Computer Vision Techniques for Complete 3D Model Reconstruction

A Dissertation Presented

by

Huei-Yung Lin

to

The Graduate School

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

State University of New York
at Stony Brook

May 2002

State University of New York
at Stony Brook

The Graduate School

Huei-Yung Lin

We, the dissertation committee for the above candidate for the

Doctor of Philosophy degree,

hereby recommend acceptance of this dissertation.

Muralidhara Subbarao, Advisor, Professor
Department of Electrical and Computer Engineering

Armen H. Zemanian, Chairman, Distinguished Professor
Department of Electrical and Computer Engineering

Petar M. Djurić, Professor
Department of Electrical and Computer Engineering

Joseph S. B. Mitchell, Professor
Department of Applied Mathematics and Statistics

This dissertation is accepted by the Graduate School.

Dean of the Graduate School

Abstract of the Dissertation
Computer Vision Techniques for Complete 3D Model Reconstruction

by

Huei-Yung Lin

Doctor of Philosophy

in

Electrical and Computer Engineering

State University of New York
at Stony Brook

2002

This dissertation addresses the problem of automatic 3D model reconstruction of real objects. It has a number of applications in both computer vision and computer graphics areas such as industrial inspection, reverse engineering, Internet Web content and E-commerce. Current 3D modeling techniques require significant manual intervention and use expensive hardware systems for data collection. In this research, we present a complete and low-cost digital vision system to create photo-realistic 3D models. In our approach, the reconstruction of 3D models involves four major steps: 1) data acquisition, 2) registration, 3) surface integration, and 4) texture mapping. Partial 3D shapes and texture information are acquired from multiple viewpoints using rotational stereo and shape from focus (SFF). Rotational stereo model is first introduced in this work to acquire the depth information. Rotational stereo provides the flexibility of controlling the effective stereo baseline and it can incorporate existing stereo matching techniques. It provides a fast and accurate approach for 3D shape recovery. The resulting range images are registered

to a common coordinate frame according to the acquisition viewpoints. The accuracy of the initial registration is improved by finding the rotation and translation matrices iteratively by a least-squares error minimization approach. The registered range data are then integrated into a surface model using three different approaches. A new algorithm named Region-of-Construction is developed for fast surface integration. It directly exploits the structure of the raw range images and determines regions corresponding to non-redundant surfaces which can be stitched along the boundaries to construct the complete 3D surface model. The algorithm is computationally efficient and has low sensitivity to noise and registration error. A final photo-realistic 3D model is obtained by mapping the texture information recovered by SFF onto the complete surface model representing 3D shape. A digital vision system has been built and the algorithms have been implemented on the system. The results of 3D model acquisition for several real objects are presented.

To Chih-Fen

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my sincere gratitude to my advisor Professor Murali Subbarao for suggesting this topic to me and for his guidance and inspiration during the entire period of my research. I have greatly benefited from his support and enthusiasm over the past four years. In the long way towards my dissertation, he encouraged creative thinking, gave me necessary freedom and offered his advice at each step of the research. I am also grateful to Professors Armen Zemanian, Petar Djurić and Joseph Mitchell for serving on my thesis committee and for their valuable suggestions and comments.

I would like to express my appreciation to my colleagues in Computer Vision Laboratory. In particular, thanks to Ta Yuan for his generous help in the early stage of this research; and thanks to Soon-Yong Park for setting up the hardware systems and the valuable discussions about stereo vision. I also thank Mrs. Maria Krause and Mrs. Deborah Kloppenburg, for their kindness and help during my study at Stony Brook.

Finally, I would like to thank my family. My parents, Hsiu-Hsiung and Mei-Hwei have always been a source of constant support and encouragement. My son, Robert has brought me joyful moments during my study. My last and final thanks go to my wife, Chih-Fen. She has made my life happy, exciting and fun. Without her I would have had many more stressful and worrisome moments.

The support of this research in part by the Olympus Optical Corporation is gratefully acknowledged.

Chapter 1

Introduction

1.1 Problem Statement

Reconstructing 3D computer models from existing objects is an important problem in many computer vision research areas such as reverse engineering (reverse CAD), pattern recognition, and industrial inspection. More recently, as a result of the availability of fast, inexpensive graphics hardware and technologies such as VRML-ready Internet browsers, the reconstruction of 3D models has become one of the most interesting subjects in both computer vision and computer graphics applications. However, in the past few decades 3D models of real objects were often created manually by users. This process is usually time-consuming and expensive. Therefore, techniques using low-cost equipments to obtain 3D models automatically from real objects could have great significance in practical applications.

In this dissertation we present a complete system for reconstructing 3D models with both geometric and photometric information of real objects. The problems addressed in this dissertation include:

- How can we get the 3D shape as well as the corresponding texture information of an object using a low-cost camera system instead of laser range scanners?

- Since a viewpoint can reveal only the information in the scene that is visible from the viewpoint, complete information has to be acquired from several viewing directions. How can we obtain a complete description of those data points in a common coordinate system?

- When combining partly overlapped information from different acquisition viewpoints, how can we remove redundant data and create a single surface

representation? How can we create the complete 3D model without any loss of detail in the original raw data?

- Given the wireframe model (with only geometric information) of an object, how can we map the acquired color images on its surface to create a photo realistic 3D model?

1.2 Applications

This research is motivated by a number of applications including consumer marketing, Web and Internet applications, manufacturing, and virtual simulation. Some application areas that would benefit from an efficient and reliable methods for 3D model reconstruction are:

- **Internet applications**

  Internet applications such as e-commerce can benefit from 3D model reconstruction for their products. The 3D model of an object can be displayed on a web browser with arbitrary viewpoint interactively chosen by a user instead of some fixed viewpoints. A format called VRML (Virtual Reality Modelling Language) has become the industrial standard for visualizing 3D models.

- **Reverse engineering**

  Computer Aided Design (CAD) tools are currently used to design and manufacture physical objects such as machine parts. However, some custom-made or existing old mechanical parts may not have the CAD models. If a replica is needed, one has to start from an existing object, reconstruct a computer model and then use it to reproduce the object.

- **Industrial inspection**

  3D models of manufactured objects can be reconstructed automatically and the data can be compared to the specifications rather than semi-automatic metrology techniques. Detected deviations from the reconstructed models can be used to calibrate a new manufacturing process and control the quality of final manufactured parts.

- **3D fax**

Scan an object and transmit the digitized data on a phone line. The replica is then realized at the receiving station by reconstructing the 3D model using a rapid prototyping technique such as stereo-lithography[1].

- **Authoring virtual environments**

  Creating virtual environments for entertainment such as video games, and other graphics applications requires models of real-world objects. The realistic models can be quickly built from their real counterparts or from sculptures created by artists.

1.3 Our Solution to the Problem

Our solution to the 3D model reconstruction problem is illustrated in Figure 1-1. It consists of four key stages [17]: (i) data acquisition, (ii) registration, (iii) surface integration, and (iv) texture mapping. A prototype vision system, Stonybrook VIsion System 2 (SVIS-2), is built and implemented to acquire the image sequences for 3D shape and focused image recovery. The information from different viewpoints is obtained by rotating the object placed on a rotation stage. The 3D shape and texture information are then used to construct the complete and photo realistic 3D model.

1.3.1 Data Acquisition

Data acquisition stage is to obtain the range and intensity images for different viewpoints. The object is placed on a rotation stage in front of a stationary camera. Input image sequences from different viewpoints are acquired by rotating the stage with known rotation angles. The rotation matrix and translation vector of the rotation axis are calibrated for 3D shape recovery. For each viewpoint, two sequences of images with different focus setting are used to construct a focused stereo image pair by *shape from focus* (SFF). The stereo image pair is then used to recover the 3D shape by *rotational stereo*. This stage provides the partial 3D shape and the corresponding texture map of a viewpoint. The partial 3D models (which include both 3D shape and texture information) from different viewpoints are obtained by rotating the object with known rotation angles and repeating the same acquisition process.

[1] Stereo-lithography is a “rapid-prototyping” process which produces a physical, three-dimensional object from a 3D CAD file. A stereolithography machine uses a computer controlled laser to cure a photo-sensitive resin, layer by layer, to create the 3D part.

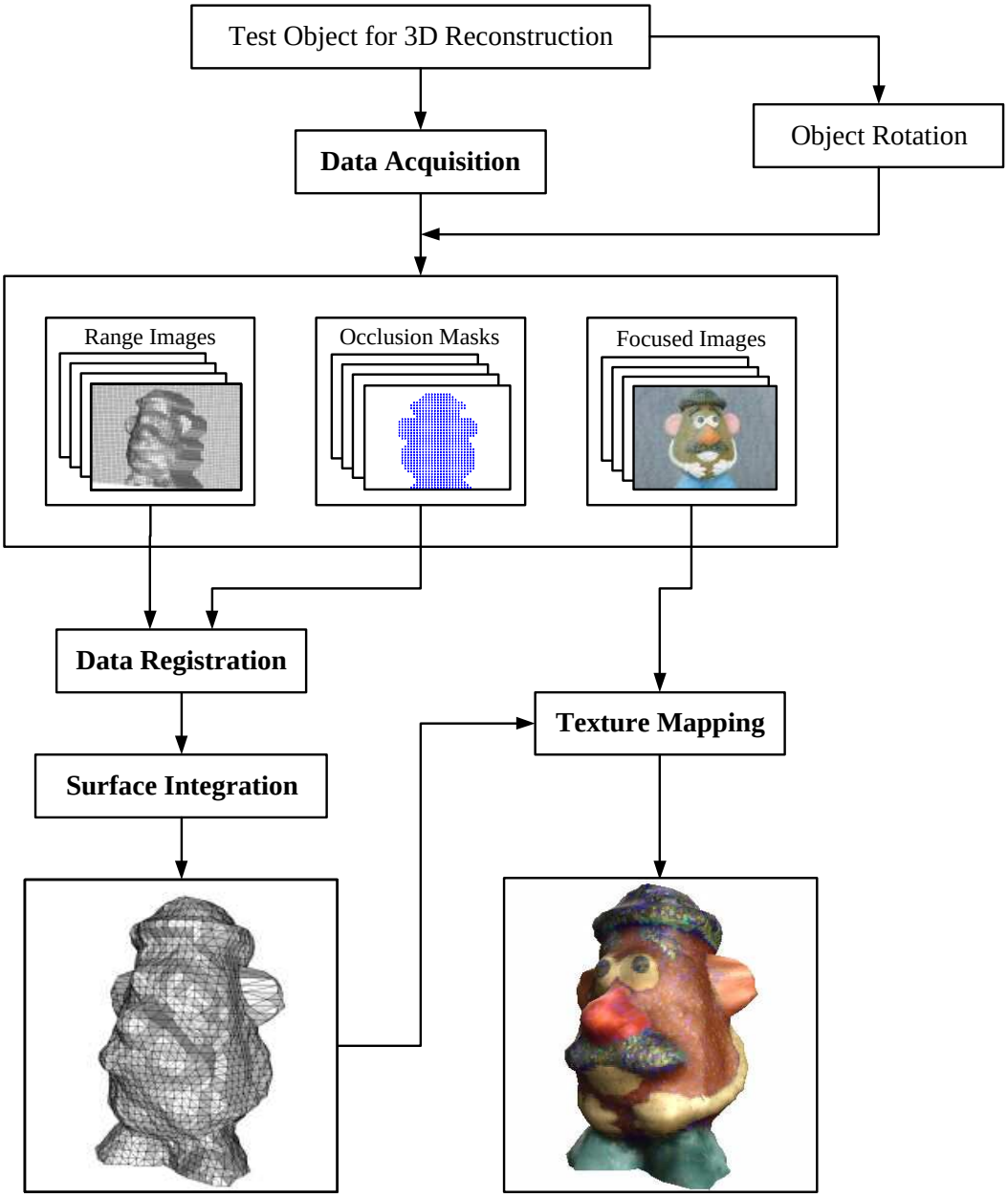


Figure 1-1: System flow chart of complete 3D model reconstruction

In our more recent work , both the accuracy and precision of 3D shape are greatly improved by *multiple base-angle rotational stereo* combined with multi-resolution stereo matching. Similar to a multiple-baseline stereo [57], a sequence of four images with different stereo rotation angles are taken instead of a stereo image pair. Error analysis indicates that we are able to extract depth with an average error of less than 1 mm error in a working range of 700 mm to 900 mm from the camera.

