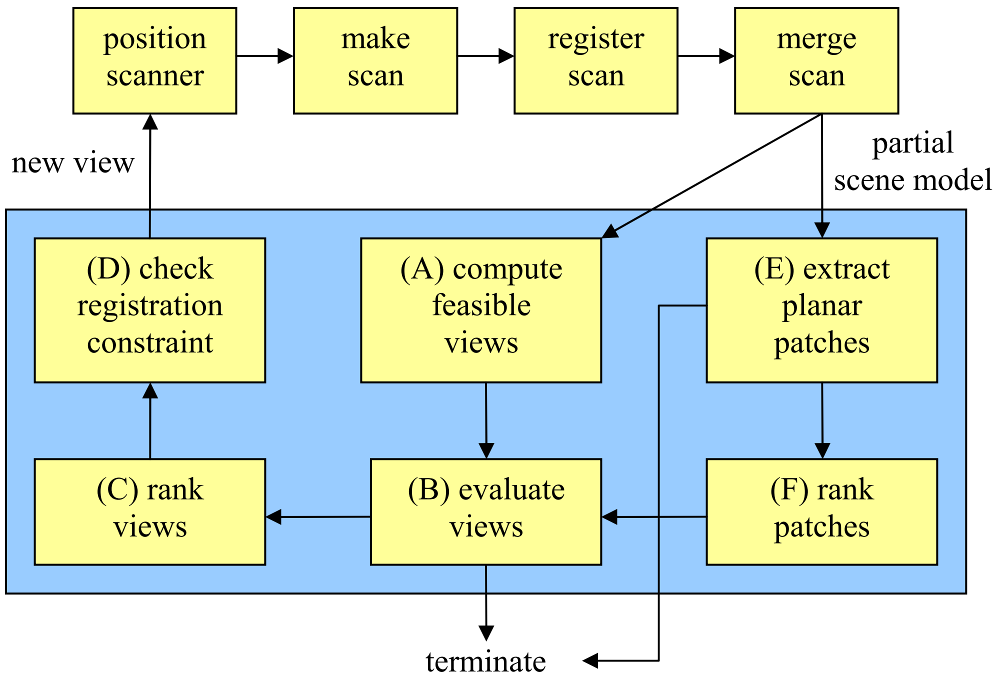


Figure 3.11: The model acquisition cycle with the view planning stage expanded to reveal the major steps and the execution flow of the proposed next-best-view algorithm.

(A) **Compute feasible views.** Using the partial scene model, this step computes the set of views that satisfy the positioning constraint function $f(v)$. The reason that the feasible views are computed first is that this step is less expensive to compute and the infeasible views should be eliminated as early as possible to save computation time in the main view evaluation step.

(B) **Evaluate views.** This step evaluates the integral part of Equation (3.1) for all the feasible views. From here onwards, the integral part is referred to as $g(v)$, i.e.

$$g(v) = \int_{p \in S} w(p) \cdot c(v, p) \cdot \max(0,\ \min(s(v, p),\ D) - q(p))\ dp \tag{3.2}$$

where v is a feasible view. One of the main contribution of this work is the use of a hierarchical view evaluation method to efficiently evaluate $g(v)$ for a large set of views. After all the feasible views have been evaluated, if they all have metric values (scores) below a threshold value, the view planner will suggest termination of the acquisition process. The details of this step are presented in Chapter 4.

(C) **Rank views.** The feasible views are ranked in descending order of their current metric values or scores.

(D) **Check registration constraint.** Starting from the highest-score view to the lowest, the registration constraint function, $r(v)$, is evaluated to determine whether the registration constraint is satisfied by each view. A view is checked for the registration constraint only if it is at least a specified distance away from all the previously-checked views, otherwise it is skipped. This is because views with similar scores tend to cluster near each other and they usually have similar registration accuracies too. The first view found to satisfy the constraint is output as the next best view.

The reason for checking the registration constraint last is that it is the most expensive to evaluate, and it is hoped that it is evaluated for as few views as possible. The details of this step are presented in Chapter 5.

(E) **Extract planar patches.** To support the hierarchical view evaluation in Step (B), surface points in the partial scene model are grouped into planar patches. Only surface points on false surfaces, and surface points whose recorded sampling densities are less than the sampling density requirement *D*, are grouped into patches. If all the surfaces have already reached the sampling density requirement, or all the patches are less than a specified size, the scene is considered satisfactorily acquired and the acquisition process can be terminated.

(F) **Rank patches.** The planar patches are then ranked in descending order of their importance. The reason for ranking the patches is that there may not be enough time to consider all the patches for the evaluation of the views. Sometimes, only a limited duration is allotted for the view planning computation, and, when time is limited, only the most important patches will be used to evaluate the views.

3.4 Partial Model Representation

Section 3.3.1 has shown some of the basic requirements on the partial model representation. A practical representation of the partial model should have the following qualities: (1) it can represent surfaces as well as volumes of space, (2) it must be space-efficient, (3) it supports efficient and robust merging operations, (4) it supports surface registration with new scans, and (5) it facilitates view evaluation.

An octree-based [Samet1989a, Samet1989b] volumetric representation of the partial model has been chosen for the implementation of the next-best-view solution. As we see later in the section, this representation possesses the required qualities listed above. Even though the surfaces have to be approximated by discrete voxels in the octree, the accuracy is sufficient for the application. Octree-based representations of partial models have been used in some previous next-best-view algorithms [Connolly1985, Sanchiz1999] and in model-based view planning [Tarbox1995].

Boundary representation (B-rep) of a solid model [Hoffmann1989] is an alternative for the partial model representation. In comparison to an octree-based representation, the B-rep can represent surfaces more accurately. However, practical implementations of geometric modeling operations (e.g. Boolean set operations) on B-rep solids have always been plagued by robustness issues [Hoffmann1989]. In practice, the surface mesh obtained from a range image produced by the Deltasphere-3000 scanner may be self-intersecting at the seam connecting the beginning and the end of a horizontally 360° scan, and this degeneracy may cause the B-rep operations to fail. Nevertheless, B-rep solid representation has been used in previous next-best-view algorithms [Reed1997].

The storage requirement of an octree model is proportional to the total surface area in the represented object [Hunter1978]. This is a great improvement over uniform 3D occupancy maps, which require storage space proportional to the volume of the represented objects. Uniform 3D occupancy maps have been used in some of the previous next-best-view algorithms [Banta1995, Massios1998] to represent partial models, and unsurprisingly they are restricted to simple and small objects, and the low-resolution partial models often result in low-quality view planning. The octree representation and its operations are generally easier to implement than those of B-rep, and it is more robust with degenerated triangle meshes. Boolean set operations, which are needed for the merging of two partial models, are also straightforward and robust for the octree representation [Hunter1979]. Moreover, the octree models also directly support acceleration of ray-casting [Foley1992], which is performed very frequently during view evaluation.

3.4.1 Creating Octree Partial Models

Each range image is first converted to a closed triangle mesh with added false surfaces, and an octree model is then created from the triangle mesh. If the range image is not from the very first scan, the triangle mesh has to be first registered with the cumulative partial model of the previous scans before it is converted to an octree, and the new octree model is then merged with the cumulative octree model.

Before the range image is converted to a triangle mesh, it is filtered to reduce noise and outliers. In the implementation, a median filter is used to reduce outliers and then anisotropic diffusion [Black1998] is applied to further smooth the range image. Next, small isolated regions of drop-outs are filled in using simple linear interpolation of the surrounding samples. If many of these small holes are left intact, they will create an excessive number of cone-like volumes (see Figure 3.9(b)) of unknown status. This can greatly reduce the amount of large contiguous empty space for feasible views, and thus may exclude many potentially good views from being considered for the next view.

The filtered range image is converted to a triangle mesh by connecting neighboring range sample points to form triangles. The false surface triangles are added as described in Section 3.3.1.1. Next, an octree is constructed to voxelize the closed triangle mesh. An octree cell is recursively subdivided if it contains or intersects any triangle of the mesh, and the subdivision stops when the cell reaches the specified minimum cell size. My implementation of the voxelization uses an efficient triangle-box overlap test algorithm described in [Akenine-Möller2001].

Six types of cells can be in an octree, and they are all leaf nodes of the octree. They are listed in decreasing precedence below.

(1) **Empty-space cells.** Each empty-space cell represents part of the 3D space that has been identified as empty.

(2) **True surface cells.** Each true surface cell contains or intersects at least a true surface triangle of the mesh. If the cell is also intersected by false surface triangles, the true surface will take precedence and the cell is still classified as a true surface cell. Each cell contains two attributes—a surface normal, and the best surface sampling density (recorded sampling density) at the surface region represented by the cell. The cell's surface normal is estimated by summing the normal vectors of the true surface triangles that intersect the cell. The best sampling density is approximated by $0.5/A_{\min}$ where $A_{\min}$ is the area of the smallest true surface triangle that intersects the cell. The sampling density is measured in the number of samples per unit area.

(3) **Occlusion surface cells.** An occlusion surface cell contains or intersects at least an occlusion surface triangle of the mesh. The occlusion surfaces are given precedence over the other two types of false surfaces, so if the cell is also intersected by hole-boundary or image-boundary surface triangles, it is still classified as an occlusion surface cell. The cell's surface normal is computed similarly to that of a true surface cell, and its recorded sampling density is 0.

(4) **Hole-boundary surface cells.** Similar to an occlusion surface cell. The hole-boundary surfaces have precedence over image-boundary surfaces. The cell has a surface normal and its recorded sampling density is 0.

(5) **Image-boundary surface cells.** Similar to an occlusion surface cell. The cell has a surface normal and its recorded sampling density is 0.

(6) **Solid-space cells.** Each solid-space cell represents a volume of 3D space that is either solid or is of unknown status and has to be resolved by additional scans.

During the conversion of the triangle mesh to an octree, after the surfaces have been voxelized, the remaining cells in the octree are either empty-space or solid-space, and they have to be distinguished. This can be done as follows. From the scanner viewpoint where the range image is acquired, a ray is shot towards the center of the cell. If the ray intersects the cell before the triangle mesh, then the cell is an empty-space cell, otherwise it is a solid-space cell.

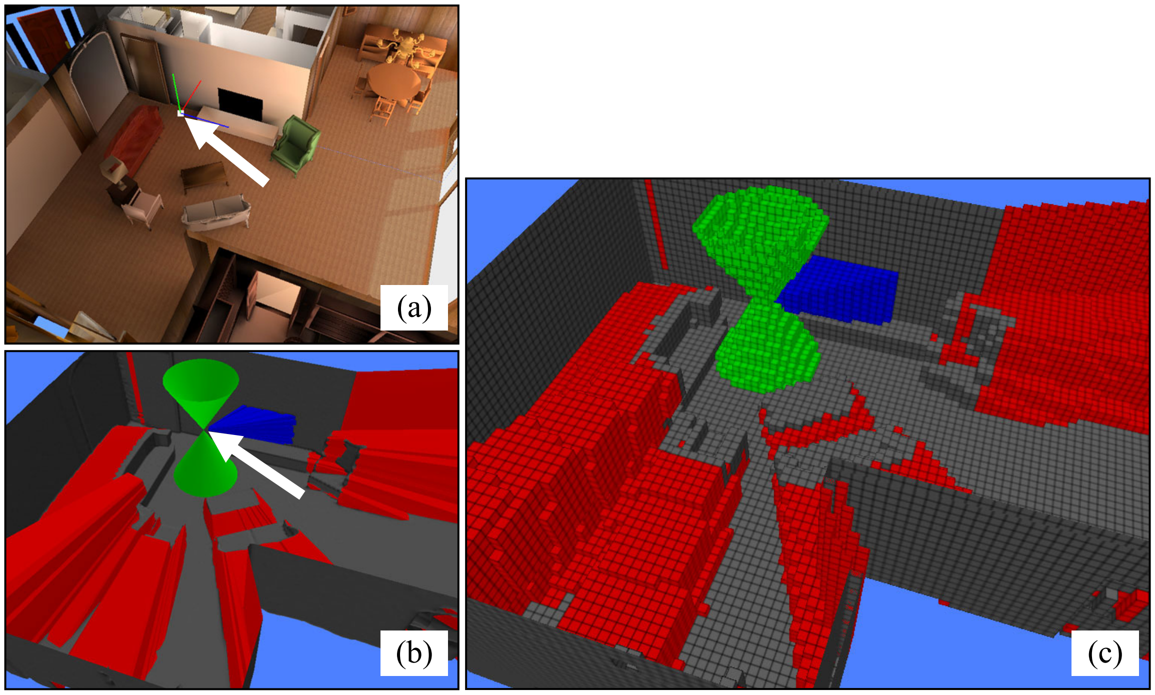


Figure 3.12: (a) A horizontally 360° scan of a synthetic scene is made from the position specified by the coordinate frame (pointed at by a white arrow). The ceiling of the room is not shown in all three images. (b) The triangle mesh created from the range image. The gray-shaded surfaces are the true surfaces, red are the occlusion surfaces, blue are the hole-boundary surfaces, and green are the image-boundary surfaces. The fireplace in the scene is too dark and results in a region of drop-outs in the range image. This region is bounded by hole-boundary surfaces (blue) in the triangle mesh. (c) An octree model is created by voxelizing the triangle mesh. Each surface voxel is 4 inches wide.

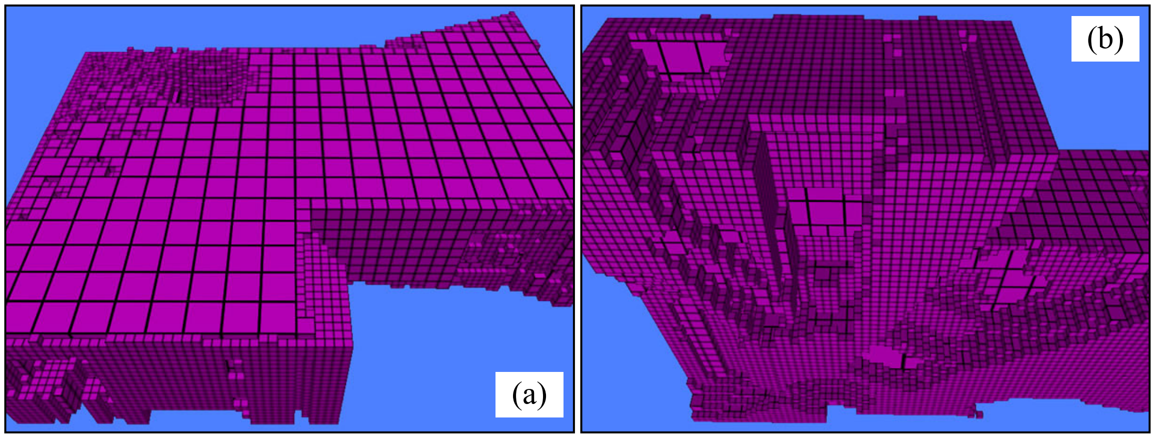


Figure 3.13: The empty-space cells of the octree model shown in Figure 3.12(c). (a) A view from the top. (b) A view from the bottom.

Figure 3.14 shows three 2D examples of octree models. All the surface cells are at the specified minimum cell size, and the empty-space and solid-space cells can have different cell sizes. Surface cells are also referred to as surface *voxels* in this dissertation. During merging of two octrees, the precedence of the cells follows the order in which they are listed above, with empty-space cells having the highest precedence and solid-space cells having the lowest. Even though the false surfaces are given precedence during creation of the different types of surface cells, the precedence among the false surfaces is actually irrelevant and can be arbitrarily assigned.

An octree model of a scan of a synthetic room is shown in Figure 3.12 and Figure 3.13. Figure 3.12(c) shows the surface cells and Figure 3.13 shows the empty-space cells of the model. It can be seen that all the surface cells are of the same size, but the empty-space cells can be of different sizes.

3.4.2 Merging Partial Models

After a new scan is made, its triangle mesh is registered, converted to an octree model as described in Section 3.4.1, and then merged with the cumulative octree model of the previous scans.

The top-level nodes of the two octrees to be merged must bound exactly the same volume in the 3D space. The merging starts by traversing both octrees top-down in parallel, and when the same volumes of space from the two octrees are compared, the one with the higher precedence (according to the precedence of the cell types) will appear in the result. The details of the algorithm are very similar to those of the Boolean set operations on octrees described in [Foley1992]. Figure 3.14 shows a 2D analogy of the merging of two octree models. Since empty-space cells have the highest precedence, the empty space in the cumulative partial model will always remain empty regardless of the new scans that will be merged. This is equivalent to the union of empty space of the partial models as mentioned in Section 3.3.1.2.

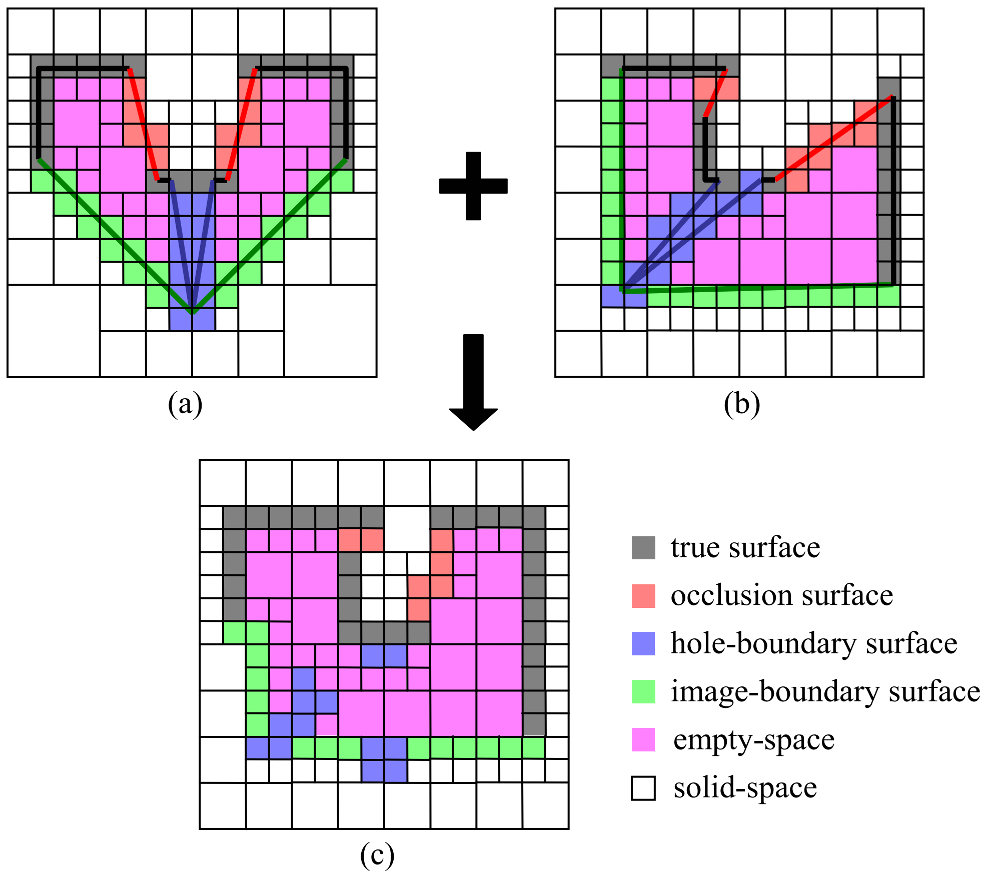


Figure 3.14: A 2D analogy of the merging of two octree models. (a) and (b) are the input octree models and (c) is the result.

When a surface cell in one octree coincides with a surface cell of the same type in the other octree, their attributes should be combined. In one of the ways to do that, the resulting surface cell's surface normal is the sum of the surface normals of the input cells, and the resulting recorded sampling density is the higher of the two inputs. When two surface cells of different types coincide, the resulting cell type and attributes are those of the higher precedence cell.

In practice, because of range measurement errors and registration error, two true surface cells representing the same surface region may not coincide. When the two octrees are merged as described by the above method, one of two true surface cells will most likely coincide with empty space and be eliminated from the result, and its attributes will just be discarded without being used. The other true surface cell will most likely coincide with solid space. The eliminated true surface cell's attributes should be combined with those of the other true surface cell that is not eliminated. To do that, for each true surface cell c_A in octree A, the nearest true surface cell c_B in a small neighborhood in octree B is found. If the surface normal of c_A and the surface normal of c_B are within an angular threshold, their attributes are combined and put back into c_A. This is repeated conversely for each true surface cell in octree B, but the old attributes of the true surface cells in octree A are used. Finally, octrees A and B are merged normally as described earlier.

For more efficient merging of two octrees, each interior node of the octree records the types of cells in its descendants. This can be done using a bit vector, where each type of cell is assigned a different bit position in the bit vector, and they are combined bottom-up in the ancestor nodes by bit-wise OR operations.

3.5 Computing Feasible Views

Feasible views are views that satisfy all the positioning constraints. As described in the overview of the next-best-view algorithm in Section 3.3.3, for efficiency, the integral part of the view metric in Equation (3.1) is evaluated only for feasible views. The positioning constraints considered in this specific view planning problem are (1) the scanner viewpoint must be placed at least a minimum clearance distance away from any object in the environment, (2) the scanner viewpoint can be positioned only between a minimum and a maximum heights above the floor, and (3) the scanner must never be placed above any object. This section describes how feasible views are computed and represented.

First, it is assumed that the floor is approximately parallel to, say, the *x-z* plane, and the *y*-axis is pointing vertically upwards. Then, the height of the objects in the environment and the height of the scanner viewpoint above the floor are measured only in the *y* direction. When the scanner is making the first scan, it may not be positioned perfectly vertically in the real environment. The acquired model must then be oriented so that the floor becomes parallel to the *x-z* plane. In the implementation, it is assumed that the bottom-most rows of the first range image are always the measurements of the floor only. A plane is then fitted to these "floor samples" and the model is oriented so that the fitting plane becomes parallel to the *x-z* plane. This needs to be performed only for the first scan because subsequent scans will be oriented correctly when they are registered to the cumulative partial model.

The set of feasible views are represented as 3D volumes in which every 3D point satisfies all the three positioning constraints. Figure 3.15 shows the resulting volumes of feasible views for an example model.

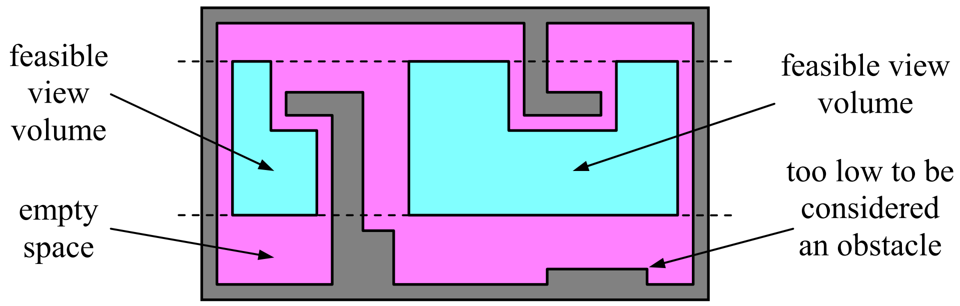


Figure 3.15: A side-view cross section of a room. The cyan regions are the feasible view volumes that satisfy the three positioning constraints. The two dotted lines indicate the minimum and maximum heights between which the scanner viewpoint can be positioned. To be considered an obstacle, an object must be over a threshold height above the floor

In the implementation, an octree is used to represent the feasible view volumes. Initially, this *feasible view octree* is a cube that encloses all the empty space in the cumulative partial model. The cube is then "carved" until it becomes the feasible view volumes. The non-empty-space cells in the partial octree model are used to "carve" the feasible view octree. First, every non-empty-space cell, whose top side is above the *obstacle threshold height* from the floor, is extruded upwards to infinity. The obstacle threshold height is used to prevent errors on the floor measurements and small low-lying objects from being considered as obstacles that will keep the scanner from being positioned on top of them. The extruded cells are then enlarged by the minimum clearance distance on all sides. Next, a large box is placed just above the scanner viewpoint maximum height, and another one just below the minimum height. These two boxes are large enough to enclose the empty space above the maximum height and below the minimum height. Finally, the feasible view octree cells are recursively subdivided until each cell does not intersect any of the modified non-empty-space cells and the two boxes. The subdivision also stops when the cell reaches a *minimum viewcell size*.

Figure 3.16 shows an example of the feasible view volumes computed by the implemented algorithm.

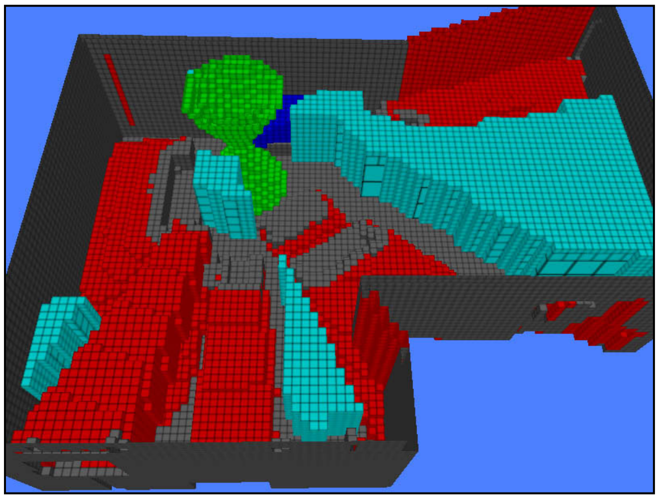


Figure 3.16: A view of the feasible view volumes for the octree partial model shown in Figure 3.12(c). The feasible view volumes are shown in cyan color. The minimum viewcell size is 4 inches wide.

3.6 Extracting Planar Patches

Referring to the integral part of the view metric in Equation (3.2), we recall that each feasible view v is evaluated with every surface point whose recorded sampling density $q(p)$ is less then the sampling density requirement D. For a view v, the values of $c(v,p)$ (the binary function for the sensing constraints) and $s(v,p)$ (the new scan sampling density) may remain constant or vary only slightly within some relatively large surface region. Similarly, for a surface point *p*, the values of $c(v,p)$ and $s(v,p)$ may remain constant or vary only slightly within some large, contiguous set of views. One of the major contributions of this work is the exploitation of this two-way spatial coherence to efficiently evaluate the view metric function. To exploit the spatial coherence, similar surface points and similar views are first grouped together.

In the implementation of the next-best-view algorithm, the surface cells in the cumulative partial model are grouped into a set of planar patches. Only surface cells whose best sampling densities (or recorded sampling densities) are less than the sampling density requirement *D* are grouped. These include the false surface cells. The surfaces cells are grouped into four types of patches, according to the type of the surface cells. A cell may only be grouped with other cells of the same type.

The planar patches are created in the following way. First, a surface cell is used as the seed of the new patch, and the cell's center point and surface normal are used as the initial fitting plane. The patch is grown in a "flood-fill" manner by including ungrouped cells that are 26-neighbor adjacent to cells that are already in the patch. A cell is added to the patch only if it is of the same type, has not reached the sampling density requirement, has a surface normal that is not too different from that of the fitting plane, and is not too far from the plane. The fitting plane is constantly updated when new cells are added to the patch. It is computed by fitting a plane to the center points of the cells already in the patch. When no new cell can be added to the patch, the patch is considered completed, and a new ungrouped cell is used as the seed to start a new patch.

Each planar patch has the following attributes that are needed during view evaluation: (1) a bounding rectangle, (2) an approximate surface area, (3) the average recorded sampling density, and (4) the *sampling deficit*. The bounding rectangle is used to make easy the evaluation of the sensing constraints and the subdivision of the patch. A bounding rectangle of the patch can be computed by applying principal component analysis [Kendall1977] on the cells' center points to generate the two axes of the rectangle in 3D space. The cells are then projected onto the plane defined by the two axes, and a bounding rectangle is computed on the plane.

An approximate surface area of each patch can be computed by "rasterizing" the patch's surface cells on the bounding rectangle. This can be done by laying a 2D grid on the bounding rectangle, and then project the surface cells' center points onto the grid. The area of the occupied grid cells are summed up to get the approximate surface area of the patch. Care must be taken to ensure that the resolution of the grid is not so high that holes will appear on the grid even though the patch is contiguous. To increase accuracy of the surface area approximation, one can increase the resolution of the grid and project multiple points from each surface cell onto the grid. The computation described here may be accelerated using graphics hardware, where the patch's surface cells are drawn onto a frame-buffer and the number of pixels that are switched on are used to compute the approximate surface area of the patch.

The average recorded sampling density is computed by summing up the recorded sampling densities of all the cells in the patch and then dividing the sum by the number of cells.

The sampling deficit is computed as

$$\text{sampling deficit} = (D - \text{average recorded sampling density}) \cdot \text{approximate surface area}$$

where D is the sampling density requirement. Since the average recorded sampling density of the patch is less than or equal to D, the sampling deficit is non-negative. The sampling deficit is the number of samples that still need to be acquired so that the average recorded sampling density of the patch can reach D. It actually corresponds to $\max\left(0,\ \min\left(s(v,p),\ D\right) - q(p)\right)$ of Equation (3.2) when $s(v,p) \geq D$ and $q(p) < D$.

In the implementation, patches that are too large are subdivided until their bounding rectangles are smaller than a threshold maximum length, which is typically set at 5 to 10 feet in the experiments. Figure 3.17 shows examples of the planar patches extracted from the octree model shown in Figure 3.12(c).

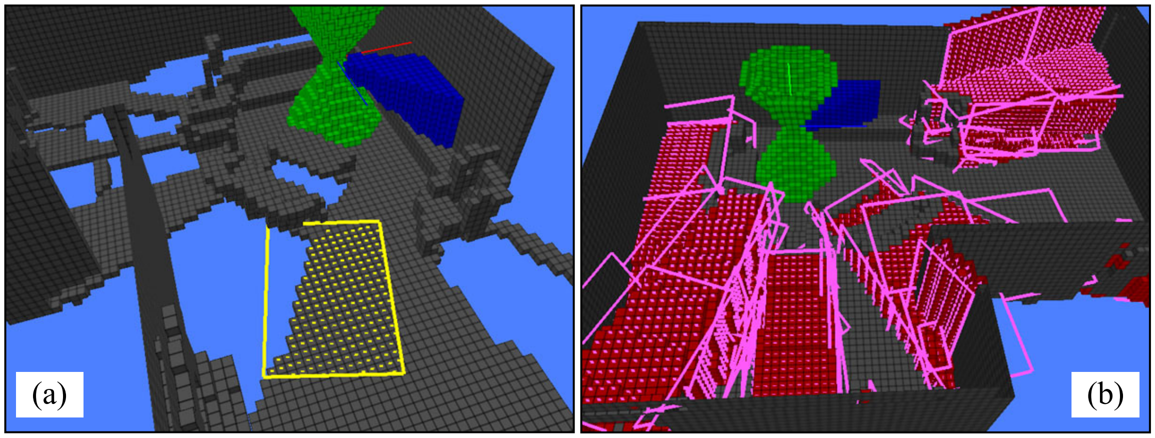


Figure 3.17: (a) A true surface patch consisting of true surface cells whose recorded sampling densities are less than the requirement. The yellow rectangle is the bounding rectangle of the patch and the yellow dots indicate the cells that are included in the patch. (b) A set of occlusion surface patches, whose bounding rectangles are shown in magenta color.

3.7 Ranking Planar Patches

We recall that the reason for ranking the patches is that there may not be enough time to consider all the patches for the evaluation of the views, and therefore one would want to evaluate the views using the most important patches. Here, the patches are ranked in descending order of their importance. The most important patch should potentially have the greatest impact on the value of $g(v)$ in Equation (3.2). This leads to the use of the following patch importance value:

$$\text{patch importance value of } P = w(P) \cdot \text{sampling deficit of } P$$

where P is the patch, and $w(P)$ is the weight assigned to the surface type of P, which is the same as the weight $w(p)$ in Equation (3.2). The patches are then sorted in descending order on their patch importance values.

Chapter 4

View Evaluation

One of the main steps in the next-best-view algorithm is the evaluation of the integral part of the view metric function in Equation (3.1) for each feasible view. This is termed *view evaluation*. The major challenge to a practical next-best-view solution is to develop an efficient method to evaluate a large set of views, using information provided by the cumulative partial model of the environment. The evaluation of each view can be computationally very expensive, since a large amount of information of the partial model may be involved, and visibility computations and other sensing constraint evaluations are expensive. This is the main difficulty that has limited many previous next-best-view algorithms to an incomplete search space, simple and small objects, incomplete sets of constraints, and low-quality acquisition.

A major contribution of this work is the novel application of a hierarchical approach to greatly accelerate view evaluations by exploiting the various spatial coherences in the acquisition constraints and reconstruction quality requirements in the view metric. In this work, the hierarchical view evaluation approach is applied to the specific next-best-view problem in which the range scanner is monostatic and the views are 3D positions with fixed orientation. The hierarchical approach is described in detail in the earlier part of this chapter. In the later part of the chapter, the hierarchical approach is generalized for more general scanning poses, acquisition constraints and quality requirements.

4.1 Hierarchical View Evaluation

We recall from Equation (3.2), the integral part of the view metric is

$$g(v) = \int_{p \in S} w(p) \cdot c(v,p) \cdot \max\left(0,\ \min\left(s(v,p),\ D\right) - q(p)\right)\ dp,$$

in which each feasible view v is evaluated with every surface point whose recorded sampling density $q(p)$ is less than the sampling density requirement D. These surfaces are referred to as *under-sampled surfaces*, and they include false surfaces. Equation (3.2) can be evaluated by first uniformly discretizing the feasible view volumes and the under-sampled surfaces at sufficiently high resolutions. Unfortunately, a brute-force approach of evaluation of Equation (3.2) for all discrete views would be prohibitively expensive.

The amount of computation can actually be reduced by exploiting the spatial coherences in the sensing constraints and the sampling quality function in Equation (3.2). The basic idea is that if a constraint is satisfied between a view v_1 and a surface point p_1 on the partial model, very likely the same constraint is also satisfied between another view v_2 and p_1, provided v_2 is near to v_1. The same constraint is also very likely to be satisfied between v_1 and another surface point p_2 that is near p_1. These spatial coherences can be exploited using a hierarchical approach. Neighboring views are first grouped into view volumes, and neighboring surface points are grouped into surface patches. The constraint is evaluated between each view volume V and each patch P. If it is entirely satisfied or entirely not satisfied between V and P, then the constraint evaluation is considered completed between every view in V and every surface point in P. If the constraint is partially satisfied between V and P, then we subdivide either V or P, and continue the evaluation on the children.

4.1.1 Formulation

Suppose all the under-sampled surfaces in the partial model have been partitioned into $\{P_i \mid i = 1,\ldots,N\}$, where each P_i is a patch. Then, Equation (3.2) can be rewritten as

$$g(v) = \sum_{i=1}^{N} g(v, P_i) \tag{4.1}$$

where

$$g(v, P) = \int_{p \in P} w(p) \cdot c(v, p) \cdot \max(0,\ \min(s(v, p),\ D) - q(p))\ \ dp\,. \tag{4.2}$$

Now, we will focus on evaluating views with respect to a patch, instead of with all the surfaces in the partial model. Suppose the values of $w(p)$, $c(v, p)$, $s(v, p)$, and $q(p)$ remain constant between a view volume *V* and a patch *P*, where $v \in V$ and $p \in P$. Then $g(v, P)$ can be computed as

$$g(v, P) = g(V, P) = w(P) \cdot c(V, P) \cdot \max(0,\ \min(s(V, P),\ D) - q(P)) \cdot a(P) \tag{4.3}$$

where

- $w(P)$ is the weight or importance value assigned to the surface type of the surface points that made up patch *P*. It is assumed that all the surface points in *P* are of the same type. The four different surface types are the true surfaces, the occlusion surfaces, the hole-boundary surfaces, and the image-boundary surfaces;
- $c(V, P)$ indicates whether all the sensing constraints are satisfied between every view $v \in V$ and every surface point $p \in P$. $c(V, P) = 1$ if $c(v, p) = 1$ for all $v \in V$ and $p \in P$, or $c(V, P) = 0$ if $c(v, p) = 0$ for all $v \in V$ and $p \in P$, otherwise $c(V, P)$ is undefined;
- $s(V, P)$ is the new scan sampling density on patch *P* if it is scanned from any view $v \in V$;
- $q(P)$ is the recorded sampling density of patch *P*. It is just the average of the recorded sampling densities of all the surface points in *P*;
- $a(P)$ is the surface area of patch *P*.

In actual fact, the value of $s(v, p)$ does not stay constant between *V* and *P*. However, if every $s(v, p)$ between *V* and *P* is bounded within a small interval, then it is considered approximately constant. The value of $s(v,p)$ between *V* and *P* is considered approximately constant when

$$\frac{s_{\max}(V,P) - s_{\min}(V,P)}{s_{\max}(V,P)} \le \varepsilon_s \tag{4.4}$$

where $s_{\max}(V,P)$ and $s_{\min}(V,P)$ are the maximum and the minimum new scan sampling densities, respectively, between *V* and *P*, and $\varepsilon_s \ge 0$ is a user-specified relative error bound. When this is the case, one can let $s(V,P)$ be any $s(v,p)$ between *V* and *P*. For conservative satisfaction of the sampling quality requirement, $s(V,P)$ is chosen to be $s_{\min}(V,P)$. Then, the *relative representative error* of $s(V,P)$ is defined as

$$\frac{s_{\max}(V,P) - s(V,P)}{s_{\max}(V,P)}. \tag{4.5}$$

With the condition in Equation (4.4) satisfied and $s(V,P) = s_{\min}(V,P)$, the value of $g(V,P)$ can be computed using Equation (4.3). If any sensing constraint in $c(V,P)$ is found entirely not satisfied between *V* and *P*, then $s(V,P)$ need not be computed and the result is $g(V,P) = 0$.

If $c(v,p)$ is not constant or $s(v,p)$ is not approximately constant between *V* and *P*, then $g(V,P)$ cannot be computed using Equation (4.3). In this case, one can subdivide either patch *P* or view volume *V*, and apply Equation (4.3) on the sub-patches or the sub-volumes. If patch *P* is subdivided, then

$$g(V,P) = g(V,P^1) + \cdots + g(V,P^k) \tag{4.6}$$

where $P^1,\ldots,P^k$ are the sub-patches of patch *P*. If view volume *V* is subdivided, then $g(V,P)$ is replaced with $g(V^1,P),\ldots,g(V^m,P)$, where $V^1,\ldots,V^m$ are the sub-volumes of *V*. Then, $g(v,P) = g(V^i,P)$ if $v \in V^i$. The subdivision stops when $c(v,p)$ is constant and

$s(v,p)$ is approximately constant between the view volume and the patch. The subdivision is thus adaptive to the rates of change of $c(v,p)$ and $s(v,p)$

4.1.2 Algorithm and Implementation

The success of the hierarchical view evaluation approach depends on the efficiency of evaluating $c(V,P)$ and $s(V,P)$. In the specific next-best-view problem addressed by this work, $c(v,p)$ consists of the four separate sensing constraints described in Section 3.2.6, which are

- The **maximum-range constraint**, represented by $c_0(v,p)$. If the distance between view *v* and surface point *p* is more than the maximum effective range of the range sensor, then $c_0(v,p)=0$, otherwise $c_0(v,p)=1$;
- The **vertical-field-of-view constraint**, represented by $c_1(v,p)$. If the surface point *p* is outside the vertical field of view of the scanner at view *v*, then $c_1(v,p)=0$, otherwise $c_1(v,p)=1$;
- The **angle-of-incidence constraint**, represented by $c_2(v,p)$. If the angle between the surface normal vector at *p* and the direction vector from *p* to *v* is greater than a threshold angle, then $c_2(v,p)=0$, otherwise $c_2(v,p)=1$;
- The **visibility constraint**, represented by $c_3(v,p)$. If the line of sight from *v* to *p* is occluded, then $c_3(v,p)=0$, otherwise $c_3(v,p)=1$.

The binary function $c(v,p)$ is defined as $c(v,p)=c_0(v,p)\cdot c_1(v,p)\cdot c_2(v,p)\cdot c_3(v,p)$.

Similarly, $c(V,P)$ is also made up of $c_0(V,P)$, $c_1(V,P)$, $c_2(V,P)$, and $c_3(V,P)$, where for $i=0,\ldots,3$, $c_i(V,P)=1$ if $c_i(v,p)=1$ for all $v\in V$ and $p\in P$, or $c_i(V,P)=0$ if $c_i(v,p)=0$ for all $v\in V$ and $p\in P$, otherwise $c_i(V,P)$ is undefined. The function $c(V,P)$ is defined as $c(V,P)=c_0(V,P)\cdot c_1(V,P)\cdot c_2(V,P)\cdot c_3(V,P)$, but $c(V,P)$ is undefined if at least one $c_i(V,P)$ is undefined and all the other are either 1 or undefined. If at least one $c_i(V,P)$ is 0, $c(V,P)$ will be 0. Intuitively, when $c_i(V,P)$ is undefined, it means that the corresponding constraint is only partially satisfied between *V* and *P*.

To avoid redundant computations, an important point to note from Equation (4.3) is that as soon as any one of $c_i(V,P)$ or $\max(0,\ \min(s(V,P),\ D) - q(P))$ has been determined to be 0, the computations of the others can be skipped because $g(V,P)$ can already be determined to be 0.

Methods are needed to group views into volumes and surface points into patches, which should be appropriately represented to facilitate the subdivisions of view volumes and patches. As described in Chapter 3, the feasible view volumes are represented in a feasible view octree in the implementation. The under-sampled surfaces are represented as voxels in a partial octree model, and they are then grouped into planar patches, where each patch is tightly bounded by a rectangle. The feasible view octree and the rectangle-bounded planar patches provide efficient ways to group and subdivide view volumes and surface patches. A node in the feasible view octree is referred to as a *viewcell*.

In Figure 4.1 is a simplified C-like procedure to evaluate $g(V,P)$. Here, input viewcell V is a feasible view volume. Each input Boolean element `c_in[i]` is `true` if $c_i(V,P)$ is already known to be 1, otherwise `c_in[i]` is `false` to indicate that $c_i(V,P)$ is unknown. The input Boolean argument `s_uniform` is `true` if the relative representative error of $s(V,P)$ is known to be bounded by ε_s, otherwise `s_uniform` is `false`. The input argument `s` is $s(V,P)$ if `s_uniform` is `true`. Initially, the procedure `EvaluateView()` is called with all `c_in[i]=false` and `s_uniform=false`.

```
EvaluateView( Viewcell *V, Patch *P,
              bool c_in[4], bool s_uniform, float s )
{
   if ( s_uniform  &&  MIN( s, D ) - q(P) <= 0 ) return;

   bool c[4] = { c_in[0], c_in[1], c_in[2], c_in[3] };

   for ( int i = 0; i < 4; i++ )
      if ( !c[i] )
      {
         int t = EvaluateConstraint( i, V, P );
         if ( t == 0 ) return;
         if ( t == 1 ) c[i] = true;
      }

   if ( !s_uniform )
   {
      float sMin, sMax;
      EvaluateSamplingDensity( V, P, &sMin, &sMax );
      if ( sMax <= q(P) ) return;
      if ( ( sMax - sMin ) / sMax <= epsilon_s )
      {
         s_uniform = true;
         s = sMin;
         if ( MIN( s, D ) - q(P) <= 0 ) return;
      }
   }

   if ( c[0] && c[1] && c[2] && c[3] && s_uniform )
   {
      V->score += w(P) * ( MIN( s, D ) - q(P) ) * a(P);
   }
   else if ( ToSubdividePatchFirst( V, P ) )
   {
      SubdividePatch(P);
      for ( int k = 0; k < P->numChildren; k++ )
         EvaluateView( V, P->child[k], c, s_uniform, s );
   }
   else
   {
      SubdivideViewcell(V);
      for ( int m = 0; m < V->numChildren; m++ )
         EvaluateView( V->child[m], P, c, s_uniform, s );
   }
}
```

Figure 4.1: A procedure `EvaluateView()` to evaluate $g(V,P)$.