1.3.2 Registration

The range images (partial 3D shapes) obtained from data acquisition stage have to be registered to a common coordinate frame for integration. To register a pair of range images, we have to find the rigid 3D transformation between them so that the overlapping areas of different range images covering the same part of the object surface are aligned with each other. Assuming the rotation angle between two viewing directions is known, finding the transformation is equivalent to finding the rotation axis of the rotation stage. Given the initial rotation matrix and translation vector from system calibration, two registration methods are proposed to refine the rotation axis. The first method searches the rotation centers on object's cross sections by minimizing the least squared error on global resampled data points. The second method directly computes the new rotation matrix and translation vector using the selected *matching points* after current registration. The rotation axis is updated iteratively until it converges.

1.3.3 Surface Integration

Given a set of registered range images, an integration algorithm is presented for combining the partly overlapping data sets into a complete non-redundant 3D data set without loss of details in the original raw data. Three computational algorithms, integration by global resampling, slice-by-slice algorithm and *Region-of-Construction* algorithm, are developed for integrating the range images to a complete 3D model. The first algorithm is based on (r, θ, y) space representation. It uniformly resamples the registered range images on both θ and y directions. The resolution of 3D model can be controlled by sampling rate and higher order interpolation. Slice-by-slice algorithm is commonly used to construct 3D model from a series of contours. It uses all of the data points provided by data acquisition stage and gives a dense mesh of the 3D model.

A new surface integration algorithm based on *Region-of-Construction* is developed. It uses only part of each range image to *stitch* the partial 3D models from different viewpoints and takes advantage of the known topology of each region-of-construction to perform fast triangulation. Because continuous regions from same

range images are used for integration, the registration error is always limited to the boundaries of regions-of-construction. This algorithm is also extended to construct complex objects with a hole. This integration method is computationally efficient in the sense that no searching is required for mesh triangulation.

1.3.4 Texture Mapping

The color images used for texture mapping are constructed by shape from focus with several different focus positions (typically 4 different focus steps). Having a 3D wireframe model created in the previous stage, image texture of an object is modeled in two ways. In the first method image texture of each polygon is specified by the projected pattern on one of the focused images. The projection of polygon can be either on the same viewpoint as where the 3D points come from, or on the viewpoint according to the best viewing direction determined by the surface normal of the polygon. In the second method we provide a whole (individual) image for each partial 3D model to map the texture and then stitch the textured partial 3D models together. Bounding boxes just enclosing each region-of-construction are used to extract the image texture to reduce the size of the textured 3D model. Depending on the quality requirement, the images are down-sampled to further reduce the size of the 3D model.

1.4 Dissertation Overview

This dissertation is organized as follows. In Chapter 2, we deal with data acquisition of an object from fixed viewpoints. The theoretical backgrounds of shape from focus and rotational stereo are described, followed by the implementation on our SVIS-2 camera system. Experimental results and accuracy analysis are given for partial 3D shapes. In Chapter 3, we first discuss the previous work on range image registration and then describe our registration methods. In Chapter 4, we begin with the review of some previous work on 3D model reconstruction. Then we describe the surface integration algorithms developed in this dissertation. We also demonstrate the ability of our algorithms for dealing with complex objects with a hole and multiple objects. The accuracy of our methods is evaluated by testing on a simple object. In Chapter 5, we describe the algorithms for mapping texture onto the surface of 3D models. The final results of reconstructed 3D models with texture map are shown in this chapter. Finally, we summarize the contribution of this research and indicate some avenues for future work in Chapter 6.

Chapter 2

Data Acquisition

2.1 Introduction

3D model of an object consists of two types of information– (i) the 3D shape of the object (geometric information), and (ii) the image texture on the outer visible surface of the object (photometric information). Recovering the first type of information (3D shape) is a difficult problem in the computer vision area [42, 2, 34]. Some popular techniques are shape from shading, shape from motion, shape from focus, structured light analysis, photometric stereo, etc [86]. In addition, there also exist active sensing devices such as range finders, which can measure the 3D shape directly [9]. As for the second type of information (image texture), it is usually recovered from the image recorded by a camera.

Most researchers in 3D model reconstruction area acquire the 3D input data using active range acquisition methods such as laser range scanner [21, 17, 66, 75] or computer tomography (CT) [6, 13]. These devices usually provide dense and highly accurate range data. However, using these active methods has several disadvantages– (i) they are usually expensive, (ii) acquiring data set is a time-consuming process, and (iii) the acquired range data usually do not provide the texture information[1] and therefore the intensity image has to be acquired separately.

In this chapter we describe our solutions to acquiring both range and intensity images simultaneously. A passive camera system, Stonybrook VIsion System 2 (SVIS-2) is built for acquiring color images. The intensity images not only provide the texture information but they are also used to recover the 3D shape of the object. The object is placed on a rotation stage in front of a stationary camera. The partial 3D shape and the corresponding texture map of the object are recovered using *rotational stereo* [49, 50] and *shape from focus* (SFF) [82]. Two sequences of images

[1] Although some sensors provide the intensity values for each data point [9], they are usually not dense enough to generate realistic output.

with different focus positions are taken with a small rotation angle to obtain stereo image pairs. The stereo rotation angle is adjusted by a PC controlled motor. Each sequence of images is used to construct the focused image and a rough depth map by SFF. A more accurate 3D shape is then obtained by matching the focused image pair using rotational stereo model with a matching constraint provided by rough depth maps.

In Sections 2.2 and 2.3, we provide the theoretical background of shape from focus and rotational stereo, respectively. The calibration of rotation axis is given in Section 2.3.1. Section 2.4 describes the experimental setup of our SVIS-2 camera system and the detailed algorithms of the implementation. Section 2.5 shows some results of recovered partial 3D shapes and focused images of several test objects. Error analysis of our camera system is also given in this section. Section 2.6 describes a multiple base-angle approach for rotational stereo followed by conclusion in Section 2.7.

2.2 Shape from Focus (SFF)

In SFF, a large sequence of image frames of a 3D scene is recorded with different camera parameters (e.g. focal length or/and lens to image detector distance) [87]. In each image frame, different objects in the scene will be blurred by different degrees depending on their distance from the camera lens. Each object will be in best focus in only one image frame in the image sequence. The entire image sequence is processed to find the best focused image of each object in the 3D scene. The distance of each object in the scene is then found from the camera parameters that correspond to the image frame that contains the best focused image of the object. The SFF methods are based on the fact that for an aberration-free convex lens, (i) the radiance at a point in the scene is proportional to the irradiance at its *focused image* [42] (photometric constraint), and (ii) the position of the point in the scene and the position of its focused image are related by the *lens formula* (geometric constraint)

$$\frac{1}{f} = \frac{1}{u} + \frac{1}{v} \tag{2.1}$$

where f is the focal length, u is the distance of the object from the lens plane, and v is the distance of the focused image from the lens plane (see Figure 2-2).

Given the irradiance and the position of the focused image of a point, its radiance and position in the scene are uniquely determined. In a sense, the positions of a point-object and its image are *interchangeable,* i.e. the image of the image is the object itself. Now, if we think of an object surface in front of the lens to be comprised of a set of points, then the focused images of these points define another

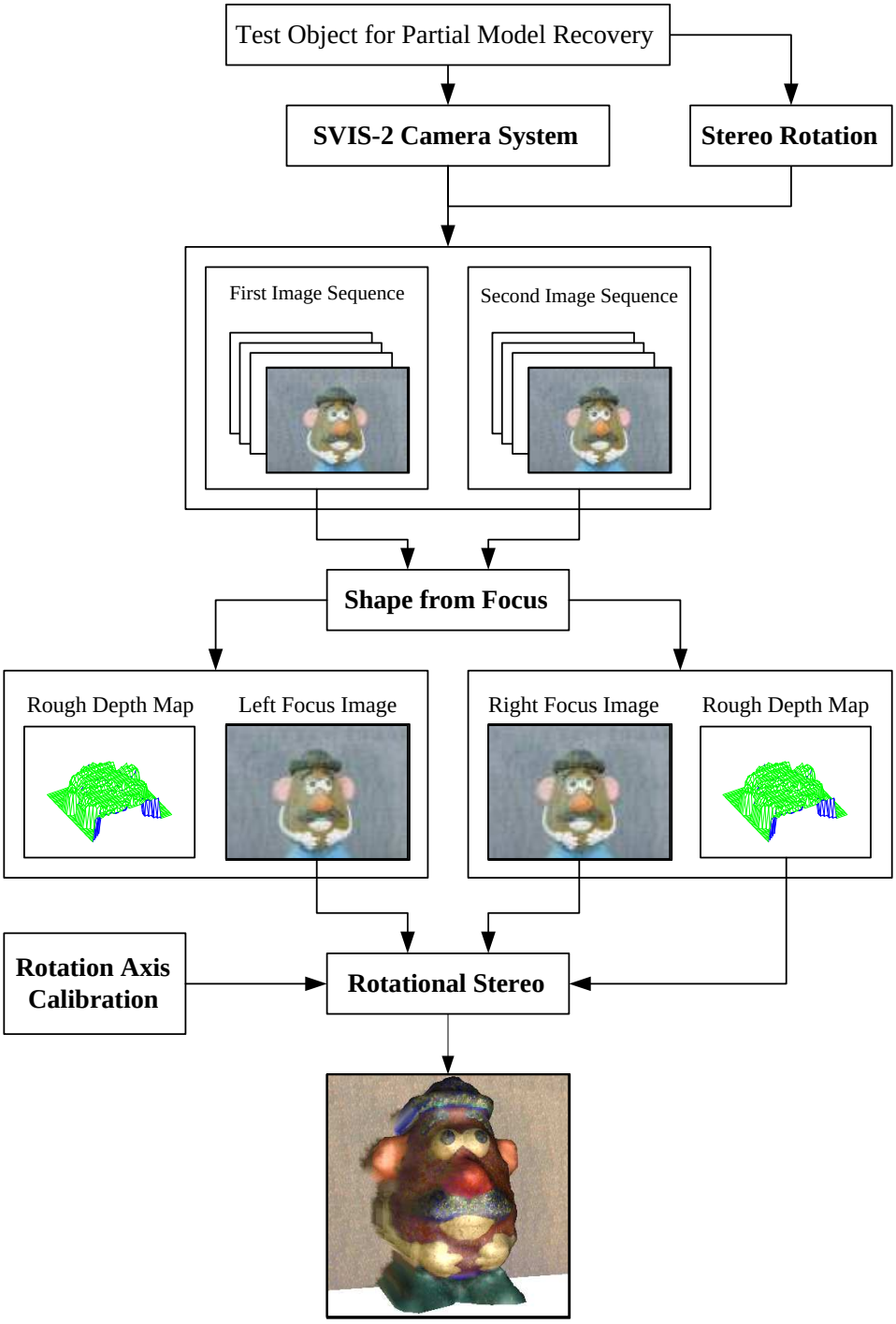


Figure 2-1: Block diagram of data acquisition

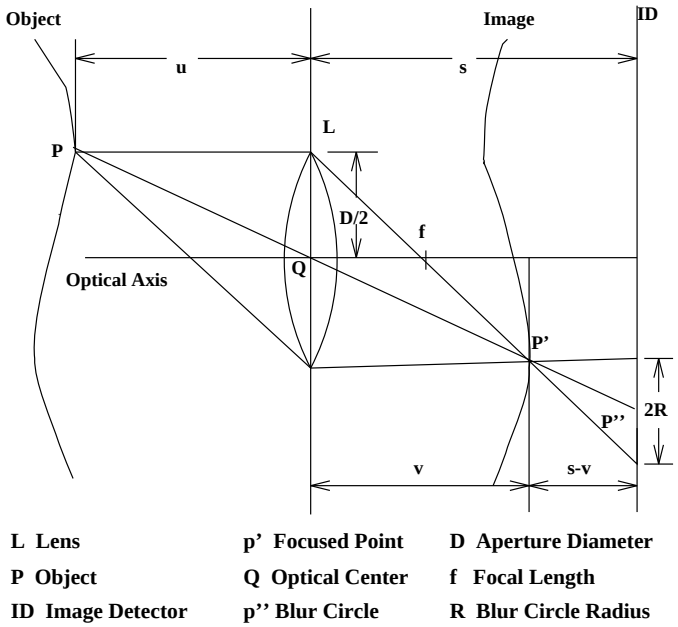


Figure 2-2: Image formation in a convex lens

surface behind the lens (see Figure 2-2). This surface is defined to be the *Focused Image Surface* (FIS) and the image irradiance on this surface to be the *focused image.* There is a *one to one correspondence* between the FIS and the object surface. The geometry (i.e. the 3D shape information) and the radiance distribution (i.e. the photometric information) of the object surface are uniquely determined by the FIS and the focused image. In traditional SFF methods (e.g. [46, 82, 81]) a sequence of images are obtained by continuously varying the distance s between the lens and the image detector or/and the focal length f (see Figure 2-3).

For each image in the sequence, a focus measure is computed at each pixel (i.e. each direction of view) in a small (about 15×15) image neighborhood around the pixel. At each pixel, that image frame among the image sequence which has the maximum focus measure is found by a search procedure. The grey level (which is proportional to image irradiance) of the pixel in the image frame thus found gives the grey level of the focused image for that pixel. The values of s and f for this image frame are used to compute the distance of the object point corresponding to the pixel. An example of a focus measure is the grey level variance. SFF methods involve a search for the values of s or/and f that results in a maximum focus measure and these methods require the acquisition and processing of a large number of images.

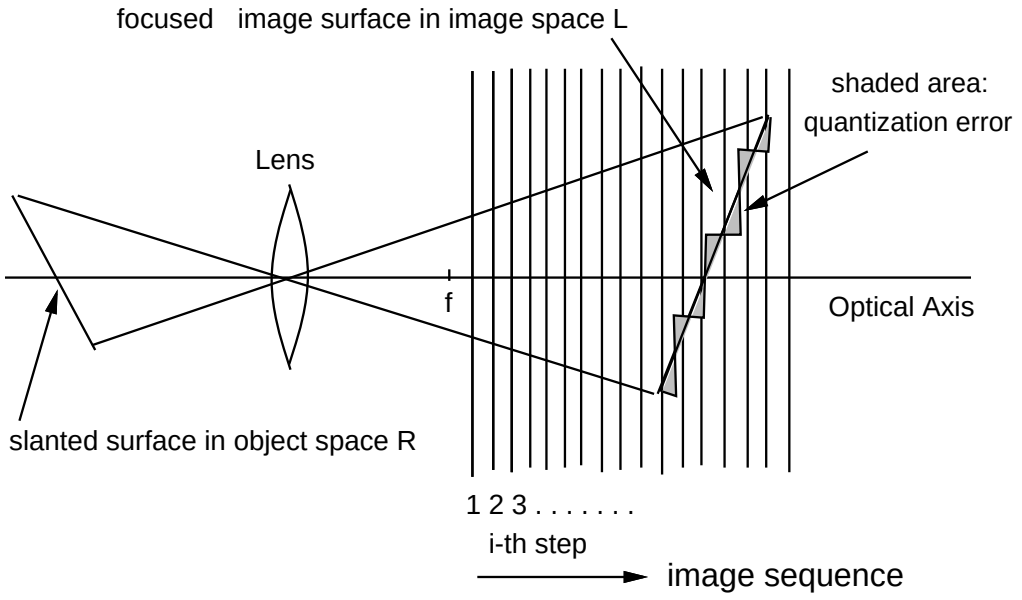


Figure 2-3: Focused image surface

2.3 Rotational Stereo

2.3.1 Shape from Stereo

Shape from stereo is one of the widely studied topics in computer vision [22, 4]. The basic principle involved in the recovery of depth is triangulation. In stereopsis, there are two major problems to be solved. One is the *correspondence problem*– given two (or more) images of a scene, which parts of them are projections of the same scene elements? This is also know as *stereo matching*– finding the corresponding points of the image pair. The other problem is the *reconstruction problem*– given a number of corresponding parts of an image pair, how to determine the 3D location and structure of the observed object?

Stereo matching is the most important stage in stereo computation. A number of different techniques have been proposed. They can be classified into two groups, *area-based* and *feature-based* methods. Area-based stereo approaches use correlation among brightness (intensity) patterns in the local neighborhood of a pixel in one image with brightness patterns in a corresponding neighborhood of a pixel in the other image. For a fixed point in one image, a cross-correlation measure is used to search for a point with a matching neighborhood in the other image. Feature-based approaches restrict the search for correspondence using symbolic features

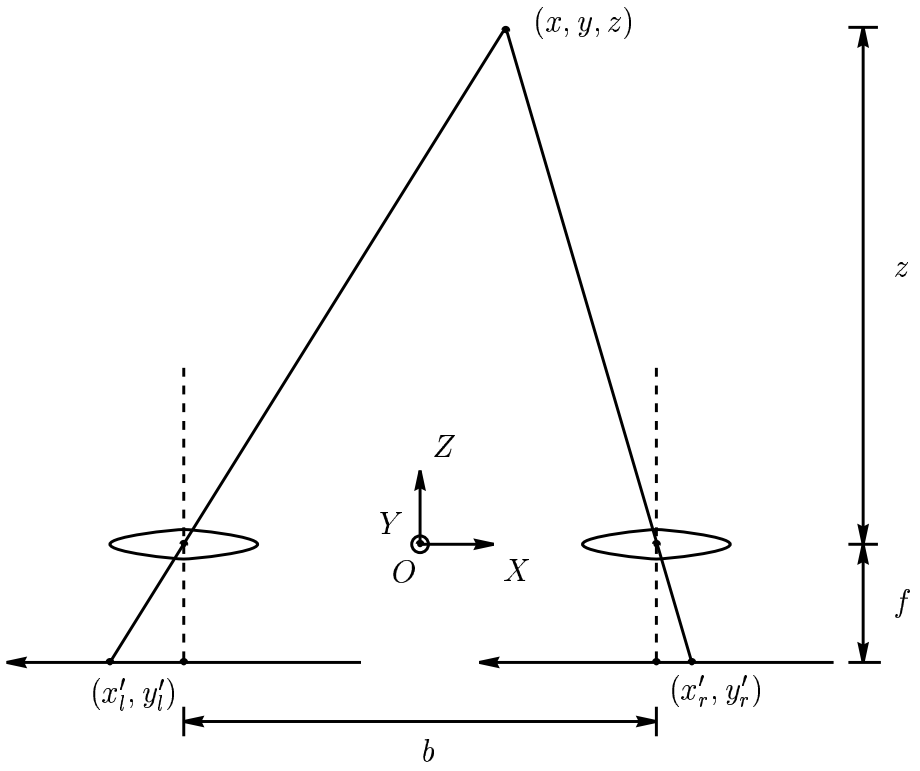


Figure 2-4: Conventional stereo system with parallel camera configuration

derived from intensity images. Corresponding elements are given by the most similar feature pair. The features used most commonly are either edge points or edge segments derived from connected edge points.

The stereo matching method used in SVIS-2 camera system is an area-based technique. For each small image area in the right image, the best match in the left image is found by minimizing the Sum-of-Squared-Difference (*SSD*) measure. Let (i, j) be the point of interest and $2k + 1$ be the matching window size, then *SSD* is given by

$$SSD(i,j) = \sum_{m=-k}^{k} \sum_{n=-k}^{k} |f_r(i,j) - f_l(i+m, j+n)|^2 \tag{2.2}$$

where f_r and f_l are image gray-levels of the right and left images, respectively.

Assuming both the intrinsic and extrinsic parameters of a stereo system are known, the depth reconstruction problem can be solved unambiguously by triangulation. Conventional stereo system uses a pair of cameras with their optical axes

mutually parallel and separated by a horizontal distance denoted as the stereo baseline b. The cameras have their optical axes perpendicular to the stereo baseline, and their image scanlines parallel to the baseline. A two-dimensional model is shown in Figure 2-4. Let image coordinate systems (x_l, y_l) and (x_r, y_r) be defined on the left and right image plane with origin on the optical axis respectively. By considering similar triangles, the world coordinates of the scene point (x, y, z) can be easily obtained as

$$x = \frac{b(x'_l - x'_r)}{2d}, \quad y = \frac{b(y'_l - y'_r)}{2d}, \quad \text{and} \quad z = \frac{bf}{d} \tag{2.3}$$

where d is the *disparity* defined as $x'_l - x'_r$ and f is the effective focal length of the camera.

In the data acquisition stage, since the object has to be rotated in order to obtain the information from different viewpoints, a small rotation angle can be used to create a stereo image pair from a single camera. The acquired image pair forms a non-parallel axis stereo geometry and can be converted to the conventional stereo by image rectification [2]. However, in our approach it is more preferable to find the epipolar geometry pixelwise since we do not match each pixel on the images. Therefore, the computation is reduced by finding the epipolar lines of the pixels of interest according to the rotation transformation. We refer to this as *rotational stereo* [49, 50], or *vergence stereo*.