The program code in Figure 4.1 is explained below:

- **Line 4**: If the relative representative error of $s(V,P)$ is already known to be bounded by ε_s, and if $\max(0,\ \min(s(V,P),\ D)-q(P))$ is 0, then the procedure can be quickly exited because $g(V,P)=0$;
- **Lines 8–14**: Evaluate each $c_i(V,P)$ if $c_i(V,P)$ is unknown. The function `EvaluateConstraint(i,V,P)` evaluates $c_i(V,P)$ and returns 0 or 1 to indicate $c_i(V,P)=0$ or $c_i(V,P)=1$, respectively, or returns any other integer values to indicate $c_i(V,P)$ is undefined. If it is found that any $c_i(V,P)=0$, then the procedure can be quickly exited because $g(V,P)=0$. The implementation of the function `EvaluateConstraint()` is described in the next section;
- **Lines 16–27**: If it is not known whether the relative representative error of $s(V,P)$ is bounded by ε_s (`epsilon_s`), the function `EvaluateSamplingDensity (V,P,&sMin,&sMax)` evaluates the minimum and maximum new scan sampling densities between *V* and *P*. In line 20, the procedure is exited if it is certain that $\max(0,\ \min(s(V,P),\ D)-q(P))$ will be 0. Using the results `sMin` and `sMax`, the relative representative error of $s(V,P)$ is checked to see whether it is bounded by ε_s. If so, $s(V,P)$ is assigned the value of `sMin`. Again, the procedure is exited if $\max(0,\ \min(s(V,P),\ D)-q(P))$ is 0. The implementation of the function `EvaluateSamplingDensity()` is described in the next section;
- **Lines 29–32**: If all $c_i(V,P)=1$ and the relative representative error of $s(V,P)$ is bounded by ε_s, then $g(V,P)$ is computed as in Equation (4.3) and the result added to the score of viewcell *V*. The function `q(P)` returns the average recorded sampling density of patch *P*, and the function `a(P)` returns the approximate surface area of *P*;
- **Lines 33–38**: If some of the $c_i(V,P)$ is unknown (or undefined) or the relative representative error $s(V,P)$ is not bounded by ε_s, then either the viewcell *V* or the patch *P* needs to be subdivided. Here, the patch is subdivided because the function `ToSubdividePatchFirst(V,P)` returns `true`. Figure 4.2 shows an

implementation of the function `ToSubdividePatchFirst()`. The function `SubdividePatch(P)` subdivides patch *P* into sub-patches if it is not already subdivided. Next, the procedure `EvaluateView()` is recursively called with *V* and every sub-patch of *P*. Note that all $c_i(V,P)$ and $s(V,P)$ are passed to `EvaluateView()` for the evaluation of $g(V,P^k)$, where P^k is a sub-patch of *P*.

- **Lines 40–44**: Similarly, here, the viewcell is subdivided instead of the patch. The function `SubdivideViewcell(V)` subdivides viewcell *V* into sub-viewcells if it is not already subdivided. Then, the procedure `EvaluateView()` is recursively called with *P* and every sub-viewcell of *V*.

```
bool ToSubdividePatchFirst( Viewcell *V, Patch *P )
{
   if ( !Subdivisible(V) && Subdivisible(P) )
      return true;
   else if ( Subdivisible(V) && !Subdivisible(P) )
      return false;
   else if ( V->width <= P->length )
      return true;
   else
      return false;
}
```

Figure 4.2: An implementation of the function `ToSubdividePatchFirst()`.

In the actual implementation, when a patch is subdivided, it is subdivided into four, or perhaps fewer, sub-patches by splitting its bounding rectangle at the center. An example is shown in Figure 4.3. The bounding rectangles of the sub-patches are recomputed, but they remain in the same orientation as the bounding rectangle of the parent patch. The orientation is kept the same so that the bounding rectangles of the sub-patches do not overlap. The average recorded sampling density and approximate surface area of each sub-patch are recomputed as described in Section 3.6. When a viewcell *V* is subdivided, it is subdivided into eight equal octants, which are added to the feasible view octree as eight child nodes of viewcell *V*. Each sub-viewcell will have an initial score of 0. The score of each viewcell *V* will be the sum of $g(V,P)$ over all the patches and sub-patches P that have been successfully evaluated with V.

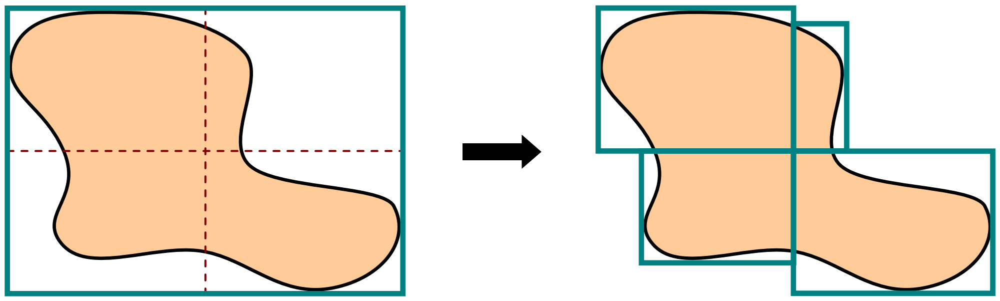


Figure 4.3: The bounding rectangles of the four sub-patches after a patch is subdivided. The orientation of each sub-patch's bounding rectangle remains the same as that of the original patch.

The subdivision of a patch or a viewcell stops when it has reached a user-specified minimum size. When an indivisible patch is passed to the function `EvaluateConstraint()` and `EvaluateSamplingDensity()`, it is treated as a point at the center of the patch. Similarly, an indivisible viewcell is treated as a point at the center of the viewcell.

It is important to note that when $c_i(V,P)=1$ for some *i*, it is also true that $c_i\left(V,P^k\right)=1$ and $c_i\left(V^m,P\right)=1$ for all sub-patches P^k of P and all sub-viewcells V^m of V. Therefore, when $c_i(V,P)=1$ for some *i*, and `EvaluateView()` is called with the sub-patches P^k of P or the sub-viewcells V^m of V, there is no need to recompute $c_i\left(V,P^k\right)$ and $c_i\left(V^m,P\right)$. This is similar for $s(V,P)$, when its relative representative error has been determined to be bounded by ε_s. This important observation can eliminate a large amount of computation and demonstrates the great benefit of evaluating the four constraints separately as $c_i(V,P)$ for $i=0,\ldots,3$, instead of combining them into one $c(V,P)$. This is because once a constraint is determined to be 0 or 1 for V and P, it needs not be evaluated anymore for their descendents.

The relative representative error bound ε_s determines how accurate $s(V,P)$ is to be computed. Smaller ε_s results in more accurate $s(V,P)$ but also results in more levels of subdivision of the viewcell and patch. This property allows one to vary the value of ε_s to trade-off between efficiency and accuracy.

Another optimization that can be added to the algorithm is to order the evaluations of the constraints and the sampling density so that the most efficient one is first and the least efficient last. This is to exploit the many early exits in the algorithm, so that the more inefficient evaluations are less likely to be executed. Generally, the visibility constraint $c_3(V, P)$ is the most expensive to evaluate, thus it should be placed at the end.

The procedure `EvaluateView()` in Figure 4.1 evaluates a feasible viewcell with respect to only one patch. To complete the view planning step in the model acquisition cycle, all feasible viewcells in the initial feasible view octree have to be evaluated with respect to all patches in the partial model. We recall that the time for the view planning step may be limited, and in this case one would want to evaluate the views with respect to the most important patches. In Section 3.7, the patches are ranked based on their importance values. When execution time is limited, all initial feasible viewcells are evaluated with the most important patches until the allotted time expires.

When the entire view evaluation is completed or when the allotted time has expired, some of the initial feasible viewcells will have descendent viewcells created by subdivisions. Each viewcell V has a score that is the accumulated value of $g(V, P)$ over all the patches and sub-patches P that have been successfully evaluated with V. In the above implementation, a viewcell's score is not propagated down to its children. Since each child viewcell contains part of the view volume of its parent viewcell, the scores in the children should include the parent's score. Therefore, the score of each viewcell should be updated by adding the scores of its ancestors. The updates are done after all patches have been evaluated with the views, or after the allotted time has expired. Figure 4.4 illustrates how the scores are propagated and updated.

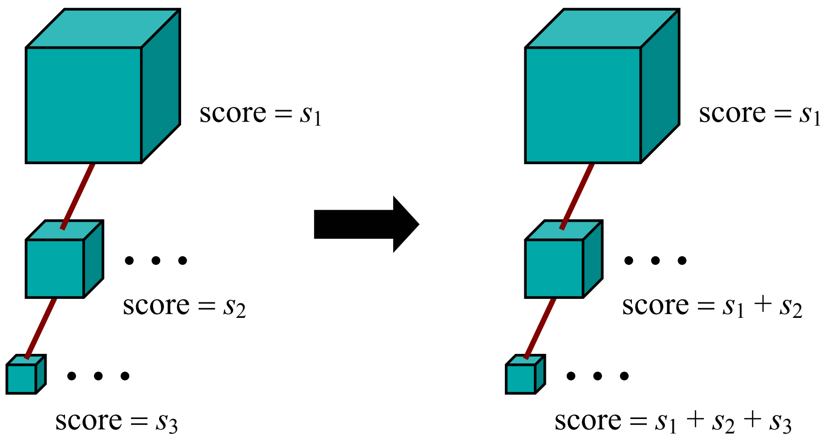


Figure 4.4: After the entire view evaluation is completed or after the allotted time has expired, the score of each viewcell is updated by adding the scores of its ancestors.

All the feasible viewcells that are leaf nodes in the resulted feasible view octree are collected into the *candidate viewcell set*. The center of each candidate viewcell is one of the *candidate views* that will be considered for the next best view. These candidate views are sorted in decreasing order on their updated scores. If the scores of all candidate views are below a specified threshold value, the view planning system will suggest the termination of the acquisition process.

In the next step of the next-best-view algorithm, the candidate views are checked for the registration constraint. Starting from the highest-score candidate view to the lowest, the registration constraint is checked to determine whether it is satisfied by each view. A view is checked for the registration constraint only if it is at least a specified distance away from all the previously-checked views, otherwise it is skipped. This is because views with similar scores tend to cluster near each other and they usually have similar registration accuracies too. The first view found to satisfy the constraint is output as the next best view.

There is one inefficiency in the procedure `EvaluateView()` listed in Figure 4.1. The procedure accepts only feasible viewcells. However, many of the initial feasible viewcells in the feasible view octree may already be very small when the octree is created. It is very inefficient to call `EvaluateView()` with a large number of small viewcells, and evaluate the constraints and sampling quality on them. An improvement to this is to allow `EvaluateView()` to also accept partially feasible viewcells as input. Each partially feasible viewcell has some descendent viewcells that are feasible. The constraints and sampling quality evaluations are applied to a partially feasible viewcell as if it were a feasible viewcell. After that, a partially feasible viewcell will always be subdivided, and its non-feasible sub-viewcells will be ignored, but the other sub-viewcells will be passed to `EvaluateView()` for evaluation.

An issue not yet addressed by the above view evaluation algorithm is the effects on the views' scores caused by the potential scanner's physical pose errors. The hierarchical algorithm can be slightly modified to take into consideration these effects, and a simple solution is proposed in Section 4.1.4.

4.1.3 Constraint and Sampling Density Evaluations

This section describes the implementation of `EvaluateConstraint()` and `EvaluateSamplingDensity()` to evaluate $c_i(V,P)$ and $s(V,P)$, respectively. The success of the hierarchical view evaluation depends on how efficiently they can be evaluated.

In actual fact, `EvaluateConstraint(i,V,P)` need not evaluate $c_i(V,P)$ precisely, in the sense that when `EvaluateConstraint(i,V,P)` returns 1 or 0, it implies that $c_i(V,P)=1$ or $c_i(V,P)=0$, respectively, but the inverse implication may not be true. When $c_i(V,P)=1$ or $c_i(V,P)=0$, `EvaluateConstraint(i,V,P)` may return undefined. This is preferred when it is expensive to precisely determine whether $c_i(V,P)=1$ or $c_i(V,P)=0$. By returning undefined, the precise evaluation of the constraint is left to the sub-patches of *P* or the sub-viewcells of *V*, and because of their smaller sizes, they are more likely to belong to one of the easy cases. Of course, when $c_i(V,P)$ is undefined, `EvaluateConstraint(i,V,P)` must return undefined.

For the same purpose, `EvaluateSamplingDensity(V,P,&sMin,&sMax)` need not return the precise minimum and maximum new scan sampling densities between *V* and *P*. It is allowed to underestimate the minimum new scan sampling density and overestimate the maximum new scan sampling density.

The following sections describe the algorithms. There are certainly some other efficient ways to accomplish these operations. When *V* is indivisible or *P* is indivisible, where they are treated as points, the algorithms are generally trivial, so they are not described here.

4.1.3.1 Maximum-Range Constraint

Let the maximum effective range of the range sensor be $R_{\max}$. When $c_0(V,P)=1$, the distance between any point in patch *P* and any view in *V* is equal to or less than $R_{\max}$. To determine this, four spheres of radius $R_{\max}$ are centered at the four corners of the patch's bounding rectangle. If the entire viewcell *V* is inside all the four spheres, then `EvaluateConstraint(0,V,P)` returns 1. The viewcell can be approximated with a bounding sphere to speed up the computation a little, but the result is less precise. If the viewcell (or its bounding sphere) intersects or is inside some but not all the four spheres, undefined is returned.

When $c_0(V,P)=0$, the distance between any point in patch *P* and any view in *V* is greater than $R_{\max}$. This can be determined as follows. Let *C* be the convex hull of the four spheres of radius $R_{\max}$ that are centered at the four corners of the patch's bounding rectangle. The convex hull is shown in Figure 4.5. If the viewcell is entirely outside the convex hull *C*, then $c_0(V,P)=0$. By approximating the viewcell with a sphere, it is not hard to efficiently determine whether the sphere is outside the convex hull. In the extreme, one can even approximate the convex hull *C* with a bounding box. When the viewcell (or its bounding sphere) has been determined to be outside the convex hull, `EvaluateConstraint(0,V,P)` returns 0.

For all other cases, `EvaluateConstraint(0,V,P)` returns undefined to indicate that the actual value of $c_0(V,P)$ is still uncertain, or that the constraint is satisfied by only some, but not all, pairs of views and patch points.

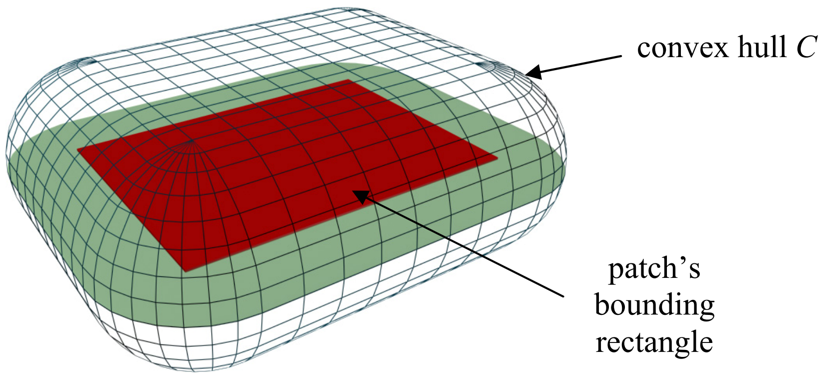


Figure 4.5: The convex hull of the four spheres of radius $R_{\max}$ that are centered at the four corners of the patch's bounding rectangle. The red rectangle is the patch's bounding rectangle.

4.1.3.2 Vertical-Field-of-View Constraint

The scanner used in the specific view planning problem has a 360° horizontal field of view, but the vertical field of view is limited, as shown in Figure 4.6. To determine whether a surface point is within the vertical field of view, we compute the angle between the *y*-axis (vertical axis) and the vector from the view position to the surface point. If the angle is less than θ_{top} or more than $(180° - \theta_{\text{bot}})$, then the surface point is outside the vertical field of view.

When $c_1(V,P) = 0$, every point in patch *P* is outside the field of view of every view in *V*. Figure 4.7 illustrates a method to determine whether $c_1(V,P) = 0$. If the directions of all the four directed lines in Figure 4.7(a) are in the bottom outside region of the vertical field of view, then `EvaluateConstraint(1,V,P)` returns 0. Similarly, in Figure 4.7(b), if the directions of all the four directed lines are in the top outside region, `EvaluateConstraint(1,V,P)` also returns 0. If some of these directed lines are inside and some are outside the vertical field of view, then `EvaluateConstraint(1,V,P)` returns undefined.

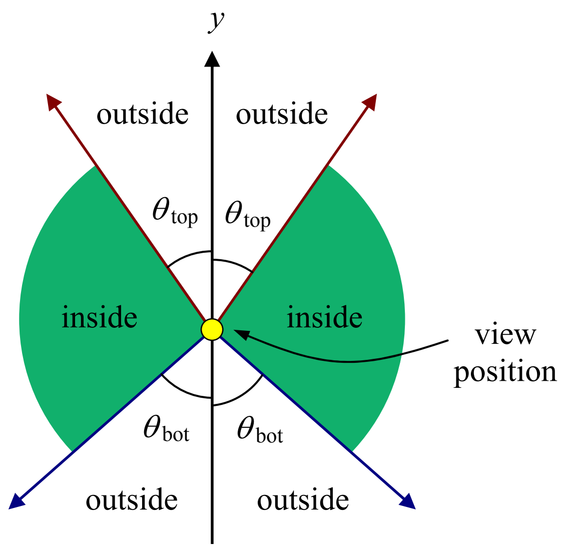


Figure 4.6: The vertical field of view of the scanner. If the direction from the view position to a surface point is in the "inside" region, then the surface point is in the field of view of the scanner at the view position.

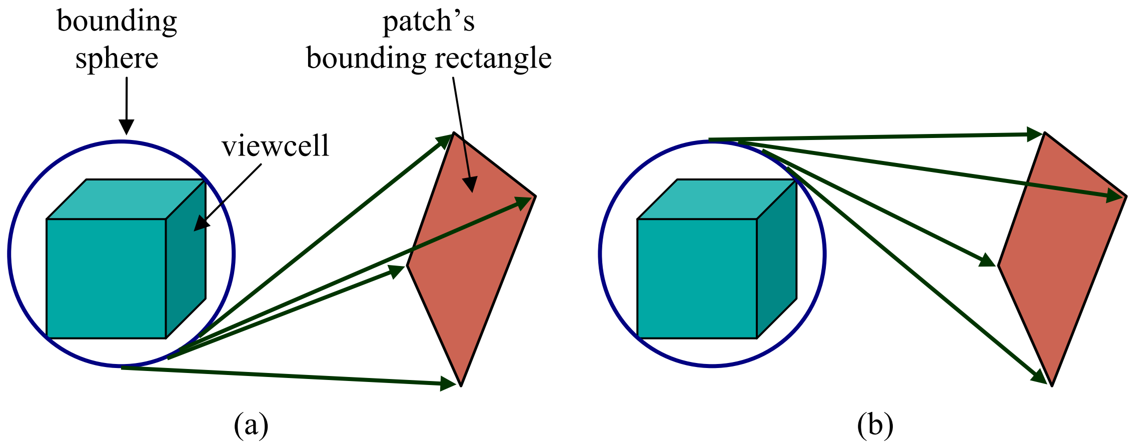


Figure 4.7: (a) The four directed lines are tangent to the sphere and point at the four corners of the patch's bounding rectangle. Each directed line touches the sphere at the lowest point where it is still tangent to the sphere. If the angles between the y-axis and all the directed lines are larger than $180° - \theta_{\text{bot}}$, then the patch is entirely in the bottom outside region of the vertical field of view of the viewcell. (b) Each of the four directed lines touches the sphere at the highest point where it is still tangent to the sphere. If the angles between the y-axis and all the directed lines are less than θ_{top}, then the patch is entirely in the top outside region of the vertical field of view of the viewcell.

To determine whether $c_1(V,P)=1$, one can use the method illustrated in Figure 4.8. If all the four corners of the patch's bounding rectangle are on the positive sides (the side where the normal vector is pointing) of both planes A and B, then the patch is entirely inside the vertical field of view of every view in the viewcell and the value to be returned by `EvaluateConstraint(1,V,P)` is 1. Otherwise, `EvaluateConstraint(1,V,P)` returns undefined.

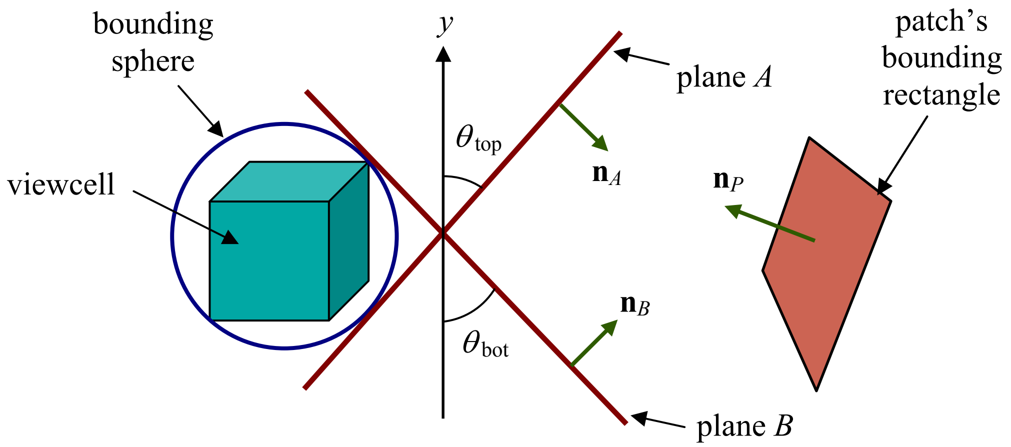


Figure 4.8: Determining whether a patch is entirely inside the vertical field of view of every view in the viewcell. Planes A and B are both tangent to the bounding sphere of the viewcell. The planes' normal vectors $\mathbf{n}_A$ and $\mathbf{n}_B$ are coplanar with the normal vector $\mathbf{n}_P$ of the patch. If all the four corners of the patch's bounding rectangle are on the positive side (the side where the normal vector is pointing) of both planes A and B, then the patch is entirely inside the vertical field of view of every view in the viewcell.

4.1.3.3 Angle-of-Incidence Constraint

The angle of incidence, ϕ, of a surface point p from a view position v is the angle between the surface normal vector at p and the direction vector from p to v. If this angle is greater than a threshold angle $\phi_{\max}$, then $c_2(v,p)=0$, otherwise $c_2(v,p)=1$.

To determine whether $c_2(V,P)=1$, four open-ended cones are set up at the four corners of the patch's bounding rectangle as shown in Figure 4.9. The base of each cone extends infinitely in the direction of the patch's normal vector, and the half angle at the apex of each cone is $\phi_{\max}$. If the viewcell V (or its bounding sphere) is entirely inside all four cones, then

$c_2(V,P)=1$, and `EvaluateConstraint(2,V,P)` returns 1. If the viewcell intersects or is inside some but not all four cones, then `EvaluateConstraint(2,V,P)` returns undefined.

To determine whether $c_2(V,P)=0$, the viewcell V (or its bounding sphere) must be entirely outside the open-ended convex hull that encloses all the four cones in Figure 4.9. In this case, `EvaluateConstraint(2,V,P)` returns 0, otherwise it returns undefined.

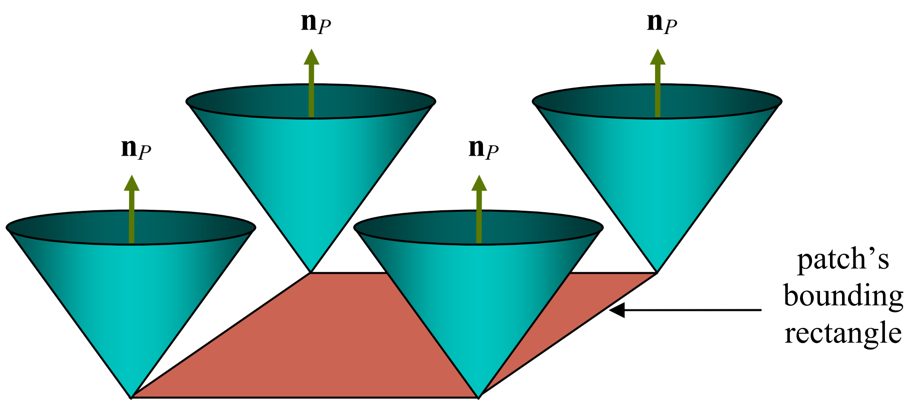


Figure 4.9: The four open-ended cones set up at the four corners of the patch's bounding rectangle. The base of each cone extends infinitely in the direction of the patch's normal vector, and the half angle at the apex of each cone is $\phi_{\max}$. $\mathbf{n}_P$ is the normal vector of the patch.

4.1.3.4 Visibility Constraint

Visibility between the scanner viewpoint and each surface point seems to be the most important, and sometimes the only, sensing constraint that is evaluated in almost all existing next-best-view algorithms. Here, we are testing the visibility between a viewcell and a rectangle-bounded patch.

When $c_3(V,P)=1$, it implies that every point in the viewcell can see every point on the patch without obstruction, i.e. the viewcell and the patch are *totally visible* to each other. To determine that, one has to ensure that there is no occluder in the *shaft* [Haines1994] between the viewcell and the patch, which is the 3D volume occupied by the line segments connecting every point in the viewcell to every point on the patch. A 2D analogy of the shaft between a viewcell and a patch's bounding rectangle is shown in Figure 4.10.

To determine $c_3(V,P)=1$, bounding planes are constructed to enclose the shaft between the viewcell and the patch's bounding rectangle. For efficiency, one can reduce the number of bounding planes needed by enclosing an approximation volume larger than the exact shaft. The surface and solid-space voxels in the partial octree model are then tested, in a top-down traversal manner, against all the bounding planes to find out if any of the voxels is inside or intersects the volume bounded by the bounding planes. The optimizations used in hierarchical view frustum culling [Assarsson2000] are applied to test the voxels against the bounded volume.

If no surface and solid-space voxel is found to be inside or intersect the bounded volume, `EvaluateConstraint(3,V,P)` returns 1 to indicate *total visibility* between the viewcell and the patch.

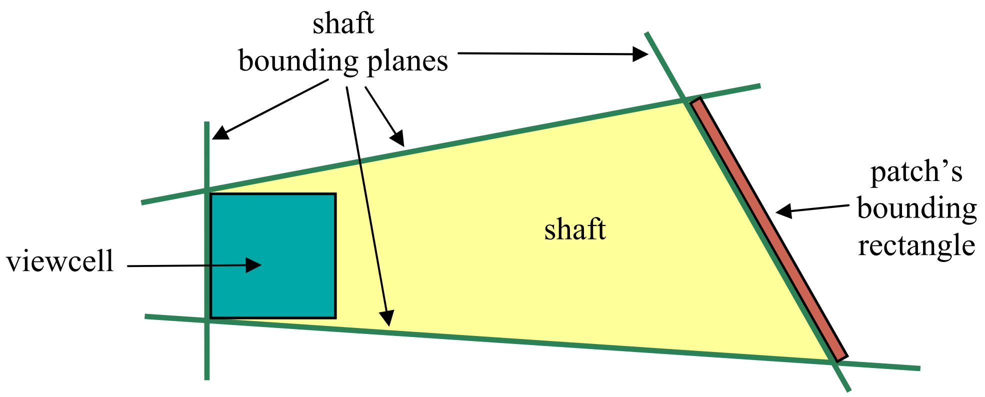


Figure 4.10: Bounding planes are constructed to enclose the shaft between the viewcell and the patch's bounding rectangle.

When $c_3(V,P)=0$, the viewcell and the patch are *totally invisible* to, or *totally occluded* from, each other. Determining *total invisibility* or *total occlusion* between two extended objects is a difficult problem because the total occlusion may be caused by multiple occluders that are not connected to each other [Cohen-Or2003]. This is made worse in the case when there are many individual small occluders. In computer graphics, most of the methods that deal with visibility between extended objects are for occlusion culling in interactive walkthrough [Cohen-Or2003], for computing shadows and for global illumination computation [Teller1993, Durand1999]. These methods are mostly used in preprocessing stages and are usually very computationally expensive, and many cannot sufficiently combine the occlusion of individual small occluders to quickly determine total occlusion.

In my case, many small voxels of the partial octree model may appear in the shaft between *V* and *P*. If it can be determined that, within the shaft, there is no path, straight or non-straight, between some point in the viewcell and some point on the patch, then one can conclude that *V* and *P* are totally occluded from each other. Of course, the inverse implication is not true because even if a path exists, *V* and *P* may still be totally occluded. This is fine for the function `EvaluateConstraint(3,V,P)` because it is allowed to return undefined even if *V* and *P* are actually totally occluded. The method of finding a path seems very suitable for indoor environments where there are large walls and partitions, and it will be able to correctly detect total occlusion when *V* and *P* are on opposite sides of some large wall or in different rooms. This can avoid many unnecessary subdivisions of the viewcells or the patches due to the inability to detect total occlusion early.

However, finding a path from *V* to *P* takes time quadratic to the number of voxels in the shaft [O'Rourke1998]. When a large shaft is obstructed or almost obstructed, there will be a large number of voxels in it, and it will take a long time to determine whether a path exists.

In the absence of an efficient algorithm, I have chosen to use a probabilistic approach to estimate total occlusion between a viewcell and a patch. The method is illustrated in Figure 4.11. On the viewcell's bounding sphere, the great circle parallel to the patch is first identified. Then, an equal number of random points is generated in each quadrant of the disc bounded by the great circle, and in each quadrant of the patch's bounding rectangle. Rays are shot from the random points on the disc to the points on the patch's bounding rectangle. In the implementation, 16 random rays are generated in this way. If all the random rays are occluded, then it is estimated that the patch is totally occluded from the viewcell, and `EvaluateConstraint(3,V,P)` returns 0. Otherwise, it is assumed that the patch is *partially visible* from the viewcell, and `EvaluateConstraint(3,V,P)` returns undefined. The hierarchical structure of the partial octree model is exploited to accelerate the determination of whether a ray intersects any non-empty-space voxels [Foley1992].

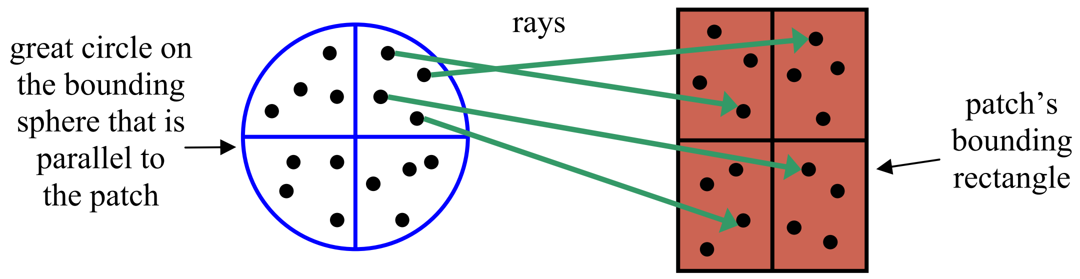

great circle on the bounding sphere that is parallel to the patch
rays
patch's bounding rectangle

Figure 4.11: Generating random rays from the viewcell's bounding sphere to the patch's bounding rectangle to estimate total occlusion.

There is no ill-effect when total occlusion is erroneously declared as partial visibility, except that it may cause an unnecessary subdivision of the viewcell or the patch. On the other hand, it may be undesirable when partial visibility is erroneously declared as total occlusion, since the patch will be disregarded even though it may increase the score of the viewcell. If the occluders between the viewcell and the patch are few and small, then this method is less likely to make a mistake. The most difficult case is when the viewcell and the patch are almost, but not totally, occluded. An example is when the viewcell and the patch are on opposite sides of a large wall that has a very small window. Most likely, the method will declare total occlusion for this case. However, since the method is probabilistic, the chance that the patch will not be "missed" by the same viewcell is relatively higher if many model acquisition cycles are considered instead of just one.

4.1.3.5 New Scan Sampling Densities

The function `EvaluateSamplingDensity(V,P,&sMin,&sMax)` outputs the minimum and the maximum new scan sampling densities between *V* and *P*. First, I define the new scan sampling density between a view position *v* and a surface point *p*. Let α be the one-dimensional angle interval between two successive samples acquired by the range scanner, and *r* be the distance from the view position *v* to the surface point *p*. Figure 4.12(a) shows a case where the angle of incidence from the view position to the surface point is 0. In this case, $\tan(\alpha/2) = l_1/2r$, where l_1 is the distance between two consecutive samples around *p*. Since α is very small, $\tan(\alpha/2) \approx \alpha/2$. Then, $\alpha/2 \approx l_1/2r \Rightarrow l_1 \approx \alpha r$. The *one-dimensional sampling density* around point *p* is $d = 1/l_1 \approx 1/\alpha r$.

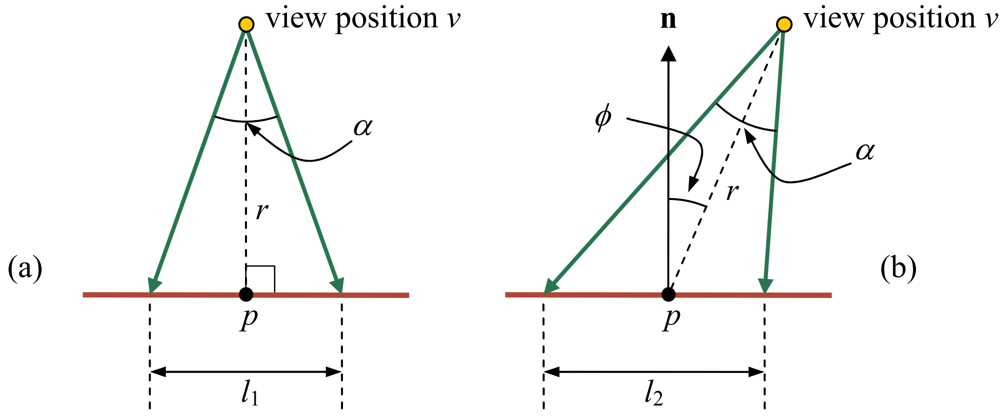


Figure 4.12: α is the one-dimensional angle interval between two successive samples acquired by the range scanner, and r is the distance from the view position v to the surface point p. In (b), ϕ is the angle of incidence from v to p.

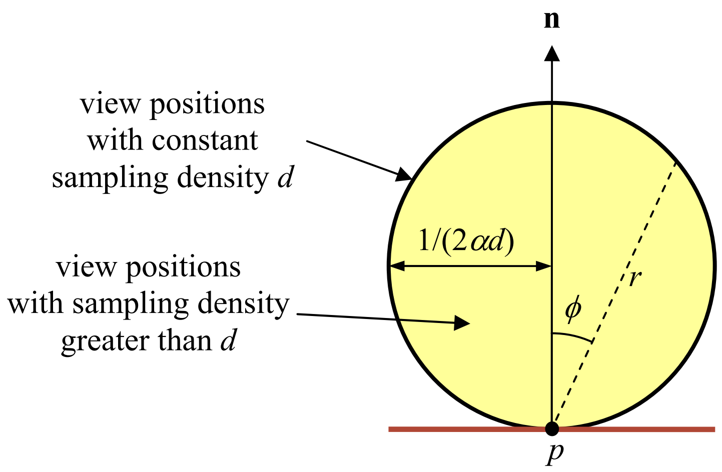
In Figure 4.12(b), the angle of incidence is ϕ, and l_2 can be approximated as $l_2 \approx l_1/\cos\phi \approx \alpha r/\cos\phi$. Therefore, in general, the one-dimensional sampling density around point p is

$$d = \frac{1}{l_2} \approx \frac{\cos\phi}{\alpha r}. \tag{4.7}$$

The 2D new scan surface sampling density at p is approximated by

$$s(v, p) = d^2 \approx \left(\frac{\cos\phi}{\alpha r}\right)^2. \tag{4.8}$$

When the values of α and d are fixed, the locus of the view position is the surface of a sphere with radius $1/(2\alpha d)$, and the sphere touches p as shown in Figure 4.13. All points inside the sphere have one-dimensional sampling densities greater than d, and points outside have one-dimensional sampling densities less than d. This sphere is referred to as the *sampling density sphere* of p.



n
view positions with constant sampling density d
$1/(2\alpha d)$
view positions with sampling density greater than d
ϕ
r
p

Figure 4.13: The sampling density sphere of p. The sphere has radius $1/(2\alpha d)$, and its surface is the set of view positions where the one-dimensional sampling density at p is always d. The set of view positions inside the sphere has one-dimensional sampling densities greater than d.

To compute the minimum sampling density from the viewcell V to the patch P, for each patch point, the smallest sampling density sphere is constructed to entirely enclose the viewcell. Let S be the largest of these smallest sampling density spheres, and let its radius be R. The one-dimensional sampling density represented by S is therefore $1/(2\alpha R)$, and the minimum 2D sampling density from the viewcell V to the patch P is $1/(2\alpha R)^2$.

Since the function `EvaluateSamplingDensity()` is allowed to under-estimate the minimum sampling density, it is sufficient to construct, for each corner of the patch's bounding rectangle, the smallest sampling density sphere that encloses the viewcell, and let S be the largest of the four spheres. The minimum 2D sampling density from the viewcell V to the patch P is estimated as $1/(2\alpha R)^2$, where R is the radius of S. The viewcell may be approximated by a bounding sphere to reduce computation.

Using the same idea, to compute the maximum sampling density from the viewcell V to the patch P, for each patch point, the largest sampling density sphere is constructed such that it touches a point of the viewcell but does not enclose the viewcell. Let S be the smallest of these largest sampling density spheres, and let its radius be R. The one-dimensional sampling density represented by S is therefore $1/(2\alpha R)$, and the maximum 2D sampling density from the viewcell V to the patch P is $1/(2\alpha R)^2$.

The function `EvaluateSamplingDensity()` is allowed to over-estimate the maximum sampling density. Let p be the point on the patch's bounding rectangle that is closest to the viewcell's bounding sphere. Then, let S be the largest sampling density sphere of p that touches a point of the viewcell's bounding sphere but does not enclose the bounding sphere. The maximum 2D sampling density from the viewcell V to the patch P is estimated as $1/(2\alpha R)^2$, where R is the radius of S.

4.1.4 View Sensitivity to Pose Errors

When the scanner is being positioned at a planned view, there may be pose error. Some views are very sensitive to pose errors, in that the view scores in the neighborhood of such a view varies greatly. The problem with using a sensitive view as the planned view is that many surfaces expected to be acquired at the planned pose may not be acquired at the actual pose.

Pose error sensitivity can be easily incorporated into the hierarchical view evaluation algorithm. When evaluating a constraint or the new scan sampling density between a viewcell and a patch, the viewcell is first enlarged by the expected or maximum pose error in each of its dimensions. The effect of this is a lower viewcell score that accounts for only the surface points that can be acquired by every view within the pose error bound of the views in the original viewcell. A surface point that can be acquired by some but not all views in the enlarged viewcell will not be included in the score.

4.2 Results

This section presents some example results of the hierarchical view evaluation algorithm described in the previous section. The results are compared to those of the straightforward view evaluation method. In a straightforward method, when the feasible view volumes are evaluated with a patch, all the feasible viewcells and the patch are first subdivided to their highest resolutions, and then every smallest viewcell is evaluated with every smallest patch element. Table 4.1 shows some of the parameter values used in the examples. The results were obtained on a laptop computer with an Intel Pentium 4-M 1.6GHz CPU, and 768 MB of DDR RAM.

Parameter	**Value**
Scanner angular sampling density, $1/\alpha$	5 samples/degree
Surface sampling density requirement, D	20 samples/inch^2
Scanner maximum range, $R_{\max}$	600 inches = 50 feet
Angle-of-incidence threshold, $\phi_{\max}$	70°
Surface voxel width	2 inches
Smallest viewcell width	4 inches
Smallest patch element length	≤ 4 inches

Table 4.1: Some parameter values used in the experiment.

The scene is a synthetic midsize living room, represented as a polygonal model. Figure 4.14(a) shows the triangle mesh created from the first range scan. The image shows the scanner position where the scan is made and shows all the false surfaces. The ceiling of the room is not shown. Figure 4.14(b) shows the feasible view volumes and the true surface voxels in the partial scene model. The feasible view volumes are made up of viewcells of different sizes. There are 8,481 initial viewcells, and the total volume is equivalent to 28,802 smallest viewcells (each is 4 inches wide).

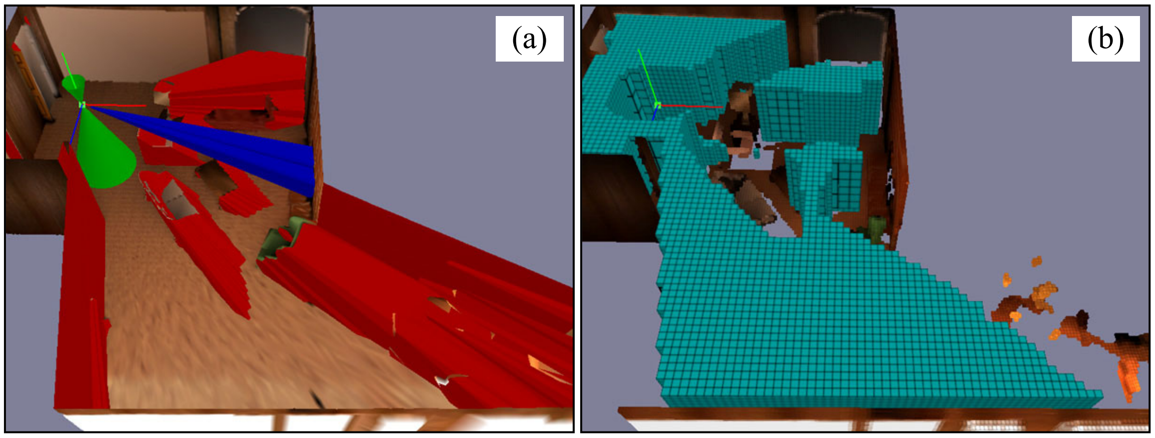


Figure 4.14: (a) The triangle mesh created from the first range scan of a living room. The scanner position is shown at the origin of the axes. Occlusion surfaces are shown in red, hole-boundary surfaces in blue, and image-boundary surfaces in green. The ceiling of the room is not shown. (b) The octree partial scene model is shown with the feasible view volumes (cyan). The feasible view volumes are made up of viewcells of different sizes. There are 8,481 initial viewcells, and the total volume is equivalent to 28,802 smallest viewcells (each is 4 inches wide).

The under-sampled surface voxels in the octree scene model are then grouped into 971 planar patches, each no larger than 10 feet × 10 feet. One of the occlusion surface patches is shown in Figure 4.15(a) in magenta color. The patch has 1020 smallest elements. This patch is then evaluated with the feasible view volumes. Figure 4.15(a) and (b) show the result when the straightforward and the hierarchical view evaluation methods are used, respectively. The best 500 candidate viewcells are shown. The three axes originate from the highest-score viewcell. The results appear to be almost identical, but the computation time of the straightforward method is more than 20 times that of the hierarchical method. In the hierarchical method, the user-specified relative representative error bound of $s(V,P)$ is $\varepsilon_s = 25\%$.

Figure 4.16 shows the accumulated results when a second patch is evaluated with the feasible view volumes. This patch has 1023 smallest elements. Again, the best 500 candidate viewcells are shown. The results appear identical, but the difference in the computation times is large.

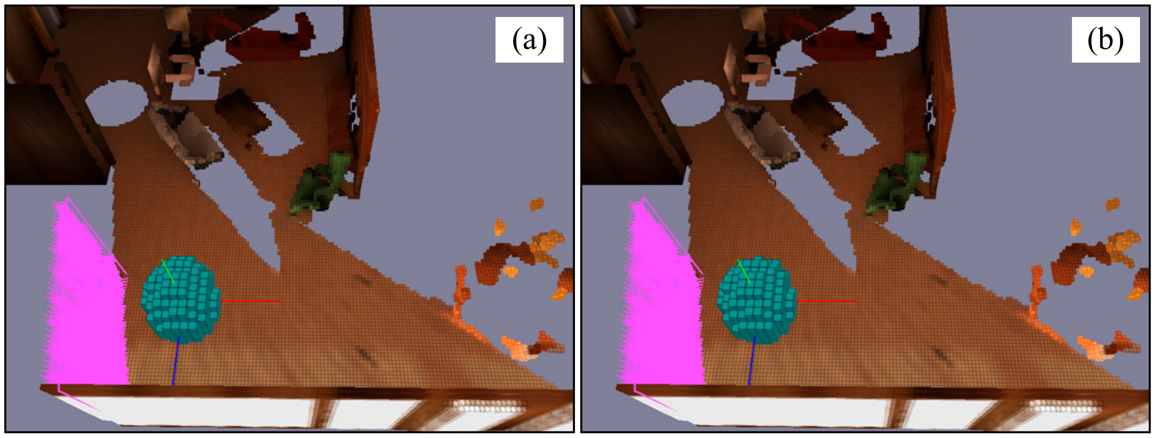


Figure 4.15: The results of evaluating the patch with the feasible view volumes. The best 500 candidate viewcells are shown. The three axes originate from the highest-score viewcell.

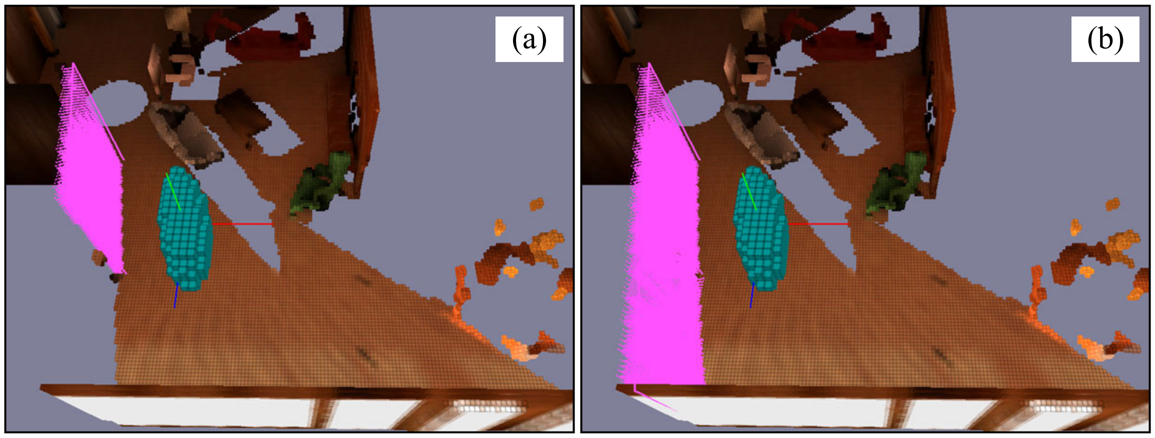


Figure 4.16: The accumulated results of evaluating another patch with the feasible view volumes. The best 500 candidate viewcells are shown.

In the next experiment, the hierarchical view evaluation method is run with different relative representative error bounds, ε_s, on the new scan sampling density evaluations. The tests are on the patch shown in Figure 4.15. The hierarchical method is run using $\varepsilon_s = 50\%$, $\varepsilon_s = 25\%$,and $\varepsilon_s = 5\%$. The resulting viewcells' scores are compared to those computed by the straightforward method, and the relative errors in their scores are shown in Figure 4.17.

In the graphs, there appears to be some anomaly in the score relative errors for $\varepsilon_s = 25\%$, and $\varepsilon_s = 5\%$, where some relative errors are not bounded by ε_s. Actually, this is not unexpected because ε_s is only a bound on the relative error of $s(V,P)$, not on $\max(0,\ \min(s(V,P),\ D) - q(P))$. The main source of the large score relative errors is the "cut-off" effect of the min function. Despite this anomaly, the majority of the score relative errors are within their respective bounds.

Table 4.2 shows the computation times when the patch shown in Figure 4.15 is evaluated with different values of ε_s. The higher the value of ε_s, the shorter the computation time. This demonstrates the ability of the hierarchical method to trade-off between computational accuracy and speed.

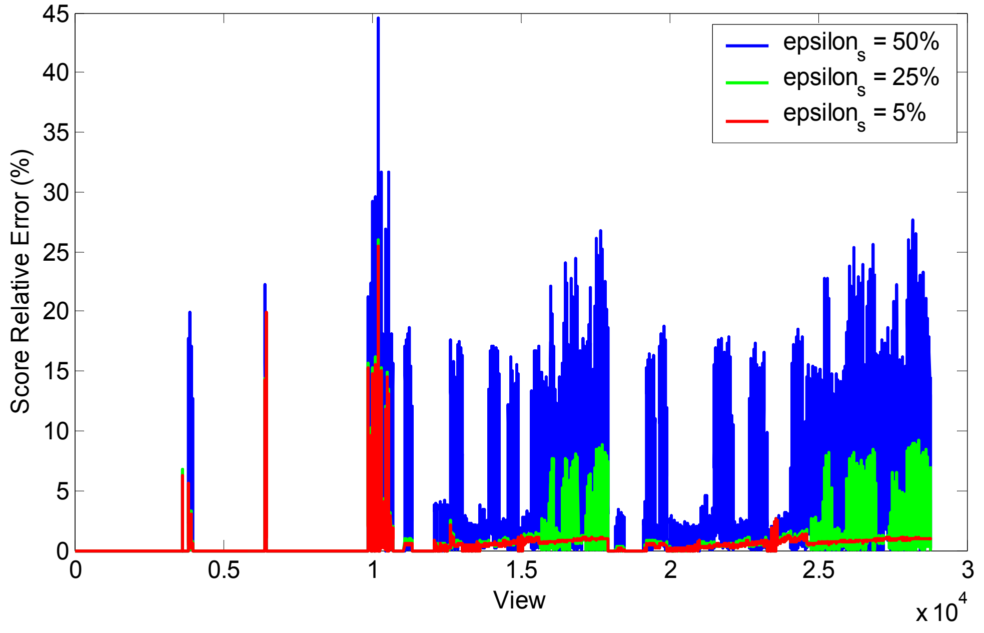


Figure 4.17: The relative errors of the viewcells' scores when they are evaluated with relative representative error bounds of $\varepsilon_s = 50\%$, $\varepsilon_s = 25\%$ and $\varepsilon_s = 5\%$ on $s(V,P)$.

	$\varepsilon_s = 5\%$	$\varepsilon_s = 25\%$	$\varepsilon_s = 50\%$
Computation time	14.8 sec	11.9 sec	10.4 sec
Number of sampling density evaluations	18,260,821	7,562,709	2,765,184

Table 4.2: The effects of using different values of ε_s to bound the relative representative error of $s(V,P)$.

4.3 Generalization

The hierarchical approach for view evaluation is not only applicable to views of 3D translational poses. With the same approach as for the case of 3D translational pose, it can be generalized to views with general 6D poses. In practice, a range scanner is seldom rolled or rotated about the axis that goes through the viewpoint and the center of the field of view. Therefore, the most general pose in practice is 5D: a 3D translation of the viewpoint, a horizontal pan and a vertical tilt.

In this section, I first present how the hierarchical approach can be applied to two example view planning problems. In the first problem, the views are 2D poses located on a sphere, and in the second, the views are 3D poses, each consisting of a 2D horizontal translation and a rotation about the vertical axis. Next, the view metric and the corresponding view evaluation algorithm are generalized for views with general poses. The generalization provides a framework to formulate and implement practical solutions for greedy next-best-view planning. I also describe how the hierarchical approach can be extended to bistatic range sensors and discuss the issues and possible difficulties that may arise.

4.3.1 Example View Planning Problems

4.3.1.1 2D Pose on a Sphere

In the previous next-best-view planning algorithms for range acquisition of objects, many researchers have greatly simplified their problems by assuming that (1) the scanner is monostatic, (2) the scanner's viewpoint is always located on a sphere (called the view sphere or viewing sphere), and (3) the object to be scanned is placed approximately at the center of the sphere and the scanner is always directed towards the center [Connolly1985, García1998, Banta1995, Massios1998, Reed1997]. Figure 4.18 shows such a setup. By fixing the roll of the scanner and other scanning parameters, each view of the scanner is basically a 2D pose. In the previous work, to find the best view for the next scan, the sphere is uniformly discretized into discrete views, which are then exhaustively evaluated according to some view metric. The most popular sphere discretization is to recursively subdivide the faces of a regular icosahedron. A regular icosahedron is a polyhedron made up of 20 faces that are equilateral triangles. Each level of subdivision of the triangles replaces each triangle with four smaller ones.

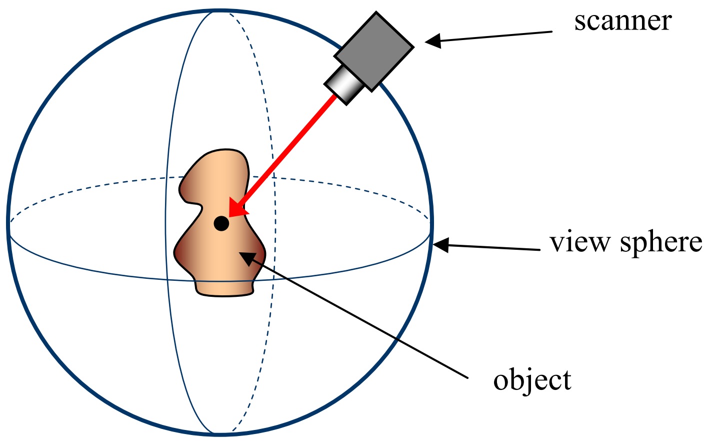


Figure 4.18: The commonly-assumed setup to scan an object. The viewpoint of the scanner is assumed to be on a view sphere and the scanner's field of view is always centered at the center of the sphere.

This 2D view planning problem is a good candidate for applying the hierarchical approach. Each triangle on the regular icosahedron represents a viewcell. Each viewcell is actually not the triangle, but the "triangular" portion of the sphere (I refer to it as the "curved triangle") approximated by the triangle. The subdivision of the triangle provides a means to subdivide the curved triangle. There may be many sensing constraints that need to be tested for each view, and here I describe only the visibility constraint evaluation and the sampling density estimation.

To determine total visibility between a viewcell and a patch, a bounding volume is constructed to enclose the shaft between the patch's bounding rectangle and the viewcell. If the object being scanned is much smaller than the view sphere, then the patch will be small and very close to the view sphere's center, and the triangle can be used as an approximation of the corresponding curved triangle when constructing the shaft. The rest of the procedure to test for total visibility and total occlusion is similar to that described in Section 4.1.3.4.

To determine the minimum and maximum new scan sampling densities between a viewcell and a patch, one can use methods similar to those in Section 4.1.3.5. For minimum sampling density, for each corner of the patch's bounding rectangle, the smallest sampling density sphere is set up to enclose the triangle. Only the three vertices of the triangle need to be tested to find each smallest sampling density sphere. The largest of the four sampling density spheres represents the minimum sampling density. To determine the maximum sampling density, one needs to find the smallest sampling density sphere that touches the triangle. This will be an over-estimate of the actual maximum sampling density when the curved triangle is used instead, and therefore is acceptable.

4.3.1.2 2D Translation and 1D Rotation

In this problem instance, the scanner has fixed and limited vertical and horizontal fields of view. The scanner's viewpoint is fixed at a specified height above the floor, and therefore the y-coordinate of the scanner's position is fixed. The scanner can be oriented by rotating about the vertical axis. The pose of each view is 3D. The first two dimensions are the (x, z) coordinates of the scanner's viewpoint, and the third dimension represents ω, the horizontal direction in which the scanner is pointed.

Before view evaluations, the positioning constraints are tested for each view to determine if it is feasible. Whether a view is feasible depends only on its (x, z) coordinates, not its horizontal direction ω. The result is a set of feasible view volumes.

Each viewcell is a 3D cuboid in its parameter space. When evaluating a viewcell with a patch, not all sensing constraints are affected by all three parameters of the views. For example, the maximum-range constraint, the vertical-field-of-view constraint, the angle-of-incidence constraint, and the visibility constraint are affected only by the (x, z) coordinates, and not by the viewing direction ω. Even the new scan sampling densities are affected by (x, z) but not ω. In this example, only the horizontal-field-of-view constraint is affected by all (x, z) and ω.

Besides the horizontal-field-of-view constraint, all the other sensing constraints and the new scan sampling densities between a viewcell and a patch can be evaluated similarly as in Section 4.1.3. For the horizontal-field-of-view constraint, the function `EvaluateConstraint()` returns 1 when the patch is entirely inside all horizontal fields of view represented in the viewcell, and returns 0 when the patch is entirely outside every horizontal field of view in the viewcell. Otherwise, `EvaluateConstraint()` returns undefined.

It can be seen that it is not necessary to subdivide in all three dimensions when a viewcell is subdivided. For example, if the visibility constraint is the only one causing the viewcell subdivision, it is more efficient to subdivide only in the x and z dimensions. On the other hand, if the horizontal-field-of-view constraint is the only one causing the viewcell subdivision, one may choose to subdivide only in the ω dimension or to subdivide in all three dimensions. Furthermore, ω is of a different entity type from x and z, and the desired subdivision resolution for it may be different from those of x and z. Therefore, it may be possible that the viewcell cannot be subdivided in all three dimensions because ω has reached its highest resolution but x and z have not.

For the above reasons, the data structure of the viewcell hierarchy must be improved to support the subdivision in only a subset of dimensions. A possible solution is to let each non-leaf viewcell to have one, two, or all three of the following groups of children:

(7) **Group 1**: 4 children; produced by subdivision in the x and z dimensions.

(8) **Group 2**: 2 children; produced by subdivision in the ω dimension.

(9) **Group 3**: 8 children; produced by subdivision in the x, z and ω dimensions.

During view evaluation, each of the three groups is independently evaluated. However, after all the evaluations, the scores in all three subtrees must be combined, and propagated down from the highest level to the leaves. Although this approach can be more efficient for viewcell evaluation, the memory requirement is higher.

Heuristics are also needed to decide whether to subdivide the viewcell or the patch. For example, when `EvaluateConstraint()` returns undefined for the horizontal-field-of-view constraint, and if the patch can fit entirely inside some horizontal field of view in the set represented by the viewcell, then the viewcell may be chosen to be subdivided, otherwise the patch is chosen.

4.3.2 Generalized Hierarchical View Evaluation

The hierarchical view evaluation can be applied to the following generalized integral part of the view metric function:

$$g(v) = \int_{p \in S} c_0(v,p) \cdots c_{K-1}(v,p) \cdot f_0(v,p) \cdots f_{M-1}(v,p) \; dp \tag{4.9}$$

where each $c_i(v,p)$ is a binary function indicating whether a sensing constraint is satisfied, and each $f_i(v,p)$ is a continuous-value function. Of course, $c_0(v,p)$ to $c_{K-1}(v,p)$ can be combined into one binary function $c(v,p)$. However, as explained in Section 4.1.2, evaluating the constraints separately has the benefit that when one individual constraint is already satisfied for a viewcell *V* and *P*, it does not need to be re-evaluated when *V* or *P* is subdivided. The same reasoning applies to $f_0(v,p)$ to $f_{M-1}(v,p)$. The function $w(p)$ that appears in the original view metric function in Equation (3.1) can be absorbed into one of the $f_i(v,p)$.

Similar to Equation (4.3), $g(V,P)$ is defined as

$$g(V,P) = c_0(V,P) \cdots c_{K-1}(V,P) \cdot f_0(V,P) \cdots f_{M-1}(V,P) \cdot a(P) \tag{4.10}$$

where $c_i(V,P) = 1$ if $c_i(v,p) = 1$ for all $v \in V$ and $p \in P$, or $c_i(V,P) = 0$ if $c_i(v,p) = 0$ for all $v \in V$ and $p \in P$, otherwise $c_i(V,P)$ is undefined; and each $f_i(V,P)$ is a representative of all the values $f_i(v,p)$ between *V* and *P*. Let $f_{i,\min}(V,P)$ and $f_{i,\max}(V,P)$ be the minimum and maximum $f_i(v,p)$ between *V* and *P*, then $f_i(V,P)$ is a value ranging from $f_{i,\min}(V,P)$

to $f_{i,\max}(V,P)$. To stop the subdivision of V or P, one has to make sure that the relative representative error of $f_i(V,P)$ must be bounded by ε_i, that is

$$\frac{f_{i,\max}(V,P)-f_i(V,P)}{f_{i,\max}(V,P)} \le \varepsilon_i \quad \text{and} \quad \frac{f_i(V,P)-f_{i,\min}(V,P)}{f_i(V,P)} \le \varepsilon_i . \tag{4.11}$$

The generalized view evaluation algorithm is shown in Figure 4.19. It is assumed that the input viewcell V is feasible. Each input Boolean element `c_in[i]` is `true` if $c_i(V,P)$ is already known to be 1, otherwise `c_in[i]` is `false` if $c_i(V,P)$ is unknown. The input Boolean argument `f_uniform_in[i]` is `true` if the relative representative error of $f_i(V,P)$ is known to be bounded by ε_i, otherwise `f_uniform_in[i]` is `false`. The input argument `f_in[i]` is $f_i(V,P)$ if `f_uniform_in[i]` is `true`. Initially, the procedure `EvaluateView()` is called with all `c_in[i]=false` and `f_uniform_in[i]=false`.

```
EvaluateView( Viewcell *V, Patch *P,
              bool c_in[K], bool f_uniform_in[M], float f_in[M] )
{
   bool c[K], f_uniform[M];
   float f[M];
   CopyArray( c, c_in, K );
   CopyArray( f_uniform, f_uniform_in, M );
   CopyArray( f, f_in, M );

   bool all_c = true;

   for ( int i = 0; i < K; i++ )
   {
      if ( !c[i] )
      {
         int t = EvaluateConstraint( i, V, P );
         if ( t == 0 ) return;
         if ( t == 1 ) c[i] = true;
      }
      all_c = ( all_c && c[i] );
   }

   bool all_f_uniform = true;
   float all_f = 1.0;

   for ( int i = 0; i < M; i++ )
   {
      if ( !f_uniform[i] )
      {
         float fMin, fMax;
         EvaluateFunction( i, V, P, &fMin, &fMax );
         f[i] = Representative( fMin, fMax );
         if ( RelativeError( f[i], fMin, fMax ) <= epsilon[i] )
            f_uniform[i] = true;
      }
      all_f_uniform = ( all_f_uniform && f_uniform[i] );
      all_f = all_f * f[i];
   }

   if ( all_c && all_f_uniform )
   {
      V->score += all_f * a(P);
   }
   else if ( ToSubdividePatchFirst( V, P, c, f_uniform, f ) )
   {
      SubdividePatch(P);
      for ( int i = 0; i < P->numChildren; i++ )
         EvaluateView( V, P->child[i], c, f_uniform, f );
   }
   else
   {
      SubdivideViewcell( V, c, f_uniform, f );
      for ( int i = 0; i < V->numChildren; i++ )
         EvaluateView( V->child[i], P, c, f_uniform, f );
   }
}
```

Figure 4.19: A generalized procedure `EvaluateView()` to evaluate $g(V,P)$.

The function `EvaluateFunction(i,V,P,&fMin,&fMax)` returns estimates of $f_{i,\min}(V,P)$ and $f_{i,\max}(V,P)$ in `fMin` and `fMax`, respectively. The function `Representative(fMin,fMax)` returns a value between `fMin` and `fMax`, and `RelativeError(f[i],fMin,fMax)` returns the maximum relative representative error of `f[i]`. Now, the function `ToSubdividePatchFirst()` may need information about which constraints $c_i(V,P)$ have been satisfied and which functions $f_i(V,P)$ have become uniformly-valued to decide to subdivide the viewcell or patch. As mentioned in the last section, it may not be necessary to subdivide the viewcell in all dimensions, so the function `SubdivideViewcell()` may need information about $c_i(V,P)$ and $f_i(V,P)$ to make the decision.

The major difficulty in extending the hierarchical view evaluation algorithm to views with 5D poses may be the memory requirement.

4.3.3 Extension to Bistatic Range Sensors

The next-best-view problem and solution have been described with the assumption that the range sensor is monostatic, and each range image is made from a single viewpoint or center of projection. In this section, I attempt to extend the hierarchical view evaluation method to a bistatic range sensor. In a bistatic range sensor, the light source and the light detector are located at different positions, and this results in two separate "viewpoints". The major difference of a bistatic sensor from a monostatic sensor is that a surface point must be visible to both viewpoints in order to be measured.

An example, similar to that in Section 4.3.1.2, is used to describe the possible solution for the bistatic case. The triangulation-based scanner is assumed to have fixed and limited vertical and horizontal fields of view. The scanner's position is fixed at a specified height above the floor, and therefore the y-coordinate of the scanner's position is fixed. Here, the midpoint between the two viewpoints is arbitrarily chosen as the origin of the local coordinate frame of the scanner, and its (x, z) position in the global coordinate frame is the view position of the scanner. This is shown in Figure 4.20. The positions and orientations of the two viewpoints are fixed relative to the local coordinate frame of the scanner. The scanner can be oriented by rotating about the vertical axis, and its horizontal direction ω is measured from the x-axis. The pose of each view is 3D. The first two dimensions are the (x, z) coordinates of the scanner's view position, and the third dimension represents ω, the horizontal direction in which the scanner is pointed.

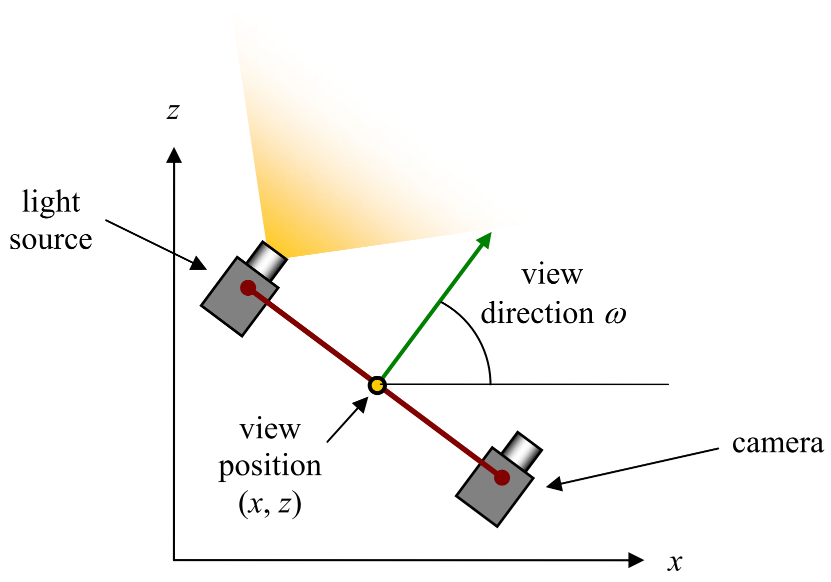


Figure 4.20: Top view of a bistatic range scanner. The scanner consists of a light source and a camera located at different locations. The view position is assumed to be at the midpoint between the centers of projection of the light source and the camera.

In this case, each 3D viewcell represents a square area of (x, z) view positions and the range of horizontal directions. To test the visibility constraint on a viewcell V and a patch P, one has to first find the set of all possible positions each viewpoint can have when the scanner is at the views inside the viewcell. Figure 4.21 shows a viewcell and the swept volumes of the two viewpoints induced by the views in the viewcell. In the example, each swept volume is a bounded area on a plane parallel to the x-z plane.

Each swept volume is tested for visibility with the patch. If any swept volume is totally occluded from the patch, then the viewcell is declared totally occluded from the patch and can be skipped over for this patch. If both swept volumes are totally visible from the patch, then the viewcell is declared totally visible from the patch and no further visibility test is needed between the children of the viewcell and those of the patch. If any swept volume is partially visible from the patch, then the viewcell or the patch will have to be subdivided. The viewcell has to keep track of which swept volume is totally visible and which is partially visible, so that the totally visible one will not be tested again. It may not be necessary to compute the exact swept volume for each viewpoint. A simple bounding rectangle can be used as an approximation.

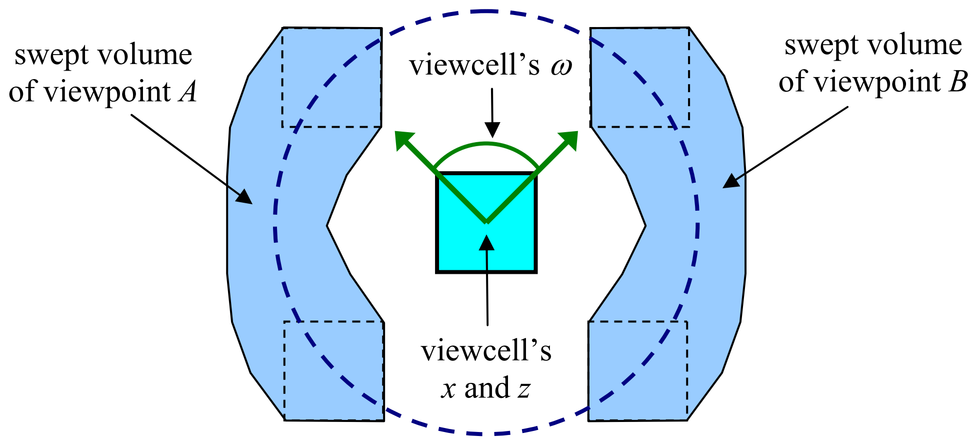


Figure 4.21: The swept volumes of the two viewpoints represented by the given viewcell.

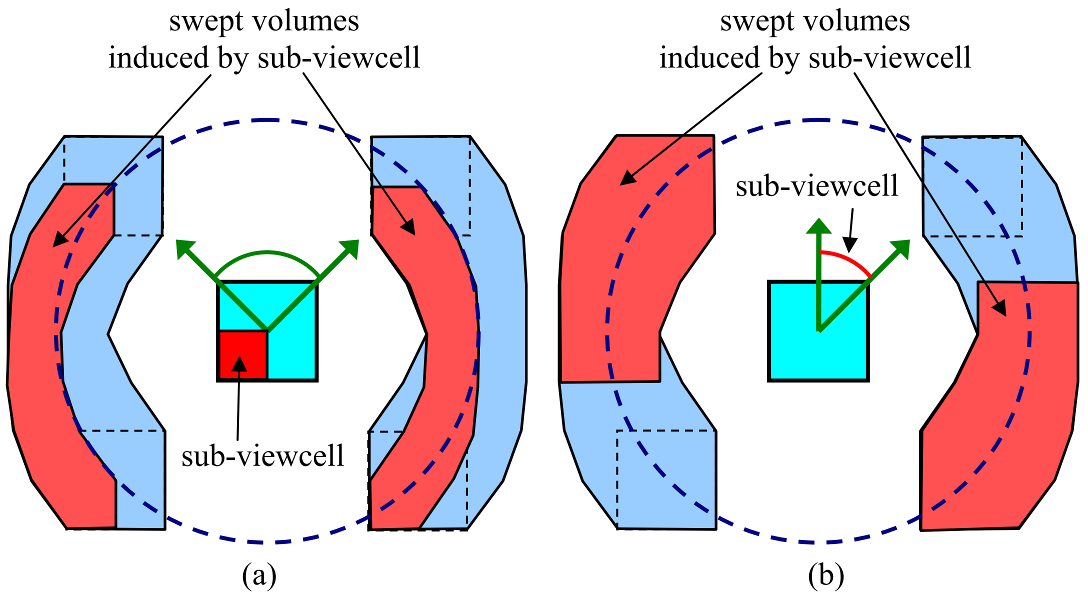


Figure 4.22: The swept volumes of the two viewpoints induced by (a) a sub-viewcell resulted from the subdivision in the x and z dimensions, (b) a sub-viewcell resulted from the subdivision in the ω dimension.

The swept volumes induced by the sub-viewcells are subsets of their parent's swept volumes. This is illustrated in Figure 4.22. It is a necessary property for the hierarchical view evaluation to work. In Figure 4.22(a), the original viewcell is subdivided in the x and z dimensions, and the red swept volumes are induced by one of the four sub-viewcells. In Figure 4.22(b), the subdivision is in the ω dimension, and the red swept volumes are induced by one of the two sub-viewcells. However, it is easy to observe from Figure 4.22 that the swept volumes of different sub-viewcells are actually overlapping each other. This is an undesirable property, and although the hierarchical view evaluation can still work correctly, it can greatly reduce the performance of the algorithm because the overlapped parts of the swept volumes will be tested multiple times.

Another inefficiency can be observed from Figure 4.21 and Figure 4.22. Some of the swept volumes are very elongated and even non-convex, and when such a swept volume is approximated with a simple shape, such as a bounding rectangle, the rectangle may be much larger than the actual swept volume. A heuristic to reduce the problem is to subdivide in the ω dimension when the swept volumes are too elongated. Another undesirable property is that the swept volumes of a viewcell can overlap with the swept volumes of another disjoint viewcell, thus incurring more duplicated computation.

In the example problem given in Section 4.3.1.2, many sensing constraints are affected by only the x and z dimensions of the viewcell, and not by the horizontal pan direction. In the bistatic case, this is not true because the range of horizontal pan angles of the viewcell affects the size and shape of the swept volumes. It is between the swept volumes and the patch that the constraints are actually evaluated. Therefore, partial satisfaction of one of these sensing constraints may cause the viewcell to subdivide in the ω dimension too.

The above inefficiencies with the swept volumes also appear in cases where the views are 4D and 5D poses. It is interesting to note that, with the bistatic range scanner model in Figure 4.20, the vertical tilt of the scanner does not have any effect on the swept volumes of the viewpoints. This is because the two viewpoints always stay in the same horizontal plane regardless of the tilt. However, the vertical-field-of-view constraint must still be evaluated by considering the range of vertical tilt angles represented in the viewcell, and the viewcell may need to be subdivided in the dimension of the vertical tilt angle.

In terms of computational correctness, the hierarchical view evaluation approach can be applied to bistatic range scanners. However, it is not clear how the above undesirable properties may affect the performance improvement that can be gained from the hierarchical view evaluation approach.

4.4 Discussion and Related Work

Exhaustive evaluation of all feasible views. The fact that the all the feasible views are exhaustively evaluated during the computation of every next best view implies that every surface point in the environment that is accessible to at least a feasible view will eventually be acquired. It also guarantees that in every scan, new information is expected to be acquired. This is also true when only a limited amount of time is allowed for the computation of each next best view, and only a subset of patches are evaluated. In randomized next-best-view algorithms [González-Baños1999, Nüchter2003], random trial views are generated, evaluated, and the best is chosen as the next best view. In contrast to my approach, if time is limited or the number of trial views is limited, the randomized methods cannot guarantee to find a view that is expected to acquire new information. This is also a problem in [Sanchiz1999], where a numerical optimization approach is used to search the neighborhood of the current view. If the objective function is "flat" in the neighborhood of the current view, then the method may not find a new view that is expected to add new information.

Consideration of pose errors in solution. Section 4.1.4 describes how the hierarchical approach can be extended to take into account the effects on the views' scores caused by the potential scanner's physical pose errors. To my knowledge, this is the first next-best-view algorithm that directly incorporates scanner's pose errors in the computation of new views. The work by Tarbox and Gottschlich [Tarbox1995] seems to be the only other one that has attempted to address the pose error issue, but in the context of a model-based view planning problem. Their method is very inefficient. The object's surfaces and the view space are first discretized. For each surface element, the set of views that can measure the surface element is computed. For each set of views, morphological erosion is applied to remove views that are near the set boundaries. These removed views are considered not robust to pose variations.

Trade-off between speed and accuracy. A nice property of the hierarchical view evaluation algorithm is that it provides a mechanism to trade-off between computation speed and accuracy. This can be done by adjusting the relative representative error bound, ε_i, of each $f_i(V,P)$ in the generalized integral part of the view metric function in Equation (4.9). A larger value of ε_i favors accuracy over speed. However, the speedup may sometimes be insignificant, because even though the function $f_i(V,P)$ is no longer evaluated as often, the sensing constraints $c_i(V,P)$ and other $f_i(V,P)$ may still continue to cause subdivisions of the viewcells and patches.

Improvement of total occlusion determination. One of the main problems with the current implementation of the hierarchical view evaluation algorithm is the use of the probabilistic method to determine total occlusions. If the viewcell and patch are actually partially visible, but are erroneously declared totally occluded, then evaluation of the viewcell and patch will be prematurely terminated. If the viewcell contains the only feasible views that can see the patch, then the patch will be totally missed by all views.

A possible improvement is to segment the partial scene model into large polygons. These polygons are then used for checking total occlusion. For example, if one of the polygons blocks the entire shaft between the viewcell and the patch, then total occlusion is reported, otherwise the viewcell or patch is subdivided. Of course, this cannot detect all total occlusions. The initial polygons segmented from the partial scene model are usually complex. They may be concave, have holes, and have many vertices. To have efficient total occlusion tests, each polygon must be first simplified or replaced with a simple shape, such as a rectangle or a disc. The simple shape must be totally enclosed by the original polygon. Then, these simplified polygons must be spatially sorted so that the polygons that intersect a shaft can be quickly found. The main challenge of this method is an efficient design and implementation.

This method may be more suitable for indoor environments that have many large flat walls and have multiple rooms or partitions.

Indoor environments. One of the reasons that the hierarchical approach is implemented and tested on indoor environments is that indoor environments usually have large planar surfaces, such as walls, ceilings and floors. These result in large planar patches, which potentially allow better exploitation of spatial coherence by the hierarchical view evaluation algorithm.

4.4.1 Related Work

The main idea of the proposed hierarchical view evaluation algorithm comes from the hierarchical radiosity algorithm [Hanrahan1991]. Hanrahan et al. introduced a hierarchical approach to rapidly compute the form factors between surface patches. Their algorithm can be generalized to evaluate pair-wise interactions between extended objects. The hierarchical approach was later applied by Stuerzlinger to a model-based view planning problem to compute a set of 3D camera positions to maximize the amount of surface area of a synthetic scene model that can be imaged by the cameras [Stuerzlinger1999]. The problem is highly simplified in that only the visibility constraint is considered and each camera is assumed to have a full spherical field of view.

The main difference of this work from Stuerzlinger's is the extension of the hierarchical approach to include many considerations, constraints and requirements to enable practical view planning for real-world range acquisition. Besides for 3D monostatic views, the idea is extended to views with other poses, and to bistatic views. A general view metric has been formulated and is accompanied by a generalized hierarchical view evaluation algorithm to provide a computation framework for greedy next-best-view solutions. The hierarchical algorithm has also been extended to take into consideration the effects of the scanner's pose errors on the views' scores.

Chapter 5

Surface Registration

When the scanner is being positioned at the pose suggested by the view planner, errors in the positioning and pose measurement system may cause the actual pose to be different from the planned pose. As a result of this unknown pose error, the range scan acquired from the new scanner pose will be misaligned with the current partial scene model. In order to correctly merge the new scan into the partial scene model, they must first be registered or aligned with each other. The pose error is usually much worse than the range measurement errors and the sampling densities of the scanner. Therefore, by matching points in the range image with those in the partial scene model, one may be able to compute a good estimate of the correct pose to bring the range scan into alignment with the partial scene model.

The purpose of the alignment is not only for the merging of the scan into the partial scene model. For autonomous range acquisition, where the scanner is mounted on a mobile robot, the registration is able to provide a much more accurate localization of the mobile robot, and greatly reduces the effect of accumulated drift in the robot's motion.

In Sections 5.1 and 5.2, I describe a commonly-used shape registration method, the *iterative closest point* (ICP) algorithm, and show how it is used in my view planning system to align a new scan to the scene model.

However, registration of two surfaces is not guaranteed to be successful every time. Registration failures can occur for several reasons, for example,

(1) when there is insufficient overlap between the two surfaces,

(2) when there is insufficient shape constraint on the 3D rigid-body transformation between the two surfaces,

(3) when the range measurement errors are too large,

(4) when the initial relative pose between the two surfaces is too large, and

(5) when the "frequencies" of the surfaces are too high for the initial relative pose [Low2003].

Therefore, when planning a new view to make the next scan, the view planner must take the above factors into consideration, to ensure that the new range scan acquired from the planned view can be successfully registered with the current scene model to within a certain error tolerance. In Sections 5.3, I present a novel method that allows the view planner to analyze a candidate view for Factors (1), (2) and (3), so that the next scan will have a high probability of being registered successfully to within a specified error bound.

Factors (4) and (5) are not considered here. They should not cause a problem for the application of view planning for indoor environments, since the initial pose errors are usually small in comparison to the size of the scene, and there are many large low-frequency surfaces, such as walls, floor and ceiling. The relationship between the ICP algorithm, the initial relative pose, and the "frequencies" of the surfaces is explored in [Low2003], which also proposes to address the issue of Factor (5) by smoothing the surfaces.

5.1 The Iterative Closest Point (ICP) Algorithm

Since its introduction by Besl and McKay [Besl1992], and by Chen and Medioni [Chen1992], the *Iterative Closest Point* (ICP) algorithm and its many variants have become the most widely-used approaches for aligning three-dimensional surfaces, especially for surfaces created from range images. Because there are many variants that do not use the closest corresponding points when matching points from the two surfaces, it has been suggested that "Iterative Corresponding Point" is a better term to describe the class of algorithms based on the original ICP algorithm [Rusinkiewicz2001]. A comprehensive survey of the many ICP variants can be found in [Rusinkiewicz2001].

For 3D surface registration, the inputs to the ICP algorithm are (1) a set of 3D points on the source surface, (2) a set of 3D points on the target surface, and (3) an initial guess of the 3D rigid-body transformation to transform the source surface to align with the target. In many ICP variants, surface normals at the 3D points are also required. The initial guess of the relative pose can be obtained by many different methods, such as from the scanner positioning and tracking devices, from user input, or from using one of the object recognition methods, such as those that use surface feature identification [Faugeras1986], spin-image-based surface signatures [Johnson1997, Ruiz-Correa2001], the Hough transform [Stockman1982], or principal component analysis of the range images [Skočaj2001]. In the view planning system presented in this dissertation, the planned pose is used as the initial guess because of the fact that the actual pose will be close to the planned pose.

The ICP algorithm iteratively refines the 3D rigid-body transformation by matching points on one surface to points on the other, and minimizes an error metric. This is repeated until a termination condition is reached, such as when the change in the result is small enough or until the number of iterations reaches a specified limit. More specifically, each iteration consists of the following steps:

(1) select a subset of points on one or both surfaces;

(2) match each point in the subset to a point on the other surface to form a point pair;

(3) reject bad point pairs;

(4) minimize an error metric formed by the accepted point pairs to obtain a 3D rigid-body transformation; and

(5) transform all points in the source surface by the 3D rigid-body transformation.

Step (1) is not necessary if all the points are to be used. However, range images usually have a huge number of points, and it is too computationally expensive to use all of them. The most common methods to select the subset of points are uniform subsampling and random sampling of the range image. Points may be selected in only one or in both surfaces.

In Step (2), the original ICP algorithm [Besl1992] matches each selected point with the closest point on the other surface. A *k*-d tree is often used to speed up the closest point search [Friedman1977]. Many other matching methods have been proposed, and some of them are compared in [Rusinkiewicz2001] to study their effects on the convergence rate of the ICP algorithm.

The purpose of Step (3) is to eliminate outliers, which may have a large effect on the minimization result in Step (4) when a least-squares error metric is used. Many strategies have been used to reject point pairs. They include rejecting a point pair if the distance between them is more than a user-specified threshold, rejecting a point pair if their surface normals are too different (the angle between the surface normals is greater than a threshold), and rejecting a point pair if any of the points is on a surface boundary [Turk1994]. When the closest-point matching is used in Step (2), the last rejection strategy is very useful for eliminating erroneous pairings at the boundaries of the surfaces. An example of such cases is shown in Figure 5.1.

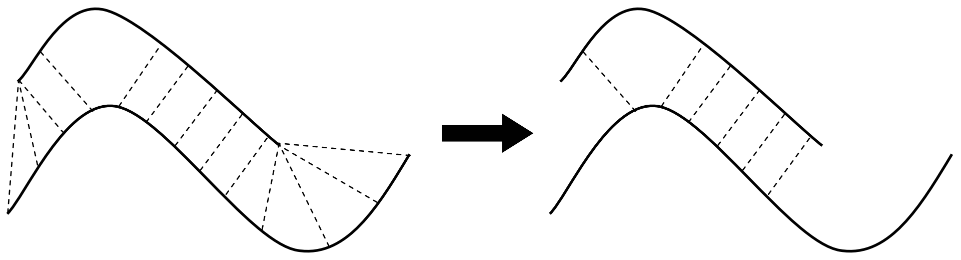


Figure 5.1: Elimination of erroneous pairings at the boundaries of surfaces. Points on the lower surface are matched with the closest points on the upper surface.

In Step (4), an error function is formed using the remaining point pairs, and the goal of the minimization is to find the 3D rigid-body transformation that minimizes the error function. Two error metrics are commonly used. One is known as the *point-to-point metric* and the other the *point-to-plane metric*. Suppose $\mathbf{s}_i = [s_{i,x}, s_{i,y}, s_{i,z}, 1]^{\mathrm{T}}$ are the homogeneous coordinates of a selected point on the source surface, $\mathbf{t}_i = [t_{i,x}, t_{i,y}, t_{i,z}, 1]^{\mathrm{T}}$ are the homogeneous coordinates of its corresponding point on the target surface, $\mathbf{n}_i = [n_{i,x}, n_{i,y}, n_{i,z}, 0]^{\mathrm{T}}$ is the unit normal vector at $\mathbf{t}_i$, and $\mathbf{M}$ is a 4×4 3D rigid-body transformation matrix, then the point-to-point error metric is

$$E_{\text{point-to-point}} = \sum_i \left(\mathbf{M} \cdot \mathbf{s}_i - \mathbf{t}_i\right)^2 \tag{5.1}$$

and the point-to-plane metric is

$$E_{\text{point-to-plane}} = \sum_i \left(\left(\mathbf{M} \cdot \mathbf{s}_i - \mathbf{t}_i\right) \cdot \mathbf{n}_i\right)^2 . \tag{5.2}$$

The goal of the minimization is to find the $\mathbf{M}$ that minimizes either $E_{\text{point-to-point}}$ or $E_{\text{point-to-plane}}$. For the point-to-point metric, the optimal $\mathbf{M}$ can be computed efficiently using the SVD method [Arun1987] or the unit quaternion method [Horn1987]. Unlike the point-to-point metric, which has a closed-form solution, the point-to-plane metric is usually solved using standard nonlinear least-squares methods, such as the Levenberg-Marquardt method [Press1992]. Although the point-to-plane metric takes more time to solve, it has been observed to have significantly better convergence rates than the point-to-point metric [Rusinkiewicz2001, Pulli1999, Low2003]. A theoretical analysis of the convergence of the point-to-plane metric is described in [Pottmann2002].

Fortunately, when the relative orientation (rotation only) between the two input surfaces is small, one can approximate the nonlinear least-squares optimization problem with a linear one, and use linear optimization methods to compute the solution more quickly. This approximation is based on the substitution of sin θ by θ and cos θ by 1 in the rotation matrix [Rusinkiewicz2001]. Details of this approximation are found in [Low2004].