In our early implementation, we considered a special case of rotational stereo with rotation axis perpendicular to the plane determined by image scanlines and the optical axis. In this case the rotational stereo model is simplified to a verged stereo system as shown in Figure 2-5. The baseline b is defined as the distance between two camera optical centers. The angle between two camera optical axes is called *camera convergence angle* and denoted by β. Similar to the parallel stereo, the world coordinates of the scene point (x, y, z) can be obtained as

$$\begin{aligned}
x &= \frac{b\left\{\tan\left[\tan^{-1}\left(\frac{x'_r}{f}\right) - \frac{\beta}{2}\right] + \tan\left[\tan^{-1}\left(\frac{x'_l}{f}\right) + \frac{\beta}{2}\right]\right\}}{2\left\{\tan\left[\tan^{-1}\left(\frac{x'_r}{f}\right) + \frac{\beta}{2}\right] + \tan\left[\tan^{-1}\left(\frac{x'_l}{f}\right) - \frac{\beta}{2}\right]\right\}} \\
y &= \frac{y'_l\left\{\cos\left(\frac{\beta}{2}\right) + b\tan\left[\tan^{-1}\left(\frac{x'_l}{f}\right) + \frac{\beta}{2}\right]\sin\left(\frac{\beta}{2}\right)\right\}}{f\left\{\tan\left[\tan^{-1}\left(\frac{x'_r}{f}\right) + \frac{\beta}{2}\right] + \tan\left[\tan^{-1}\left(\frac{x'_l}{f}\right) - \frac{\beta}{2}\right]\right\}} \\
z &= \frac{b}{\tan\left[\tan^{-1}\left(\frac{x'_r}{f}\right) + \frac{\beta}{2}\right] + \tan\left[\tan^{-1}\left(\frac{x'_l}{f}\right) - \frac{\beta}{2}\right]}
\end{aligned} \tag{2.4}$$

Note that the above equations are simplified to Eqs. (2.3) when the convergence angle β equals 0.

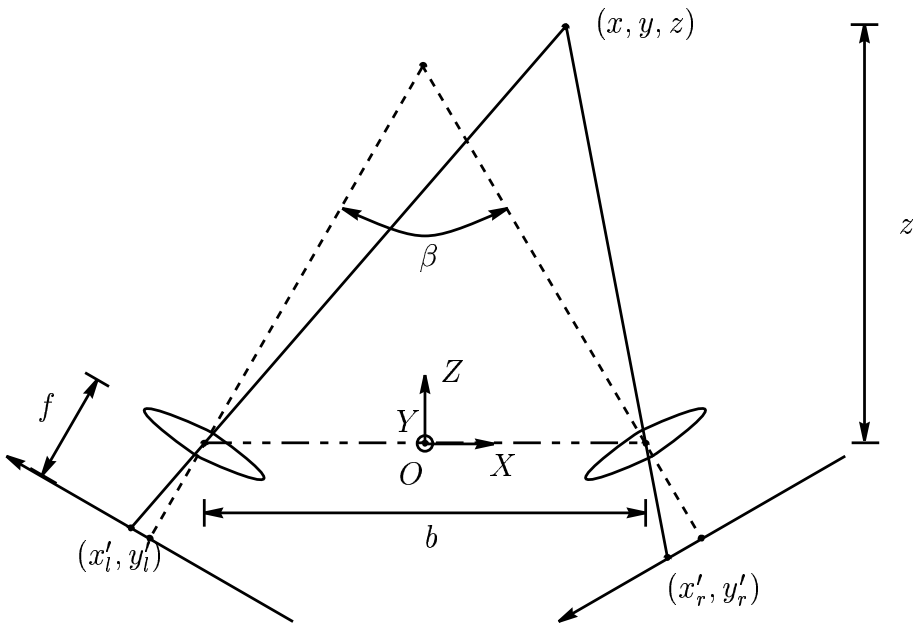


Figure 2-5: Conventional stereo system with verged camera configuration

Rotational stereo offers some important advantages compared to conventional stereo. A single camera is used instead of two, the stereo matching is easier as the field of view remains almost the same for the camera. The camera calibration is easier since only a single stationary camera is used. For a single camera system [94], stereo image pairs can be obtained without camera movement along the stereo baseline.

2.3.2 Rotational Stereo Model and Epipolar Geometry

The rotational stereo model with arbitrary rotation axis is shown in Figure 2-6. The rotation axis is described by the unit vector $\vec{u} = (u_1, u_2, u_3)$ and the translation vector $\vec{d} = (d_1, d_2, d_3)$ in the camera coordinate system. The image pair used for stereo matching is obtained by rotating the object an angle θ with respect to the rotation axis. Let (x_1, y_1, z_1) and (x_2, y_2, z_2) be the same object point before and after rotation, respectively. Let $(\hat{x}_1, \hat{y}_1)$, $(\hat{x}_2, \hat{y}_2)$ denote the corresponding projected points on the image plane. The images taken before and after rotation are referred to as the first and second image, or right and left image. For each pixel $(\hat{x}_1, \hat{y}_1)$ in the second (right) image, the corresponding *epipolar line* in the first (left) image is

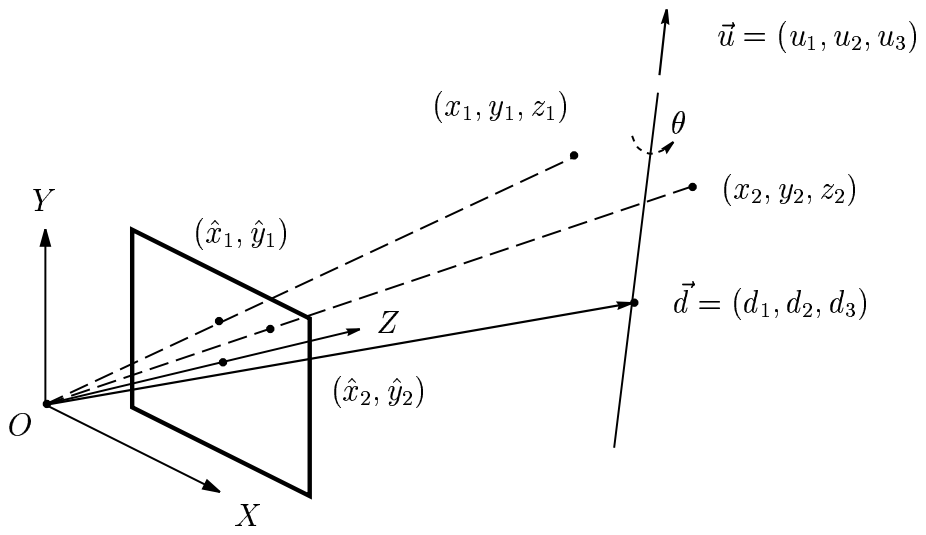


Figure 2-6: Rotational stereo model

calculated as follows (see Figure 2-7 and Figure 2-8).

In order to efficiently deal with arbitrary orientation of rotation axis with six degrees of freedom, the *quaternion representation* is used to define the rotation transformation (see Appendix A). Let $\vec{u}$ be a unit vector along the selected rotation axis and θ be the specified rotation angle about this axis, then the rotation matrix is given by [33]

$$\mathbf{M}_R(\theta) = \begin{pmatrix} 1-2b^2-2c^2 & 2ab-2sc & 2ac+2sb \\ 2ab+2sc & 1-2a^2-2c^2 & 2bc-2sa \\ 2ac-2sb & 2bc+2sa & 1-2a^2-2b^2 \end{pmatrix} \quad (2.5)$$

where

$$s = \cos\frac{\theta}{2} \quad \text{and} \quad \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \sin\frac{\theta}{2} \quad (2.6)$$

To compute the required rotation matrix about any rotation axis with a translation vector $\vec{d}$, we need to include the translation matrix

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & 0 & d_1 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2.7)$$

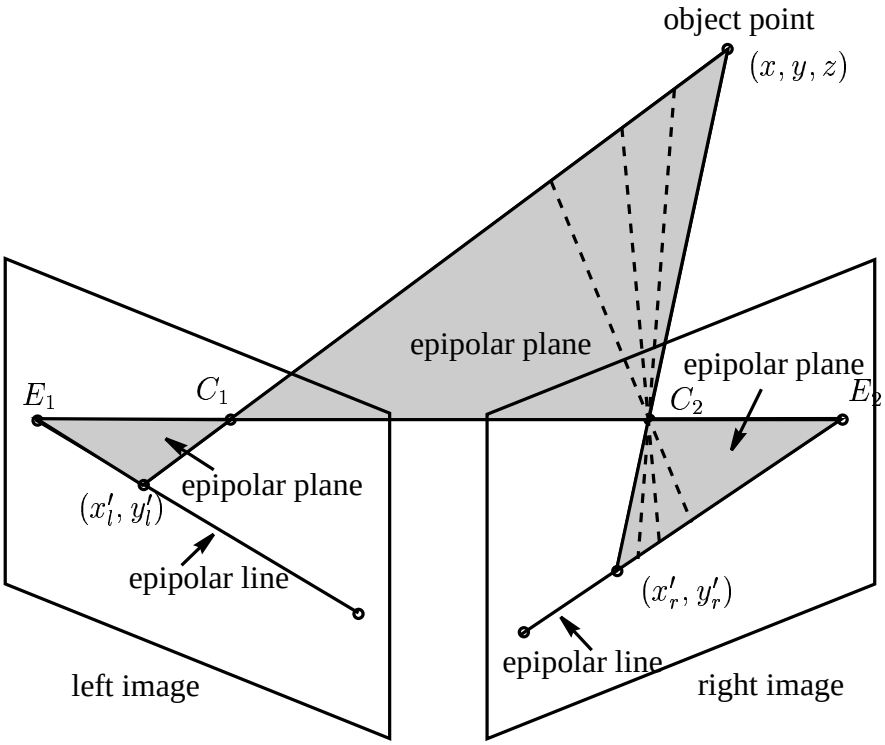


Figure 2-7: Epipolar geometry

Thus, the rotation matrix of any rotation axis is given by

$$\mathbf{R}(\theta) = \mathbf{T}^{-1} \cdot \begin{pmatrix} \mathbf{M}_R(\theta) & 0 \\ 0 & 1 \end{pmatrix} \cdot \mathbf{T} \tag{2.8}$$

According to the perspective projection, the relationship between an object point (x_i, y_i, z_i) and an image point $(\hat{x}_i, \hat{y}_i)$ can be written as

$$x_i = \hat{x}_i t_i, \quad y_i = \hat{y}_i t_i, \quad \text{and} \quad z_i = f t_i \tag{2.9}$$

for $i = 1, 2$ and t_i's are unknown scaling parameters. Hence, we have the equations

$$\begin{pmatrix} \hat{x}_2 t_2 \\ \hat{y}_2 t_2 \\ f t_2 \\ 1 \end{pmatrix} = \mathbf{R}(\theta) \cdot \begin{pmatrix} \hat{x}_1 t_1 \\ \hat{y}_1 t_1 \\ f t_1 \\ 1 \end{pmatrix} \tag{2.10}$$

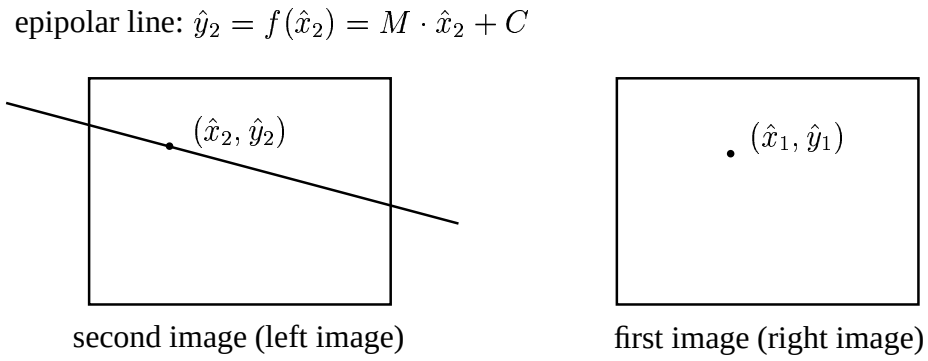


Figure 2-8: For each pixel on the right image, stereo matching is done along the pre-calculated epipolar line $\hat{y}_2 = M \cdot \hat{x}_2 + C$ on the left image.