In light of the above, it is possible to achieve more efficient registration by using both metrics in the ICP algorithm. The point-to-point metric is used in the first few iterations of the ICP algorithm until the relative orientation between the two input surfaces is small enough. Then, the "linearized" point-to-plane metric is used in the subsequent iterations. In this way, one may be able to enjoy the efficiency and high convergence rates of the "linearized" point-to-plane metric without subjecting it to large relative orientation when the approximation errors of the "linearization" may be large. This hybrid approach is used in the view planning system described in this dissertation, and has been observed, in almost all cases, to converge significantly faster than using the point-to-point metric alone. The "linearized" point-to-plane metric has been very robust even when the relative orientation between the two input surfaces is fairly large, sometimes as large as 30°.

5.2 Surface Registration in the View Planner

During range acquisition using views planned by the view planner, each new range scan, except the first one, is registered to the partial scene model as soon as it is available. This section describes the implementation of the surface registration used in the view planner.

The inputs to the surface registration are

(1) a triangle mesh constructed from the range image (the source surface),

(2) the set of all true surface voxels in the octree partial scene model (the target surface), and

(3) the current planned pose (the initial guess of the 3D rigid-body transformation).

The triangle mesh is originally constructed with respect to the local coordinate frame of the scanner. Just before the first ICP iteration starts, it is transformed by the current planned pose into the coordinate frame of the partial scene model, and into rough alignment with the surfaces of the partial scene model.

In the first step of an ICP iteration, the center of every true surface voxel is paired with the closest vertex on the triangle mesh. To accelerate the closest-point searches, a *k*-d tree is built for the vertices of the original triangle mesh. Note that the *k*-d tree need not be rebuilt every ICP iteration even though the triangle mesh is repeatedly being transformed. All we need to do is to transform the query point (a voxel center) into the original coordinate frame of the triangle mesh before making the closest-point search in the *k*-d tree.

Next, some of the point pairs are rejected according to the following conditions. A pair is rejected

(1) if the two points are more than a threshold distance apart (typically one to two feet),

(2) if the angle between the surface normal vectors at the two points is larger than a threshold (typically 45°), or

(3) if the vertex is on a boundary of the triangle mesh.

For checking the third condition, each vertex of the triangle mesh has been assigned a flag during mesh creation to indicate whether it is on a mesh boundary. There can be three types of boundaries in a mesh: (1) hole boundaries, (2) depth boundaries, and (3) image boundaries. Hole boundaries are the result of holes or missing range samples in the range image, depth boundaries occur at depth discontinuities between adjacent samples, and image boundaries occur at the boundaries of the range image. Figure 5.2 shows a simple example that includes all three types of boundaries.

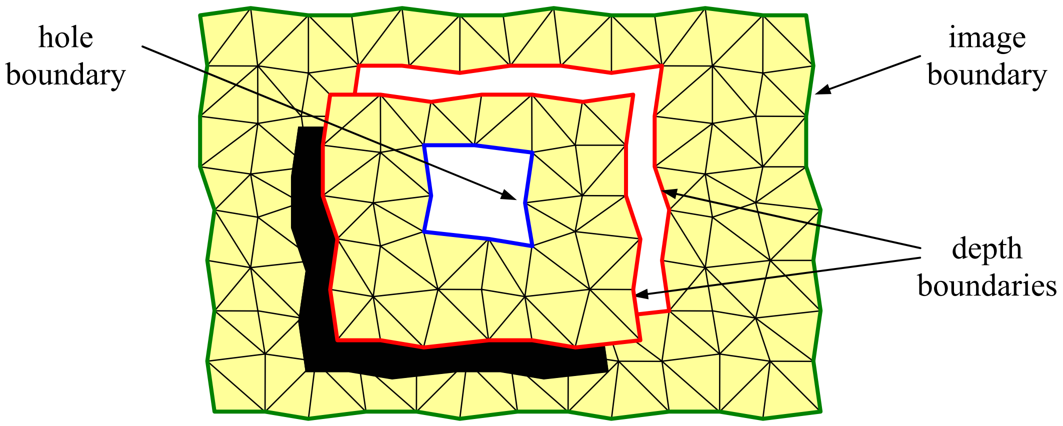


Figure 5.2: There are three types of boundaries in a triangle mesh created from a range image: (1) hole boundaries (blue lines), (2) depth boundaries (red lines), and (3) image boundaries (green lines).

Both the point-to-point and the point-to-plane metrics are used for solving the 3D rigid-body transformation. The point-to-point metric is used until the orientation difference between the results of two successive iterations is less than a threshold. After that, the “linearized” point-to-plane metric is used. This threshold is typically a 10° rotation about each coordinate axis.

The ICP registration stops when the number of iterations reaches a limit or when the change in the results of two successive iterations is small enough. The change is small enough when the translation is less than a distance threshold and the rotation about every coordinate axis is less than an angle threshold.

5.3 Predetermining Registration Accuracy of a View

The view planner must try its best to avoid the situation in which the newly acquired range scan fails to register, or fails to register accurately enough, with the partial scene model. To do that, the view planner must examine each candidate view before the actual scan is made, to help ensure that the planned view will produce a range scan that can be accurately registered with the current partial scene model. Three of the factors[3] that affect registration accuracies are:

(1) the range measurement errors of the scanner,

(2) the amount of overlap between the two surfaces, or the number of point pairs that can be used in the registration, and

(3) the amount of shape constraint on the 3D rigid-body transformation between the two surfaces.

The lack of shape constraint on the 3D rigid-body transformation can result in catastrophic registration failure. For example, when a plane is being registered to another plane (see Figure 5.3), the first plane can "slide" and "spin" on the second plane without being constrained in those motions, and thus cannot be registered successfully.

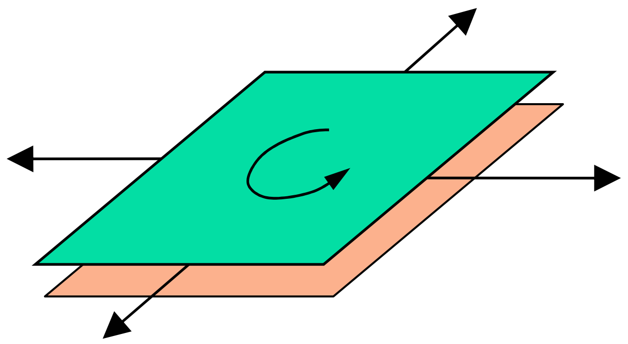


Figure 5.3: Registering two planes can result in catastrophic failure because one plane can "slide" and "spin" on the other.

In this section, I derive a *registration accuracy metric* to estimate registration accuracy, and demonstrate how it is applied to test a candidate view to determine whether it satisfies the registration constraint. The metric consists of two *registration accuracy conditions*, and they take into consideration all three factors listed above.

The derivations of the registration accuracy conditions are built upon the constraint analysis in [Simon1996]. In his doctoral dissertation, Simon presented a means to measure

[3]Registration errors can also be caused by nonlinear distortion of distances in the range measurements. This is the result of calibration error in the range scanner.

the relative amount of constraint on the 3D rigid-body transformation exerted by the shape of a surface. More specifically, his method is able to compute, for a set of surface points, a value that represents the amount of constraint on the 3D rigid-body transformation when the surface is being aligned with itself. Being a relative measurement, this value is only useful for comparing with the values of other point sets, so as to determine which point set has the best constraint on the transformation. It does not indicate the absolute accuracy that can be achieved with each point set during registration. My registration accuracy conditions extend Simon's constraint analysis to estimate absolute registration accuracies.

Before I present my derivations of the registration accuracy conditions, I review Simon's constraint analysis in detail. I adopt the notation used in [Simon1996].

5.3.1 Simon's Constraint Analysis

The basic idea of the constraint analysis is as follows. Let P be a set of points on a surface. Suppose a small 3D rigid-body transformation is applied to P to produce the point set P', in which some of the points may no longer be on the original surface. Let $E_{P'}$ be the sum of the squared distance between each point in P' and the original surface. The constraint analysis tries to quantify the sensitivity of $E_{P'}$ to each component of the 3D rigid-body transformation. The higher the sensitivity is with respect to a certain component, the stronger the constraint on the motion corresponding to the component.

The following describes how $E_{P'}$ is computed. Since, given an arbitrary surface, there is no closed-form analytical expression for the distance between a point $\mathbf{x}$, and the surface, a first-order approximation of the true point-to-surface distance is used:

$$D(\mathbf{x}) = \frac{F(\mathbf{x})}{\|\nabla F(\mathbf{x})\|} \tag{5.3}$$

where $F(\mathbf{x}) = 0$ is the implicit equation of the surface, $\|\nabla F(\mathbf{x})\|$ is the magnitude of the gradient to the surface, $\mathbf{x}$ is a point which may or may not lie on the surface and $D(\mathbf{x})$ is the approximate distance.

Let $\mathbf{x}_s$ be a point that lies on the surface, i.e. $D(\mathbf{x}_s)=0$. This point can be perturbed by applying a differential transformation $\boldsymbol{T}$ to it. $\boldsymbol{T}$ can be represented by a homogeneous transformation which is a function of the 6 parameters (t_x, t_y, t_z, ω_x, ω_y, ω_z), where (ω_x, ω_y, ω_z) are rotations about the *x*, *y* and *z* axes, respectively, and (t_x, t_y, t_z) are the translations along the newly rotated *x*, *y* and *z* axes. The rate of change of *D* with respect to an arbitrary transformation $\boldsymbol{T}$ of the point $\mathbf{x}_s$ is given by

$$V(\mathbf{x}_s)=\frac{\partial}{\partial \mathbf{t}}D(\boldsymbol{T}(\mathbf{x}_s))=\begin{bmatrix}\mathbf{n}_{\mathbf{x}_s}\\ \mathbf{x}_s\times\mathbf{n}_{\mathbf{x}_s}\end{bmatrix} \tag{5.4}$$

where $\mathbf{t}=[t_x,t_y,t_z,\omega_x,\omega_y,\omega_z]^{\mathrm{T}}$ and $\mathbf{n}_{\mathbf{x}_s}$ is the unit normal to the surface evaluated at the point $\mathbf{x}_s$. Equation (5.4) can be written as

$$D(\boldsymbol{T}(\mathbf{x}_s))=\boldsymbol{V}^{\mathrm{T}}(\mathbf{x}_s)\,d\mathbf{t}=\begin{bmatrix}\mathbf{n}_{\mathbf{x}_s}\\ \mathbf{x}_s\times\mathbf{n}_{\mathbf{x}_s}\end{bmatrix}^{\mathrm{T}}d\mathbf{t}\,. \tag{5.5}$$

By squaring Equation (5.5), we get

$$D^2(\boldsymbol{T}(\mathbf{x}_s))=d\mathbf{t}^{\mathrm{T}}\,\boldsymbol{V}(\mathbf{x}_s)\boldsymbol{V}^{\mathrm{T}}(\mathbf{x}_s)\,d\mathbf{t}=d\mathbf{t}^{\mathrm{T}}\,\boldsymbol{M}(\mathbf{x}_s)\,d\mathbf{t} \tag{5.6}$$

where $\boldsymbol{M}(\mathbf{x}_s)=\boldsymbol{V}(\mathbf{x}_s)\boldsymbol{V}^{\mathrm{T}}(\mathbf{x}_s)$ is a symmetric, positive semi-definite 6×6 matrix. By summing the quantity in Equation (5.6) over a set, *P*, of discrete surface points, we get

$$E_P(\boldsymbol{T}(\mathbf{x}_s))=\sum_{\mathbf{x}_s\in P}D^2(\boldsymbol{T}(\mathbf{x}_s))=d\mathbf{t}^{\mathrm{T}}\left(\sum_{\mathbf{x}_s\in P}\boldsymbol{M}(\mathbf{x}_s)\right)d\mathbf{t}=d\mathbf{t}^{\mathrm{T}}\,\boldsymbol{\Psi}\,d\mathbf{t}\,. \tag{5.7}$$

The $E_{P'}$ that was mentioned in the beginning of the section is actually $E_{P'}=E_P(\boldsymbol{T}(\mathbf{x}_s))$, where $P'=\{\boldsymbol{T}(\mathbf{x}_s):\mathbf{x}_s\in P\}$. The matrix $\boldsymbol{\Psi}$ is a scatter matrix that contains information about the distribution of the original $\boldsymbol{V}(\mathbf{x}_s)$ over all points in *P*. Now, by performing principal component analysis [Kendall1977] to $\boldsymbol{\Psi}$, we can factorize it into the form

$$\boldsymbol{\Psi} = \mathbf{Q}\,\boldsymbol{\Lambda}\,\mathbf{Q}^{\mathrm{T}} = \begin{bmatrix}\mathbf{q}_1 & \mathbf{q}_2 & \mathbf{q}_3 & \mathbf{q}_4 & \mathbf{q}_5 & \mathbf{q}_6\end{bmatrix} \begin{bmatrix} \lambda_1 & & & & & \\ & \lambda_2 & & & 0 & \\ & & \lambda_3 & & & \\ & & & \lambda_4 & & \\ & 0 & & & \lambda_5 & \\ & & & & & \lambda_6 \end{bmatrix} \begin{bmatrix} \mathbf{q}_1^{\mathrm{T}} \\ \mathbf{q}_2^{\mathrm{T}} \\ \mathbf{q}_3^{\mathrm{T}} \\ \mathbf{q}_4^{\mathrm{T}} \\ \mathbf{q}_5^{\mathrm{T}} \\ \mathbf{q}_6^{\mathrm{T}} \end{bmatrix} \qquad (5.8)$$

where $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4 \geq \lambda_5 \geq \lambda_6$ are the eigenvalues of $\boldsymbol{\Psi}$, and $\mathbf{q}_i$ are the corresponding unit eigenvectors. Each eigenvector, $\mathbf{q}_i$, represents a differential transformation where the first three elements are the translation components and the last three elements are the rotation components.

Each eigenvalue, λ_i, is proportional to the rate of change of the error, E_P, induced by a transformation in the direction specified by $\mathbf{q}_i$. This implies that $\mathbf{q}_1$, which corresponds to the largest eigenvalue, represents the transformation of maximum constraint. When an eigenvalue, λ_i, is close to or equal to zero, transforming the points in the direction specified by $\mathbf{q}_i$ will not change E_P. This means that the set of points has no constraint on the transformation in the direction specified by $\mathbf{q}_i$. For example, the plane shown in Figure 5.3 will have three zero eigenvalues, one corresponds to the rotation about the vertical axis, and the other two correspond to the translation in the horizontal plane.

Let P_1 and P_2 be two different sets of discrete points on a surface. To determine which point set has a better "overall" constraint on the transformation, the quantity $\lambda_6 / \sqrt{\lambda_1}$ (called the *noise amplification index*) of one point set is compared to that of the other point set. The point set with the higher noise amplification index is considered better.

One problem with the result in Equation (5.8) is that the rotation components are dependent on the scale of the surface being analyzed. This is the consequence of having $\mathbf{x}_s$ in the cross product in Equation (5.5). Simon addressed this problem by shifting the centroid of the point set to the origin and scaling all points so that the average distance from the points to the origin is 1.

Equation (5.8) and the noise amplification index do not provide enough information to estimate the absolute registration errors when the points in the point set have been perturbed by noise. In Section 5.3.2, I present an extension to Simon's constraint analysis to estimate absolute bounds on the registration errors.

5.3.2 Estimating Absolute Registration Errors

In the following derivations, we consider translation and rotation separately. This makes sense because translation and rotation are parameterized by different entity types, i.e. translation is parameterized by distance and rotation by angle. Having both entity types in the same eigenvectors $\mathbf{q}_i$ in Equation (5.8) makes the absolute values of the eigenvalues λ_i hard to interpret. Moreover, by considering translation and rotation separately, as we will see, the arbitrary scale of the object is no longer a problem for the rotation analysis.

5.3.2.1 Translational Alignment Error

Following the derivation in Section 5.3.1, the translation component of $\boldsymbol{V}(\mathbf{x}_s)$ is

$$\boldsymbol{V}_{\boldsymbol{\tau}}(\mathbf{x}_s) = \frac{\partial}{\partial \boldsymbol{\tau}} D(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{x}_s)) = \mathbf{n}_{\mathbf{x}_s} \tag{5.9}$$

where $\boldsymbol{\tau} = [t_x, t_y, t_z]^{\mathrm{T}}$. Then

$$D(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{x}_s)) = \boldsymbol{V}_{\boldsymbol{\tau}}^{\mathrm{T}}(\mathbf{x}_s)\, d\boldsymbol{\tau}\,. \tag{5.10}$$

By squaring Equation (5.10), we get

$$D^2(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{x}_s)) = d\boldsymbol{\tau}^{\mathrm{T}}\, \boldsymbol{V}_{\boldsymbol{\tau}}(\mathbf{x}_s) \boldsymbol{V}_{\boldsymbol{\tau}}^{\mathrm{T}}(\mathbf{x}_s)\, d\boldsymbol{\tau} = d\boldsymbol{\tau}^{\mathrm{T}}\, \boldsymbol{M}_{\boldsymbol{\tau}}(\mathbf{x}_s)\, d\boldsymbol{\tau} \tag{5.11}$$

where $\boldsymbol{M}_{\boldsymbol{\tau}}(\mathbf{x}_s) = \boldsymbol{V}_{\boldsymbol{\tau}}(\mathbf{x}_s) \boldsymbol{V}_{\boldsymbol{\tau}}^{\mathrm{T}}(\mathbf{x}_s)$ is a symmetric, positive semi-definite 3×3 matrix. By summing the quantity in Equation (5.11) over the set, *P*, of discrete surface points, we get

$$E_P(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{x}_s)) = \sum_{\mathbf{x}_s \in P} D^2(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{x}_s)) = d\boldsymbol{\tau}^{\mathrm{T}} \left(\sum_{\mathbf{x}_s \in P} \boldsymbol{M}_{\boldsymbol{\tau}}(\mathbf{x}_s) \right) d\boldsymbol{\tau} = d\boldsymbol{\tau}^{\mathrm{T}}\, \boldsymbol{\Psi}_{\boldsymbol{\tau}}\, d\boldsymbol{\tau}\,. \tag{5.12}$$

Using principal component analysis, $\boldsymbol{\Psi}_{\boldsymbol{\tau}}$ can be factorized into

$$\boldsymbol{\Psi}_{\boldsymbol{\tau}} = \mathbf{Q}_{\boldsymbol{\tau}} \boldsymbol{\Lambda}_{\boldsymbol{\tau}} \mathbf{Q}_{\boldsymbol{\tau}}^{\mathrm{T}} = \mathbf{Q}_{\boldsymbol{\tau}} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathbf{Q}_{\boldsymbol{\tau}}^{\mathrm{T}} \tag{5.13}$$

where $\lambda_1 \geq \lambda_2 \geq \lambda_3$ are the eigenvalues of $\boldsymbol{\Psi}_{\boldsymbol{\tau}}$, and the columns of $\mathbf{Q}_{\boldsymbol{\tau}}$ are the corresponding unit eigenvectors. Since the sum of the eigenvalues is equal to the trace of the original matrix [Strang1988],

$$\begin{aligned} \lambda_1 + \lambda_2 + \lambda_3 &= \mathrm{tr}(\boldsymbol{\Psi}_{\boldsymbol{\tau}}) \\ &= \mathrm{tr}\left(\sum_{\mathbf{x}_s \in P} \boldsymbol{M}_{\boldsymbol{\tau}}(\mathbf{x}_s) \right) = \mathrm{tr}\left(\sum_{\mathbf{x}_s \in P} \boldsymbol{V}_{\boldsymbol{\tau}}(\mathbf{x}_s) \boldsymbol{V}_{\boldsymbol{\tau}}^{\mathrm{T}}(\mathbf{x}_s) \right) \\ &= \sum_{\mathbf{x}_s \in P} \| \boldsymbol{V}_{\boldsymbol{\tau}}(\mathbf{x}_s) \|^2 = \sum_{\mathbf{x}_s \in P} \| \mathbf{n}_{\mathbf{x}_s} \|^2 = \sum_{\mathbf{x}_s \in P} 1 = |P| = M \end{aligned} \tag{5.14}$$

where M is the number of points in P.

For the alignment of the surface to be successful, λ_i must be greater than 0 for $i = 1, 2, \text{and } 3$. Ideally, we wish to select the set of points for P such that $\lambda_1 = \lambda_2 = \lambda_3 = M/3$, which means that the surface alignment is constrained equally in all three orthogonal translation directions. The minimum requirement for alignment is that $M = 3$, and that the three surface normals must span the 3D space. In practice, because of errors in the range measurements, it is necessary to have $M \geq M_{\min} > 3$, such that the translational alignment can be performed to a certain desired accuracy, assuming the surface is already correctly oriented.

The translation resulting from the alignment can be decomposed into three orthogonal directions. In the following, we investigate the relationship between $M_{\min}$ and the translational alignment accuracy by looking at the alignment errors in the three orthogonal directions. Without loss of generality, we choose the *x*, *y*, and *z* directions.

Let $[x_i, y_i, z_i]^{\mathrm{T}}$ be the 3D coordinates of the *i*th true surface point, where $i = 1, 2, \ldots, M$. Suppose there are two sets of measurements of the surface points, producing the point set *A* with coordinates $[x_i + u_{Ai}, y_i + v_{Ai}, z_i + w_{Ai}]^{\mathrm{T}}$ and the point set B with coordinates $[x_i + u_{Bi}, y_i + v_{Bi}, z_i + w_{Bi}]^{\mathrm{T}}$, where ($u_{Ai}$, u_{Bi}), (v_{Ai}, v_{Bi}), and (w_{Ai}, w_{Bi}) are measurement errors in the *x*, *y*, and *z* directions, respectively.

Suppose point set *B* is to be translated so that it is aligned with point set *A*. For the alignment, the correspondences between points in point sets *A* and *B* are known. The alignment uses the least-squares (least-sum-of-squares) error metric, where we want to find the translation vector $\boldsymbol{\tau} = [t_x, t_y, t_z]^{\mathrm{T}}$ to translate point set *B* to minimize

$$\begin{aligned} SSE &= \sum_{i=1}^{M} \left\| \begin{bmatrix} x_i + u_{Ai} \\ y_i + v_{Ai} \\ z_i + w_{Ai} \end{bmatrix} - \begin{bmatrix} x_i + u_{Bi} \\ y_i + v_{Bi} \\ z_i + w_{Bi} \end{bmatrix} - \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \right\|^2 \\ &= \sum_{i=1}^{M} (u_{Ai} - u_{Bi} - t_x)^2 + \sum_{i=1}^{M} (v_{Ai} - v_{Bi} - t_y)^2 + \sum_{i=1}^{M} (w_{Ai} - w_{Bi} - t_z)^2 \end{aligned} \tag{5.15}$$

Differentiating *SSE* with respect to t_x, t_y, and t_z, we have

$$\begin{aligned} \frac{\partial(SSE)}{\partial t_x} &= -2\sum_{i=1}^{M} (u_{Ai} - u_{Bi} - t_x) \\ \frac{\partial(SSE)}{\partial t_y} &= -2\sum_{i=1}^{M} (v_{Ai} - v_{Bi} - t_y) \\ \frac{\partial(SSE)}{\partial t_z} &= -2\sum_{i=1}^{M} (w_{Ai} - w_{Bi} - t_z) \end{aligned} \tag{5.16}$$

SSE is minimum when $\frac{\partial(SSE)}{\partial t_x} = \frac{\partial(SSE)}{\partial t_y} = \frac{\partial(SSE)}{\partial t_z} = 0$, and this is true when

$$\begin{aligned} & -2\sum_{i=1}^{M} (u_{Ai} - u_{Bi} - t_x) = 0 \\ & \Rightarrow M t_x = \sum_{i=1}^{M} (u_{Ai} - u_{Bi}) \\ & \Rightarrow t_x = \frac{1}{M}\sum_{i=1}^{M} u_{Ai} - \frac{1}{M}\sum_{i=1}^{M} u_{Bi} = \overline{u_A} - \overline{u_B} \end{aligned} \tag{5.17}$$

and similarly

$$\begin{aligned} t_y &= \frac{1}{M}\sum_{i=1}^{M} v_{Ai} - \frac{1}{M}\sum_{i=1}^{M} v_{Bi} = \overline{v_A} - \overline{v_B} \\ t_z &= \frac{1}{M}\sum_{i=1}^{M} w_{Ai} - \frac{1}{M}\sum_{i=1}^{M} w_{Bi} = \overline{w_A} - \overline{w_B} \end{aligned} \tag{5.18}$$

We now first consider only the translation in the x direction. Let u_{Ai} and u_{Bi} be the values of the independent random variables U_{Ai} and U_{Bi}, respectively, and $\overline{u_A}$ and $\overline{u_B}$ be the values of the random variables $\overline{U_A}$ and $\overline{U_B}$, respectively. Suppose each U_{Ai} has normal distribution with mean μ_{U_A} and standard deviation σ_{U_A}, and each U_{Bi} has normal distribution with mean μ_{U_B} and standard deviation σ_{U_B}, i.e.

$$U_{Ai} \sim N(\mu_{U_A}, \sigma_{U_A}) \quad \text{and} \quad U_{Bi} \sim N(\mu_{U_B}, \sigma_{U_B}) \tag{5.19}$$

Then, the sampling distributions [Walpole1993] of $\overline{U_A}$ and $\overline{U_B}$ are

$$\overline{U_A} \sim N\left(\mu_{U_A}, \frac{\sigma_{U_A}}{\sqrt{M}}\right) \quad \text{and} \quad \overline{U_B} \sim N\left(\mu_{U_B}, \frac{\sigma_{U_B}}{\sqrt{M}}\right). \tag{5.20}$$

Since $t_x = \overline{u_A} - \overline{u_B}$, we have

$$T_x \sim N\left(\mu_{U_A} - \mu_{U_B}, \sqrt{\frac{\sigma_{U_A}^2}{M} + \frac{\sigma_{U_B}^2}{M}}\right) \tag{5.21}$$

where t_x is the value of the random variable T_x.

We assume that there is no bias in the measurement errors, therefore $\mu_{U_A} = \mu_{U_B} = 0$. With the above assumptions, we can be $(1-\alpha)100\%$ confident that the translational alignment error in the x direction will not exceed $\varepsilon_{\mathbf{\tau}} > 0$ when the following condition is true [Walpole1993]:

$$\varepsilon_{\boldsymbol{\tau}}^2 M \geq \left(z_{\alpha/2} \sqrt{\sigma_{U_A}^2 + \sigma_{U_B}^2} \right)^2 \tag{5.22}$$

where

$$P\left(-z_{\alpha/2} < Z < z_{\alpha/2}\right) = 1 - \alpha \tag{5.23}$$

and Z is a random variable that has the standard normal distribution.

If U_{Ai} and U_{Bi} are not normal distributions, then the condition in Equation (5.22) is still a good approximation as long as M is greater than 30 [Walpole1993].

Normally, σ_{U_A}, σ_{U_B}, σ_{V_A}, σ_{V_B}, σ_{W_A}, and σ_{W_B} are not constant and they may vary depending on factors such as the choice of the coordinate system with respect to the surface's orientation, the incident angle of the laser to the surface point, the surface reflectance properties, and the distance between the sensor and the surface point. To simplify the analysis, and the fact that we can be more conservative in this case, we can assume all the above standard deviations are less than or equal to the worst possible RMS error in range measurement, e_{RMS}. Then from Equation (5.22), we obtain the more conservative condition

$$\varepsilon_{\boldsymbol{\tau}}^2 M \geq 2\left(z_{\alpha/2} e_{\mathrm{RMS}}\right)^2. \tag{5.24}$$

However, the condition in Equation (5.24) is only true in a special case. When a point $\mathbf{p}_i = [x_i, y_i, z_i]^{\mathrm{T}}$ is translated by a small distance t_x in the x direction, its contribution to the energy function (the constraint), E_P in Equation (5.12), is

$$D^2\left(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{p}_i)\right) = d\boldsymbol{\tau}^{\mathrm{T}}\, \boldsymbol{V}_{\boldsymbol{\tau}}(\mathbf{p}_i) \boldsymbol{V}_{\boldsymbol{\tau}}^{\mathrm{T}}(\mathbf{p}_i)\, d\boldsymbol{\tau} = \left(\mathbf{n}_i \cdot \begin{bmatrix} t_x \\ 0 \\ 0 \end{bmatrix} \right)^2 = \left(\begin{bmatrix} n_{i,x} \\ n_{i,y} \\ n_{i,z} \end{bmatrix} \cdot \begin{bmatrix} t_x \\ 0 \\ 0 \end{bmatrix} \right)^2 = t_x^2 n_{i,x}^2 \tag{5.25}$$

where $\mathbf{n}_i$ is the unit normal at $\mathbf{p}_i$. We can easily see that $D^2\left(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{p}_i)\right)$ is maximum only when $\mathbf{n}_i = [\pm 1, 0, 0]^{\mathrm{T}}$, and in this case we get

$$E_P(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{p}_i)) = \sum_{\mathbf{p}_i \in P} D^2(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{p}_i)) = t_x^2 \sum_{i=1}^{M} n_{i,x}^2 = t_x^2 \sum_{i=1}^{M} (\pm 1)^2 = t_x^2 M \tag{5.26}$$

The R.H.S. of Equation (5.26) is basically the same as the L.H.S. of Equation (5.24). This makes sense because Equation (5.24) is derived on the assumption that each point in point set B is matched correctly with the corresponding point in point set A. Therefore, the vector between each pair of corresponding points is parallel to the direction of the translation. When the translation is $[t_x, 0, 0]^{\mathrm{T}}$, this is equivalent to having a surface normal $\mathbf{n}_i = [\pm 1, 0, 0]^{\mathrm{T}}$ at every point of one of the point sets. Therefore, we can say that the condition in Equation (5.24) is true only in the special case when each point $\mathbf{p}_i$ is at its maximum constraint on the translation in the x direction, that is when $\mathbf{n}_i = [\pm 1, 0, 0]^{\mathrm{T}}$. However, in the general case when each $\mathbf{n}_i$ can be any unit vector, we have

$$E_P(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{p}_i)) = \sum_{\mathbf{p}_i \in P} D^2(\boldsymbol{T}_{\boldsymbol{\tau}}(\mathbf{p}_i)) = t_x^2 \sum_{i=1}^{M} n_{i,x}^2 . \tag{5.27}$$

Now, for the general case, we can be $(1-\alpha)100\%$ confident that the translational alignment error in the x direction will not exceed $\varepsilon_{\boldsymbol{\tau}} > 0$ when the following condition is true:

$$\varepsilon_{\boldsymbol{\tau}}^2 \sum_{i=1}^{M} n_{i,x}^2 \geq 2\left(z_{\alpha/2} e_{\mathrm{RMS}}\right)^2 . \tag{5.28}$$

Similarly, we can obtain the following conditions for translations in the y and z directions, respectively:

$$\varepsilon_{\boldsymbol{\tau}}^2 \sum_{i=1}^{M} n_{i,y}^2 \geq 2\left(z_{\alpha/2} e_{\mathrm{RMS}}\right)^2 \quad \text{and} \quad \varepsilon_{\boldsymbol{\tau}}^2 \sum_{i=1}^{M} n_{i,z}^2 \geq 2\left(z_{\alpha/2} e_{\mathrm{RMS}}\right)^2 . \tag{5.29}$$

By combining the results in Equations (5.28), (5.29), and (5.14), we get

$$\varepsilon_{\boldsymbol{\tau}}^2 \sum_{i=1}^{M} \left(n_{i,x}^2 + n_{i,y}^2 + n_{i,z}^2\right) \geq 3 \cdot 2 \left(z_{\alpha/2} e_{\mathrm{RMS}}\right)^2$$
$$\Rightarrow \quad \varepsilon_{\boldsymbol{\tau}}^2 M \geq 6 \left(z_{\alpha/2} e_{\mathrm{RMS}}\right)^2 \quad \Rightarrow \quad M \geq M_{\min} = 6 \left(\frac{z_{\alpha/2} e_{\mathrm{RMS}}}{\varepsilon_{\boldsymbol{\tau}}} \right)^2 \tag{5.30}$$
$$\Rightarrow \quad \lambda_1 + \lambda_2 + \lambda_3 \geq M_{\min} = 6 \left(\frac{z_{\alpha/2} e_{\mathrm{RMS}}}{\varepsilon_{\boldsymbol{\tau}}} \right)^2$$

Finally, with the results in Equations (5.28), (5.29) and (5.30), for the general case when $\mathbf{n}_i$ can be any unit vector, we get the following condition:

Translational Alignment Error Condition. We can be $(1-\alpha)^3 100\%$ confident that the translational alignment error in *any* direction will not exceed $\sqrt{\varepsilon_{\boldsymbol{\tau}}^2 + \varepsilon_{\boldsymbol{\tau}}^2 + \varepsilon_{\boldsymbol{\tau}}^2} = \sqrt{3}\,\varepsilon_{\boldsymbol{\tau}}$ when $\lambda_1 \geq M_{\min}/3$, $\lambda_2 \geq M_{\min}/3$ and $\lambda_3 \geq M_{\min}/3$, where $M_{\min} = 6\left(z_{\alpha/2} e_{\mathrm{RMS}} / \varepsilon_{\boldsymbol{\tau}}\right)^2$.

Given that a confidence interval has been selected, the translational alignment error condition can be used to estimate the value of $\varepsilon_{\boldsymbol{\tau}}$, e_{RMS}, or $M_{\min}$, given that the other two values are already provided. For example, suppose we are given point sets *A* and *B*, which are two different sets of measurements of the same set of true surface points, and the RMS measurement error is e_{RMS}. We first use one of the two point sets to compute the values of λ_1, λ_2 and λ_3 as in Equations (5.9)–(5.13). Then, let $M_{\min} = 3\lambda_3$, and solve for the value of $\varepsilon_{\boldsymbol{\tau}}$ in the equation $M_{\min} = 6\left(z_{\alpha/2} e_{\mathrm{RMS}} / \varepsilon_{\boldsymbol{\tau}}\right)^2$. According to the error condition, we can be $(1-\alpha)^3 100\%$ confident that, if we register point sets *A* and *B* to each other, the translational alignment error in *any* direction will not exceed $\sqrt{3}\,\varepsilon_{\boldsymbol{\tau}}$.

For analyzing candidate views for registration accuracies, e_{RMS} and $\varepsilon_{\boldsymbol{\tau}}$ are already specified, and these allow $M_{\min}$ to be calculated. If a view produces $\lambda_3 < M_{\min}/3$, then it is rejected. Later, In Section 5.3.3, I describe how to compute, for my view planner, a value for e_{RMS}, and show, given a candidate view, how λ_1, λ_2 and λ_3 are computed.

5.3.2.2 Rotational Alignment Error

The rotation component of $\boldsymbol{V}(\mathbf{x}_s)$ can be written as

$$\boldsymbol{V}_{\boldsymbol{\theta}}(\mathbf{x}_s) = \frac{\partial}{\partial \boldsymbol{\theta}} D(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{x}_s)) = \mathbf{x}_s \times \mathbf{n}_{\mathbf{x}_s} \tag{5.31}$$

where $\boldsymbol{\theta} = [\omega_x, \omega_y, \omega_z]^{\mathrm{T}}$. Then

$$D(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{x}_s)) = \boldsymbol{V}_{\boldsymbol{\theta}}^{\mathrm{T}}(\mathbf{x}_s)\, d\boldsymbol{\theta}\,. \tag{5.32}$$

By squaring Equation (5.32), we get

$$D^2(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{x}_s)) = d\boldsymbol{\theta}^{\mathrm{T}}\, \boldsymbol{V}_{\boldsymbol{\theta}}(\mathbf{x}_s) \boldsymbol{V}_{\boldsymbol{\theta}}^{\mathrm{T}}(\mathbf{x}_s)\, d\boldsymbol{\theta} = d\boldsymbol{\theta}^{\mathrm{T}}\, \boldsymbol{M}_{\boldsymbol{\theta}}(\mathbf{x}_s)\, d\boldsymbol{\theta}\,. \tag{5.33}$$

By summing the quantity in Equation (5.33) over the set, *P*, we get

$$E_P(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{x}_s)) = \sum_{\mathbf{x}_s \in P} D^2(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{x}_s)) = d\boldsymbol{\theta}^{\mathrm{T}} \left(\sum_{\mathbf{x}_s \in P} \boldsymbol{M}_{\boldsymbol{\theta}}(\mathbf{x}_s) \right) d\boldsymbol{\theta} = d\boldsymbol{\theta}^{\mathrm{T}}\, \boldsymbol{\Psi}_{\boldsymbol{\theta}}\, d\boldsymbol{\theta}\,. \tag{5.34}$$

Using principal component analysis, $\boldsymbol{\Psi}_{\boldsymbol{\theta}}$ can be factorized into

$$\boldsymbol{\Psi}_{\boldsymbol{\theta}} = \mathbf{Q}_{\boldsymbol{\theta}}\, \boldsymbol{\Lambda}_{\boldsymbol{\theta}}\, \mathbf{Q}_{\boldsymbol{\theta}}^{\mathrm{T}} = \mathbf{Q}_{\boldsymbol{\theta}} \begin{bmatrix} \gamma_1 & 0 & 0 \\ 0 & \gamma_2 & 0 \\ 0 & 0 & \gamma_3 \end{bmatrix} \mathbf{Q}_{\boldsymbol{\theta}}^{\mathrm{T}} \tag{5.35}$$

where $\gamma_1 \geq \gamma_2 \geq \gamma_3$ are the eigenvalues of $\boldsymbol{\Psi}_{\boldsymbol{\theta}}$, and the columns of $\mathbf{Q}_{\boldsymbol{\theta}}$ are the corresponding unit eigenvectors. Since the sum of the eigenvalues is equal to the trace of the original matrix [Strang1988],

$$
\begin{aligned}
S &= \gamma_1 + \gamma_2 + \gamma_3 \\
&= \operatorname{tr}(\boldsymbol{\Psi}_{\boldsymbol{\theta}}) = \operatorname{tr}\left(\sum_{\mathbf{x}_s \in P} \boldsymbol{M}_{\boldsymbol{\theta}}(\mathbf{x}_s) \right) = \operatorname{tr}\left(\sum_{\mathbf{x}_s \in P} \boldsymbol{V}_{\boldsymbol{\theta}}(\mathbf{x}_s) \boldsymbol{V}_{\boldsymbol{\theta}}^{\mathrm{T}}(\mathbf{x}_s) \right) \\
&= \sum_{\mathbf{x}_s \in P} \| \boldsymbol{V}_{\boldsymbol{\theta}}(\mathbf{x}_s) \|^2 = \sum_{\mathbf{x}_s \in P} \left\| \mathbf{x}_s \times \mathbf{n}_{\mathbf{x}_s} \right\|^2 = \sum_{\mathbf{x}_s \in P} \| \mathbf{x}_s \|^2 \sin^2 \phi_{\mathbf{x}_s}
\end{aligned}
\tag{5.36}
$$

where $\phi_{\mathbf{x}_s}$ is the angle between the vector $\mathbf{x}_s$ and $\mathbf{n}_{\mathbf{x}_s}$.

For the alignment of the surface to be successful, γ_i must be greater than 0 for $i = 1, 2, \text{and } 3$. Ideally, we wish to select the set of points for *P* such that $\gamma_1 = \gamma_2 = \gamma_3 = S / 3$, which means that the surface alignment is constrained equally in the rotations about all three orthogonal axes. The minimum requirement for alignment is that $M = 3$, where *M* is the number of points in *P*, and that the three vector in the set $\{ \mathbf{x}_s \times \mathbf{n}_{\mathbf{x}_s} \mid \mathbf{x}_s \in P \}$ must span the 3D space. In practice, with errors in the range measurements, it is necessary to have $M \geq M_{\min} > 3$ and $S \geq S_{\min} > 0$, such that the rotational alignment can be performed to a certain desired accuracy, assuming the surface is already correctly translated. In the following, we investigate the conditions necessary to attain a specified angular accuracy in the rotational alignment.

Let $[x_i, y_i, z_i]^{\mathrm{T}}$ be the 3D coordinates of the *i*th true surface point, where $i = 1, 2, \ldots, M$. Suppose there are two sets of measurements of the surface points, producing the point set *A* with coordinates $[x_i + u_{Ai}, y_i + v_{Ai}, z_i + w_{Ai}]^{\mathrm{T}}$ and the point set *B* with coordinates $[x_i + u_{Bi}, y_i + v_{Bi}, z_i + w_{Bi}]^{\mathrm{T}}$, where ($u_{Ai}$, u_{Bi}), (v_{Ai}, v_{Bi}), and (w_{Ai}, w_{Bi}) are measurement errors in the *x*, *y*, and *z* directions, respectively.