for the same object point projected on different images. In the above equations, there are four unknowns $\hat{x}_2, \hat{y}_2, t_1, t_2$, but only three independent equations. They can be used to find the relationship between $\hat{x}_2$ and $\hat{y}_2$. Therefore, the epipolar line equation on the left image is solved as

$$\hat{y}_2 = M \cdot \hat{x}_2 + C \tag{2.11}$$

where

$$\begin{aligned} M &= \frac{-\hat{y}_1 \left[d_1 (sb - ac) + d_2 (-sa - bc) + d_3 (a^2 + b^2)\right]}{X_1 + Y_1 + C_1} \\ C &= \frac{-\hat{y}_1 f \left[d_1 (-c^2 - b^2) + d_2 (-sc + ab) + d_3 (ac + sb)\right]}{X_1 + Y_1 + C_1} \end{aligned}$$

and

$$\begin{aligned} X_1 &= \hat{x}_1 \Big[d_1 (ac - sb) + d_2 \Big(sa - 2b^3 c + bc - 2bc^3 - 2a^2 bc - 2s^2 cb\Big) \\ &\quad + d_3 \Big(2b^4 - b^2 - a^2 + 2s^2 b^2 + 2b^2 c^2 + 2a^2 b^2\Big)\Big] \\ Y_1 &= \hat{y}_1 \Big[d_1 \Big(2a^2 cb + 2b^3 c + 2c^3 b + 2s^2 bc\Big) \\ &\quad + d_3 \Big(-2s^2 ba - 2a^3 b - 2b^3 a - 2ac^2 b\Big)\Big] \\ C_1 &= f \Big[d_1 \Big(-2b^4 - 2s^2 b^2 + b^2 + c^2 - 2c^2 b^2 - 2b^2 a^2\Big) \\ &\quad + d_2 \Big(2bc^2 a + 2s^2 ab + 2a^3 b + 2ab^3 + sc - ab\Big) + d_3 (-sb - ac)\Big] \end{aligned}$$

Given the rotation angle θ, and the translation and direction vectors of the rotation axis, both parameters M and C are constants for any fixed $(\hat{x}_1, \hat{y}_1)$.

In the rotational stereo, the epipolar line equation (2.11) is computed only for the pixels to be used for stereo matching. The matching is then done along the given equation in the first image for every fixed point $(\hat{x}_1, \hat{y}_1)$ in the second image. Since we are only interested in the foreground (object) region for stereo matching, it is more computationally efficient compared to image rectification, which usually processes the whole image.

Now let's consider a special case when the rotation axis is "vertical", i.e. it intersects the camera optical axis and is perpendicular to both optical axis and image scanlines. The rotational stereo model is simplified to two-dimensional case. It is equivalent to a stereo system with verged camera configuration (see Figure 2-9). The convergence angle is the same as the object rotation angle. The stereo baseline is determined by the rotation angle and the distance from camera to the object rotation axis. In this case, the translation and direction vectors of the rotation axis are given by $\vec{u} = (0, 1, 0)$ and $\vec{d} = (0, 0, d_3)$, and the epipolar line equation can be simplified as

$$\hat{y}_2 = \frac{\hat{y}_1 (\cos\theta - 1)}{(1 - \cos\theta)\hat{x}_1 + f \sin\theta}\hat{x}_2 + \frac{\hat{y}_1 f \sin\theta}{(1 - \cos\theta)\hat{x}_1 + f \sin\theta} \tag{2.12}$$

or

$$\hat{y}_2 = m \cdot \hat{x}_2 + c \tag{2.13}$$

where

$$m = \frac{\hat{y}_1 (\cos\theta - 1)}{(1 - \cos\theta)\hat{x}_1 + f \sin\theta} \tag{2.14}$$

and

$$c = \frac{\hat{y}_1 f \sin\theta}{(1 - \cos\theta)\hat{x}_1 + f \sin\theta} \tag{2.15}$$

It can be seen that the resulting epipolar line is independent of d_3, which is the distance between the camera and the rotation axis. This result also gives us a good approximate solution to the case when the rotation axis is almost *vertical* with less computation.

2.3.3 Rotation Axis Calibration

In this section, we describe a simple 3-point calibration method to find the rotation axis of the rotational stereo model. Although only 3 points are needed in this calibration, it can be extended to use $3N$ points to obtain a more accurate result by taking the average. The calibration takes three images of a planar object with three fixed points on it. Three images are taken with different rotation angles, 0, θ and 2θ degrees, of the planar object. The known distances between any two points

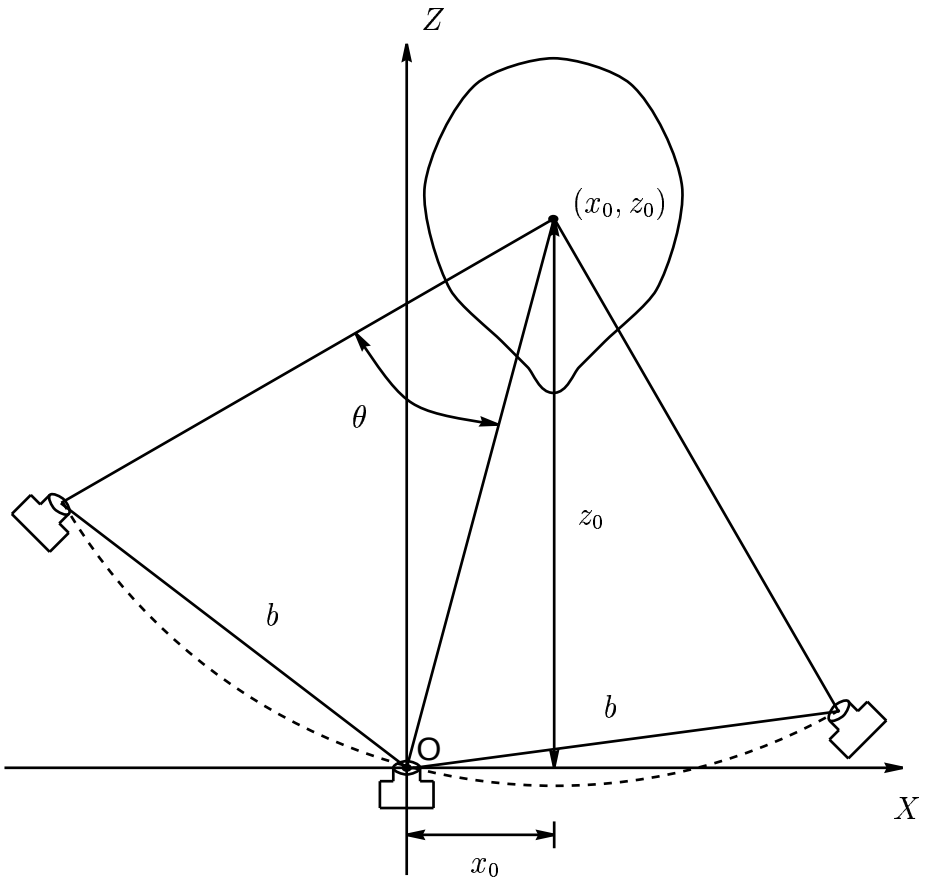


Figure 2-9: Equivalent verged camera configuration of rotational stereo with rotation axis perpendicular to image scanlines and camera optical axis.

on the object are used to determine the 3D points. Let $\mathbf{p}_i$ refer to the image point of an object point $\mathbf{P}_i$, where $\mathbf{p}_i = (\hat{x}_i, \hat{y}_i, f)$ and $\mathbf{P}_i = (x_i, y_i, z_i)$ for $i = 1, 2, 3$. Then we have

$$\left\| \frac{z_1}{f}\mathbf{p}_1 - \frac{z_2}{f}\mathbf{p}_2 \right\| = d_{12}, \tag{2.16}$$

$$\left\| \frac{z_2}{f}\mathbf{p}_2 - \frac{z_3}{f}\mathbf{p}_3 \right\| = d_{23}, \tag{2.17}$$

$$\left\| \frac{z_3}{f}\mathbf{p}_3 - \frac{z_1}{f}\mathbf{p}_1 \right\| = d_{31} \tag{2.18}$$

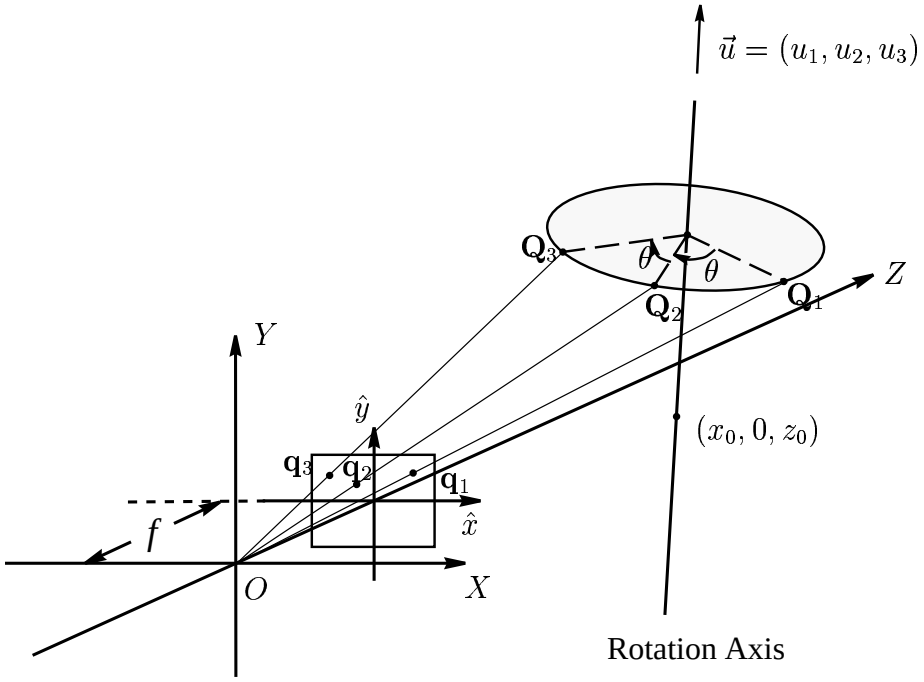


Figure 2-10: 3-point rotation axis calibration

where f is the focal length, d_{ij} is the distance between two points $\mathbf{p}_i$, $\mathbf{p}_j$, and z_i is the depth of object point $\mathbf{p}_i$. In the above equations, three unknowns z_1, z_2, z_3 have more than one solution, but can be reduced to a unique solution by examining the physical constrains. The solution of z_1, z_2, z_3 gives us the coordinates of the object points.

As illustrated in Figure 2-10, let $\mathbf{q}_1$ be the projection of any object point obtained above and $\mathbf{q}_2$, $\mathbf{q}_3$ refer to the projections of the same object point on the second and third image after rotating angles θ and 2θ. Then the object points are determined by the equations:

$$\left\| \frac{z_1}{f}\mathbf{q}_1 - \frac{z_2}{f}\mathbf{q}_2 \right\| = \left\| \frac{z_2}{f}\mathbf{q}_2 - \frac{z_3}{f}\mathbf{q}_3 \right\| = 2\cos\frac{\theta}{2} \left\| \frac{z_1}{f}\mathbf{q}_1 - \frac{z_3}{f}\mathbf{q}_3 \right\| \tag{2.19}$$

Again, there is more than one solution but can be reduced to a unique solution by examining the physical constrains. The resulting 3D points consist of a plane, whose plane normal is the same as the unit vector $\mathbf{u}$ of the rotation axis. To uniquely determine the rotation axis, we have to find the translation vector as well. Assume the rotation axis intersects the above plane at a point $\mathbf{C}$, then it can be determined

using the fact that $\mathbf{C}$ belongs to the plane and

$$\|\mathbf{C} - \mathbf{Q}_1\| = \|\mathbf{C} - \mathbf{Q}_2\| = \|\mathbf{C} - \mathbf{Q}_3\| \tag{2.20}$$

where $\mathbf{Q}_1$, $\mathbf{Q}_2$, $\mathbf{Q}_3$ are the corresponding object points of $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$. That is, the line equation of the rotation axis is completely determined by the six components of $\mathbf{u}$ and $\mathbf{C}$.