Suppose point set *B* is to be rotated about the origin so that it is aligned with point set *A*. For the alignment, the correspondences between points in point sets *A* and *B* are known. The alignment uses the least-squares (least-sum-of-squares) error metric, where we want to find the rotation vector $\boldsymbol{\theta} = [\omega_x, \omega_y, \omega_z]^{\mathrm{T}}$ to rotate point set *B* to minimize

$$\begin{aligned} SSE &= \sum_{i=1}^{M} \left\| \begin{bmatrix} x_i + u_{Ai} \\ y_i + v_{Ai} \\ z_i + w_{Ai} \end{bmatrix} - \boldsymbol{R_z}(\omega_z) \cdot \boldsymbol{R_y}(\omega_y) \cdot \boldsymbol{R_x}(\omega_x) \cdot \begin{bmatrix} x_i + u_{Bi} \\ y_i + v_{Bi} \\ z_i + w_{Bi} \end{bmatrix} \right\|^2 \\ &= \sum_{i=1}^{M} \left\| \begin{bmatrix} x_i + u_{Ai} \\ y_i + v_{Ai} \\ z_i + w_{Ai} \end{bmatrix} - \boldsymbol{R}(\omega_x, \omega_y, \omega_z) \cdot \begin{bmatrix} x_i + u_{Bi} \\ y_i + v_{Bi} \\ z_i + w_{Bi} \end{bmatrix} \right\|^2 \end{aligned} \tag{5.37}$$

where $\boldsymbol{R_x}(\omega_x)$, $\boldsymbol{R_y}(\omega_y)$ and $\boldsymbol{R_z}(\omega_z)$ are the rotation matrices for rotations of ω_x, ω_y and ω_z radians about the *x*-axis, *y*-axis and *z*-axis, respectively. Since the rotations will be small, we can approximate the matrix $\boldsymbol{R}(\omega_x, \omega_y, \omega_z)$ by using the approximations $\sin\theta \approx \theta$ and $\cos\theta \approx 1$ when $\theta \approx 0$. With this approximation, we get

$$\begin{aligned} SSE &\approx \sum_{i=1}^{M} \left\| \begin{bmatrix} x_i + u_{Ai} \\ y_i + v_{Ai} \\ z_i + w_{Ai} \end{bmatrix} - \begin{bmatrix} 1 & \omega_x\omega_y - \omega_z & \omega_x\omega_z + \omega_y \\ \omega_z & \omega_x\omega_y\omega_z + 1 & \omega_y\omega_z - \omega_x \\ -\omega_y & \omega_x & 1 \end{bmatrix} \cdot \begin{bmatrix} x_i + u_{Bi} \\ y_i + v_{Bi} \\ z_i + w_{Bi} \end{bmatrix} \right\|^2 \\ &\approx \sum_{i=1}^{M} \left\| \begin{bmatrix} x_i + u_{Ai} \\ y_i + v_{Ai} \\ z_i + w_{Ai} \end{bmatrix} - \begin{bmatrix} 1 & -\omega_z & \omega_y \\ \omega_z & 1 & -\omega_x \\ -\omega_y & \omega_x & 1 \end{bmatrix} \cdot \begin{bmatrix} x_i + u_{Bi} \\ y_i + v_{Bi} \\ z_i + w_{Bi} \end{bmatrix} \right\|^2 \end{aligned} \tag{5.38}$$

To simplify the analysis, we first consider only the rotation about the *x*-axis. This gives us

$$\begin{aligned} SSE_x &\approx \sum_{i=1}^{M} \left\| \begin{bmatrix} x_i + u_{Ai} \\ y_i + v_{Ai} \\ z_i + w_{Ai} \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -\omega_x \\ 0 & \omega_x & 1 \end{bmatrix} \cdot \begin{bmatrix} x_i + u_{Bi} \\ y_i + v_{Bi} \\ z_i + w_{Bi} \end{bmatrix} \right\|^2 \\ &= \sum_{i=1}^{M} \begin{pmatrix} (u_{Ai} - u_{Bi})^2 + (v_{Ai} - v_{Bi} + \omega_x(z_i + w_{Bi}))^2 + \\ (w_{Ai} - w_{Bi} - \omega_x(y_i + v_{Bi}))^2 \end{pmatrix} \end{aligned} \tag{5.39}$$

Differentiating SSE_x with respect to ω_x, we have

$$\frac{\partial(SSE_x)}{\partial\omega_x} \approx 2\sum_{i=1}^{M} \begin{pmatrix} (z_i + w_{Bi})(v_{Ai} - v_{Bi} + \omega_x(z_i + w_{Bi})) - \\ (y_i + v_{Bi})(w_{Ai} - w_{Bi} - \omega_x(y_i + v_{Bi})) \end{pmatrix}. \tag{5.40}$$

Normally $|z_i| >> |w_{Bi}|$ and $|y_i| >> |v_{Bi}|$, so we can further simplify the above by

$$\frac{\partial(SSE_x)}{\partial\omega_x} \approx 2\sum_{i=1}^{M}\left(z_i\left(v_{Ai} - v_{Bi} + \omega_x z_i\right) - y_i\left(w_{Ai} - w_{Bi} - \omega_x y_i\right)\right). \tag{5.41}$$

SSE_x is minimum when $\frac{\partial(SSE_x)}{\partial\omega_x} = 0$, and this is true when

$$\begin{aligned} &2\sum_{i=1}^{M}\left(z_i\left(v_{Ai} - v_{Bi} + \omega_x z_i\right) - y_i\left(w_{Ai} - w_{Bi} - \omega_x y_i\right)\right) = 0 \\ &\Rightarrow \omega_x \sum_{i=1}^{M}\left(y_i^2 + z_i^2\right) = \sum_{i=1}^{M}\left(y_i\left(w_{Ai} - w_{Bi}\right) - z_i\left(v_{Ai} - v_{Bi}\right)\right) \end{aligned} \tag{5.42}$$

Let v_{Ai}, v_{Bi}, w_{Ai} and w_{Bi} be the values of the independent random variables V_{Ai}, V_{Bi}, W_{Ai} and W_{Bi}, respectively. We also let $\sum_{i=1}^{M}\left(y_i\left(w_{Ai} - w_{Bi}\right) - z_i\left(v_{Ai} - v_{Bi}\right)\right)$ be the value of the random variable Q_x. Assume that each of V_{Ai}, V_{Bi}, W_{Ai} and W_{Bi} has normal distribution with mean μ_{V_A}, μ_{V_B}, μ_{W_A} and μ_{W_B}, and standard deviation σ_{V_A}, σ_{V_B}, σ_{W_A} and σ_{W_B}, respectively, i.e.

$$\begin{aligned} V_{Ai} &\sim N\left(\mu_{V_A}, \sigma_{V_A}\right) \quad \text{and} \quad V_{Bi} \sim N\left(\mu_{V_B}, \sigma_{V_B}\right), \\ W_{Ai} &\sim N\left(\mu_{W_A}, \sigma_{W_A}\right) \quad \text{and} \quad W_{Bi} \sim N\left(\mu_{W_B}, \sigma_{W_B}\right) \end{aligned} \tag{5.43}$$

then the mean [Walpole1993] of Q_x is

$$\mu_{Q_x} = \sum_{i=1}^{M}\left(y_i\left(\mu_{W_A} - \mu_{W_B}\right) - z_i\left(\mu_{V_A} - \mu_{V_B}\right)\right) \tag{5.44}$$

and the standard deviation [Walpole1993] of Q_x is

$$\sigma_{Q_x} = \sqrt{\sum_{i=1}^{M}\left(y_i^2\sigma_{W_A}^2 + y_i^2\sigma_{W_B}^2 + z_i^2\sigma_{V_A}^2 + z_i^2\sigma_{V_B}^2\right)}. \tag{5.45}$$

Q_x satisfies the Lindeberg condition (see Appendix A), and by Lindeberg's Theorem[4] [Ash1972, pp. 336–337], Q_x converges to a normal distribution, i.e.

$$\begin{aligned} & Q_x \sim N\left(\mu_{Q_x}, \sigma_{Q_x}\right) \\ \Rightarrow \; & Q_x \sim N\left(\sum_{i=1}^{M} \left(y_i \left(\mu_{W_A} - \mu_{W_B}\right) - z_i \left(\mu_{V_A} - \mu_{V_B}\right)\right), \; \sqrt{\sum_{i=1}^{M} \left(y_i^2 \sigma_{W_A}^2 + y_i^2 \sigma_{W_B}^2 + z_i^2 \sigma_{V_A}^2 + z_i^2 \sigma_{V_B}^2 \right)} \right) \end{aligned} \tag{5.46}$$

Since we are assuming there is no bias in the measurement errors, $\mu_{W_A} = \mu_{W_B} = \mu_{V_A} = \mu_{V_B} = 0$. With the above assumptions, we can be $(1-\alpha)100\%$ confident that the rotational alignment error about the *x*-axis will not exceed $\varepsilon_{\mathbf{\theta}} > 0$ radians when the following condition is true:

$$\varepsilon_{\mathbf{\theta}} \sum_{i=1}^{M} \left(y_i^2 + z_i^2 \right) \; \geq \; z_{\alpha/2} \sqrt{\sum_{i=1}^{M} \left(y_i^2 \sigma_{W_A}^2 + y_i^2 \sigma_{W_B}^2 + z_i^2 \sigma_{V_A}^2 + z_i^2 \sigma_{V_B}^2 \right)}. \tag{5.47}$$

Assuming σ_{U_A}, σ_{U_B}, σ_{V_A}, σ_{V_B}, σ_{W_A}, and σ_{W_B} are less than or equal to the worst possible RMS error in range measurement, e_{RMS}, we obtain the more conservative condition

$$\begin{aligned} & \varepsilon_{\mathbf{\theta}} \sum_{i=1}^{M} \left(y_i^2 + z_i^2 \right) \; \geq \; z_{\alpha/2} \sqrt{\sum_{i=1}^{M} \left(y_i^2 e_{\text{RMS}}^2 + y_i^2 e_{\text{RMS}}^2 + z_i^2 e_{\text{RMS}}^2 + z_i^2 e_{\text{RMS}}^2 \right)} \\ \Rightarrow \; & \varepsilon_{\mathbf{\theta}}^2 \sum_{i=1}^{M} \left(y_i^2 + z_i^2 \right) \; \geq \; 2\left(z_{\alpha/2} e_{\text{RMS}} \right)^2 \end{aligned} \tag{5.48}$$

However, the condition in Equation (5.48) is only valid for a special case. When a point $\mathbf{p}_i = [x_i, y_i, z_i]^{\text{T}}$ is rotated a small angle ω_x about the *x*-axis, its contribution to the energy function (the constraint), E_P in Equation (5.34), is

[4]Lindeberg's Theorem is a generalization of the central limit theorem. The central limit theorem states that the sum of many independent identically distributed random variables with finite variance will be approximately normally distributed. Lindeberg's Theorem does not require identical distribution, but incorporates a condition (the Lindeberg condition), which, if satisfied, implies that the sum will be approximately normally distributed.

$$\begin{aligned} D^2(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{p}_i)) &= d\boldsymbol{\theta}^{\mathrm{T}}\, \boldsymbol{V}_{\boldsymbol{\theta}}(\mathbf{p}_i)\boldsymbol{V}_{\boldsymbol{\theta}}^{\mathrm{T}}(\mathbf{p}_i)\, d\boldsymbol{\theta} \\ &= \left(\mathbf{p}_i \times \mathbf{n}_i \cdot \begin{bmatrix} \omega_x \\ 0 \\ 0 \end{bmatrix} \right)^2 = \left(\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \times \begin{bmatrix} n_{i,x} \\ n_{i,y} \\ n_{i,z} \end{bmatrix} \cdot \begin{bmatrix} \omega_x \\ 0 \\ 0 \end{bmatrix} \right)^2 = \omega_x^2 \left(y_i n_{i,z} - z_i n_{i,y} \right)^2 \end{aligned} \tag{5.49}$$

where $\mathbf{n}_i$ is the unit surface normal at $\mathbf{p}_i$. $D^2(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{p}_i))$ is maximum if $\mathbf{n}_i$ is perpendicular to both the vector $\mathbf{p}_i$ and the x-axis, i.e.

$$\mathbf{n}_i = \mathbf{normalize}\left(\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \times \begin{bmatrix} \pm 1 \\ 0 \\ 0 \end{bmatrix} \right) = \mathbf{normalize}\left(\begin{bmatrix} 0 \\ \pm z_i \\ \mp y_i \end{bmatrix} \right) = \begin{bmatrix} 0 \\ \pm z_i \Big/ \sqrt{y_i^2 + z_i^2} \\ \mp y_i \Big/ \sqrt{y_i^2 + z_i^2} \end{bmatrix}. \tag{5.50}$$

Then,

$$\begin{aligned} D^2(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{p}_i)) &= \omega_x^2 \left(y_i n_{i,z} - z_i n_{i,y} \right)^2 \\ &= \omega_x^2 \left(y_i \left(\mp y_i \Big/ \sqrt{y_i^2 + z_i^2} \right) - z_i \left(\pm z_i \Big/ \sqrt{y_i^2 + z_i^2} \right) \right)^2 = \omega_x^2 \left(y_i^2 + z_i^2 \right) \end{aligned} \tag{5.51}$$

and

$$E_P(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{p}_i)) = \sum_{\mathbf{p}_i \in P} D^2(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{p}_i)) = \omega_x^2 \sum_{i=1}^{M} \left(y_i^2 + z_i^2 \right). \tag{5.52}$$

The R.H.S. of Equation (5.52) is basically the same as the L.H.S. of Equation (5.48). This is expected because Equation (5.48) is derived on the assumption that each point in point set B is matched correctly with the corresponding point in point set A. The vector between each pair of corresponding points is parallel to the direction of the rotation. When the rotation is about the x-axis, this vector is perpendicular to both the vector $\mathbf{p}_i$ and the x-axis. This vector can be treated as the normal $\mathbf{n}_i$. Therefore, we can say that the condition in Equation (5.48) is true only in the special case when each point $\mathbf{p}_i$ is at its maximum constraint on the rotation about the x-axis, i.e. $\mathbf{n}_i$ is perpendicular to both the vector $\mathbf{p}_i$ and the x-axis. However, in the general case when each $\mathbf{n}_i$ can be any unit vector, we have

$$\begin{aligned} & D^2(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{p}_i)) = \omega_x^2 (y_i n_{i,z} - z_i n_{i,y})^2 \\ & \Rightarrow E_P(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{p}_i)) = \sum_{\mathbf{p}_i \in P} D^2(\boldsymbol{T}_{\boldsymbol{\theta}}(\mathbf{p}_i)) = \omega_x^2 \sum_{i=1}^{M} (y_i n_{i,z} - z_i n_{i,y})^2 \end{aligned} \tag{5.53}$$

Now, for the general case, we can be $(1-\alpha)100\%$ confident that the rotational alignment error about the x-axis will not exceed $\varepsilon_{\boldsymbol{\theta}} > 0$ radians when the following condition is true:

$$\varepsilon_{\boldsymbol{\theta}}^2 \sum_{i=1}^{M} (y_i n_{i,z} - z_i n_{i,y})^2 \ \geq \ 2(z_{\alpha/2} e_{\text{RMS}})^2 . \tag{5.54}$$

The conditions for rotations about the y-axis and z-axis can be similarly derived respectively as

$$\begin{aligned} & \varepsilon_{\boldsymbol{\theta}}^2 \sum_{i=1}^{M} (z_i n_{i,x} - x_i n_{i,z})^2 \ \geq \ 2(z_{\alpha/2} e_{\text{RMS}})^2 , \\ & \varepsilon_{\boldsymbol{\theta}}^2 \sum_{i=1}^{M} (x_i n_{i,y} - y_i n_{i,x})^2 \ \geq \ 2(z_{\alpha/2} e_{\text{RMS}})^2 . \end{aligned} \tag{5.55}$$

By combining the results in Equations (5.54), (5.55), and (5.36), we get

$$\begin{aligned} & \varepsilon_{\boldsymbol{\theta}}^2 \sum_{i=1}^{M} \left((y_i n_{i,z} - z_i n_{i,y})^2 + (z_i n_{i,x} - x_i n_{i,z})^2 + (x_i n_{i,y} - y_i n_{i,x})^2 \right) \ \geq \ 3 \cdot 2 (z_{\alpha/2} e_{\text{RMS}})^2 \\ \Rightarrow \quad & \varepsilon_{\boldsymbol{\theta}}^2 \sum_{\mathbf{p}_i \in P} \| \mathbf{p}_i \times \mathbf{n}_i \|^2 \ \geq \ 6 (z_{\alpha/2} e_{\text{RMS}})^2 \\ \Rightarrow \quad & \gamma_1 + \gamma_2 + \gamma_3 \ \geq \ S_{\min} = 6 \left(\frac{z_{\alpha/2} e_{\text{RMS}}}{\varepsilon_{\boldsymbol{\theta}}} \right)^2 \end{aligned} \tag{5.56}$$

Finally, with the results in Equations (5.54), (5.55), and (5.56), for the general case when $\mathbf{n}_i$ can be any unit vector, we get the following condition:

Rotational Alignment Error Condition. We can be $(1-\alpha)^3 100\%$ confident that the rotational alignment errors about the *x*-axis, *y*-axis and *z*-axis, respectively, will not exceed $\varepsilon_{\boldsymbol{\theta}} > 0$ radians when $\gamma_1 \geq S_{\min}/3$, $\gamma_2 \geq S_{\min}/3$ and $\gamma_3 \geq S_{\min}/3$, where $S_{\min} = 6\left(z_{\alpha/2} e_{\mathrm{RMS}} / \varepsilon_{\boldsymbol{\theta}}\right)^2$.

Similar to the application of the translational alignment error condition for analyzing candidate views, if a view produces $\gamma_3 < S_{\min}/3$, then it is rejected.

5.3.2.3 Remarks

The three registration accuracy factors listed at the beginning of Section 5.3 are taken into account in the two alignment error conditions. For Factor (1), the range measurement errors are incorporated into e_{RMS}. For Factor (2), if there is a lot of overlap between the two surfaces, and thus a large number of point pairs that can be used in the registration, the values of the eigenvalues λ_i and γ_i will likely be large. Larger values for λ_i and γ_i allow $M_{\min}$ and $S_{\min}$ to be higher in value, which in turn result in smaller error bounds $\varepsilon_{\boldsymbol{\tau}}$ and $\varepsilon_{\boldsymbol{\theta}}$. For Factor (3), the values of λ_i and γ_i reflect the amount of constraint the points have on the different components of the 3D rigid-body transformation, and each eigenvalue is proportional to the registration accuracy that can be achieved in the corresponding motion.

The derivations of the alignment error conditions assume that the point correspondences between the two point sets are known. In practice, these point correspondences may not be available, such as when these point sets come from range images of the same surface scanned from different scanner positions. Fortunately, in many cases, the ICP algorithm is able to provide a good approximation of the point correspondences. As the ICP algorithm iteratively refines the alignment of the two surfaces, the matching of point pairs becomes more accurate. When the two surfaces are almost correctly aligned, the accuracy of the closest-point matching is limited by the sampling spacing between points in each point set. For example, a true surface point is sampled in point set *A*, but not sampled in point set *B* because it falls between two adjacent samples in *B*. In this case, the sample in *A* will be matched with one of the two adjacent samples in B. The errors caused by the sampling spacing are considered quantization errors, and can be incorporated into e_{RMS} when using the two alignment error conditions.

Of course, there are ways that these quantization errors can be reduced. For example, we can interpolate the two adjacent samples in point set B to produce a new point to pair with the point in point set A. Another way to reduce the quantization errors is to use the point-to-plane error metric, which is implemented in the view planner. Figure 5.4 illustrates how it helps to improve the point matching. In the diagram, the voxel center, c, is first paired with the closest vertex, v, on the triangle mesh. If the point-to-point metric is used, then the distance of the pair (c, v) is used in the energy function to be minimized. On the other hand, if the point-to-plane metric is used, then the distance of the pair (c, p) is actually used in the energy function. The point p is a better "partner" for c because it is more likely to be, or closer to, the true surface point that corresponds to c.

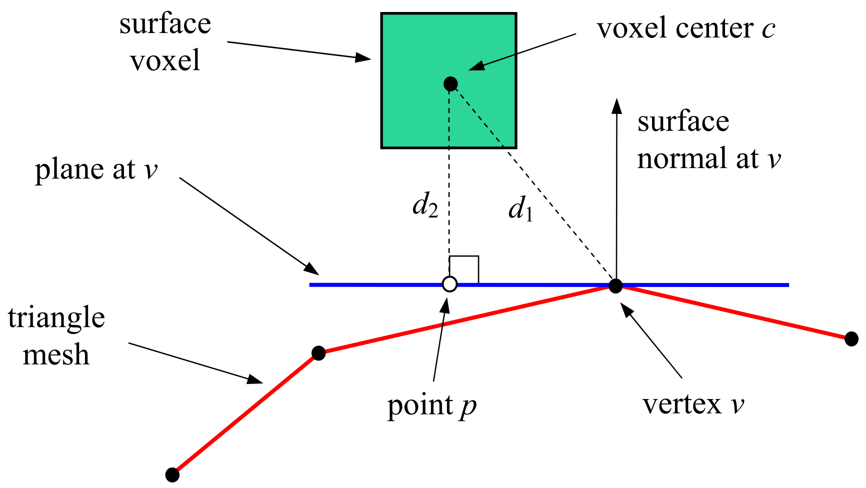


Figure 5.4: The use of the point-to-plane error metric improves the accuracy of the point matching.

A limitation of the alignment error conditions is that the errors in the surface normals are not taken into consideration. The true surface normals at the measured points are needed to compute the eigenvalues in the alignment error conditions, but they are usually not available, and estimated normals are used instead. For a range image, we can estimate these normals using the sample points by fitting a plane to the points in the neighborhood of each candidate point. The estimation is more accurate if the points are from a relatively smooth region on the surface.

As we will see in Section 5.4, the errors in the surface normals may become a significant problem when λ_3 is very small relative to λ_1, or γ_3 is very small relative to γ_1. A solution is presented in that section.

5.3.3 Application of Alignment Error Conditions

This section describes how the alignment error conditions are used to test candidate views produced by the hierarchical view evaluation. It is important to note that the validity of using the conditions to test candidate views is based on the assumption that the values of λ_i and γ_i at the planned pose will not be significantly different from those at the unknown actual pose. For my application, this is generally not an issue, as the scanner pose errors are very small in comparison to the size of the scene to be acquired.

To apply the two conditions, we have to first choose and determine the values of the several parameters, i.e. (1) the confidence interval $(1-\alpha)^3 100\%$ and the confidence limit $z_{\alpha/2}$, (2) the points' RMS position error e_{RMS}, (3) the translation error tolerance $\varepsilon_{\boldsymbol{\tau}}$, and (4) the rotation angle error tolerance $\varepsilon_{\boldsymbol{\theta}}$. These allow $M_{\min}$ and $S_{\min}$ to be calculated. Then, if a candidate view has $\lambda_3 < M_{\min}/3$ or $\gamma_3 < S_{\min}/3$, it is rejected.

In my experiments, the alignment error conditions are often applied at a 99% confidence interval. Which means $\alpha = 1 - \sqrt[3]{0.99} \approx 0.003345$ and $z_{\alpha/2} \approx 2.712$.

Sometimes, the rotational alignment error tolerance is specified as a distance instead of as an angle $\varepsilon_{\boldsymbol{\theta}}$. This distance error tolerance must then be converted to an angle tolerance $\varepsilon_{\boldsymbol{\theta}}$ before it can be used in the rotational alignment error condition. The range measurement errors are not the only source of errors that contribute to e_{RMS}. Errors can also come from the voxelization of surfaces and inexact point pairing. The following sections provide more details.

5.3.3.1 Computing the Eigenvalues

Given a candidate view, in order to compute the eigenvalues λ_i and γ_i, we need to find out which surface points will be used in the registration should the view be used to make the next range scan. We recall that, in the view planner, this new range image will be converted to a triangle mesh and registered with the center points of the set of true surface voxels in the octree partial scene model. Each surface voxel's center will be paired with the nearest vertex on the triangle mesh. Therefore, the maximum number of point pairs that can be used in the registration is the total number of surface voxels in the scene model. However, those surface voxels that will not be overlapped by the mesh will not be able to form point pairs to be used in the registration.

Even though the new scan has not been acquired, we can still estimate which surface voxels would be used in the registration. To do that, for each surface voxel in the current partial scene model, we test whether it satisfies all the scanning constraints with respect to the candidate view. If satisfied, the surface voxel is selected for the computation of λ_i and γ_i. In my implementation of the view planner, each surface voxel is tested to determine whether it is within the working range of the sensor, within the field of view of the scanner, within the threshold angle of incidence of the laser beam to the surface point, and whether it is visible from the candidate viewpoint.

The center points and the surface normals of the selected surface voxels are then used to compute λ_i as stated in Equations (5.9)–(5.13), and γ_i as in Equations (5.31)–(5.35).

5.3.3.2 Determining the RMS Error e_{RMS}

The main sources of errors that contribute to e_{RMS} are

(1) the range measurement errors,

(2) the quantization errors caused by the voxelization of the surface, and

(3) the error caused by pairing each voxel's center to a mesh vertex that does not correspond to the same true surface point as the voxel's center.

The range measurement errors are related to the precision or measurement uncertainty of the range sensor. For a time-of-flight-based sensor, theoretically, the range precision is constant throughout the sensor's range [Beraldin2000]. However, in practice, the precision actually deteriorates gradually with distance, because of factors such as the divergence of the emitted light beam and the attenuation of the returning light energy. As for triangulation-based range sensors, the precision is inversely proportional to the square of the distance. To be conservative about estimating the registration errors, we should use the lowest precision within the working range of the sensor or the precision at the furthest surface to be measured. The precision is usually specified as the RMS (or standard deviation) of the range errors and it is denoted by σ_{range} here.

The voxelization of the surface introduces a maximum error of ±½ voxel width at each voxel's center in each of the *x*, *y* and *z* directions. This quantization error is uniformly distributed between ±½ voxel width, has a mean of zero, and a standard deviation [Smith1999] of

$$\sigma_{\text{voxel-quant}} = \frac{\text{voxel width}}{\sqrt{12}}. \tag{5.57}$$

As pointed out in Section 5.3.2.3, when two surfaces are almost correctly aligned, the accuracy of the closest-point matching is limited by the spacing between adjacent vertices in the triangle mesh. More specifically, when a voxel's center is matched to a mesh vertex, the maximum error is half the sampling spacing in the immediate neighborhood of the vertex. This error can be greatly reduced by matching the voxel's center to the closest point on the triangle mesh, instead of only to the vertices. Another way to reduce this error is to use the point-to-plane error metric, as explained in Section 5.3.2.3 and Figure 5.4. This type of error can be reduced to be approximately equal to the range errors at the vertices, and therefore their standard deviation, $\sigma_{\text{matching}} = \sigma_{\text{range}}$.

Finally, the RMS error, e_{RMS}, for the alignment error conditions is

$$e_{\text{RMS}} = \sqrt{\sigma_{\text{range}}^2 + \sigma_{\text{voxel-quant}}^2 + \sigma_{\text{matching}}^2}. \tag{5.58}$$

5.3.3.3 Choosing a Rotation Angle Error Tolerance $\varepsilon_{\boldsymbol{\theta}}$

Sometimes, it is more convenient to specify the rotational alignment error tolerance as a distance instead of as an angle $\varepsilon_{\boldsymbol{\theta}}$. Since in the translational alignment error condition, the translation error tolerance is $\sqrt{3}\,\varepsilon_{\boldsymbol{\tau}}$ in any direction, it is natural to want to limit the maximum rotational-induced translation error to $\sqrt{3}\,\varepsilon_{\boldsymbol{\tau}}$ also. Suppose the coordinate frame for the scene model has been established, and the distance between the origin and the furthest point p_{furthest} in the scene is $d_{\max}$. The point p_{furthest} will potentially have the largest rotation-induced translation error. To limit this largest translation error to $\sqrt{3}\,\varepsilon_{\boldsymbol{\tau}}$, we use

$$\varepsilon_{\boldsymbol{\theta}} = 2\,sin^{-1}\left(\frac{\sqrt{3}\,\varepsilon_{\boldsymbol{\tau}}}{2\,d_{\max}}\right). \qquad (5.59)$$

However, the value of $\varepsilon_{\boldsymbol{\theta}}$ computed in Equation (5.59) can be too small and results in a very large $S_{\min}$ that makes the rotational alignment error condition too stringent. To derive a realistic value for $\varepsilon_{\boldsymbol{\theta}}$, we should consider all the points that would be used in the registration, and take into account the distribution of their distances from the origin. One such method has been implemented in the view planner, and it is described next.

Given a candidate view, let V be the set of surface voxels that satisfy all the scanning constraints with respect to the view, and let $v_i \in V$ for $i = 1, 2, \ldots, |V|$. We wish to limit the rotation-induced translation errors of these voxels to within an RMS value τ_{RMS}, i.e.

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}\tau_i^2} \le \tau_{\text{RMS}} \qquad \Rightarrow \qquad \sum_{i=1}^{N}\tau_i^2 \le N\tau_{\text{RMS}}^2 \qquad (5.60)$$

where $N = |V|$, and τ_i is the rotation-induced translation error of the center of voxel v_i. Suppose ϕ is the amount of rotation that results in τ_i for all $i = 1, 2, \ldots, |V|$, i.e.

$$\tau_i = 2d_i\,sin\frac{\phi}{2} \qquad (5.61)$$

where d_i is the distance between the origin and the center of the voxel v_i. Substituting Equation (5.61) into (5.60), we get

$$\sum_{i=1}^{N}\left(2d_i\, sin\frac{\phi}{2}\right)^2 \le N\tau_{\text{RMS}}^2 \quad\Rightarrow\quad \phi \;\le\; 2\,sin^{-1}\sqrt{\frac{N\tau_{\text{RMS}}^2}{4\sum_{i=1}^{N} d_i^2}} = \phi_{\max}\,. \tag{5.62}$$

Finally, we choose the angle tolerance $\varepsilon_{\boldsymbol{\theta}} = \phi_{\max}$ for the rotational alignment error condition.

5.4 Example

The synthetic 3D scene shown in Figure 5.5 is used to demonstrate the application of the alignment error conditions to test the registration constraint for the candidate views. The entire scene is 35.5 feet along the *x*-axis, 20 feet along the *z*-axis, and 10 feet along the *y*-axis (vertical axis). A doorway connects the two rooms, and it is 4 feet wide and 8 feet high. Both rooms have flat ceilings that are not shown in the figure. The figure also shows the scanner pose for making the first scan, where its origin is 5 feet above the floor. This pose will be used as the coordinate frame of the volumetric partial scene model.

Figure 5.6 shows the triangle mesh constructed from the first scan. Again, the ceiling is not shown. The scan is made with range precision $\sigma_{\text{range}} = 0.5$ inch. The range image has been smoothed using an anisotropic diffusion method [Black1998], so that the surface normals at the range samples can be more accurately estimated. Next, the triangle mesh is voxelized to create a volumetric partial scene model, where each surface voxel is 2 inches wide.

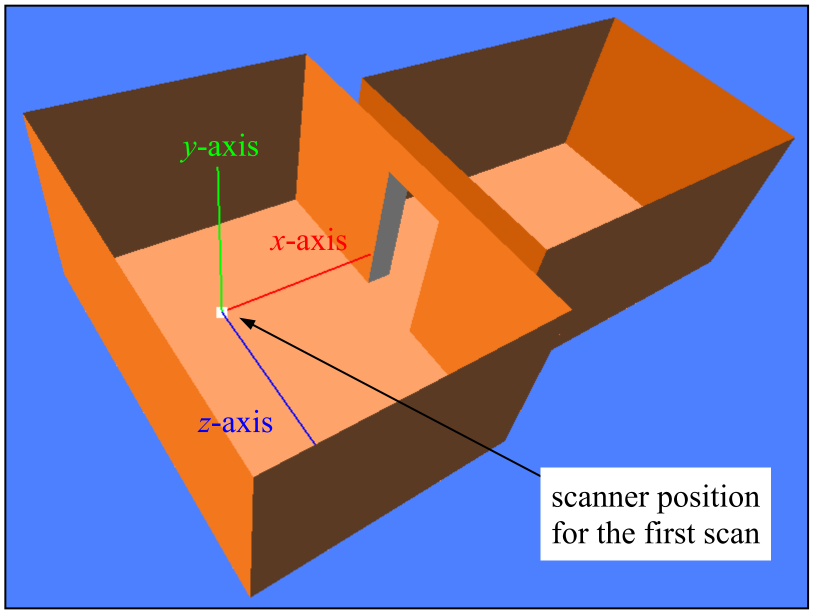


Figure 5.5: The synthetic 3D scene that is used to demonstrate the application of the alignment error conditions in view planning. The coordinate frame shown is the scanner pose for making the first scan, and it will also be used as the coordinate frame of the volumetric partial scene model. The ceilings of the two rooms are not shown.

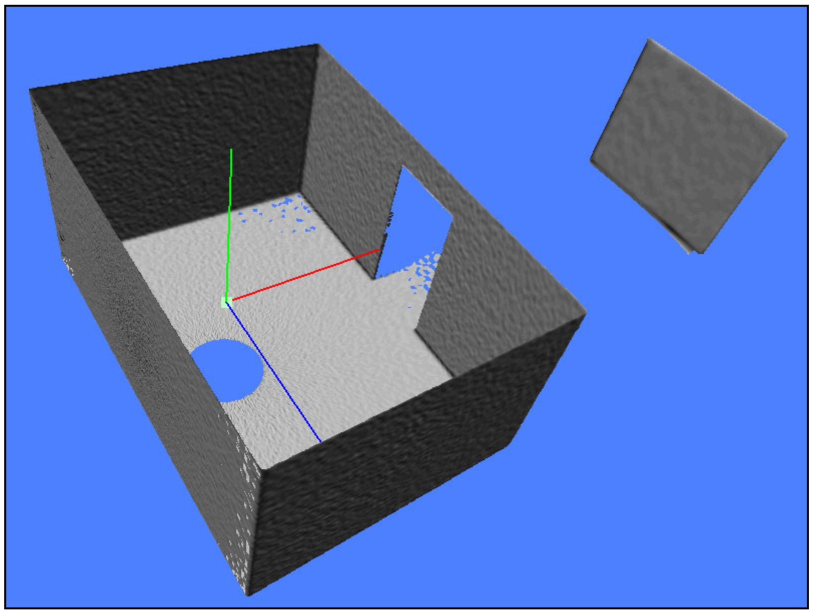


Figure 5.6: The triangle mesh of the first scan. The scan is made with range precision $\sigma_{\text{range}} = 0.5$ inch. The ceiling is not shown.

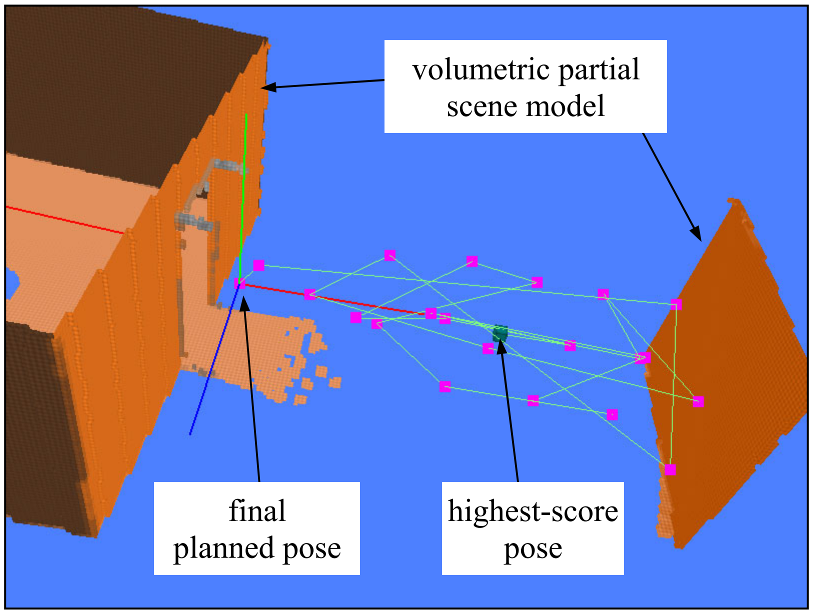


Figure 5.7: The candidate views produced by the hierarchical view evaluations are tested for the alignment error conditions. The highest-score view fails the conditions, and many other views are tested until one near to the doorway is found to satisfy both conditions.

With the volumetric partial scene model, a set of discrete candidate views are produced by the hierarchical view evaluations. Figure 5.7 shows the view (pose) with the highest score, which, however, does not satisfy the alignment conditions and is rejected. Another 21 views (shown in pink) are tested for the alignment error conditions, in descending order of their scores. Eventually, the last of them satisfies both of the alignment error conditions and is chosen as the planned pose for the next scan.

The following values are provided for computing the alignment error conditions:

(1) $z_{\alpha/2} \approx 2.712$, corresponding to a 99% confidence interval;

(2) surface voxel width is 2 inches;

(3) range precision, $\sigma_{\text{range}} = 0.5$ inch;

(4) $\varepsilon_{\boldsymbol{\tau}} = 0.5$ inch;

(5) maximum RMS value of the rotation-induced translation errors, $\tau_{\text{RMS}} = 0.5$ inch.

Since $\sigma_{\text{voxel-quant}} = (\text{voxel width})/\sqrt{12}$ and $\sigma_{\text{matching}} = \sigma_{\text{range}}$, the value of e_{RMS} is computed as

$$e_{\text{RMS}} = \sqrt{\sigma^2_{\text{range}} + \sigma^2_{\text{voxel-quant}} + \sigma^2_{\text{matching}}} \approx 0.9129 .$$

In Table 5.1, we can see the actual values of $\varepsilon_{\boldsymbol{\theta}}$, λ_i, γ_i, $M_{\min}/3$ and $S_{\min}/3$ computed at the highest-score pose and the final planned pose. The values of λ_i and γ_i are listed together with their corresponding eigenvectors. All the values, except the eigenvectors, have been rounded to 4 significant figures. The highest-score pose is rejected because $\lambda_3 < M_{\min}/3$ and $\gamma_3 < S_{\min}/3$, whereas, at the final planned pose, $\lambda_3 \geq M_{\min}/3$ and $\gamma_3 \geq S_{\min}/3$, and it is therefore accepted.

	Highest-Score Pose	**Final Planned Pose**
$\varepsilon_{\boldsymbol{\theta}}$	0.001647 radian 0.09436 degree	0.002264 radian 0.1297 degree
$M_{\min}/3$	49.03	49.03
λ_1	5,521 $[0.996, 0.002, 0.086]^{\text{T}}$	9,683 $[0.996, 0.000, 0.086]^{\text{T}}$
λ_2	38.54 $[0.002, 0.999, -0.042]^{\text{T}}$	2,219 $[0.000, 1.000, 0.000]^{\text{T}}$
λ_3	13.34 $[-0.086, 0.042, 0.995]^{\text{T}}$	364.3 $[-0.086, 0.000, 0.996]^{\text{T}}$
$S_{\min}/3$	4,520,000	2,392,000
γ_1	6,935,000 $[-0.065, 0.268, 0.961]^{\text{T}}$	37,540,000 $[-0.004, 0.997, 0.082]^{\text{T}}$
γ_2	6,483,000 $[0.018, 0.963, -0.267]^{\text{T}}$	32,280,000 $[-0.105, -0.082, 0.991]^{\text{T}}$
γ_3	48,920 $[0.998, 0.000, 0.067]^{\text{T}}$	4,403,000 $[0.995, -0.005, 0.104]^{\text{T}}$

Table 5.1: The actual values of $\varepsilon_{\boldsymbol{\theta}}$, λ_i, γ_i, $M_{\min}/3$ and $S_{\min}/3$ computed at the highest-score pose and the final planned pose. The values of λ_i and γ_i are listed together with their corresponding eigenvectors. All the values, except the eigenvectors, have been rounded to 4 significant figures.

The problem with the highest-score pose is that it is too far from the doorway and cannot acquire enough surfaces of the other room to overlap the surfaces acquired in the first scan. The only overlapping surfaces cannot constrain the translational motion along the y and z directions, and the rotational motion about the x-axis. Figure 5.8 shows the scan made at the highest-score pose. The insufficiency of the translational constraint is evident in the small values of λ_2 and λ_3. Their corresponding eigenvectors indicate the translational directions that lack constraint. Similarly, the relatively small value of γ_3 and its corresponding eigenvector indicate that the rotation about the x-axis is not sufficiently constrained.

Figure 5.9 shows the scan made at the final planned pose. We can see that the amount of overlapping surfaces with the first scan is significantly higher, especially the surfaces that can constrain the translational motion along the y and z directions, and the rotational motion about the x-axis.

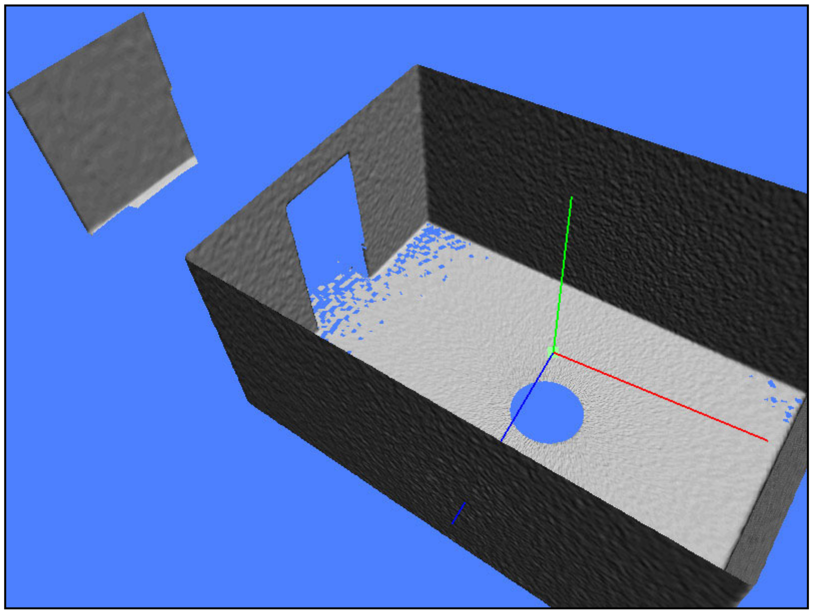


Figure 5.8: The range scan made at the highest-score pose. The scan does not acquire enough surfaces of the other room to overlap the surfaces acquired in the first scan. The only overlapping surfaces cannot constrain the translational motion along the y and z directions, and the rotational motion about the x-axis.

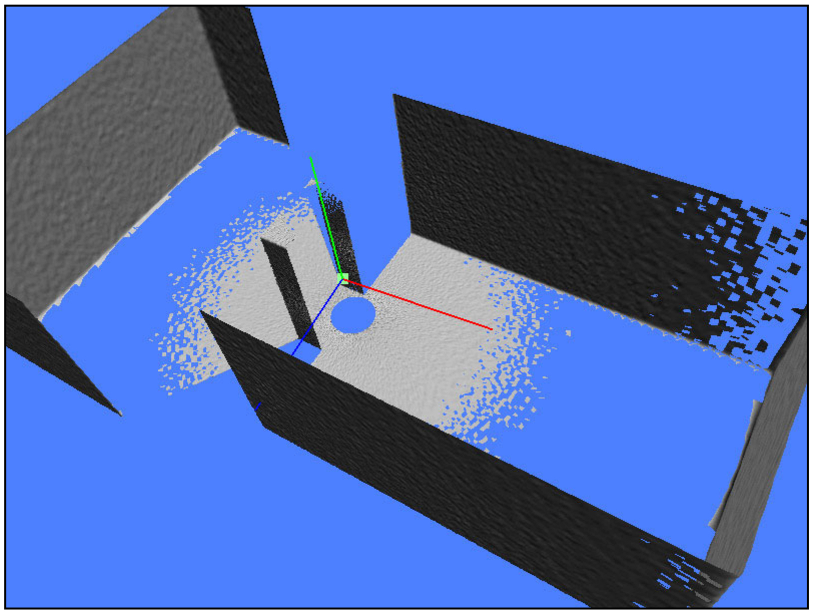


Figure 5.9: The range scan made at the final planned pose. The amount of overlapping surfaces with the first scan is significantly higher, especially the surfaces that can constrain the translational motion along the *y* and *z* directions, and the rotational motion about the *x*-axis.

Next, the scan made at each pose is registered to the volumetric partial scene model. The ICP algorithm implementation described in Section 5.2 is used. Table 5.2 shows the actual errors in the registration. All values are rounded to 3 significant figures. Figure 5.10 and Figure 5.11 show the results of the registrations.

We can see in Table 5.2 that, for the scan made at the final planned pose, the translation error in each of *x*, *y*, and *z* directions is less than the threshold $\varepsilon_{\boldsymbol{\tau}} = 0.5$ inch, while the rotation error about each of the *x*, *y*, and *z*-axis is less than the threshold $\varepsilon_{\boldsymbol{\theta}} = 0.002264$ radian. We notice that the translation error in the *y* direction is significantly larger than in the other two directions. This may be because, for the convenience of the next-best-view computation, the triangle mesh has been oriented such that the floor and the ceiling are parallel to the *x*-*z* plane. When these large floor and ceiling are voxelized, the voxel quantization errors in these voxels are almost constant and are far from being uniformly random. This can result in a much larger value of $\sigma_{\text{voxel-quant}}$ in the *y* direction. The vertical walls in the model are aligned with neither the *x*-*y* plane nor the *y*-*z* plane, therefore the voxel quantization errors in these voxels are more uniformly distributed in the *x* and *z* directions. It is not strictly necessary to have the floor and ceiling oriented to be parallel to the *x*-*z* plane, but doing it makes the implementation easier and the view planning results easier to understand. To adjust for the large constant voxel quantization errors, one may use $\sigma_{\text{voxel-quant}} = \frac{1}{2}(\text{voxel width})$, but that may make the value of e_{RMS} too large and the conditions too stringent. The large voxel quantization errors in the *y* direction has not presented problem in my experiments, and therefore I continue to use $\sigma_{\text{voxel-quant}} = (\text{voxel width})/\sqrt{12}$.

For the scan made at the highest-score pose, the ICP registration actually cannot converge correctly. We see that the translation errors in all of *x*, *y*, and *z* directions are beyond the threshold $\varepsilon_{\boldsymbol{\tau}} = 0.5$ inch, and the rotation error about the *x*-axis is beyond the threshold $\varepsilon_{\boldsymbol{\theta}} = 0.001647$ radian.

	Highest-Score Pose	**Final Planned Pose**
t_x	−1.83 inches	0.0100 inch
t_y	0.579 inch	0.260 inch
t_z	20.8 inches	−0.0138 inch
ω_x	0.00888 radian 0.509 degree	0.000373 radian 0.0214 degree
ω_y	0.000584 radian 0.0335 degree	0.000119 radian 0.00685 degree
ω_z	0.000346 radian 0.0198 degree	0.000940 radian 0.0538 degree

Table 5.2: The registration errors when the scan made at each pose is registered to the volumetric partial scene model.

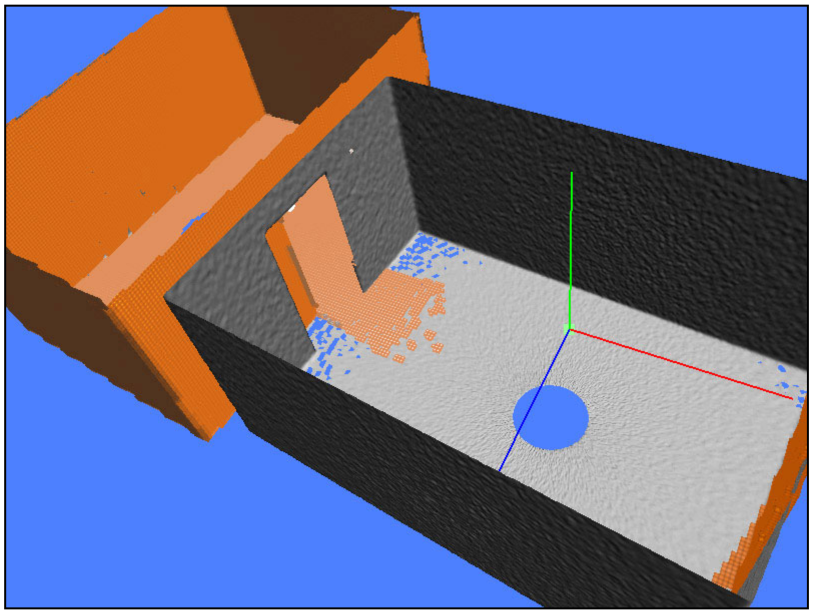


Figure 5.10: The scan made from the highest-score pose cannot be registered accurately to the volumetric partial scene model. The largest registration error is in the *z* direction, which is very evident from the misalignment of the doorway.

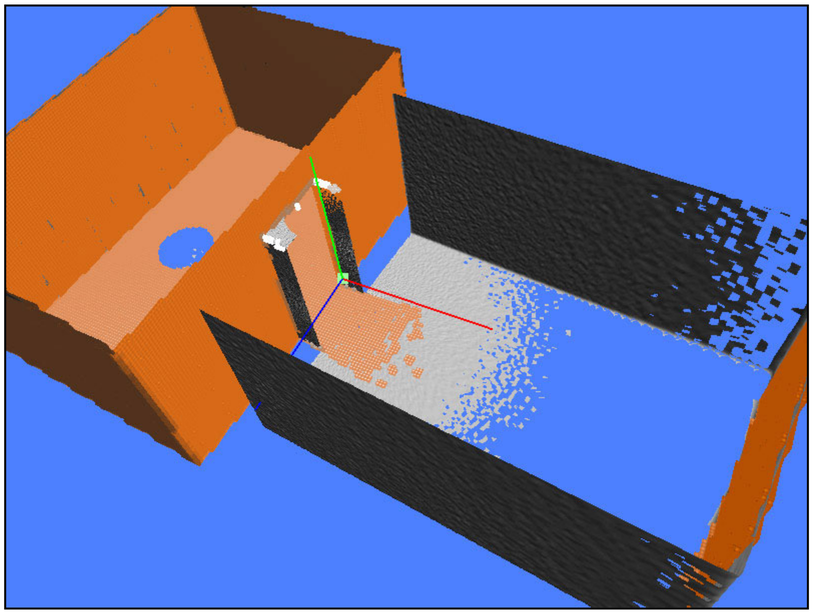


Figure 5.11: The scan made from the final planned pose can be registered accurately to within the error thresholds $\varepsilon_{\tau} = 0.5$ inch and $\varepsilon_{\theta} = 0.002264$ radian.

So, for what minimum values of ε_{τ} and ε_{θ} will the alignment error conditions still accept the final planned pose? How do these minimum threshold values compare with the actual registration errors we see in Table 5.2? For the first question, the minimum threshold values are

$$\varepsilon_{\boldsymbol{\tau},\min} \approx 0.1835 \text{ inch} \quad \text{and} \quad \varepsilon_{\boldsymbol{\theta},\min} \approx 0.001669 \text{ radian.}$$

This means that for any $\varepsilon_{\boldsymbol{\tau}} \geq \varepsilon_{\boldsymbol{\tau},\min}$ and any $\varepsilon_{\boldsymbol{\theta}} \geq \varepsilon_{\boldsymbol{\theta},\min}$, the final planned pose will satisfy both alignment error conditions. When $\varepsilon_{\boldsymbol{\theta}} = \varepsilon_{\boldsymbol{\theta},\min}$, the corresponding value of $\tau_{\text{RMS}} \approx 0.3686$ inch.

To answer the second question, we compare the value of $\varepsilon_{\boldsymbol{\tau},\min} \approx 0.1835$ to the translation errors $[t_x, t_y, t_z]^{\text{T}}$ = $[0.0100, 0.260, -0.0138]^{\text{T}}$, and the value of $\varepsilon_{\boldsymbol{\theta},\min} \approx 0.001669$ to the rotation errors $[\omega_x, \omega_y, \omega_z]^{\text{T}} = [0.000373, 0.000119, 0.000940]^{\text{T}}$. Since the relatively large value of t_y, in comparison to t_x and t_z, is caused by large constant voxel quantization errors in the y direction, it can be treated as an exception and will not be considered. Then, by comparing $\varepsilon_{\boldsymbol{\tau},\min}$ to t_x and t_z, it seems that the translational alignment error condition is quite conservative, because $\varepsilon_{\boldsymbol{\tau},\min} >> t_x$ and $\varepsilon_{\boldsymbol{\tau},\min} >> t_z$. As for the rotational alignment error condition, since $\varepsilon_{\boldsymbol{\theta},\min}$ is not much larger than ω_x, ω_y and ω_z, the condition seems relatively less conservative.

For the highest-score pose, the minimum values of $\varepsilon_{\boldsymbol{\tau}}$ and $\varepsilon_{\boldsymbol{\theta}}$ where the alignment error conditions will be satisfied are

$$\varepsilon_{\boldsymbol{\tau},\min} \approx 0.9585 \text{ inch} \quad \text{and} \quad \varepsilon_{\boldsymbol{\theta},\min} \approx 0.01584 \text{ radian.}$$

However, referring to the actual translation errors in Table 5.2, where $[t_x, t_y, t_z]^{\text{T}} = [-1.83,\ 0.579,\ 20.8]^{\text{T}}$, the value $\varepsilon_{\boldsymbol{\tau},\min} \approx 0.9585$ seems to be much too small. This is because the computed values of λ_2 and λ_3 are relatively much larger than they should be. The main cause of this is the errors in the estimated surface normals at the range samples. These errors add false constraint in the y and z directions, and since λ_2 and λ_3 have small absolute values, their relative errors become large. The smaller the value of λ_i with respect to λ_1, the more sensitive it is to errors in the surface normals. Therefore, to avoid accepting such views, we may add the condition that the ratio λ_3 / λ_1 must not be less than a threshold. In practice, a threshold ratio of 0.01 works in most cases.

It is not quite appropriate to compare $\varepsilon_{\boldsymbol{\theta},\min}$ with the rotation errors in Table 5.2 because the large translation error may have caused them to be far from correct. Similar to the problem with λ_i, the values of γ_i are also affected by errors in the surface normals. Therefore, we may also add the condition that the ratio γ_3 / γ_1 must not be less than a threshold. Again, a threshold ratio of 0.01 works well in practice.

Chapter 6

Results

Three sets of example results are presented to demonstrate the use of the next-best-view planning system for the range acquisition of indoor environments. The first two examples use simulated scanners to scan synthetic digital models of a midsize living room and a kitchen, respectively, while the third example uses the DeltaSphere-3000 3D Scene Digitizer to scan a part of an office building floor. The third example shows that the next-best-view planner is useful, practical and robust for real-world environments, where there are complicating factors such as noise, outliers, and drop-outs.

The scanners used in the three examples are assumed to have only 3D translational poses, and they both have full horizontal field of view but limited vertical field of view. In all the examples, the view evaluation step is limited to only two minutes in every cycle of the acquisition process. The importance values of the surface types, $w(p)$ in Equation (3.2), and $w(P)$ in Equation (4.5), are all assumed to be 1.

Next, Section 6.4 compares the performances of four different view plans for the kitchen model. The view plans are the results of using four arbitrarily selected and different first views.

All the timings and results were obtained on a laptop computer with an Intel Pentium 4-M 1.6GHz CPU, and 768 MB of DDR RAM.

6.1 Simulated Scans (Example 1)

In this example, a simulated range scanner is used to acquire the geometric information of the interior of a synthetic midsize living room. The living room is represented as a polygonal model. It has approximately 630 square feet of floor area, and the floor is level and flat. The largest corner-to-corner distance in the room is about 527 inches. The ceiling is about 90 inches (7.5 feet) from the floor. All the doors to the living room are closed, and its interior is totally enclosed by opaque surfaces. A top view of the living room is shown in Figure 6.1(a), in which the ceiling is not shown.

The simulated range scanner is similar to a DeltaSphere-3000 3D Scene Digitizer. It is assumed to have only 3D translational poses, a full horizontal field of view and a limited vertical field of view. Table 6.1 lists the parameter values of the scanner and the values of some other parameters used in the experiment. Gaussian noise, with standard deviation of σ_{range}, is added to the range measurements.

For the first scan of the room, the scanner is arbitrarily positioned as shown in Figure 6.1(a) (pointed by the yellow arrow). The scanner pose has been randomly perturbed, so its local vertical axis is not exactly parallel to the vertical of the living room. Figure 6.1(b) shows the partial octree model and the feasible view volumes created from the first scan. Figure 6.1(c) shows the under-sampled true surface patches extracted from the partial octree model.

Figure 6.2 shows the computed pose for the next scan. This pose is the highest-score view computed by the view evaluation step, and it satisfies the alignment error conditions. The view evaluation step is allowed to take only two minutes. Before the second scan is made, the computed pose is randomly perturbed to simulate pose error in the positioning of the scanner. The scanner is positioned at a computed pose with an RMS translational error of 8 inches in each of the *x*, *y*, and *z* directions, and a RMS rotational error of 5 degrees about each of the *x*-axis and *z*-axis, and a RMS rotational error of 10 degrees about the *y*-axis (vertical axis).

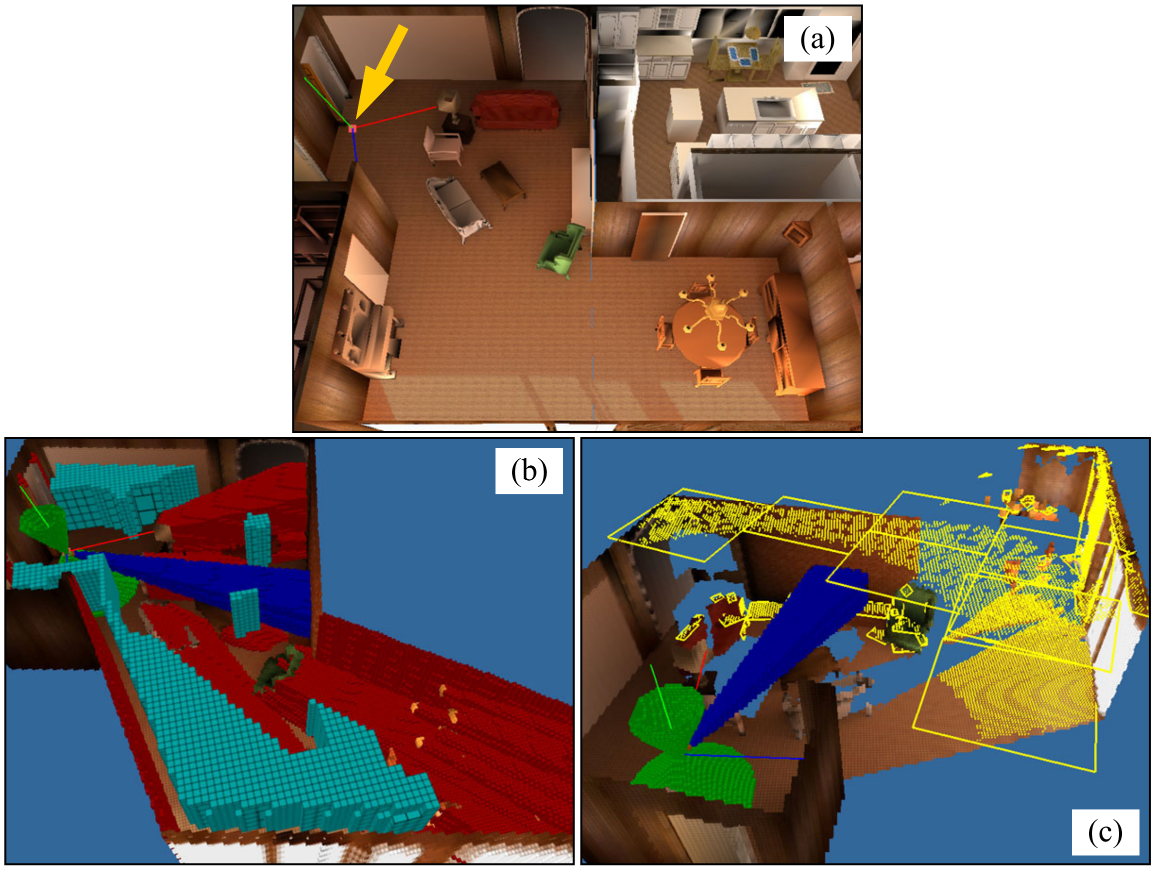


Figure 6.1: (a) A view of the living room model. The yellow arrow points to the scanner pose for the first scan. (b) The partial octree model and the feasible view volumes (light-blue) created from the first scan. (c) The under-sampled true surface patches are shown in yellow.

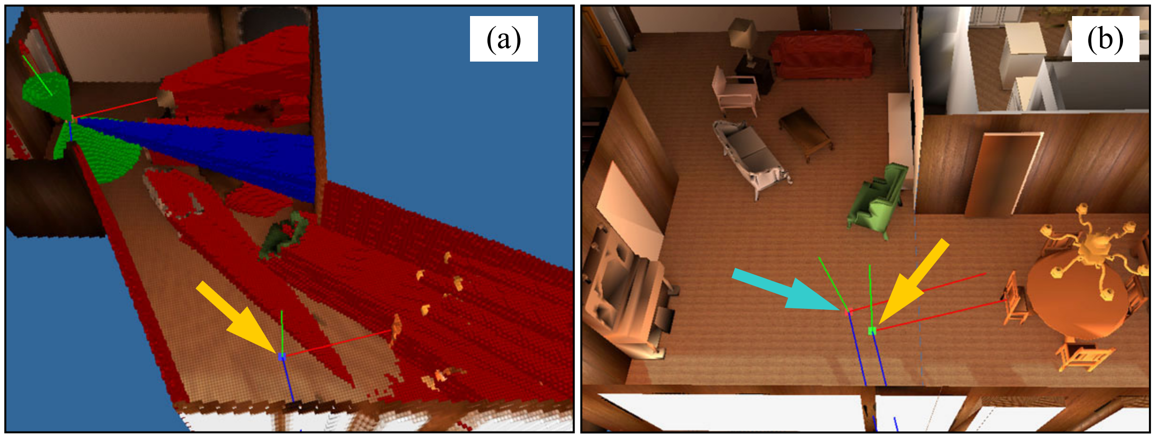


Figure 6.2: (a) The yellow arrow points to the computed pose for the second scan. (b) The light-blue arrow points to the scanner pose that is obtained by randomly perturbing the computed pose (pointed by the yellow arrow).

Parameter	**Value**
Scanner angular sampling density, $1/\alpha$	10 samples/degree
Scanner maximum range, $R_{\max}$	480 inches = 40 feet
Scanner vertical field of view	–55° to 70°
Scanner minimum clearance distance	12 inches
Scanner minimum height from floor	30 inches = 2.5 feet
Scanner maximum height from floor	78 inches = 6.5 feet
Surface sampling density requirement, D	1 sample/inch^2
Angle-of-incidence threshold, $\phi_{\max}$	70°
Surface voxel width	2 inches
Smallest viewcell width	4 inches
Smallest patch element length	≤ 4 inches
Relative representative error bound on $s(V,P)$, ε_s	15%
Range precision, σ_{range}	0.3 inch
Confidence interval for alignment error conditions, $(1-\alpha')^3 100\%$	99% $\Rightarrow$ ($z_{\alpha'/2} \approx 2.712$)
Translation error tolerance, $\varepsilon_{\boldsymbol{\tau}}$	1 inch
Maximum RMS value of the rotation-induced translation errors, τ_{RMS}	1 inch
Minimum threshold for λ_3 / λ_1	0.05
Minimum threshold for γ_3 / γ_1	0.01

Table 6.1: Some parameter values used in the first simulated-scan example.

The pose error results in misalignment between the surfaces in the partial octree model and the triangle mesh made from the second scan (see Figure 6.3(a)). The ICP registration algorithm is then applied to the surfaces to obtain the proper alignment shown in Figure 6.3(b).

Figure 6.4 shows the partial octree model and the feasible view volumes after the second scan has been merged with the first one. In comparison to Figure 6.1(b), one can see that the number of false surfaces have been reduced considerably and the feasible view volumes have increased in total volume.

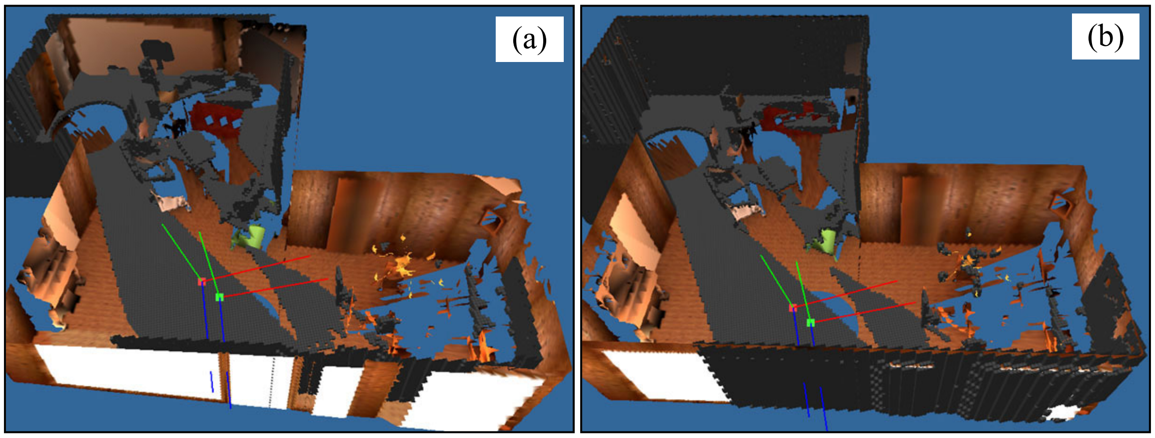


Figure 6.3: (a) Misalignment between the surfaces in the partial octree model and the triangle mesh made from the second scan. (b) The surfaces are now properly aligned after the registration step.

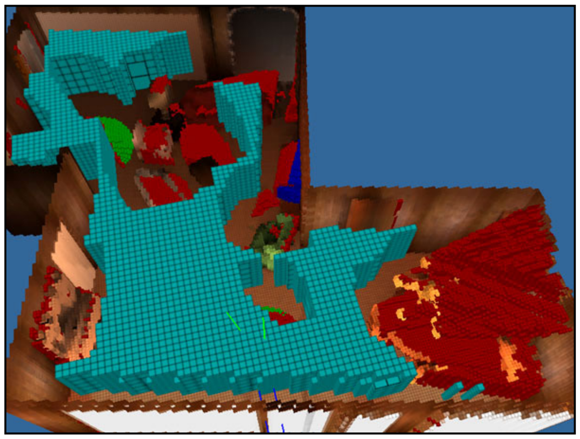


Figure 6.4: The partial octree model and the feasible view volumes (light-blue) after the second scan is merged.

This acquisition process is manually terminated after the eighth scan has been made. Figure 6.5 shows the unperturbed pose of the first scan, and the computed poses for the subsequent seven scans. Note that the eight scanner positions may not be at the same height.

The final partial octree model and feasible view volumes are shown in Figure 6.6. The living room is still not completely acquired, as one can see that there are still false surface voxels in the partial model.

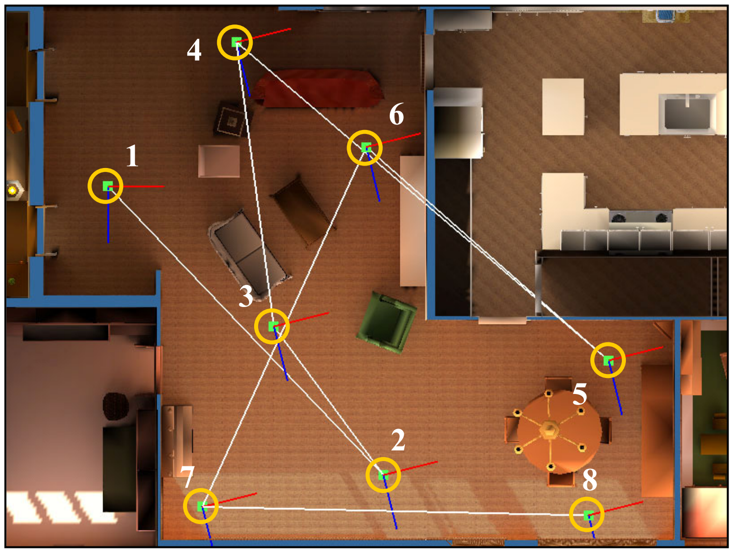


Figure 6.5: The unperturbed pose of the first scan, and the computed poses for the subsequent seven scans.

The graph in Figure 6.7(a) shows the percentage of each type of surface voxels in the partial octree model after each scan is merged. After the eighth scan, approximately 9% of the surface voxels in the partial octree model are false surface voxels, approximately 0.3% are under-sampled true surface voxels, and approximately 91% satisfy the sampling density requirement. The rate of increase of the proportion of satisfactory surface voxels diminishes with each additional scan. In general, this proportion is not always increasing. For example, if a completely new room is "discovered" later in the acquisition process, then there will be a large increase in the number of occlusion surface voxels and the proportion of satisfactory surface voxels may decrease.

The graph in Figure 6.7(b) shows the rate of increase of the total volume of the feasible view volumes. This graph is always non-decreasing.

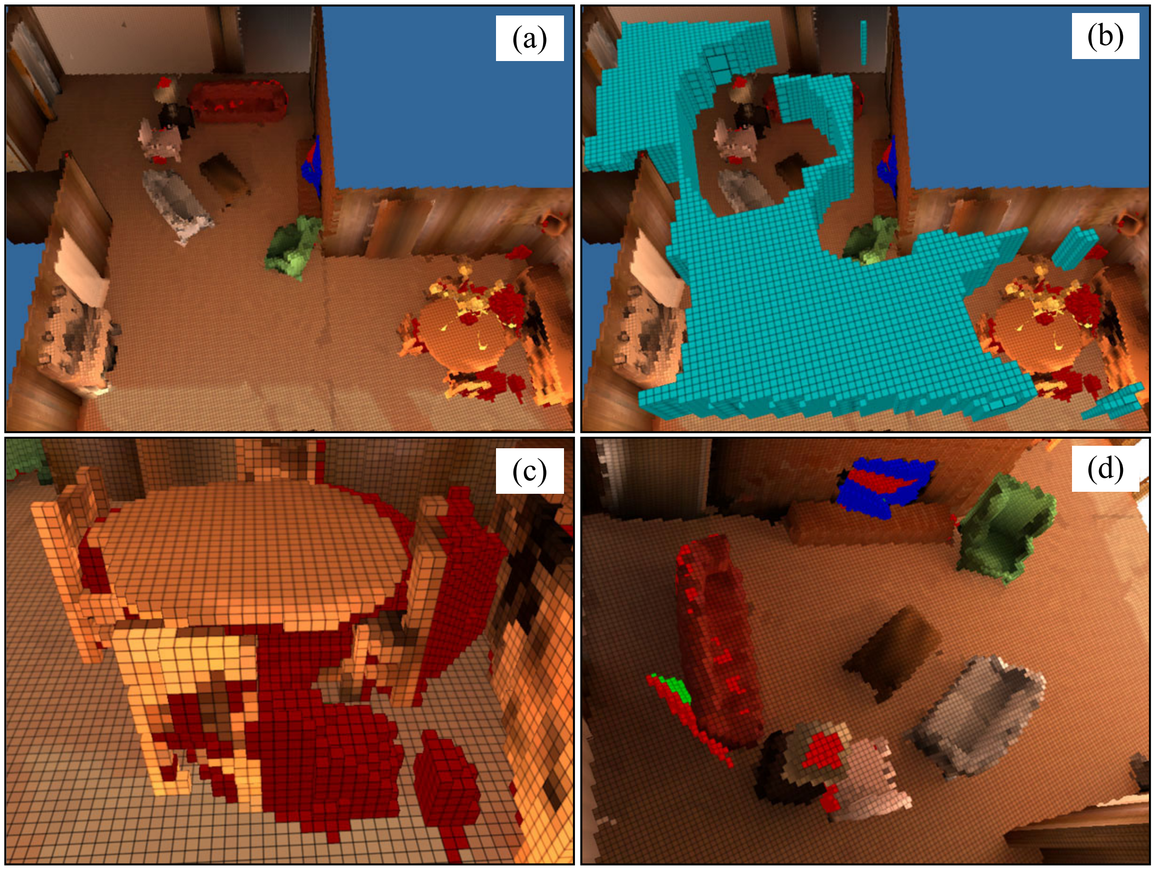


Figure 6.6: The final partial octree model and the final feasible view volumes.

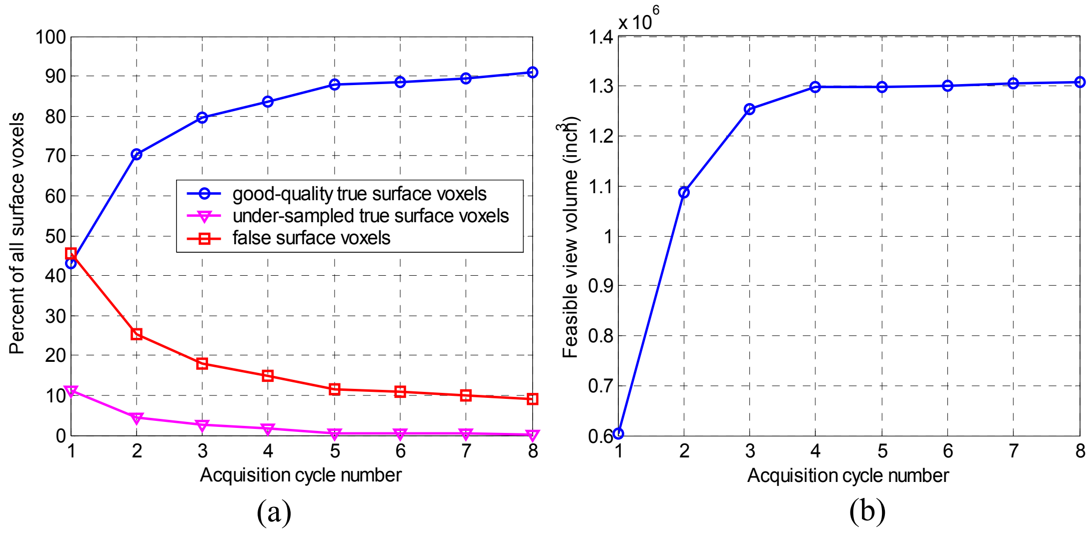


Figure 6.7: (a) The proportions of the different types of surface voxels in the partial models after each scan is merged. (b) The total volume of the feasible view volumes.

Figure 6.8 shows the triangle-mesh model of the living room constructed from the eight range images. A commercial software package, PolyWorks 8 by InnovMetric Software [InnovMetric], was used to reconstruct the model. Due to insufficient system memory for the reconstruction computation, all the eight range images were downsampled to half the resolution in each direction.

In Figure 6.8(b) and (c), it appears that the holes in the reconstructed model are more numerous than those indicated by the false surface voxels in Figure 6.6(c) and (d). There are two reasons. The first is that in the next-best-view planning system, each input range image is preprocessed to fill small holes caused by drop-outs. This speculative hole-filling is not performed in PolyWorks 8. The second reason is that the true surface voxels have higher precedence over the false surface voxels, and a voxel that intersects both false and true surfaces will be labeled a true surface voxel.

Table 6.2 shows the average execution time taken by each main step of the acquisition cycle. The view evaluation is limited to approximately two minutes. The graph in Figure 6.9(a) shows how much of the total patch area that has been evaluated by the view evaluation step in the two minutes. Besides the first two acquisition cycles, the view evaluation step was able to evaluate more than 80% of the total patch area. Figure 6.9(b) shows the total patch sampling deficit in each cycle, and how much of the sampling deficit has been evaluated.

Steps	**Average time (sec)**	**Percentage**
Range image processing	49.1	24.5
Surface registration	6.4	3.2
Partial octree model creation and merging	21.9	10.9
Feasible view octree creation	0.8	0.4
Patch extraction and ranking	1.4	0.7
View evaluation	120.4	60.0
Registration analysis	0.5	0.2
Total:	200.5	100.0

Table 6.2: The average execution time of each main step.

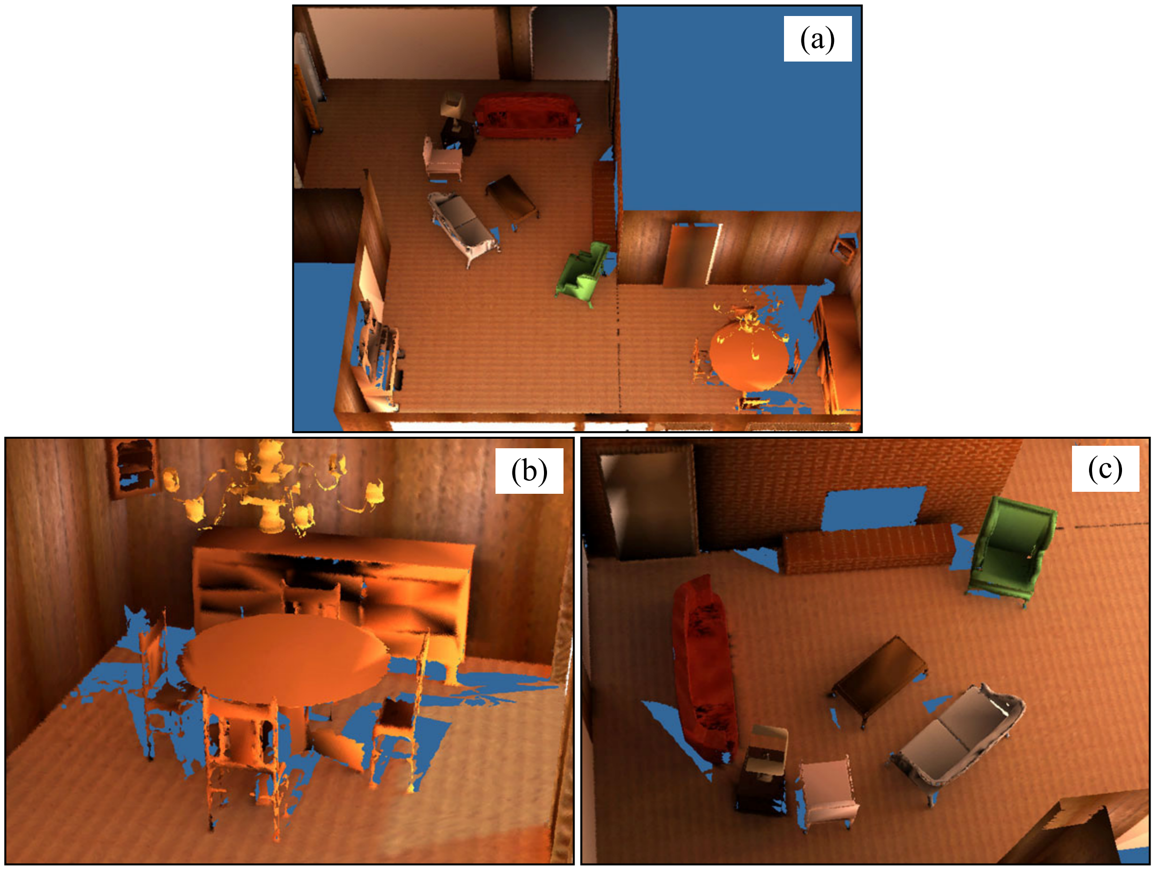


Figure 6.8: A triangle-mesh model of the living room reconstructed from the eight range images.

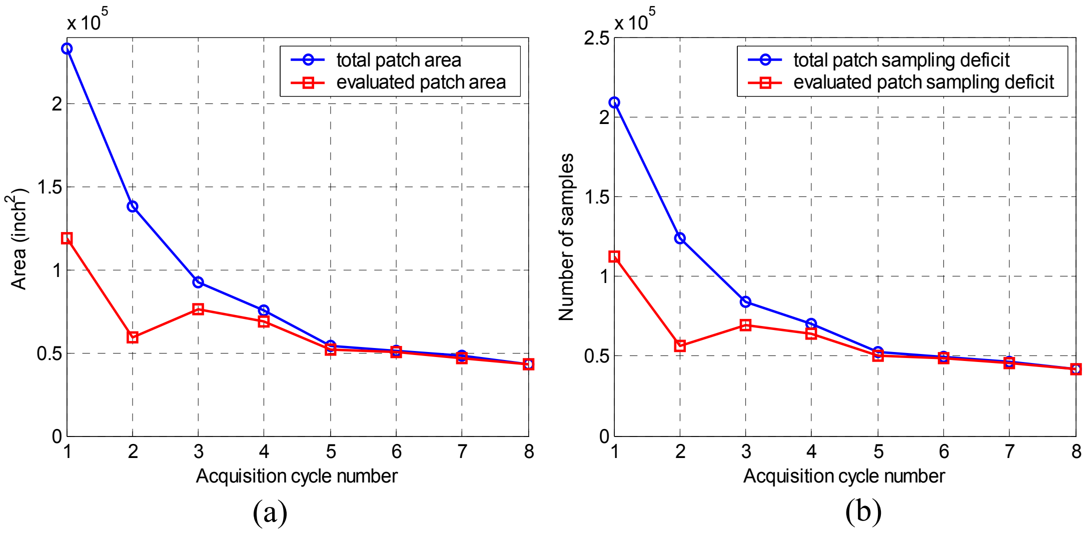


Figure 6.9: (a) The total patch area in the partial octree model after each scan has been merged, and the amount that is evaluated in the view evaluation step. (b) The total patch sampling deficit and the amount that is evaluated in the view evaluation step.

6.2 Simulated Scans (Example 2)

In the second example, a similar simulated range scanner is used to acquire the geometric information of the interior of a synthetic model of a kitchen. The kitchen is represented as a polygonal model. A top view of the kitchen is shown in Figure 6.10(a), in which the ceiling is not shown. The yellow line marks the approximate boundary of the space accessible to the scanner. The main kitchen area is connected to three small storage rooms (surrounded by green lines), and together they have approximately 350 square feet of floor area, and the floor is level and flat. The largest corner-to-corner distance in the kitchen model is about 390 inches. The ceiling is about 93 inches (7.75 feet) from the floor. All the other doors to the kitchen are closed, and its interior is totally enclosed by opaque surfaces. Figure 10(b) shows another view of the kitchen.

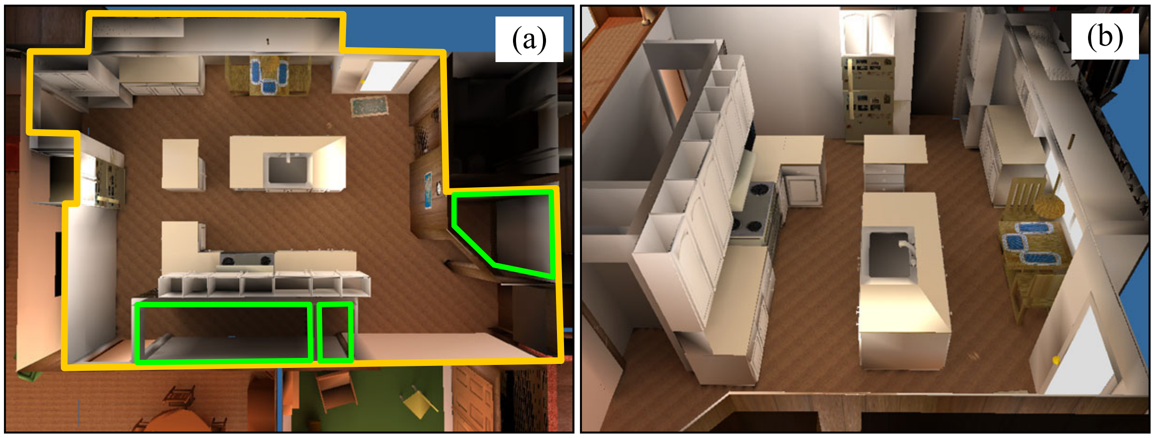


Figure 6.10: (a) A top view of the kitchen model. The yellow line marks the approximate boundary of the space accessible to the scanner, and the green lines mark the space of the three small storage rooms. (b) Another view of the kitchen model.

Similar to the first example, the simulated range scanner is assumed to have only 3D translational poses, a full horizontal field of view and a limited vertical field of view. Table 6.3 lists the parameter values of the scanner and the values of some other parameters used in the experiment.

Parameter	Value
Scanner angular sampling density, $1/\alpha$	5 samples/degree
Scanner maximum range, $R_{\max}$	600 inches = 50 feet
Scanner vertical field of view	−75° to 75°
Scanner minimum clearance distance	6 inches
Scanner minimum height from floor	24 inches = 2 feet
Scanner maximum height from floor	84 inches = 7 feet
Surface sampling density requirement, D	2 sample/inch^2
Angle-of-incidence threshold, $\phi_{\max}$	70°
Surface voxel width	2 inches
Smallest viewcell width	4 inches
Smallest patch element length	≤ 4 inches
Relative representative error bound on $s(V,P)$, ε_s	12.5%
Range precision, σ_{range}	0.3 inch
Confidence interval for alignment error conditions, $(1-\alpha')^3 100\%$	99% ⇒ ($z_{\alpha'/2} \approx 2.712$)
Translation error tolerance, $\varepsilon_{\boldsymbol{\tau}}$	1 inch
Maximum RMS value of the rotation-induced translation errors, τ_{RMS}	1 inch
Minimum threshold for λ_3 / λ_1	0.01
Minimum threshold for γ_3 / γ_1	0.01

Table 6.3: Some parameter values used in the second simulated-scan example.

The first view is arbitrarily positioned near the center of the kitchen, and altogether, ten scans are made in this simulation. The acquisition process is manually terminated after the tenth scan has been made. In each acquisition cycle, the view evaluation is limited to approximately two minutes. Figure 6.11 shows the evolution of the partial octree model as each scan is made and merged. The unperturbed poses of the ten scans are shown in Figure

6.11(j). Note that the ten scanner positions may not be at the same height. The final feasible view volumes and the final partial octree model are shown in Figure 6.12. Figure 6.13 shows the triangle-mesh model of the kitchen constructed from the ten range images. PolyWorks 8 was used to reconstruct the model.

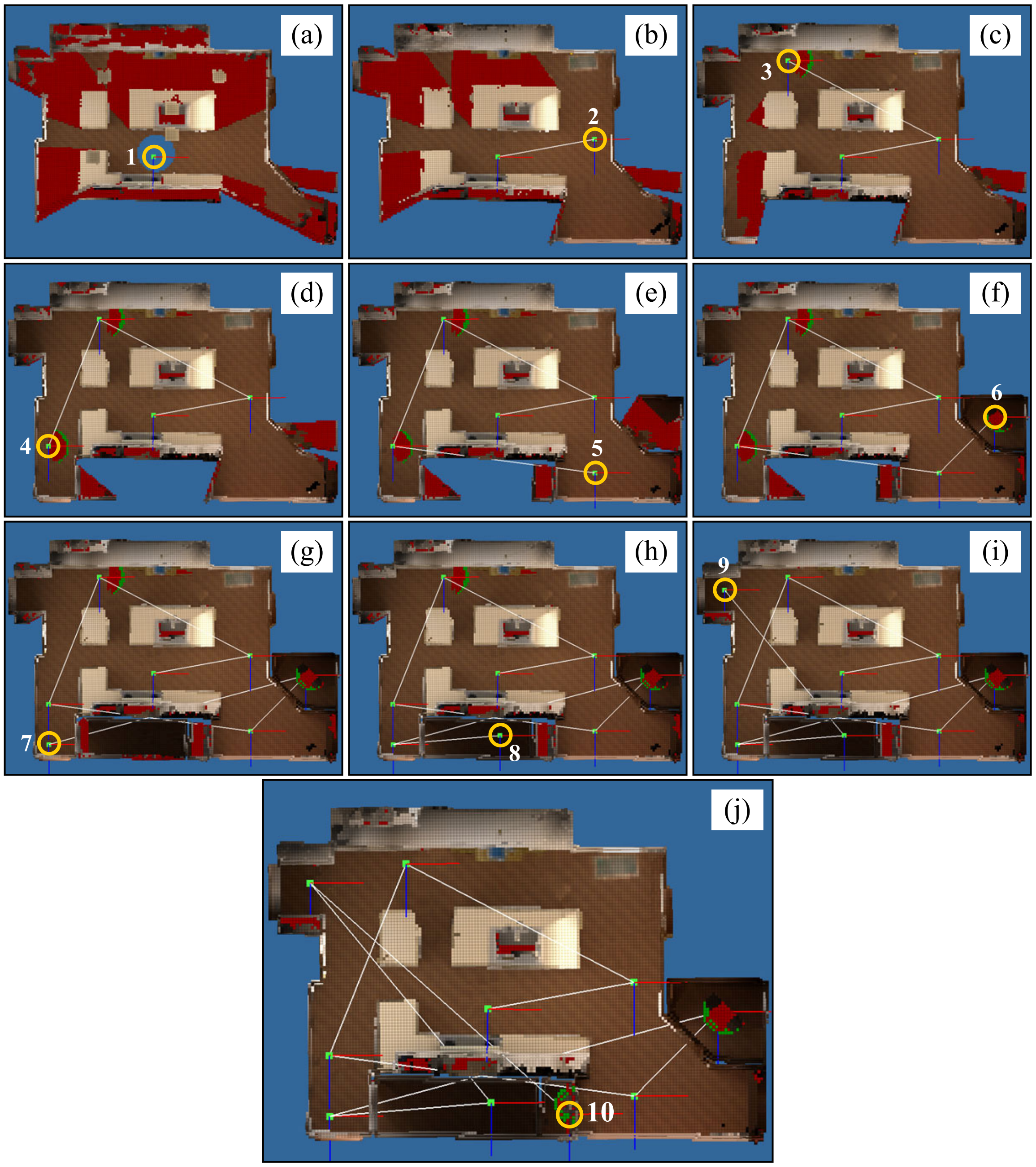


Figure 6.11: The evolution of the partial model as each scan is made and merged. The last image (j) shows a top view of the final partial model and the ten unperturbed poses in the view plan.