In the experimental setup of our system, the rotation stage is placed in front of the camera such that the rotation axis is close to the optical axis of the camera. A checkerboard pattern is used in our calibration (see Figure 2-11). Three corner points, upper-left, upper-right and bottom-right, are used to calibrate the rotation axis. The measured distances between any two points are 220, 150 and 266 mm. The focal length and pixel size of the camera are 19.35 and 0.00345 mm, respectively. Three images are taken at rotation angles 0, 60, and 120 degrees. Solving Eqs. (2.16) – (2.20) with these parameters, the line equation of the rotation axis is given by

$$\begin{aligned} x &= u_1 t + c_1 = 0.0054t + 0.93 \\ y &= u_2 t + c_2 = 0.9997t + 64.76 \\ z &= u_3 t + c_3 = -0.0245t + 831.06 \end{aligned}$$

and therefore we have the rotation center at $(0.58, 832.65)$ and the unit vector $(0.0054, 0.9997, -0.0245)$. The result indicates that the rotation axis is almost perpendicular to the image scanlines and camera optical axis.

Another calibration method is to use two 3D point pair with known coordinates (which can be derived by triangulation) to compute the rotation axis. Let P_1, Q_1 be the 3D points before rotation and P_2, Q_2 be the corresponding 3D points after rotation. Let π_P be the plane passing $\frac{P_1+P2}{2}$ and perpendicular to $\overline{P_1P_2}$, and π_Q be the plane passing $\frac{Q_1+Q2}{2}$ and perpendicular to $\overline{Q_1Q_2}$. Then the rotation axis is determined by the intersection of π_P and π_Q.

2.4 Camera System and Implementation

The vision system used for image acquisition in our experiments is called Stonybrook VIsion System 2 (SVIS-2). SVIS-2 camera system consists of a high resolution digital still camera (Olympus C-3030), a rotation stage with a stepper motor and a PC (Pentium II 450 MHz). The objects are mounted on the rotation stage and the camera is placed in front of it such that the optical axis is close to the rotation axis (see Figure 2-12). The calibration method described in the previous section is used for rotation axis estimation. The calibrated rotation axis is used for

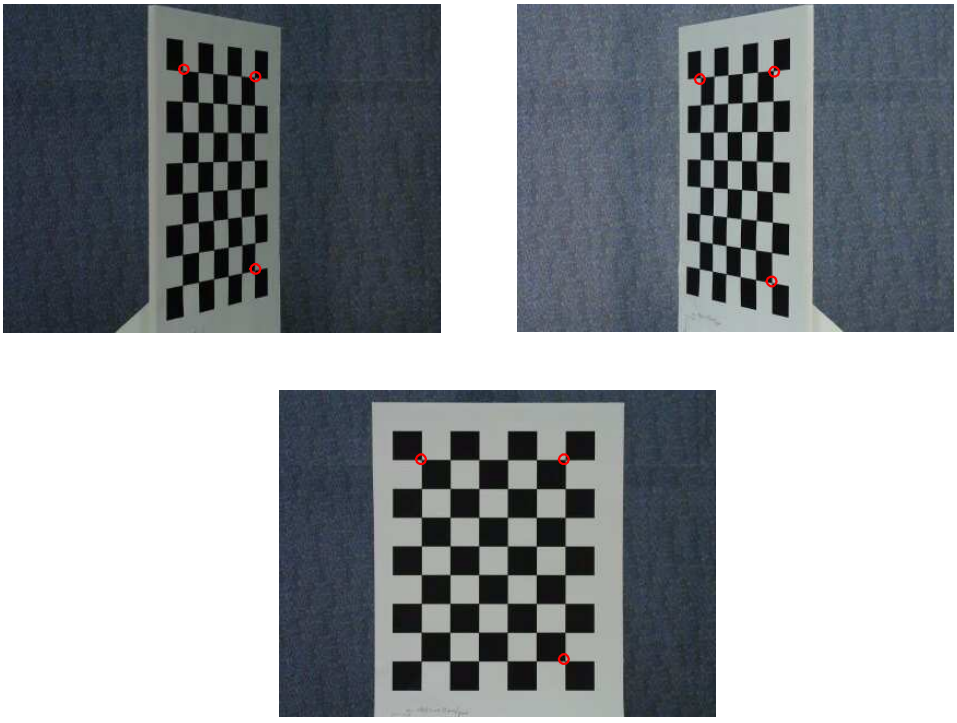


Figure 2-11: The checkerboard pattern used for rotation axis calibration. Three images are taken from 0, 60 and 120 degrees of rotation angle. Three corner points (as indicated in red circles) are used to calculate the rotation axis.

stereo matching in rotational stereo. In the experiment, the parameters of SVIS-2 are adjusted for objects that fit inside a 250 mm $\times$ 250 mm $\times$ 250 mm cube placed at a distance of about 830 mm from the camera. The camera focal length is set to 19.35 mm and F-number to 2.8. Different focus settings are obtained by moving the camera's motorized lens controlled by a PC. The image resolution is set to 1280×960 pixels. All acquired images are transferred to a PC through a USB port for off-line processing.

The steps for recovering 3D shape and focused image of each view of an object are described as follows. First, two sequences of 4 images with different focus settings are recorded before and after rotating the object by a small angle to obtain the stereo image pairs. The stereo rotation angle is set to 6 degrees, which gives the equivalent parallel stereo baseline of about 174 mm. Shape from focus is applied on each sequence of images to get the focused image and a rough depth map. 16×16 image blocks are used to obtain a 80×60 rough depth map and a 1280×960 focused image. The depth map is thresholded to segment both the depth map and

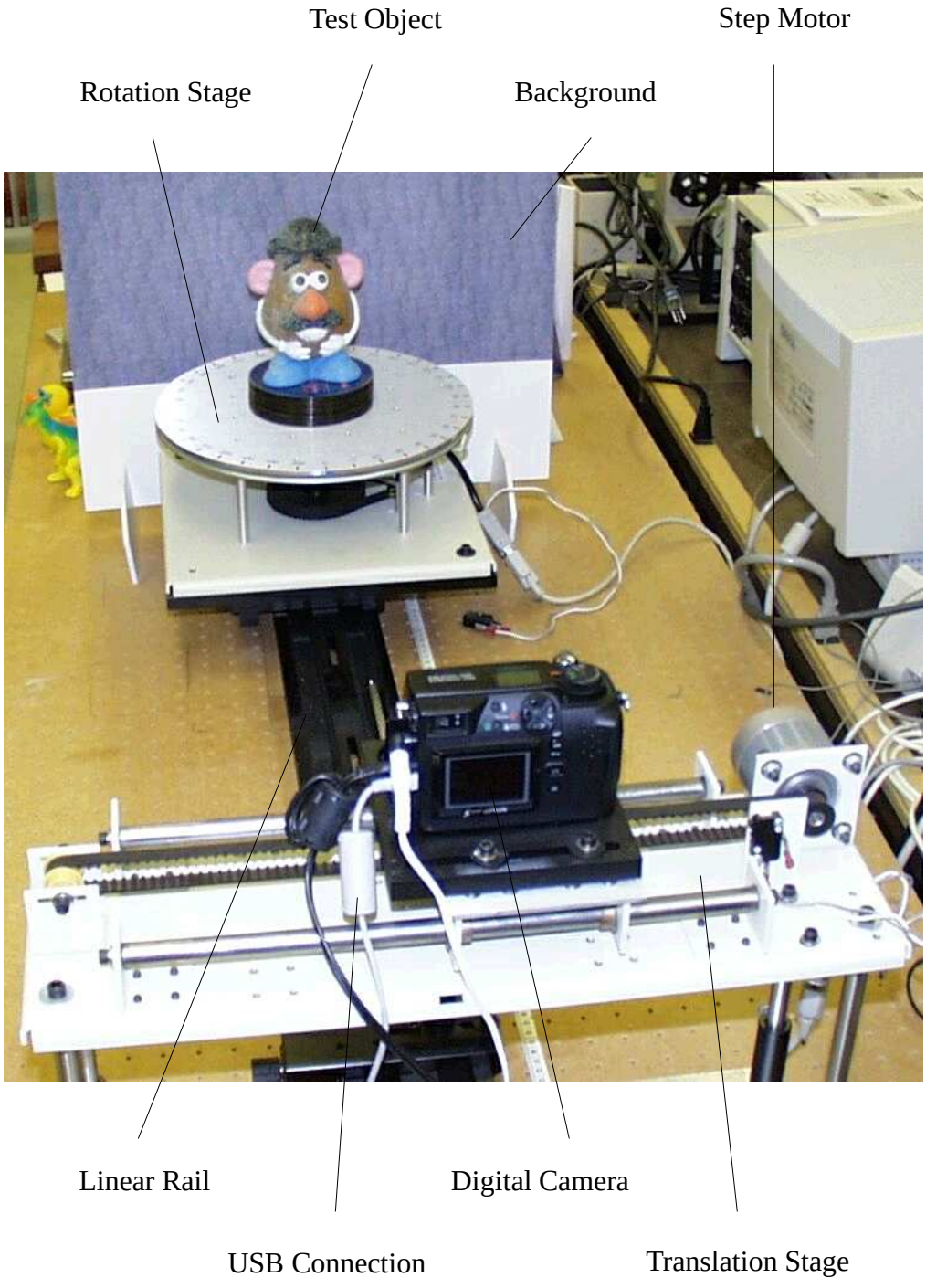


Figure 2-12: SVIS-2 (Stonybrook VIsion System 2) camera system

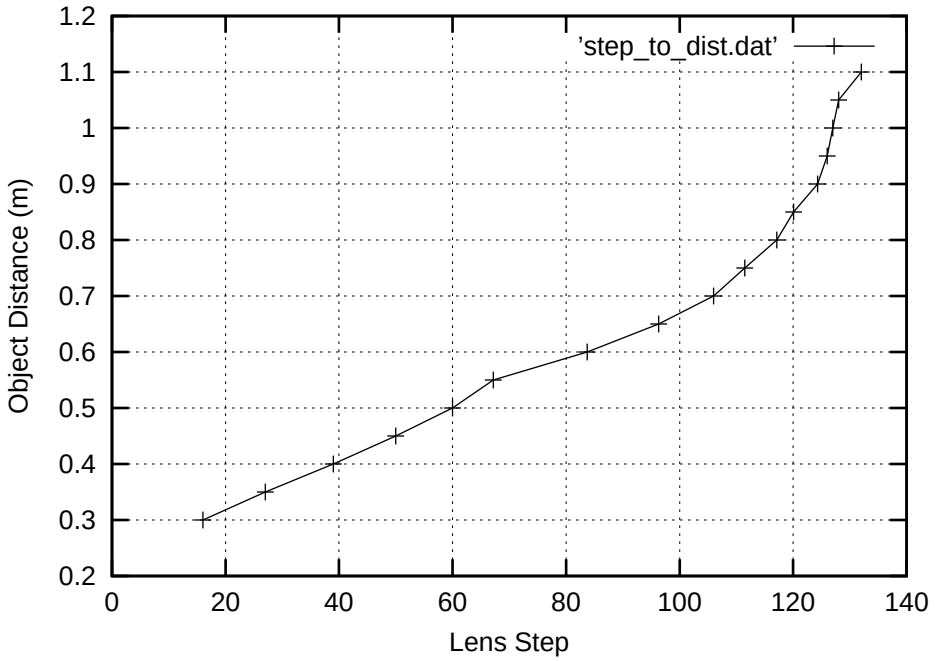
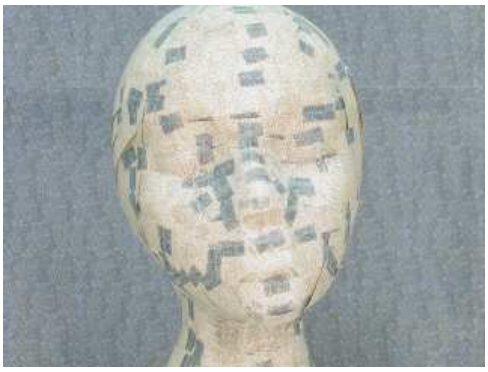