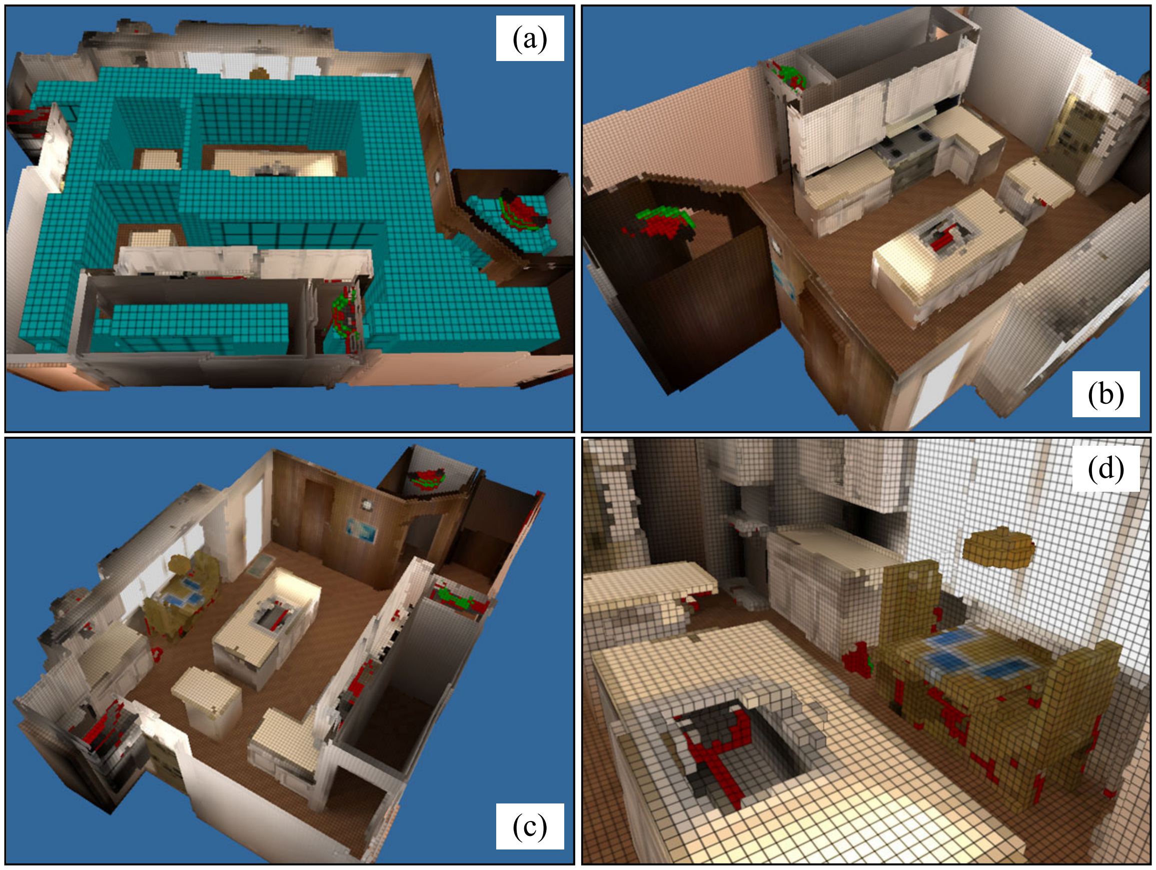


Figure 6.12: (a) The final feasible view volumes. (b), (c) and (d) Different views of the final partial octree model.

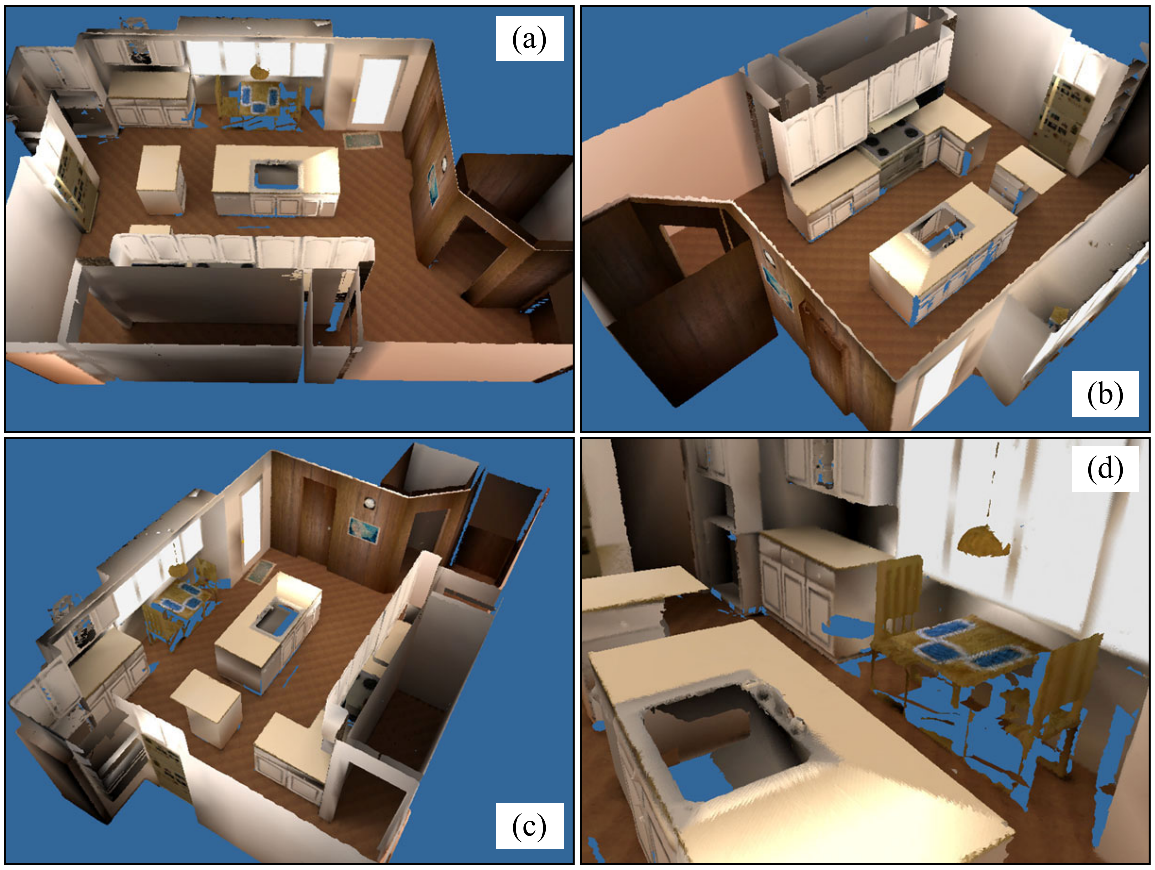


Figure 6.13: Four different views of a triangle-mesh model of the kitchen reconstructed from the ten range images.

The graph in Figure 6.14(a) shows the percentage of each type of surface voxels in the partial octree model after each scan is merged. After the tenth scan, approximately 3% of the surface voxels in the partial octree model are false surface voxels, approximately 2% are under-sampled true surface voxels, and approximately 95% satisfy the sampling density requirement. The rate of increase of the proportion of satisfactory surface voxels slows down with each additional scan. There is a slight decrease in the proportion of satisfactory surface voxels after the seventh scan is merged. This is caused by the "discovery" of the largest small room, which suddenly increases the total under-sampled true surface area.

The graph in Figure 6.14(b) shows the rate of increase of the total volume of the feasible view volumes. This graph is always non-decreasing. A sudden increase in feasible view volume can be seen after the seventh scan is merged. Again, this is due to the "discovery" of the largest small room.

Table 6.4 shows the average execution time taken by each main step of the acquisition cycle. The view evaluation is limited to at most two minutes; however, due to the smaller size of the environment, the actual time required for view evaluation is often less than the allowable two minutes. This can be observed in the graph in Figure 6.15(a), which shows how much of the total patch area has been evaluated by the view evaluation step. For the fifth and the subsequent scans, the system is able to evaluate 100% of the patches in less than two minutes. Figure 6.15(b) shows the total patch sampling deficit in each cycle, and how much of the sampling deficit has been evaluated.

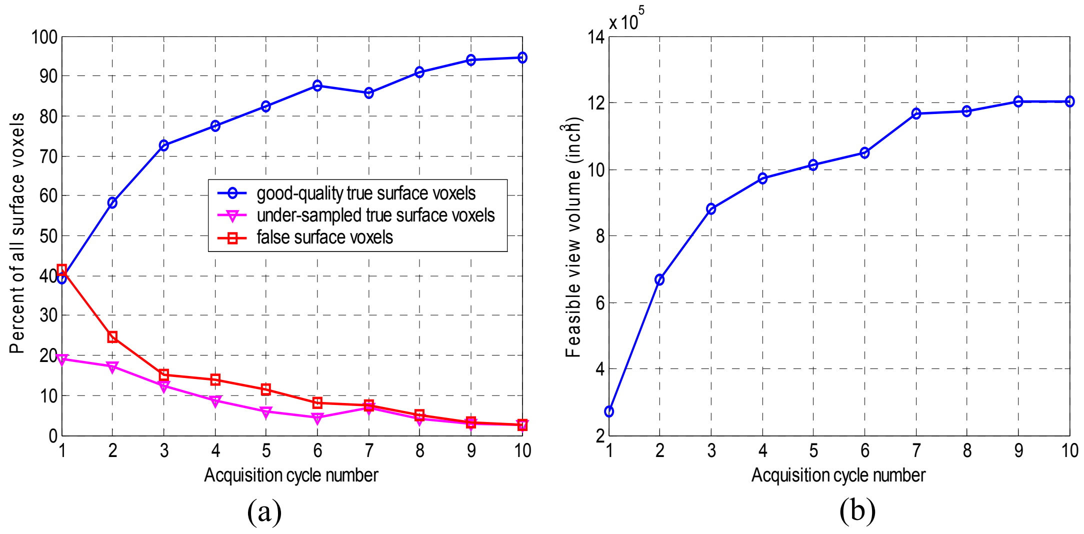


Figure 6.14: (a) The proportions of the different types of surface voxels in the partial models after each scan is merged. (b) The total volume of the feasible view volumes.

Steps	Average time (sec)	Percentage
Range image processing	13.6	9.6
Surface registration	7.9	5.6
Partial octree model creation and merging	13.9	9.8
Feasible view octree creation	0.7	0.5
Patch extraction and ranking	0.8	0.5
View evaluation	104.7	73.8
Registration analysis	0.2	0.2
Total:	141.7	100.0

Table 6.4: The average execution time of each main step.

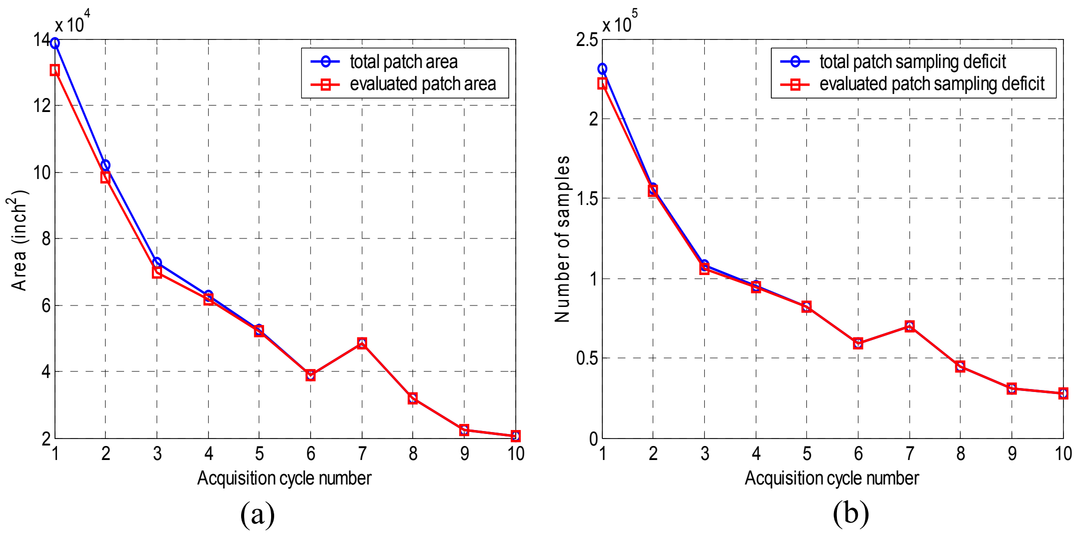


Figure 6.15: (a) The total patch area in the partial octree model after each scan has been merged, and the amount that is evaluated in the view evaluation step. (b) The total patch sampling deficit and the amount that is evaluated in the view evaluation step.

6.3 Real Scans

The purpose of this experiment is to demonstrate that the next-best-view planning system is sufficiently robust for real-world applications. Real-world indoor environments usually have higher geometric complexity than synthetic models do, and the range images of them normally contain much more noise, and many more outliers and drop-outs than in the simulated case.

The room that is used in the experiment is part of an office floor. Two photographs of the room are shown in Figure 6.16. The room contains reflective objects, such as glass on the doors and cabinet, that can cause outliers and drop-outs in the range images. The room is not totally enclosed by opaque surfaces—there are small glass windows on the doors that allow one to see through into the other rooms. The room has approximately 600 square feet of floor area, and the floor is fairly level and flat. The largest corner-to-corner distance in the room is about 680 inches. The ceiling is about 108 inches (9 feet) from the floor.

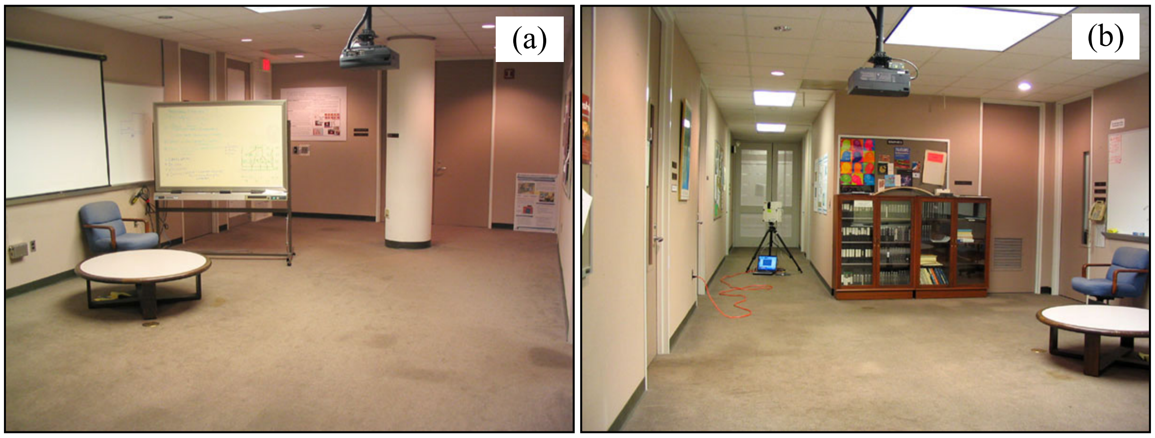


Figure 6.16: Two views of the room that was used in the experiment. It is part of an office floor.

The range scanner used is the DeltaSphere-3000 3D Scene Digitizer. Table 6.5 lists the parameter values of the scanner and the values of some other parameters used in the experiment. At 10-samples-per-degree scanning resolution, the DeltaSphere-3000 takes about eight minutes to make a full panoramic scan.

Parameter	**Value**
Scanner angular sampling density, $1/\alpha$	10 samples/degree
Scanner maximum range, $R_{\max}$	480 inches = 40 feet
Scanner vertical field of view	–55° to 70°
Scanner minimum clearance distance	12 inches
Scanner minimum height from floor	40 inches = 3.33 feet
Scanner maximum height from floor	84 inches = 7 feet
Surface sampling density requirement, D	4 sample/inch^2
Angle-of-incidence threshold, $\phi_{\max}$	70°
Surface voxel width	2 inches
Smallest viewcell width	4 inches
Smallest patch element length	≤ 4 inches
Relative representative error bound on $s(V,P)$, ε_s	15%
Range precision, σ_{range}	0.3 inch
Confidence interval for alignment error conditions, $(1-\alpha')^3 100\%$	99% $\Rightarrow$ ($z_{\alpha'/2} \approx 2.712$)
Translation error tolerance, $\varepsilon_{\boldsymbol{\tau}}$	1 inch
Maximum RMS value of the rotation-induced translation errors, τ_{RMS}	1 inch
Minimum threshold for λ_3 / λ_1	0.05
Minimum threshold for γ_3 / γ_1	0.01

Table 6.5: Some parameter values used in the real-scan example.

For the first scan of the room, the scanner is positioned as shown in Figure 6.17(a) (pointed to by the yellow arrow). The model in Figure 6.17(a) is the triangle mesh created from the first range image. Figure 6.17(b) shows the partial octree model and Figure 6.17(c) the feasible view volumes created from the first scan. Figure 6.17(d) shows the under-sampled true surface patches extracted from the partial octree model.

In the partial octree model in Figure 6.17(b), one can observe that there are a large number of erroneous occlusion surfaces, originating fairly close to the scanner's viewpoint. These erroneous occlusion surfaces are caused by outliers that appear to be very close to the scanner's viewpoint, and they can be clearly seen in Figure 6.17(c) as some very dark and isolated voxels around the scanner's viewpoint. As a result of these erroneous occlusion surfaces, the empty space becomes very fragmented and there is very little contiguous space for the feasible view volumes.

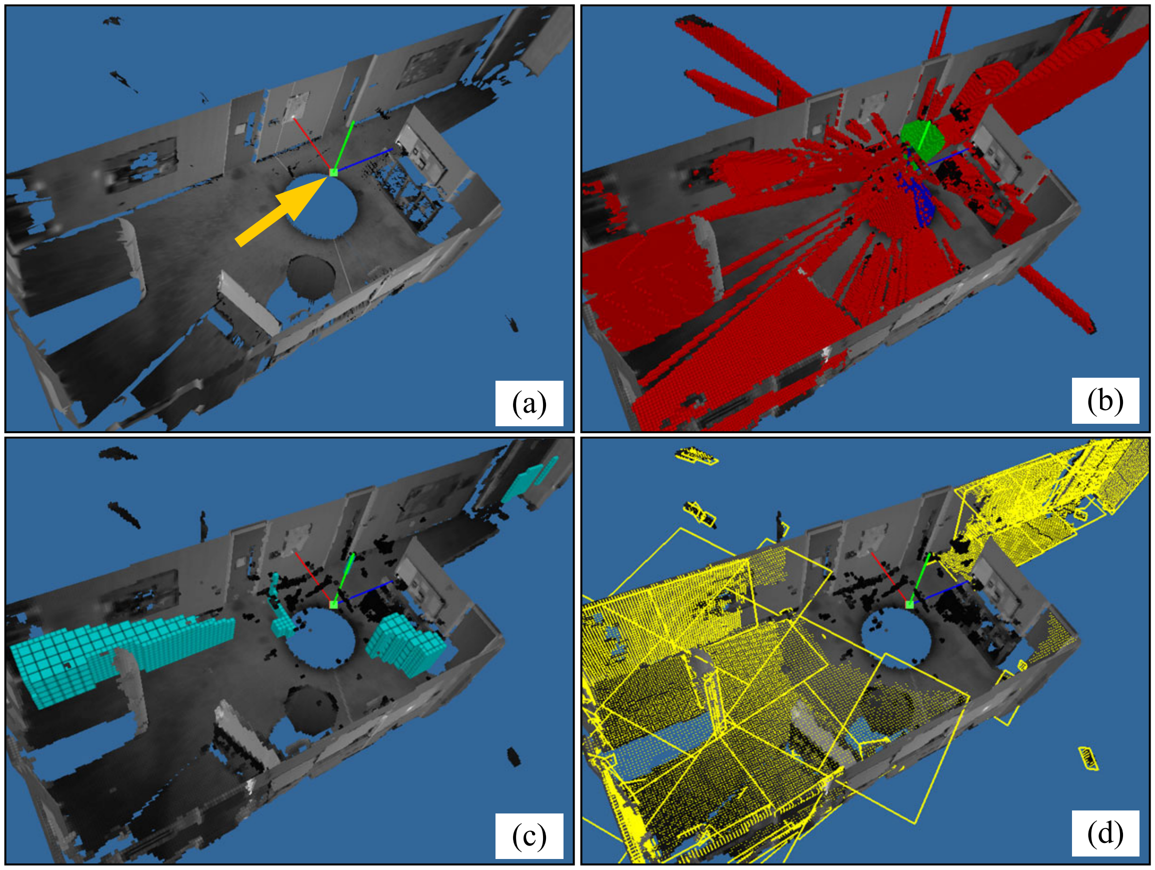


Figure 6.17: (a) The triangle mesh created from the range image acquired by the first scan. The yellow arrow points to the pose of the scanner from which the range image is acquired. The ceiling is not shown. (b) The partial octree model. (c) The feasible view volumes (light-blue). (d) The under-sampled true surface patches are shown in yellow.

As long as the set of feasible view volumes is not empty, these outliers do not pose much of a problem, since most of them will be eliminated when subsequent scans are merged with the partial octree model. This is because empty-space has higher precedence than any other type of surface. Figure 6.18(a) shows the partial octree model after the second scan is merged. The scanner pose from which the second scan is made is indicated by the yellow arrow. This pose is computed by the view evaluation step. One can see that many of the erroneous occlusion surface voxels have been eliminated, and the result is a much larger total volume in the feasible view volumes (see Figure 6.18(b)).

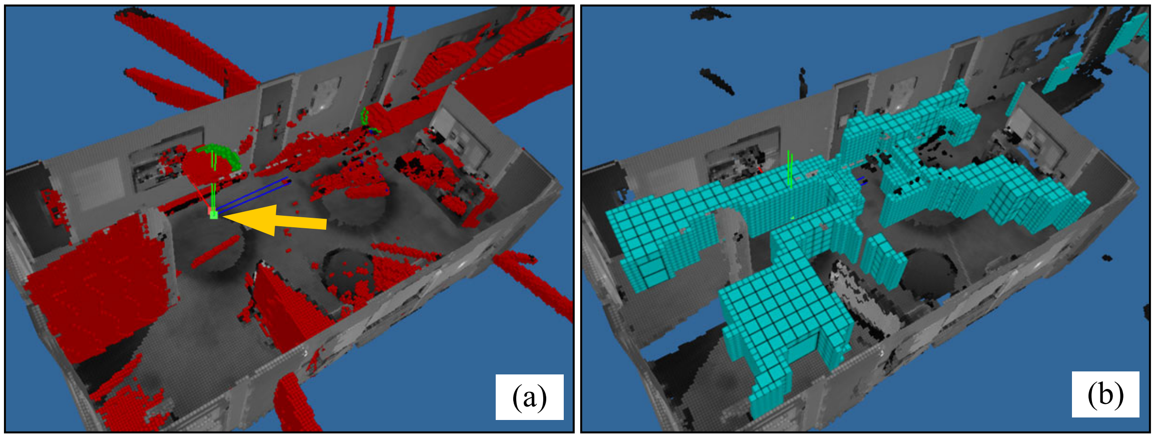


Figure 6.18: The partial octree model and feasible view volumes after the second scan is merged.

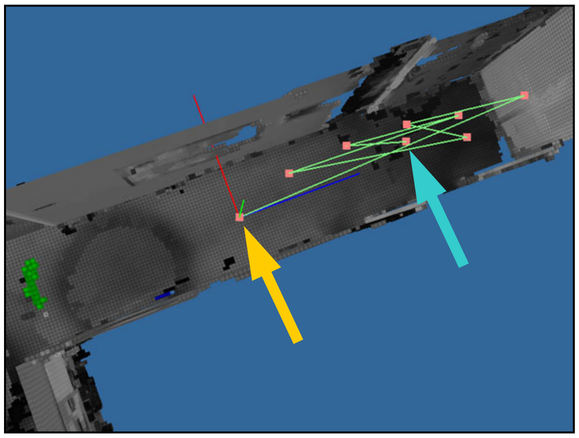


Figure 6.19: Eight candidate views are checked for the alignment error conditions when selecting the best view for the fifth scan. The light-blue arrow points to the overall highest-score view, and the yellow arrow point to the view that satisfies the alignment error conditions. Each view is checked for the alignment error conditions only if it is at least three feet away from all other views that have been checked.

In each cycle, the view evaluation is allowed only approximately two minutes. After this, the candidate views are checked for the alignment error conditions. Figure 6.19 shows that eight candidate views were checked when selecting the best view for the fifth scan. The light-blue arrow points to the overall highest-score view, and the yellow arrow points to the view that satisfies the alignment error conditions. Each view is checked for the alignment error conditions only if it is at least three feet away from all other views that have been checked.

After a new pose is computed for the next scan, the scanner is manually moved to the new location, using the graphical display from the system as a guide to the approximate position and orientation. The system also displays the exact height of the new position from the floor, and a measuring tape is used to check the actual physical height. The positioning typically takes two to five minutes for each scan. From the results of the surface registration, the positioning errors were all found to be less than five inches in each of the three orthogonal directions, and the orientation errors were all within two degrees from the three orthogonal axes.

The acquisition process was manually terminated after the fifth scan was made. Figure 6.20(a) shows the pose of the first scan, and the computed poses for the subsequent four scans. The final partial octree model and feasible view volumes are shown in Figure 6.20(b). The room is still not completely acquired, as one can see that there are still false surface voxels in the partial model. The graph in Figure 6.21(a) shows the percentage of each type of surface voxels in the partial octree model after each scan is merged. After the fifth scan, approximately 22% of the surface voxels in the partial octree model are false surface voxels, approximately 8% are under-sampled true surface voxels, and approximately 70% satisfy the sampling density requirement. The large proportion of occlusion surface voxels is partly the result of the scanner "seeing" through the small windows on the doors into the other rooms. These occlusion surfaces will never be eliminated because the openings into the other rooms are too small and the view planning system is unable to grow the feasible view volumes into the other rooms.

The graph in Figure 6.21(b) shows the rate of increase of the total volume of the feasible view volumes. The increase in volume is the greatest just after the second scan is merged to the partial octree model, as this is when most of the erroneous occlusion surface voxels are eliminated.

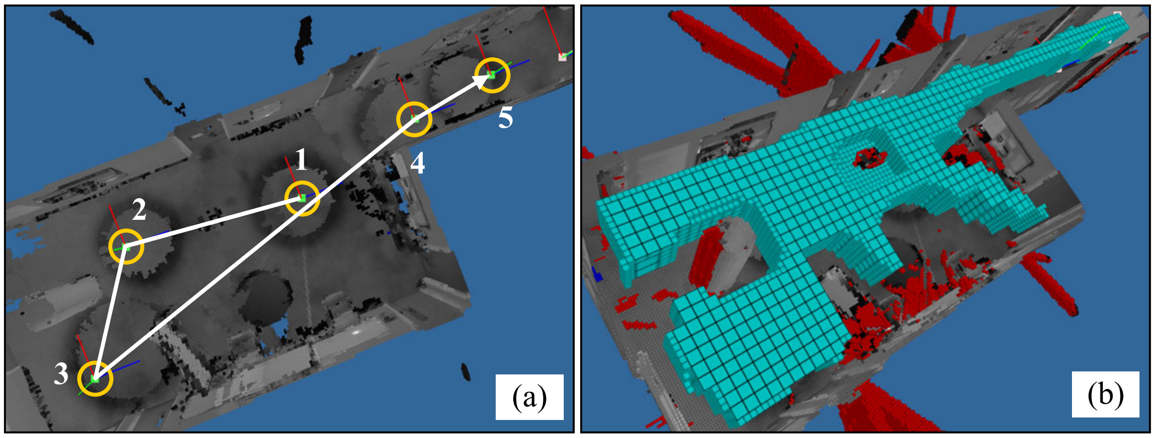


Figure 6.20: (a) The pose of the first scan and the computed poses for the subsequent four scans. (b) The final partial octree model and feasible view volumes.

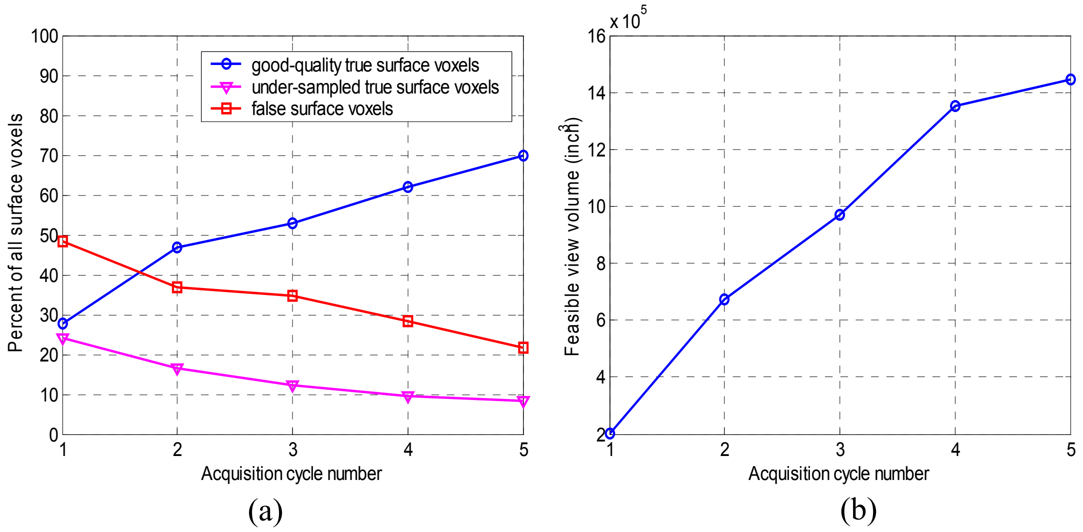


Figure 6.21: (a) The proportions of the different types of surface voxels in the partial models after each scan is merged. (b) The total volume of the feasible view volumes.

Figure 6.22 shows the triangle-mesh model of the room constructed from the five range images. PolyWorks 8 was used to reconstruct the model. Again, due to insufficient system memory for reconstruction computation, all the five range images were downsampled to half the resolution in each direction.

Table 6.6 shows the average execution time taken by each main step of the acquisition cycle. The view evaluation is limited to approximately two minutes. At 10-samples-per-degree scanning resolution, the DeltaSphere-3000 takes about eight minutes to make a full panoramic scan. The graph in Figure 6.23(a) shows how much of the total patch area that has been evaluated by the view evaluation step in the two minutes. Besides the second acquisition cycle, the view evaluation step was able to evaluate more than 70% of the total patch area. Figure 6.23(b) shows the total patch sampling deficit in each cycle, and how much of the sampling deficit has been evaluated.

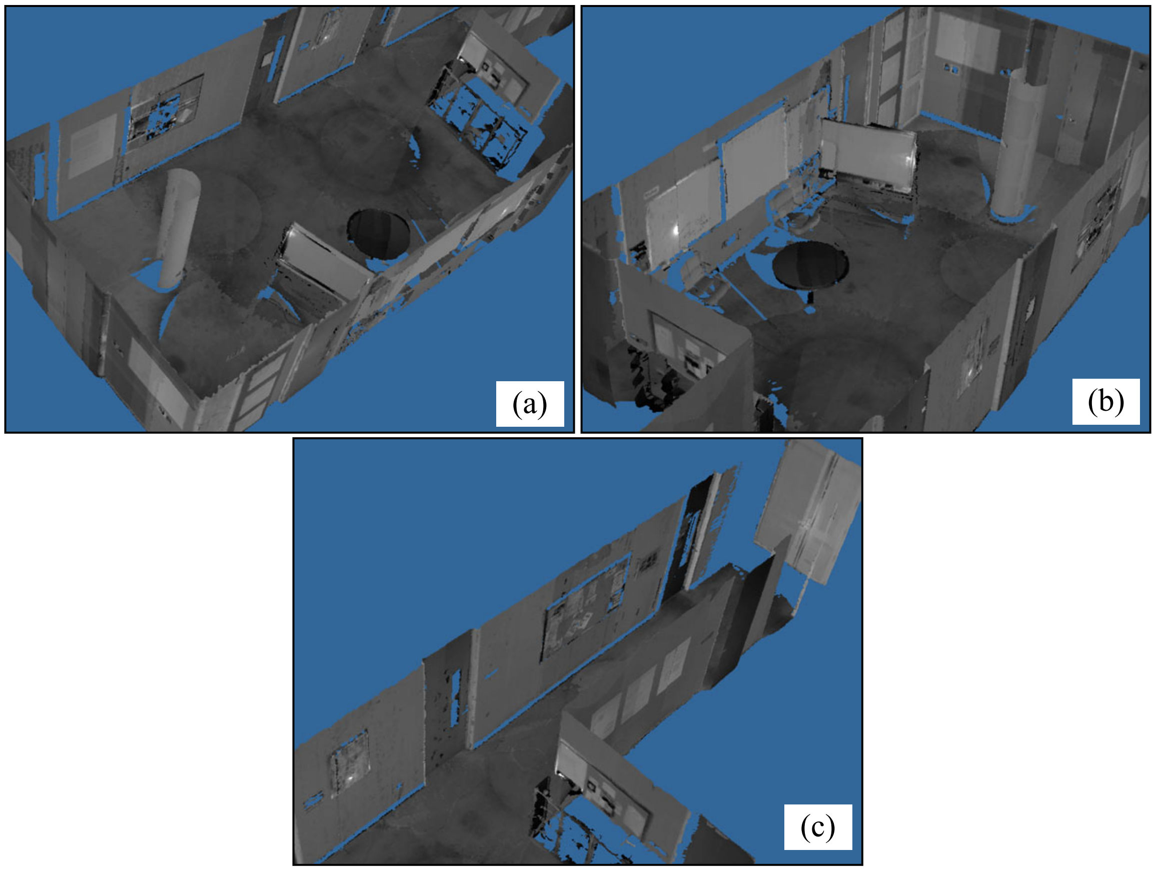


Figure 6.22: Three different views of the triangle-mesh model of the room reconstructed from the five range images.

Steps	Average time (sec)	Percentage
Range image processing	29.5	15.5
Surface registration	8.7	4.6
Partial octree model creation and merging	24.4	12.8
Feasible view octree creation	1.2	0.6
Patch extraction and ranking	4.7	2.5
View evaluation	120.5	63.1
Registration analysis	1.7	0.9
Total:	190.8	100.0

Table 6.6: The average execution time of each main step.

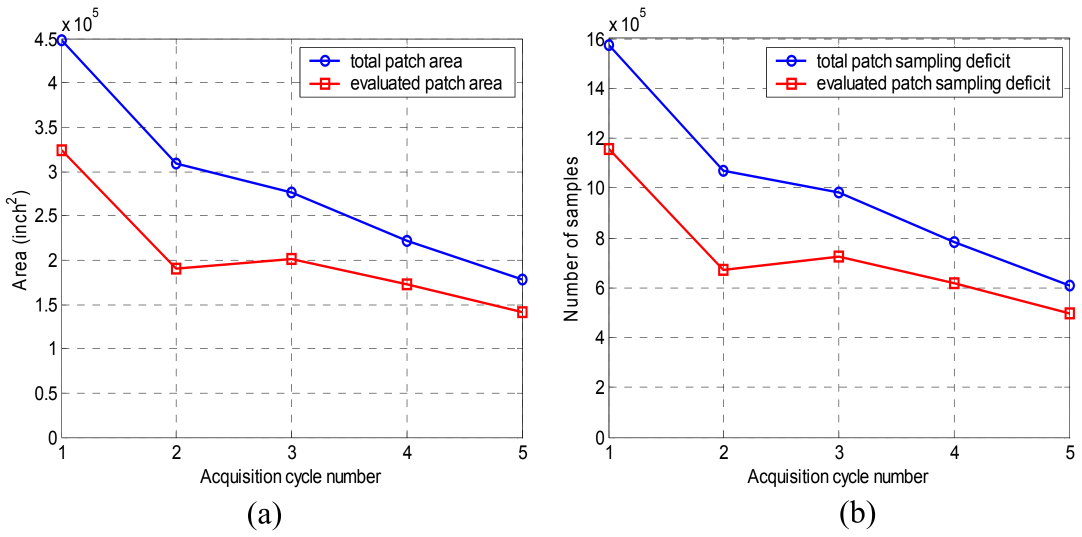


Figure 6.23: (a) The total patch area in the partial octree model after each scan has been merged, and the amount that is evaluated in the view evaluation step. (b) The total patch sampling deficit and the amount that is evaluated in the view evaluation step.

6.4 Using Different First Views

In this experiment, we compare four different view plans generated by the next-best-view planning system for the synthetic kitchen model used in Section 6.2. Each view plan is the result of using a different first view to start the model acquisition process, and every view plan consists of ten views—the first view is manually and arbitrarily chosen, and the subsequent nine views are computed by the next-best-view planner.

The simulated range scanner is identical to that in Section 6.2, and the parameter values of the scanner and the values of some other parameters used in the experiment are shown in Table 6.3.

Top views of the four view plans are shown in Figure 6.24. In View Plans 0 and 1, the first views are in the main kitchen area, whereas in View Plans 2 and 3, they are in two of the small rooms. The graph in Figure 6.25(a) compares the four view plans by looking at the number of true and false surface voxels in the partial octree model after each scan is merged. It can be seen that View Plan 2, which started from a small room, is significantly slower in "discovering" most of the surface areas in the environment. This is evident in the view plan shown in Figure 6.24. The effect is also reflected in the graphs of Figure 6.25(b) and 6.26(a).

Figure 6.26(b) compares the total patch area extracted from the partial model after each scan is merged. It is interesting to compare the order of occurrences of the "peaks" of the four view plans. The "peak" of View Plan 2 occurs later than those of the other three view plans. This is an indication that most of the information of the environment is "discovered" later than by the other view plans.

It appears that after the eighth scan, all the graphs of the four view plans begin to converge to approximately the same values. Figure 6.27 shows the final partial models generated by each view plan after the tenth scans. The differences between these models are very few and minor. The conclusion from this experiment is that, if the number of scans is small, the choice of the first view can significantly affect the quality of the reconstructed model. With a sufficient number of scans, the choice of the first view becomes less important. The minimum number of scans required, so that the choice of the first view becomes unimportant, is dependent on the geometry of the scene and the capability of the scanner.

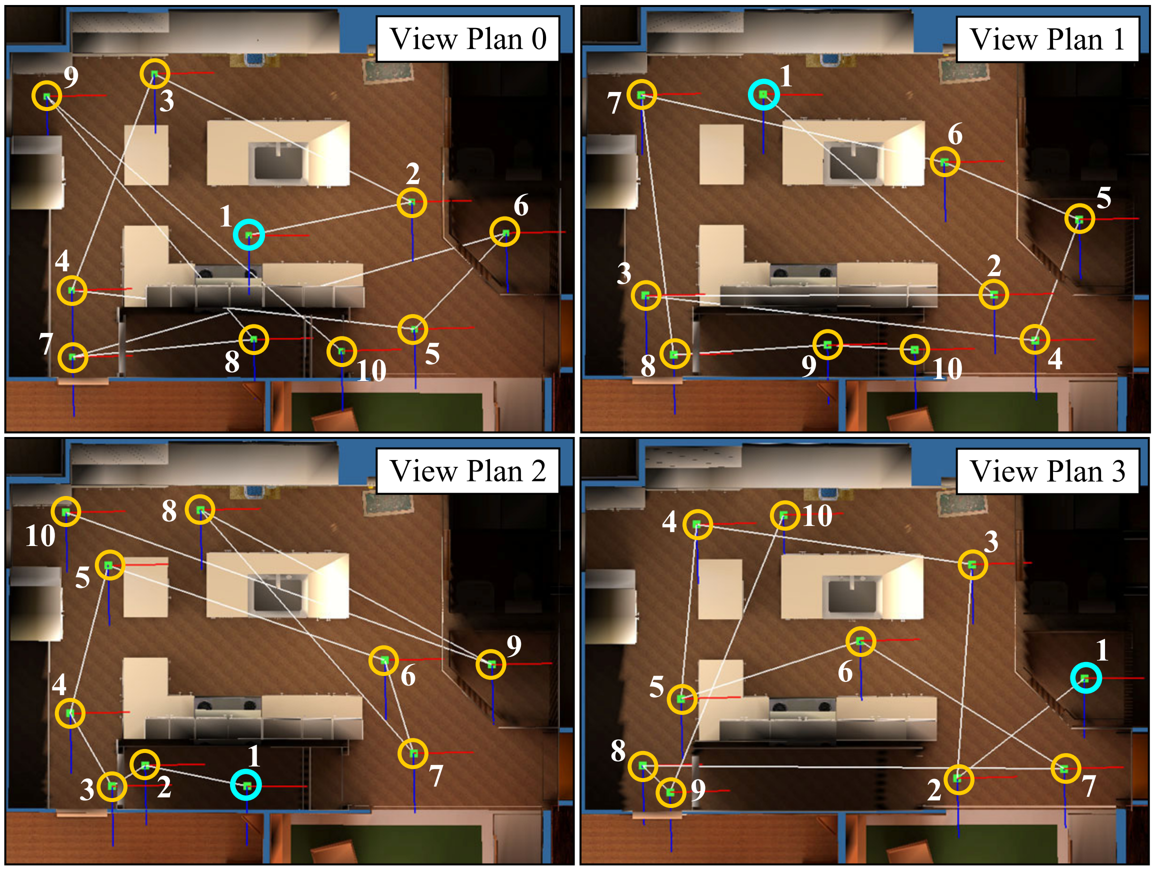
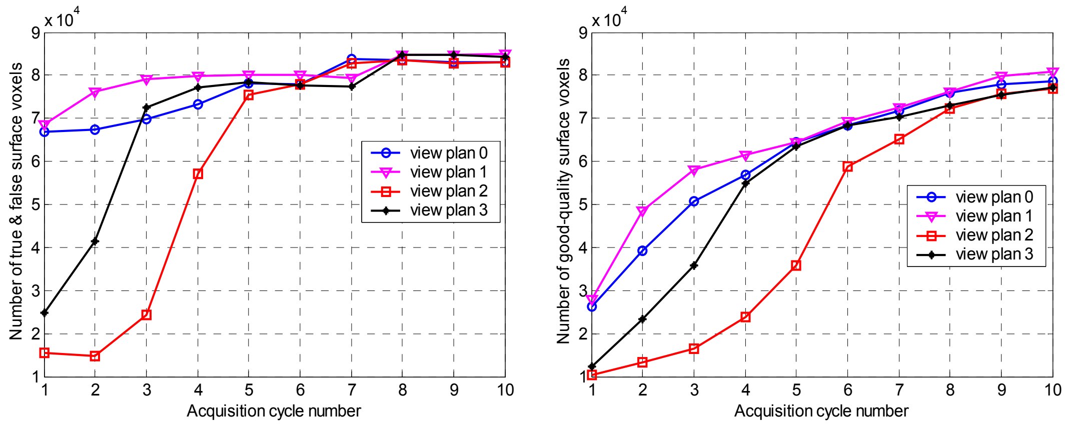


Figure 6.24: The four different view plans. Each view plan consists of ten views. The views surrounded by light-blue circles are the first views. The views may not be of the same height from the floor.



(a) (b)

Figure 6.25: Comparison of the four view plans. (a) The number of true and false surface voxels in the partial octree model after each scan is merged. (b) The number of good-quality surface voxels in the partial octree model after each scan is merged.

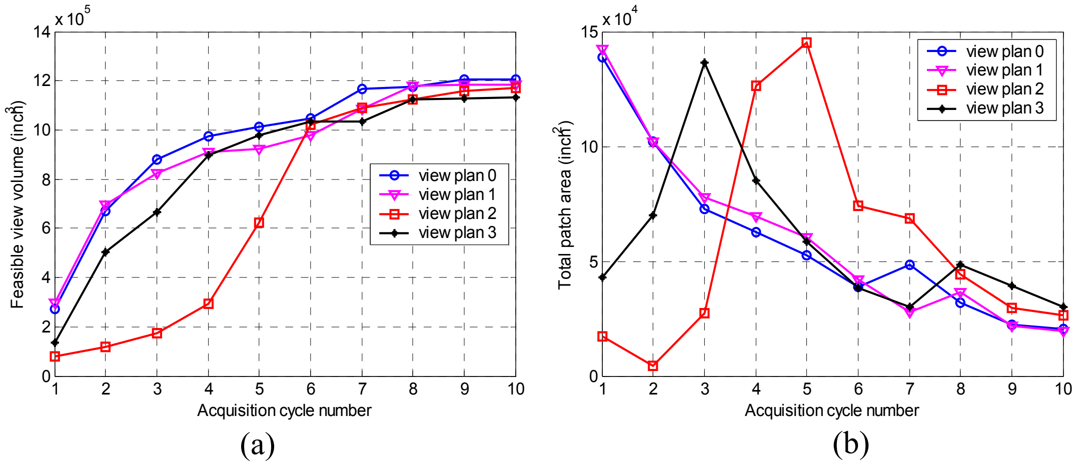


Figure 6.26: Comparison of the four view plans. (a) The total volume of the feasible view volumes after each scan is merged. (b) The total patch area extracted from the partial model after each scan is merged.