Figure 2-13: Lens step vs. best focused distance used in SFF

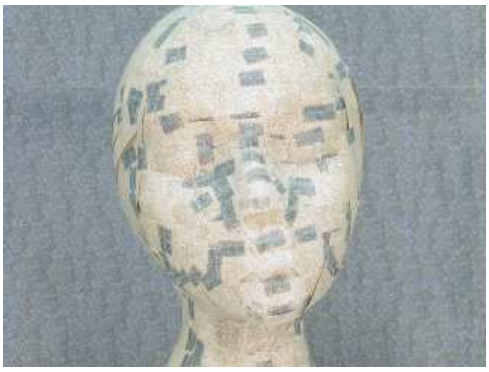
the focused image into two regions, one corresponding to background region (points farther than the expected distance of object points) and object (foreground) region.

Figure 2-13 shows a plot of the relationship between the lens step number and the best focused distance of the camera used in our experiment. The plot indicates that the lens step number and the best focused distance have an almost linear relationship in several regions (step numbers 20 to 65, 65 to 105, and 105 to 125). The sequence of 4 images used to construct the focused image and depth map are taken at lens step numbers 108, 115, 122 and 129. They are roughly in the linear region. Figure 2-14 shows the blurred images taken from the camera (Figure 2-14(a) – 2-14(d)) and the constructed focused image (Figure 2-14(e)).

The rotational stereo is then carried out using the focused images and the initial depth map estimated by SFF to get a more accurate depth map. Sum-of-squared-difference measure on 16×16 image blocks is used for matching in the foreground regions [57]. According to the rotational stereo model, the epipolar line corresponding to each matching block is calculated for fast matching instead of using image rectification. Each block from the first image searches for the matching image region in the second image along the epipolar line. The search is limited to a 2 pixel wide band in the image vertical direction. Along the epipolar line, the search is



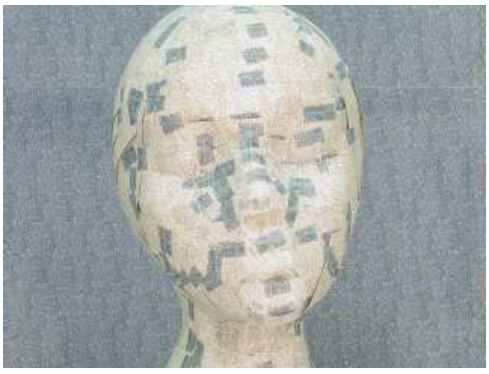
(a) Lens step 108



(b) Lens step 115

(c) Lens step 122

(d) Lens step 129



(e) Constructed focused image

Figure 2-14: A sequence of blurred images and the constructed focused image

further limited using the depth map obtained by SFF with a maximum expected depth error. For a high contrast image, SFF depth error is set to 7 lens steps in focus setting. A 3×3 median filter is applied on the resulting depth map to reduce noise and mismatching.

The partial 3D model obtained here is modelled in two parts– (i) the 3D shape of the surface specified by the depth map array, and (ii) the image texture of the surface specified by the focused image recovered by SFF. The 3D shape of the surface is modelled as follows. The camera coordinates of the points in the depth map are obtained by an inverse perspective projection for the camera and printed as (x_i, y_i, z_i) triples for $i = 1, 2, ..., n$, where n is the total number of 3D points. These triples constitute a list of vertices in the 3D space of the camera coordinate system. The rectangular grid specified by the array is used to create a list of quadrilaterals in the 3D space. For each quadrilateral, the image texture is obtained from the focused image in the corresponding rectangle in the rectangular grid.

The above procedure for obtaining partial 3D model is repeated for several different views by rotating the object using a computer controlled motor. In the experiment, typically 4 views for every 90 degrees rotation interval are acquired for complete 3D model reconstruction.

2.5 Experimental Results

In this section, we show some experimental results and the acquisition time of partial 3D models constructed by rotational stereo and shape from focus. Four different real objects– a foam head, a toy "Potato Head", a cylinder, a detergent container, and a multiple-object scene are presented. To separate the foreground and background regions by SFF, a board with high contrast pattern is placed at about 1000 mm from the camera. On the surfaces of the objects, random patterns are pasted or sprayed to facilitate stereo matching.

Figure 2-15 shows the results of the foam head object. Random dot patterns are pasted on the surface of the object. The left and right focused images constructed from the first and second image sequences are shown in Figure 2-15(a) and 2-15(b). The initial depth map and occlusion mask constructed by shape from focus are shown in Figure 2-15(c) and 2-15(d), respectively. Figure 2-15(e) shows the final depth map obtained from rotational stereo and SFF. Figure 2-15(f) shows the recovered partial 3D shape obtained from inverse perspective projection of the depth map.

Figure 2-16 shows the results of the cylinder object. The initial depth map and occlusion mask constructed by shape from focus are shown in Figure 2-16(a) and 2-16(b), respectively. Two different resolutions, 80×60 and 160×120, of the final

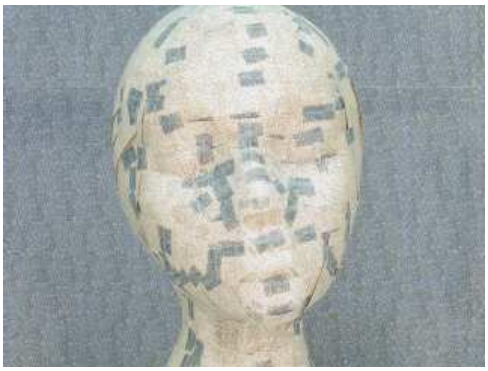
Table 2-1: Measured execution times for data acquisition

Object	Shape from Focus	80×60 resolution	160×120 resolution
Head	5.68 sec.	8.05 sec.	21.10 sec.
Toy	5.52 sec.	5.18 sec.	18.39 sec.
Cylinder	5.59 sec.	9.37 sec.	35.06 sec.
Detergent	3.03 sec.	9.87 sec.	37.02 sec.
Bottle	5.48 sec.	5.51 sec.	19.63 sec.

depth maps and partial 3D shape are shown in Figure 2-16(c) – 2-16(f). There are 7 images taken (lens step 108 to 129 with every 4 steps) for SFF instead of 4 for this object.

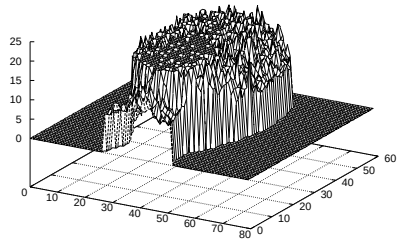
Figure 2-17 shows the results of the toy object "Potato Head". Figure 2-17(a) – 2-17(d) show the partial 3D models with texture map from different viewing directions. Four viewpoints are observed by rotating the object every 90 degrees. The shading information (vertex normals on each data point) is also given to provide a smooth surface appearance. Figure 2-18 and 2-19 gives the textured partial 3D shape results of a detergent container and a multiple objects scene respectively.

The total execution time for data acquisition includes the image acquisition time taken by SVIS-2 camera system and the processing time used for 3D shape and focused image recovery. For each viewpoint, it takes approximately 4 minutes to acquire all images including rotating the object (for a small stereo rotation angle). Totally 8 images are acquired– 2 sequences of 4 images with different focus positions. All images are saved as uncompressed TIFF format with 1280×960 image resolution. The processing times for SFF and stereo matching with different resolution depth map output are shown in Table 2-1 with several test objects (in seconds, on a Pentium II 450 MHz PC). For a complete 3D model reconstruction, we acquire 4 partial 3D models from every 90 degrees viewpoint with a total of 360 degrees. Therefore, the data acquisition time for creating a complete 3D model is about 20 minutes.

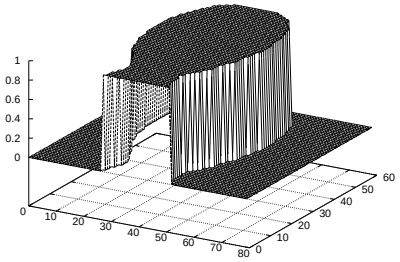


(a) Left focused image

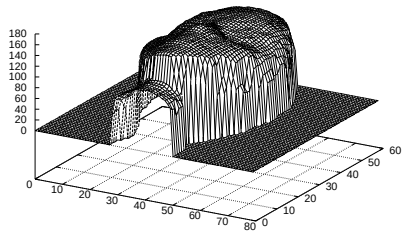
(b) Right focused image



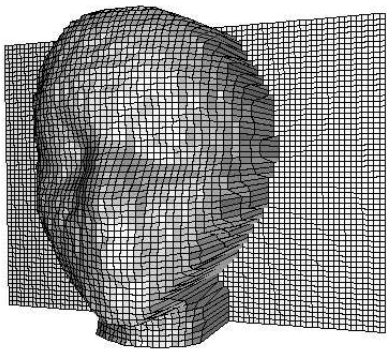
(c) Initial depth map



(d) Occlusion mask

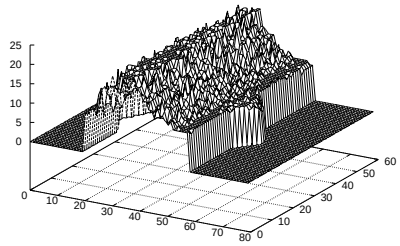


(e) Final depth map

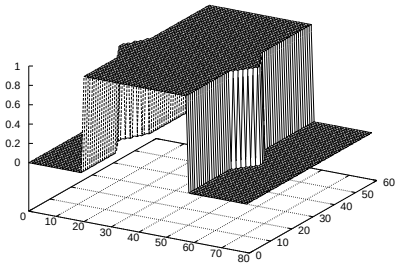


(f) Partial 3D shape

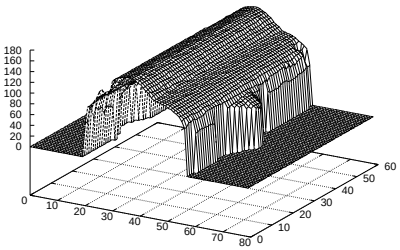
Figure 2-15: The stereo image pair (a) and (b), the rough depth map (c), and the occlusion mask (d) are obtained by SFF. These results are used to find the final depth map (e). The partial 3D shape (f) is recovered by inverse perspective projection.