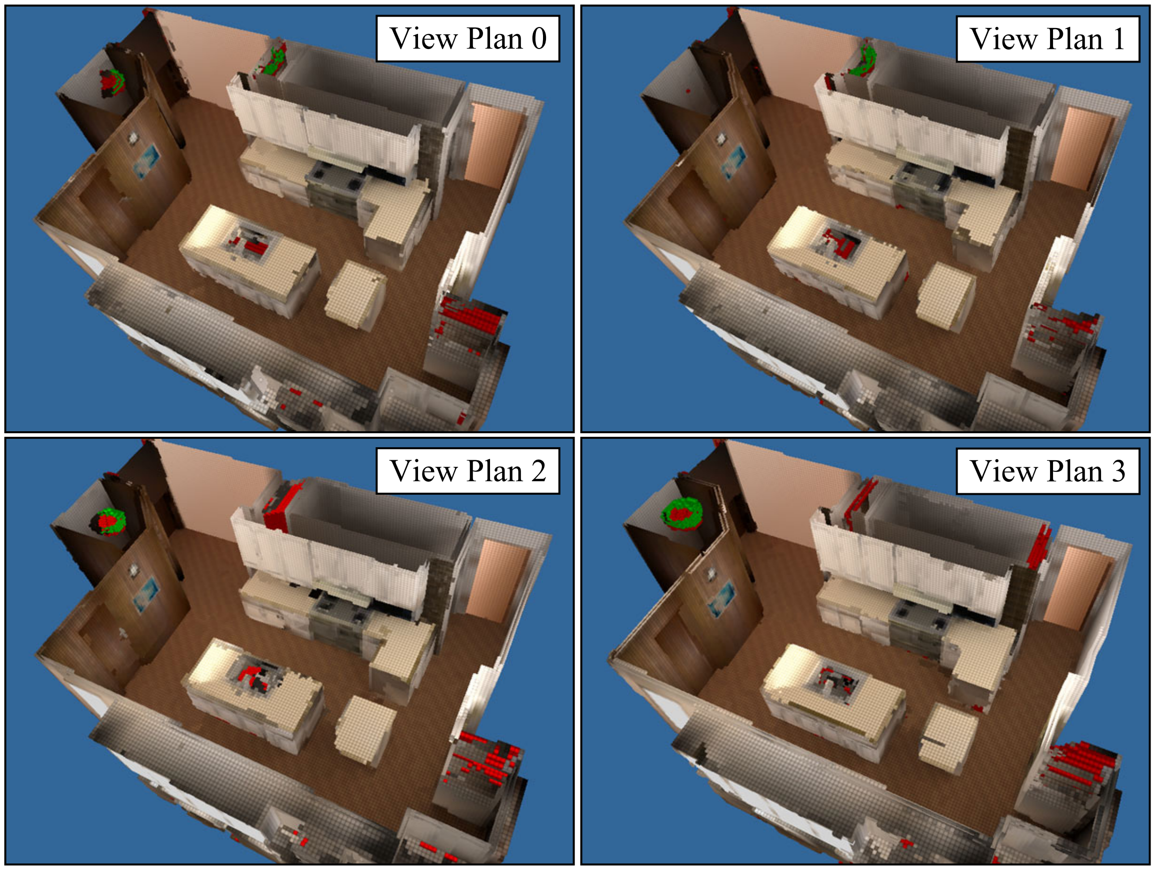


Figure 6.27: The partial octree models resulted from the four different view plans.

Chapter 7

Conclusion

The goal of this work is to devise a practical method to automate the view planning for the acquisition of range data to reconstruct 3D models of indoor environments. The dissertation has presented the view planning problem in detail, and stated the objectives and requirements of a solution to the problem.

This work has demonstrated that it is practical to use a greedy next-best-view planning approach for the range acquisition of indoor environments. The proposed hierarchical view evaluation method allows exhaustive 3D view evaluations to be possible within reasonable time. The hierarchical approach achieves computational efficiency by exploiting the various spatial coherences in the acquisition constraints and the sampling quality function in the view metric. The algorithm has been implemented in a view planning system and has demonstrated great speedups over the straightforward view evaluation method used in previous next-best-view algorithms.

The usefulness of the view planning system also comes from the inclusion of many practical real-world acquisition constraints and quality requirements in the view metric. Three quality requirements are considered—they are the surface completeness, the surface sampling quality, and the surface registration accuracy. The acquisition constraints are classified into three groups, namely, the positional constraints, the sensing constraints, and the registration constraint. I believe that the view metric and the hierarchical view evaluation method are general enough to allow many other useful constraints and requirements to be added. For example, if it is required that the new view should not be more than a specified distance from the current view, then this constraint can be implemented as one of the positional constraints in the view metric function. In another example, if we favor new views that are closer to the current one, then a weighting function can be added to scale the score of each view proportional to its distance from the current view.

Another advantage of the hierarchical view evaluation method is that view sensitivity to the potential pose errors of the scanner can be easily incorporated into the evaluation of views. With this, each view is evaluated only with the surface elements that can be acquired from every pose within the pose error bound of the view. The result is that surfaces that are expected to be acquired from a view will be acquired even if the scanner is physically positioned with pose error.

The dissertation has also shown the importance of the registration constraint and the effectiveness of the new registration accuracy metric. The constraint ensures that the range image to be acquired from the computed new view can be accurately registered to the partial model. The registration is essential to localize the actual pose of the scanner, and also to allow the new range image to be correctly merged to the partial model. Unlike all the previous next-best-view methods that consider the registration constraint, the proposed registration accuracy metric is more accurate in that it takes into account not only the amount of overlap between the two surfaces, but also the shape constraint on the 3D rigid-body transformation between the two surfaces, and the range measurement errors.

Although the hierarchical view evaluation method has only been applied to 3D views, it is potentially applicable to higher-dimensional views. It is also extensible to scanners with bistatic sensors, but it is not clear how well the spatial coherences can be exploited to gain significant performance improvement over the straightforward method.

In the following section, some of the limitations of the next-best-view method are discussed, and future work and extensions are proposed.

7.1 Limitations and Future Work

The most fundamental limitation of the proposed next-best-view method is the non-optimality of the greedy approach in terms of the number of views in the view plan. Even though complete a priori geometric information is not available, with the partial information, it is still possible to produce results that are generally better than those from a purely greedy approach. Finding a local optimal set of views for the current partial model is computationally difficult because of the exponential time complexity and, moreover, it is likely that most of the views in the local optimal solution will become obsolete as new information is obtained from subsequent scans. One potential solution may be a hybrid between a local optimal approach and a purely greedy approach. With this hybrid approach, at each iteration we first find the optimal set of *k*, or fewer, views for the current partial model, where *k* is a constant specified by the user. Using the greedy approach, the "best" view among the *k* views is chosen for the next scan. After the information of the new scan is incorporated into the partial model, a new optimal set of *k*, or fewer, views is computed for the new partial model, and the process repeats until termination.

It seems that the hybrid approach is able to avoid the obvious non-optimality of the purely greedy approach and, with a small value of *k* (for example, 3), it is able to avoid the computational intractability of the NP-complete problems. One of the major challenges to implementing this hybrid approach may be the large amount of memory required.

The hierarchical view evaluation is suitable for indoor environments because these environments usually have large planar surfaces such as walls, floors and ceilings, which contribute a great deal to the spatial coherence that can be exploited to speed up view evaluation. The hierarchical method may not be very efficient for environments or objects that have very few or no large planar surface. However, for environments with many large curved surfaces, view evaluation efficiency may not be a problem because these curved surfaces can be approximated by planar patches during patch extraction.

There are still a few major problems to be solved in order to extend the hierarchical view evaluation to higher-dimensional views. Extension to 5D poses (3D position, pan and tilt) seems to be the most useful because, in practice, that is the highest degree of positioning freedom necessary for most scanners. We have seen in Section 4.3 that some of the major challenges of extending the algorithm to 5D poses are (1) a data structure to allow independent subdivision of the solution space in any subset of dimensions, and (2) a very space-efficient representation of the solution space, possibly by exploiting the independent spatial coherences in the different dimensions. Besides extending the hierarchical algorithm to views with higher-dimensional poses, other imaging parameters may be included in the views as well. For example, the variable field of view of the scanner and the adjustable scanning resolution may be some of the parameters in the views.

Although I have discussed, in Section 4.3.3, the potential difficulties of applying the hierarchical view evaluation method to range scanners with bistatic sensors, it is still interesting to explore how we should subdivide the views and how we can approximate the swept volumes so that the hierarchical algorithm can become effective for bistatic sensors.

It seems that the evaluation of the registration constraint can also benefit from using a hierarchical evaluation approach. When evaluating the registration constraint for a view, all the true surface elements in the partial model are considered. These true surface elements can first be grouped into patches, and each candidate view is then evaluated with the patches in a hierarchical manner where patches are subdivided if necessary. If all the feasible views are to be evaluated for the registration constraint, then the patches are evaluated with the viewcells.

There are some practical extensions that we can add to the next-best-view planning system. We can have a user-interface to let the user add his knowledge of the environment to the partial model. For example, he can manually specify that a certain region is empty space. This should improve the optimality of the view planning solution, because more information is now available earlier to the view planner. Sometimes, only a limited number of scans can be afforded and, in this case, it may be useful if the user is allowed to specify which regions or surfaces are more important, so that the next-best-view planning system will give priorities to those regions. Since the complete model of the scene is not available for the user to precisely specify the important surfaces, the user may instead have to specify, over many acquisition cycles, the approximate volumetric regions that contain the surfaces.

With automated view planning, it is possible to fully-automate the whole acquisition process if the scanner can be positioned automatically. We can use a mobile robot to move the scanner. In this case, the next-best-view solution must also include unobstructed paths for the robot to move to the planned position.

Appendix A

Lindeberg's Theorem

Lindeberg's Theorem is used in the derivation of the rotational alignment error condition in Section 5.3.2.2. In this appendix, the theorem is defined, and then I show how the Lindeberg condition is satisfied in my derivation.

A.1 Definition

The central limit theorem states that the sum of many independent identically distributed random variables with finite variance will be approximately normally distributed. Lindeberg's Theorem is a generalization of the central limit theorem. It does not require identical distribution, but incorporates a condition (Lindeberg condition) that, if satisfied, implies that the sum will be approximately normally distributed. The theorem is defined in [Ash1972] as follows:

Lindeberg's Theorem. Let $S_n = X_1 + \cdots + X_n$, $n = 1, 2, \ldots$, where the X_k are independent random variables with finite mean m_k and finite variance ${\sigma_k}^2$. Let $T_n = c_n^{-1}(S_n - E(S_n))$, where ${c_n}^2 = \operatorname{Var} S_n = \sum_{k=1}^{n} {\sigma_k}^2$, and let F_k be the distribution function of X_k. If for every $\varepsilon > 0$,

$$\left(\frac{1}{{c_n}^2} \sum_{k=1}^{n} \int_{\{x : |x - m_k| \geq \varepsilon c_n\}} (x - m_k)^2 \, dF_k(x) \right) \to 0 \quad \text{as} \quad n \to \infty \qquad \begin{pmatrix} \text{Lindeberg} \\ \text{condition} \end{pmatrix},$$

then T_n converges in distribution to a random variable X^* that is normal with mean 0 and variance 1.

A.2 Application

I describe how the Lindeberg condition is satisfied to arrive at the distribution of Q_x in Equation (5.46). From Equation (5.42), we have

$$S_n = Q_x = \sum_{k=1}^{n} X_k \tag{A.1}$$

where $n = M$ and

$$X_k = y_k\left(W_{Ak} - W_{Bk}\right) - z_k\left(V_{Ak} - V_{Bk}\right). \tag{A.2}$$

Then, from Equation (5.44), the mean of X_k is

$$m_k = y_k\left(\mu_{W_A} - \mu_{W_B}\right) - z_k\left(\mu_{V_A} - \mu_{V_B}\right). \tag{A.3}$$

From Equation (5.45), we get

$$c_n{}^2 = \sigma_{Q_x}^2 = \sum_{i=1}^{M}\left(y_i^2\sigma_{W_A}^2 + y_i^2\sigma_{W_B}^2 + z_i^2\sigma_{V_A}^2 + z_i^2\sigma_{V_B}^2\right). \tag{A.4}$$

Since $c_n{}^2 \to \infty$ and $\{x : |x - m_k| \geq \varepsilon\, c_n\} \downarrow \varnothing$ as $n \to \infty$, the Lindeberg condition is satisfied, i.e.

$$\left(\frac{1}{c_n{}^2}\sum_{k=1}^{n}\int_{\{x : |x - m_k| \geq \varepsilon\, c_n\}} \left(x - m_k\right)^2 dF_k(x)\right) \to 0 .$$

Therefore, by Lindeberg's theorem, $T_n = c_n^{-1}\left(S_n - E(S_n)\right)$ converges to a normal distribution with mean 0 and variance 1, and that means $S_n = Q_x$ converges to the normal distribution $N\left(\, E(S_n), c_n\right)$ where

$$E(S_n) = \mu_{Q_x} = \sum_{i=1}^{M}\left(y_i\left(\mu_{W_A} - \mu_{W_B}\right) - z_i\left(\mu_{V_A} - \mu_{V_B}\right)\right). \tag{A.5}$$

Bibliography

[Aarts1985] E. H. L. Aarts and P. J. M. van Laarhoven, Statistical Cooling: A General Approach to Combinatorial Optimization Problems. Philips J. Research, 40, pp. 193–226, 1985.

[Aliaga1999] Daniel G. Aliaga, and Anselmo Lastra. Automatic Image Placement to Provide a Guaranteed Frame Rate. Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99), pp. 307–316, 1999.

[Akenine-Möller2001] Tomas Akenine-Möller. Fast 3D Triangle-Box Overlap Testing. Journal of Graphics Tools, 6(1):29–33, 2001. C code available at http://www.acm.org/jgt/papers/AkenineMoller01/.

[Allen1998] Peter K. Allen, Michael K. Reed, and Ioannis Stamos. View Planning for Site Modeling. Proceedings of the DARPA Image Understanding Workshop, November 1998.

[Allen2003] Peter K. Allen, Alejandro Troccoli, Benjamin Smith, and Stephen Murray, Ioannis Stamos, and Marius Leordeanu. New Methods for Digital Modeling of Historic Sites. IEEE Computer Graphics and Applications, 23(6):32–41, December 2003.

[Anderson1985] D. P. Anderson. Efficient Algorithms for Automatic Viewer Orientation. Computer & Graphics, 9(4):407–413, 1985.

[Arun1987] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-Squares Fitting of Two 3-D Points Sets. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 9(5):698–700, 1987.

[Ash1972] Robert B. Ash. Real Analysis and Probability. Academic Press, 1972.

[Assarsson2000] Ulf Assarsson and Tomas Möller. Optimized View Frustum Culling Algorithms for Bounding Boxes. Journal of Graphics Tools (JGT), 5(1):9–22, 2000.

[Banta1995] Joseph E. Banta, Y. Zhieng, X. Z. Wang, G. Zhang, M. T. Smith, and M. A. Abidi. A "Best-Next-View" Algorithm for Three-Dimensional Scene Reconstruction Using Range Images. Proceedings of the SPIE Conference on Intelligent Robots and Computer Vision XIV: Algorithms, Vol. 2588, pp. 418–429, October 1995.

[Beraldin2000] J-Angelo Beraldin, François Blais, Luc Cournoyer, Guy Godin, Marc Rioux. Active 3D Sensing. Modelli E Metodi per lo studio e la conservazione dell'architettura storica, University: Scuola Normale Superiore, Pisa. pp. 22–46, 2000. NRC 44159. (Invited paper). Also available at http://www.vit.iit.nrc.ca/References/NRC-44159.pdf.

[Bergevin1996] Robert Bergevin, Marc Soucy, Herve Gagnon, and Denis Laurendeau. Towards a General Multi-View Registration Technique. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 18(5):540–547, May 1996.

[Besl1989] Paul J. Besl. Range Image Sensors. Advances in Machine Vision, J. Sanz, Ed. Springer-Verlag, 1989.

[Besl1992] Paul J. Besl, and Neil D. McKay. A Method for Registration of 3-D Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 14(2):239–256, 1992.

[Black1998] M. J. Black, G. Sapiro, D. Marimont, D. Heeger. Robust Anisotropic Diffusion. IEEE Transactions on Image Processing, Special issue on Partial Differential Equations and Geometry Driven Diffusion in Image Processing and Analysis, 7(3):421–432, March 1998.

[Blais2003] François Blais. A Review of 20 Years of Range Sensor Development. Videometrics VII, Proceedings of SPIE-IS&T Electronic Imaging, SPIE Vol. 5013 (2003), pp. 62–76, January 2003. Also available at http://www.vit.iit.nrc.ca/References/NRC-44965.pdf.

[Boissonnat1984] Jean-Daniel Boissonnat. Geometric Structures for Three-Dimensional Shape Representation. ACM Transactions on Graphics (TOG), 3(4):266–286, 1984.

[Chen1992] Yang Chen, and Gerard Medioni. Object Modeling by Registration of Multiple Range Images. International Journal of Image and Vision Computing, 10(3):145–155, 1992.

[Chen1993] Shenchang Eric Chen, and Lance Williams. View Interpolation for Image Synthesis. Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93), pp. 279–288, 1993.

[Chen1999] Chu-Song Chen, Yi-Ping Hung, and Jen-Bo Cheng. RANSAC-Based DARCES: A New Approach to Fast Automatic Registration of Partially Overlapping Range Images. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 21(11):1229–1234, November 1999.

[Cohen1986] Michael Cohen, Donald P. Greenberg, Dave S. Immel, and Philip J. Brock. An Efficient Radiosity Approach for Realistic Image Synthesis. IEEE Computer Graphics and Applications, 6(3):26–35, 1986.

[Cohen-Or2003] Daniel Cohen-Or, Yiorgos Chrysanthou, Claudio Silva, and Frédo Durand. A Survey of Visibility for Walkthrough Applications. IEEE Transaction on Visualization and Computer Graphics, 9(3):412–431, July 2003.

[Connolly1985] C. I. Connolly. The Determination of Next Best Views. Proceedings of the IEEE International Conference on Robotics and Automation, pp. 432–435, 1985.

[Cormen1990] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. Introduction to Algorithms. MIT Press and McGraw-Hill, 1990.

[Cowan1988] C. K. Cowan, and P. D. Kovesi. Automatic Sensor Placement from Vision Task Requirements. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 10(3):407–416, May 1988.

[Curless1996] Brian Curless, and Marc Levoy. A Volumetric Method for Building Complex Models from Range Images. Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96), pp. 303–312, 1996.

[Davis2002] James Davis, Stephen R. Marschner, Matt Garr, and Marc Levoy. Filling Holes in Complex Surfaces Using Volumetric Diffusion. Proceedings of the First International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT '02), pp. 428–438, June 2002.

[Debevec1996] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach. Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96), pp. 11–12, 1996.

[DeltaSphere] DeltaSphere, Inc. http://www. deltasphere.com.

[Drettakis1997] George Drettakis, and François X. Sillion. Interactive Update of Global Illumination Using a Line-Space Hierarchy. Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97), pp. 57–64, 1997.

[Durand1999] Frédo Durand. 3D Visibility: Analytical Study and Applications. Ph.D. Dissertation, Université Joseph Fourier, Grenoble, France, July 1999.

[Faugeras1993] Olivier Faugeras. Three-Dimensional Computer Vision: A Geometric Viewpoint. The MIT Press, 1993.

[Fleishman1999] Shachar Fleishman, Daniel Cohen-Or, and Dani Lischinski. Automatic Camera Placement for Image-Based Modeling. Proceedings of the 7th Pacific Conference on Computer Graphics and Applications (PG '99), pp. 12–20, 1999.

[Foley1992] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. Computer Graphics: Principles and Practice, Second Edition. Addison-Wesley, 1992.

[Forsyth2002] David A. Forsyth, and Jean Ponce. Computer Vision: A Modern Approach. Prentice Hall, 2002.

[Friedman1977] J.H. Friedman, J. L. Bentley, and R. A. Finkel. An Algorithm for Finding Best Matches in Logarithmic Expected Time. ACM Transactions on Mathematical Software, 3(3):209–226, 1977.

[Frisken2000] Sarah F. Frisken, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones. Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics. Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2000), pp. 249–254, 2000.

[Frisken2002] Sarah F. Frisken, and Ron Perry. Simple and Efficient Traversal Methods for Quadtrees and Octrees. Journal of Graphics Tools (JGT), 7(3):1–11, 2002.

[Früh2003] Christian Früh, and Avideh Zakhor. Constructing 3D City Models by Merging Aerial and Ground Views. IEEE Computer Graphics and Applications, 23(6):52–61, December 2003.

[García1998] Miguel A. García, Susana Velázquez, and Angel D. Sappa. A Two-Stage Algorithm for Planning the Next View From Range Images. Proceedings of the 9th British Machine Vision Conference (BMVC '98), pp.720–729, 1998.

[Garland1997] Michael Garland, and Paul S. Heckbert. Surface Simplification Using Quadric Error Metrics. Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97), pp. 209–216, 1997.

[González-Baños1998] Héctor González-Baños, and Jean-Claude Latombe. Planning Robot Motion for Range-Image Acquisition and Automatic 3D Model Construction. Integrated Planning for Autonomous Agent Architectures, AAAI Fall Symposium, October 1998.

[González-Baños1999] Héctor González-Baños,Eric Mao, Jean-Claude Latombe, T. M. Murali and Alon Efrat. Planning Robot Motion Strategies for Efficient Model Construction. Proceedings of the 9th International Symposium on Robotics Research, pp. 345–352, 1999.

[González-Baños2001] Héctor González-Baños, and Jean-Claude Latombe. A Randomized Art-Gallery Algorithm for Sensor Placement. Proceedings of the 17th ACM Symposium on Computational Geometry, pp. 232–240, 2001.

[Haines1994] Eric A. Haines, and John R. Wallace. Shaft Culling for Efficient Ray-Traced Radiosity. Proceedings of the Second Eurographics Workshop on Rendering, pp.122–138, 1994.

[Hall-Holt1998] Olaf Hall-Holt. Scrutinizing Real-World Geometry: The Next Best View. http://www-graphics.stanford.edu/~olaf/nbv.html, 1998.

[Hanrahan1991] Pat Hanrahan, David Salzman, Larry Aupperle. A Rapid Hierarchical Radiosity Algorithm. ACM SIGGRAPH Computer Graphics (Proceedings of SIGGRAPH '91), 25(4):197–206, July 1991.

[Hillesland2002] Karl Hillesland, Brian Salomon, Anselmo Lastra, and Dinesh Manocha. Fast and Simple Occlusion Culling Using Hardware-Based Depth Queries. Technical Report TR02-039, Department of Computer Science, University of North Carolina at Chapel Hill, 2002.

[Hilton1997] Adrian Hilton, and John Illingworth. Multi-Resolution Geometric Fusion. Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling (3DIM '97), pp. 181–189, 1997.

[Hilton1997a] Adrian Hilton. Model Building from 3D Surface Measurements. http://www.ee.surrey.ac.uk/Research/VSSP/3DVision/model_building/model.html, 1997.

[Hoffmann1989] Christoph M. Hoffmann. Geometric and Solid Modeling: An Introduction. Morgan Kaufmann Publishers, 1989.

[Horn1987] Berthold K. P. Horn. Closed-Form Solution of Absolute Orientation Using Unit Quaternions. Journal of the Optical Society of America (JOSA), Series A, 4(4):629–642, 1987.

[Huber2001] Daniel F. Huber, and Martial Hebert. Fully Automatic Registration of Multiple 3D Data Sets. Proceedings of IEEE Computer Society Workshop on Computer Vision Beyond the Visible Spectrum (CVBVS 2001), December 2001.

[Hunter1978] G. M. Hunter. Efficient Computation and Data Structures for Graphics. Ph.D. Thesis, Department of Electrical Engineering and Computer Science, Princeton University, Princeton, NJ, 1978.

[Hunter1979] G. M. Hunter, and K. Steiglitz. Operations on Images Using Quad Trees. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 1(2):145–153, April 1979.

[Hutchinson1989] Seth A. Hutchinson, and Avinash C. Kak. Planning Sensing Strategies in a Robot Work Cell with Multi-Sensor Capabilities. IEEE Transactions on Robotics and Automation, 5(6):765–783, December 1989.

[InnovMetric] InnovMetric Software, Inc. http://www.innovmetric.com.

[Johnson1997] Andrew Johnson. Spin-Images: A Representation for 3-D Surface Matching. Ph.D. Dissertation, Robotics Institute, Carnegie Mellon University, August 1997.

[Johnson1999] Andrew E. Johnson, and Martial Hebert. Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 21(5):433–449, 1999.

[Kakusho1995] K. Kakusho, T. Kitahashi, K. Kondo, and J.-C. Latombe. Continuous Purposive Sensing and Motion for 2D Map Building. Proceedings of IEEE International Conference of Systems, Man and Cybernatics, pp. 1472–1477, 1995.

[Kendall1977] M. G. Kendall, and A. Stuart. Canonical Variables, 4th Edition. Chapter 43, pp. 320–369, Griffin, London, 1977.

[Kirkpatrick1983] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by Simulated Annealing. Science, 220, pp. 671–680, 1983.

[Klein2000] Konrad Klein, and Vítor Sequeira. The View-Cube: An Efficient Method of View Planning for 3D Modelling from Range Data. Proceedings of the Fifth IEEE Workshop on Applications of Computer Vision, pp. 186–191, December 2000.

[Laidlaw1986] David H. Laidlaw, W. Benjamin Trumbore, and John F. Hughes. Constructive Solid Geometry for Polyhedral Objects. Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '86), pp. 161–170, 1986.

[Lensch2003] Hendrik P.A. Lensch, Jochen Lang, Asla M. Sá, and Hans-Peter Seidel. Planned Sampling of Spatially Varying BRDFs. Computer Graphics Forum (Proceedings of EUROGRAPHICS 2003), 22(3):473–482, September 2003.

[Levoy1996] Marc Levoy, and Pat Hanrahan. Light Field Rendering. Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96), pp. 31–42, 1996.

[Levoy2000] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, Dave Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk, The Digital Michelangelo Project: 3D Scanning of Large Statues. Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2000), pp. 131–144, 2000.

[Levoy2003] Marc Levoy et al. The Digital Michelangelo Project. http://graphics.stanford.edu/ projects/mich/index.html, 2003.

[Lorensen1987] William E. Lorensen, and Harvey E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87), pp. 163–169, July 1987.

[Lorusso1995] A. Lorusso, D. W. Eggert, and R. B. Fisher. A Comparison of Four Algorithms for Estimating 3-D Rigid Transformations. Proceedings of the 1995 British Conference on Machine Vision (BMVC '95), Vol. 1, pp. 237–246, 1995.

[Low2003] Kok-Lim Low and Anselmo Lastra, Reliable and Rapidly-Converging ICP Algorithm Using Multiresolution Smoothing. Proceedings of the 4th IEEE International Conference on 3-D Digital Imaging and Modeling (3DIM '03), pp. 171–178, October 2003.

[Low2004] Kok-Lim Low. Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration. Technical Report TR04-004, Department of Computer Science, University of North Carolina at Chapel Hill, February 2004.

[Massios1998] Nikolaos A. Massios, and Robert B. Fisher. A Best Next View Selection Algorithm Incorporating a Quality Criterion. Proceedings of the 9th British Machine Vision Conference (BMVC '98), pp. 780–789, 1998.

[Maver1995] Jasna Maver, and Ruzena Bajcsy. Occlusions as a Guide for Planning the Next View. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 15(5):417–433, May 1995.

[McMillan1995] Leonard McMillan, and Gary Bishop. Plenoptic Modeling: An Image-Based Rendering System. Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95), pp. 39–46, 1995.

[MENSI] MENSI, Inc. http://www.mensi.com.

[Nishino2002] Ko Nishino, and Katsushi Ikeuchi. Robust Simultaneous Registration of Multiple Range Images. Proceedings of the Fifth Asian Conference on Computer Vision (ACCV 2002), pp. 454–461, January 2002.

[Nüchter2003] Andreas Nüchter, Hartmut Surmann, and Joachim Hertzberg. Planning Robot Motion for 3D Digitalization of Indoor Environments. Proceedings of the 11th IEEE International Conference on Advanced Robotics (ICAR 2003), pp. 222–227, June 2003.

[O'Rourke1987] Joseph O'Rourke. Art Gallery Theorems and Algorithms. Oxford University Press, 1987.

[O'Rourke1998] Joseph O'Rourke. Computational Geometry in C, Second Edition. Cambrigde University Press, 1998.

[Papadopoulos1997] Dimitri Papadopoulos-Orfanos, and Francis Schmitt. Automatic 3-D Digitization Using a Laser Rangefinder with a Small Field of View. Proceedings of IEEE International Conference on Recent Advances in 3-D Digital Imaging and Modeling (3DIM '97), pp. 60–67, May 1997.

[Papadopoulos2001] Dimitri Papadopoulos-Orfanos. 3-D Sensing. http://perso.club-internet.fr/dpo/ numerisation3d/index.html, 2001.

[Pito1996] Richard Pito. Automated Surface Acquisition Using Range Cameras. Ph.D. Dissertation, University of Pennsylvania, Pennsylvania, May 1996.

[Pito1996a] Richard Pito. A Sensor-Based Solution to the Next Best View Problem. Proceedings of IEEE International Conference on Pattern Recognition (ICPR '96), pp. 941–945, 1996.

[Pito1996b] Richard Pito. Mesh Integration Based on Co-Measurements. Proceedings of the International Conference on Image Processing, pp. 397–400, 1996.

[Pito1999] Richard Pito. A Solution to the Next Best View Problem for Automated Surface Acquisition. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 21(10):1016–1030, October 1999.

[Plantinga1990] Harry Plantinga, and Charles R. Dyer. Visibility, Occlusion and the Aspect Graph. International Journal of Computer Vision, 5(2):137–160, 1990.

[Pottmann2002] H. Pottmann, S. Leopoldseder, M. Hofer. Registration without ICP. Technical Report No. 91, Institute of Geometry, Vienna University of Technology, February 2002.

[Press1992] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. Numerical Recipes in C: The Art of Scientific Computing, Second Edition. Cambridge University Press, 1992.

[Pulli1997] Kari Pulli, Tom Duchamp, Hugues Hoppe, John McDonald, Linda Shapiro, and Werner Stuetzle. Robust Meshes from Multiple Range Maps. Proceedings of IEEE International Conference on Recent Advances in 3-D Digital Imaging and Modeling (3DIM '97), pp. 205–212, May 1997.

[Pulli1999] Kari Pulli. Multiview Registration for Large Data Sets. Proceedings of the International Conference on 3-D Digital Imaging and Modeling (3DIM '99), pp. 106–114, 1999.

[Reed1997] Michael K. Reed, Peter K. Allen, and Ioannis Stamos. 3-D Modeling from Range Imagery: An Incremental Method with a Planning Component. Proceedings of IEEE International Conference on Recent Advances in 3-D Digital Imaging and Modeling (3DIM '97), pp. 76–84, May 1997.

[Reed1998] Michael K. Reed. Solid Model Acquisition from Range Imagery. Ph.D. Dissertation. Columbia University, New York, 1998.

[Reed2000] Michael K. Reed, and Peter K. Allen. Constraint-Based Sensor Planning for Scene Modeling. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 22(12):1460–1467, December 2000.

[RIEGL] RIEGL USA, Inc. http://www.riegl.com.

[Roberts1998] D. R. Roberts, and A. D. Marshall. Viewpoint Selection for Complete Surface Coverage of Three Dimensional Objects. Proceedings of the 9th British Machine Vision Conference (BMVC '98), pp. 740–750, 1998.

[Roth1997] Gerhard Roth, and Eko Wibowoo. An Efficient Volumetric Method for Building Closed Triangular Meshes from 3-D Image and Point Data. Proceedings of the Conference on Graphics Interface, pp. 173–180, 1997.

[Ruiz-Correa2001] Salvador Ruiz-Correa, Linda G. Shapiro, and Marina Melia. A New Signature-Based Method for Efficient 3-D Object Recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001), Vol. 1, pp. 769–776, 2001.

[Rusinkiewicz2001] Szymon Rusinkiewicz, and Marc Levoy. Efficient Variants of the ICP Algorithm. Proceedings of the International Conference on 3-D Digital Imaging and Modeling (3DIM '01), pp. 145–152, 2001.

[Rusinkiewicz2002] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-Time 3D Model Acquisition. ACM Transactions on Graphics (Proceedings of SIGGRAPH 2002), 21(3):438–446, July 2002.

[Rutishauser1994] Martin Rutishauser, Markus Stricker, and Marjan Trobina. Merging Range Images of Arbitrarily Shaped Objects. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR '94), pp. 573–580, June 1994.

[Sakane1987] S. Sakane, M. Ishii, and M. Kakikura. Occlusion Avoidance of Visual Sensors Based on a Hand Eye Action Simulator System: HEAVEN. Advanced Robotics, 2(2):149–165, 1987.

[Sakane1991] Shigeyuki Sakane, and Tomomasa Sato. Automatic Planning of Light Source and Camera Placement for an Active Photometric Stereo System. Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1080–1087, April 1991.

[Sakane1992] S. Sakane, R. Niepold, T. Sato, and Y. Shirai. Illumination Setup Planning for a Hand-Eye System Based on an Environmental Model. Advanced Robotics, 6(4):461–482, 1992.

[Samet89a] Hanan Samet. The Design and Analysis of Spatial Data Structures. Addison-Wesley, Reading, MA, 1989.

[Samet89b] Hanan Samet. Applications of Spatial Data Structures: Computer Graphics, Image Processing and GIS. Addison-Wesley, Reading, MA, 1989.

[Sanchiz1999] José M. Sanchiz, Robert B. Fisher. A Next-Best-View Algorithm for 3D Scene Recovery with 5 Degrees of Freedom. Proceedings of the 10th British Machine Vision Conference (BMVC '99), pp. 163–172, 1999.

[Schroeder1992] William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of Triangle Meshes. Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '92), pp. 65–70, 1992.

[Scott2000] William R. Scott, Gerhard Roth, and Jean-François Rivest. Performance-Oriented View Planning for Model Acquisition. Proceedings of the International Symposium on Robotics (ISR 2000), pp. 212–219, May 2000.

[Scott2001a] William R. Scott, Gerhard Roth, and Jean-François Rivest. View Planning with a Registration Constraint. Proceedings of the 3D Imaging and Modeling Conference (3DIM '01), pp. 127–134, May 2001.

[Scott2001b] William R. Scott, Gerhard Roth, and Jean-François Rivest. View Planning with Positioning System Error. Technical Report NRC-44195. National Research Council of Canada, Institute for Information Technology, Ottawa, Ontario, Canada, May 2001. Also available at http://iit-iti.nrc-cnrc.gc.ca/iit-publications-iti/docs/NRC-44195.pdf.

[Scott2001c] William R. Scott, Gerhard Roth, and Jean-François Rivest. View Planning for Multi-Stage Object Reconstruction. Proceedings of the International Conference on Vision Interface, pp. 64–71, June 2001.

[Scott2001d] William R. Scott, Gerhard Roth, and Jean-François Rivest. View Planning as a Set Covering Problem. Technical Report NRC-44892. National Research Council of Canada, Institute for Information Technology, Ottawa, Ontario, Canada, August 2001. Also available at http://iit-iti.nrc-cnrc.gc.ca/iit-publications-iti/docs/NRC-44892.pdf.

[Scott2002] William R. Scott. Performance-Oriented View Planning for Automated Object Reconstruction. Ph.D. Dissertation, University of Ottawa, Ottawa, Ontario, Canada, 2002.

[Scott2002a] William R. Scott, Gerhard Roth, and Jean-François Rivest. Pose Error Effects on Range Sensing. Proceedings of the 15th International Conference on Vision Interface, pp. 331–338, May 2002.

[Scott2003] William R. Scott, Gerhard Roth, and Jean-François Rivest. View Planning for Automated Three-Dimensional Object Reconstruction and Inspection. ACM Computing Surveys, 35(1):64–96, March 2003.

[Sequeira1996] Vítor Sequeira, João G. M. Gonçalves, and M. Isabel Ribeiro. Active View Selection for Efficient 3D Scene Reconstruction. Proceedings of the 13th International Conference on Pattern Recognition, Track1: Computer Vision, pp. 815–819, August 1996.

[Sequeira1999] Vítor Sequeira. 3D Image Acquisition using Laser Range Finders. http://sir.jrc.it/3d/3DReconstruction/3DSensors/3DSensors.htm, 1999.

[Sequeira2002] Vítor Sequeira, and João G. M. Gonçalves. 3D Reality Modelling: Photo-Realistic 3D Models of Real World Scenes. Proceedings of the First International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT '02), pp. 776–783, June 2002.

[Simon1996] David A. Simon. Fast and Accurate Shape-Based Registration. Ph.D. Dissertation, Carnegie Mellon University, CMU-RI-TR-96-45, 1996. Also available at http://www.ri.cmu.edu/pub_files/pub1/simon_david_1996_1/ simon_david_1996_1.pdf

[Skočaj2001] Danijel Skočaj and Aleš Leonardis. Robust Recognition and Pose Determination of 3-D objects Using Range Images in Eigenspace Approach. Proceedings of the International Conference on 3-D Digital Imaging and Modeling (3DIM '01), pp. 171–178, 2001.

[Smith1999] Steven W. Smith. The Scientist and Engineer's Guide to Digital Signal Processing, Second Edition. California Technical Publishing, San Diego, California, 1999.

[Söderkvist1999] Inge Söderkvist. Introductory Overview of Surface Reconstruction Methods. Technical Report 1999-10, Department of Mathematics, Lulea University of Technology, Lulea, Sweden, 1999. Also available at http://www.sm.luth.se/~inge/publications/ surfrec.pdf.

[Soucy1995] Marc Soucy, and Denis Laurendeau. A General Surface Approach to the Integration of a Set of Range Views. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 17(4):344–358, April 1995.

[Stockman1982] G. Stockman, S. Kopstein, and S. Benett. Matching Images to Models for Registration and Object Detection via Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 4(3):229–241, 1982.

[Strang1988] Gibert Strang. Linear Algebra and Its Applications, Third Edition. Harcourt College Publishers, 1988.

[Stuerzlinger1999] Wolfgang Stuerzlinger. Imaging All Visible Surfaces. Proceedings of the 1999 Conference on Graphics Interface, pp. 115–122, June 1999.

[Surmann2003] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. An Autonomous Mobile Robot with a 3D Laser Range Finder for 3D Exploration and Digitalization of Indoor Environments. Robotics and Autonomous Systems, Vol. 45, pp. 181–198, 2003.

[Tarabanis1991] Konstantinos A. Tarabanis, and Roger Y. Tsai. Computing Viewpoints that Satisfy Optical Constraints. Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR '91), pp. 152–158, June 1991.

[Tarabanis1995] Konstantinos A. Tarabanis, Peter K. Allen, and Roger Y. Tsai. A Survey of Sensor Planning in Computer Vision. IEEE Transactions on Robotics and Automation, 11(1):86–104, February 1995.

[Tarabanis1995a] Konstantinos A. Tarabanis, Roger Y. Tsai, and Peter K. Allen. The MVP Sensor Planning System for Robotic Vision Tasks. IEEE Transactions on Robotics and Automation, 11(1):72–85, February 1995.

[Tarabanis1996] Konstantinos A. Tarabanis, Roger Y. Tsai, and Anil Kaul. Computing Occlusion-Free Viewpoints. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 18(3):279–292, March 1996.

[Tarbox1995] G.H. Tarbox and S.N. Gottschlich. Planning for Complete Sensor Coverage in Inspection. Computer Vision and Image Understanding 61(1):84–111, January 1995.

[Taubin1995] Gabriel Taubin. A Signal Processing Approach to Fair Surface Design. Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95), pp. 351–358, 1995.

[Teller1993] Seth Teller, and Pat Hanrahan. Global Visibility Algorithms for Illumination Computations. Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93), pp. 239–246, 1993.

[Thibault1987] William C. Thibault, and Bruce F. Naylor. Set Operations on Polyhedra Using Binary Space Partitioning Trees. Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87), pp. 153–162, July 1987.

[Trucco1998] Emanuele Trucco, and Alessandro Verri. Introductory Techniques for 3-D Computer Vision. Prentice Hall, 1998.

[Turk1994] Greg Turk, and Marc Levoy. Zippered Polygon Meshes from Range Images. Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '94), pp. 311–318, 1994.

[Walpole1993] Ronald E. Walpole, and Raymond H. Myers. Probability and Statistics for Engineers and Scientists, Fifth Edition. Prentice Hall, 1993.

[Wang2002] Jianning Wang, and Manuel M. Oliveira. Improved Scene Reconstruction from Range Images. Computer Graphics Forum (Proceedings of EUROGRAPHICS 2002), 21(3):521–530, 2002.

[Wang2003a] Jianning Wang, and Manuel M. Oliveira. A Hole Filling Strategy for Reconstruction of Smooth Surfaces in Range Images. Proceedings of the XVI Brazilian Symposium on Computer Graphics and Image Processing, pp. 11–18, October 2003.

[Wang2003b] Rui Wang and David Luebke. Efficient Reconstruction of Indoor Scenes with Color. Proceedings of the 4th International Conference on 3D Imaging and Modeling (3DIM '03), pp. 402–409, 2003.

[Whaite1990] Peter Whaite, and Frank P. Ferrie. From Uncertainty to Visual Exploration. Proceeding of the Third International Conference on Computer Vision (ICCV '90), pp. 690–697, December 1990.

[Whaite1991] Peter Whaite, and Frank P. Ferrie. From Uncertainty to Visual Exploration. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 13(10):1038–1049, October 1991.

[Whaite1994] Peter Whaite, and Frank P. Ferrie. Autonomous Exploration: Driven by Uncertainty. Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR '94), pp. 339–346, 1994.

[Wilson2003] Andrew Wilson, and Dinesh Manocha. Simplifying Complex Environment Using Incremental Textured Depth Meshes. ACM Transactions on Graphics (Proceedings of SIGGRAPH 2003), 22(3):678–688, July 2003.

[Woodham1980] R. J. Woodham. Photometric Method for Determining Surface Orientation from Multiple Images. Journal of Optical Engineering, 19(1):138–144, 1980.

[Yi1990] S. Yi, R. M. Haralick, and L. G. Shapiro. Automatic Sensor and Light Source Positioning for Machine Vision. Proceedings of the 10th International Conference on Pattern Recognition, pp. 55–59, 1990.

[Zha1997] Hongbin Zha, Ken'ichi Morooka, Tsutomu Hasegawa, and Tadashi Nagata. Active Modeling of 3-D Objects: Planning on the Next Best Pose (NBP) for Acquiring Range Images. Proceedings of IEEE International Conference on Recent Advances in 3-D Digital Imaging and Modeling (3DIM '97), pp. 68–75, May 1997.

[Zhang1994] Zhengyou Zhang. Iterative Point Matching for Registration of Free-Form Curves and Surfaces. International Journal of Computer Vision, 13(2):119–152, 1994.

[Zhang1997] Hansong Zhang, Dinesh Manocha, Tom Hudson, and Kenneth E. Hoff III. Visibility Culling using Hierarchical Occlusion Maps. Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97), pp. 77–88, 1997.)