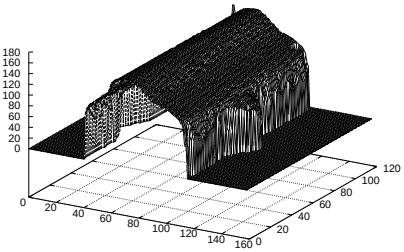
(a) Initial depth map



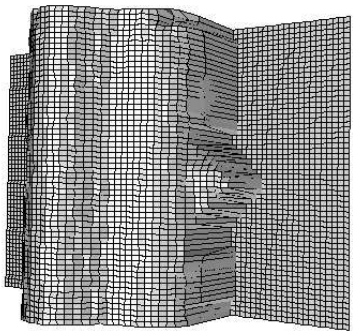
(b) Occlusion mask



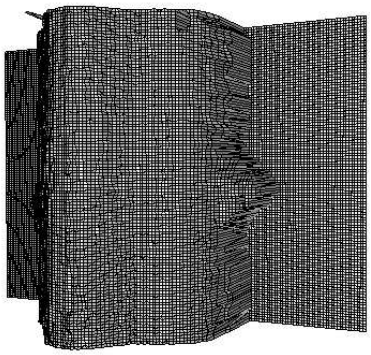
(c) Depth map (80×60)



(d) Depth map (160×120)

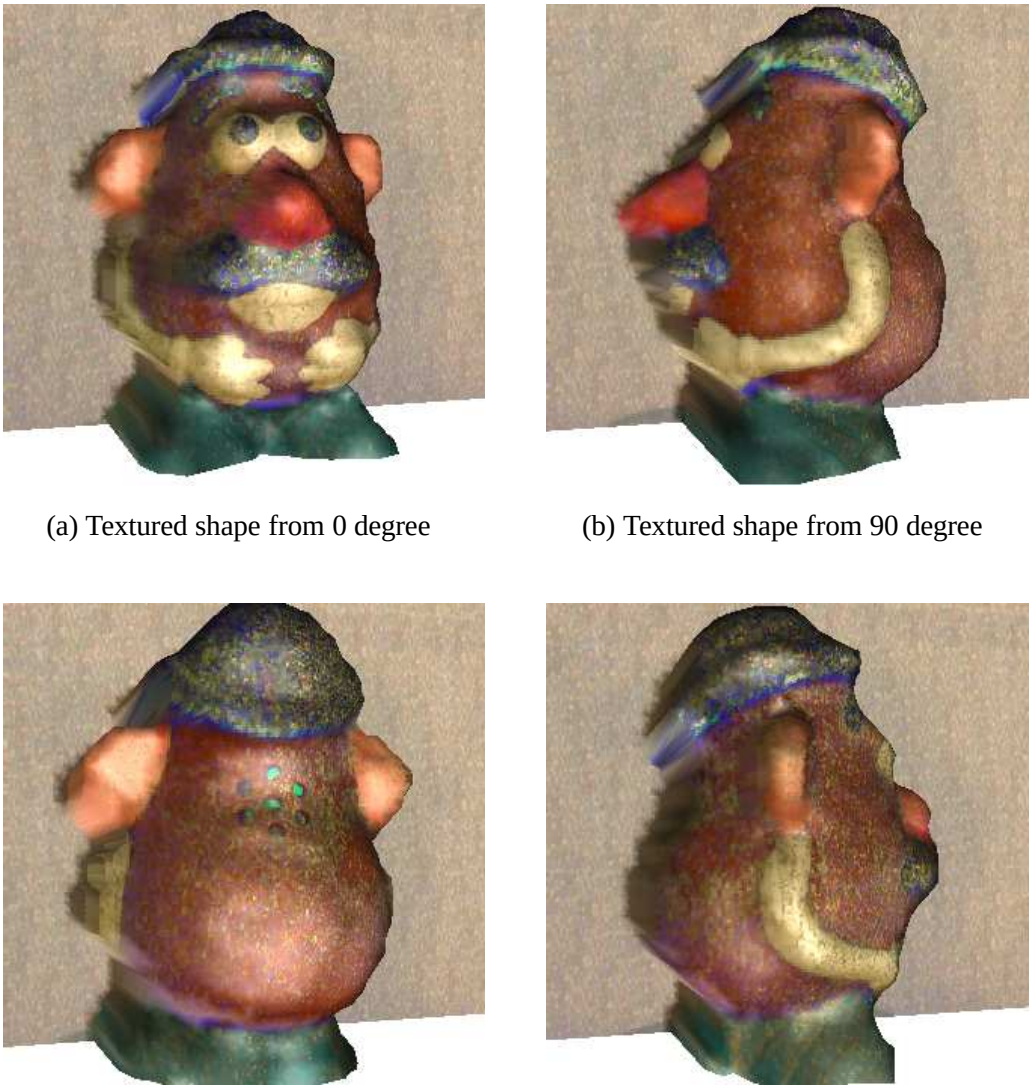


(e) Partial 3D shape (80×60)



(f) Partial 3D shape (160×120)

Figure 2-16: Although the rough depth map (a) and occlusion mask (b) obtained from SFF only provide the resolution of 80×60, it can be increased on the final depth map by stereo matching. The number of data points in (d) and (f) are increased by 4 from (c) and (e). This will also benefit data registration to be discussed in the Chapter 3.



(a) Textured shape from 0 degree

(b) Textured shape from 90 degree

(c) Textured shape from 180 degrees

(d) Textured shape from 270 degrees

Figure 2-17: Experimental results for a toy object

(a) Textured shape from 0 degree

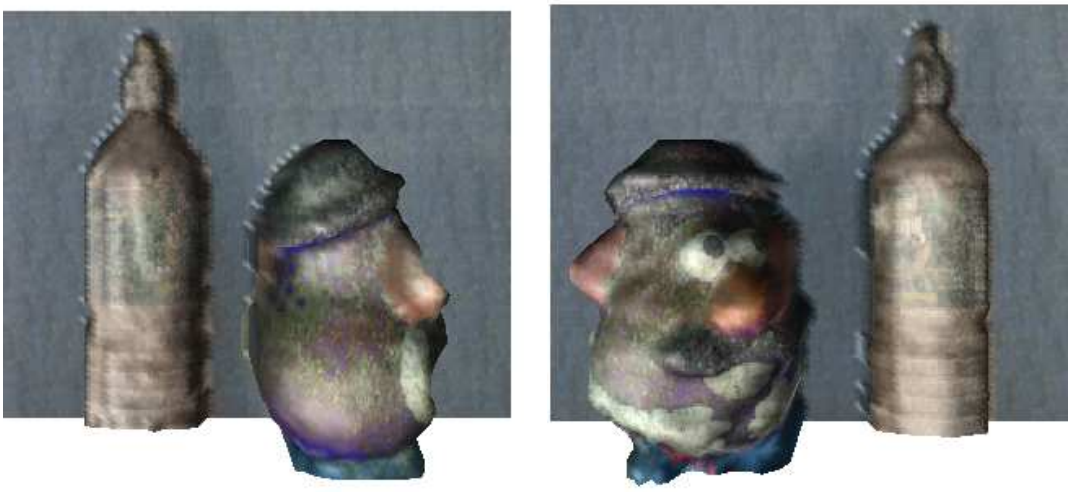
(b) Textured shape from 90 degree

(c) Textured shape from 180 degrees

(d) Textured shape from 270 degrees

Figure 2-18: Experimental results for a detergent container

(a) Textured shape from 0 degree (b) Textured shape from 90 degree



(c) Textured shape from 180 degrees (d) Textured shape from 270 degrees

Figure 2-19: Experimental results for multiple objects

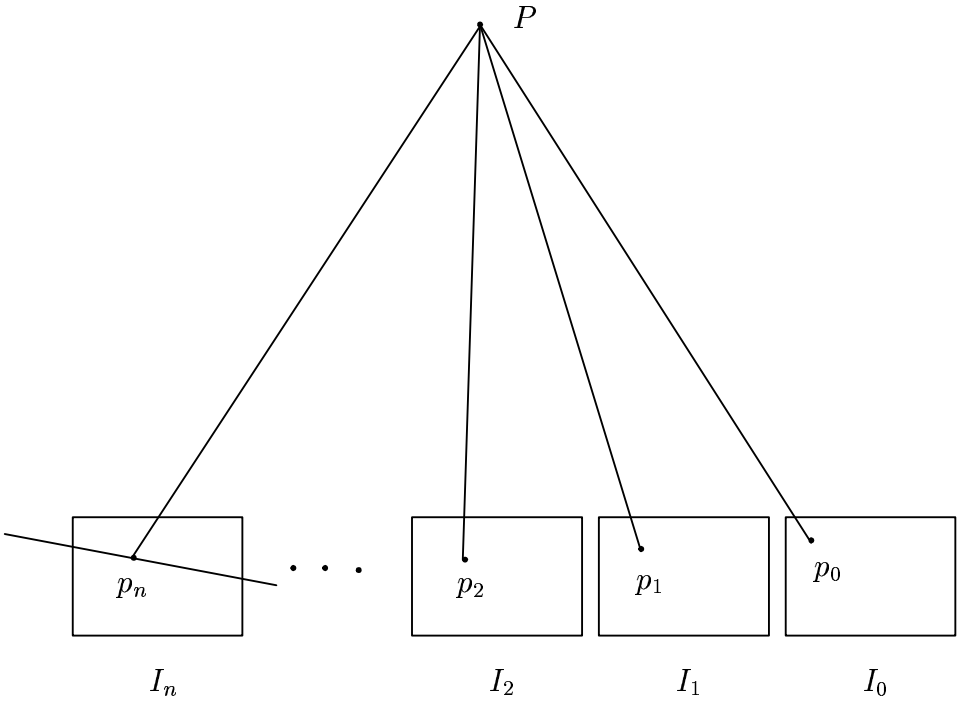


Figure 2-20: Multiple base-angle rotational stereo

2.6 Multiple Base-angle Rotational Stereo

Similar to the parallel stereo with multiple baselines [57], precise and accurate depth recovery can be achieved by multiple *base-angles*. However, unlike multiple-baseline stereo, multiple base-angles are achieved by rotating the object in front of a single fixed camera instead of using multiple cameras. In order to recover depth through triangulation, correspondence between different images has to be established. In this work, a *correlation-based* method with simple SSD (sum of squared difference) is used. The matching is performed by computing the SSD values for each image pair and summing them up to produce the sum of SSD's (SSSD). Correct matching points are then decided by the minimum SSSD values. The detailed algorithm is described as follows.

As illustrated in Figure 2-20, let $I_0, I_1, \cdots, I_n$ be the images taken at the rotation angles $0, \theta_1, \cdots, \theta_n$ with $0 < \theta_1 < \cdots < \theta_n$. This gives us n stereo image pairs $\{I_0, I_1\}$, $\cdots$, $\{I_0, I_n\}$ with base-angles $\theta_1, \cdots, \theta_n$. It should be noted that, while all the images (I_0, I_1, $\cdots$, I_n) are used to find the correspondence, only the image pair $\{I_0, I_n\}$, which possesses the largest base-angle, is used to compute the depth map. First, for each pixel (at which depth is to be computed) in I_0, the

corresponding epipolar line is computed only for image I_n. The depth for each pixel (say, p_n in Figure 2-20) on the epipolar line is then obtained by triangulation and projected back to images $I_1, I_2, \cdots, I_{n-1}$ to find the potential correspondences, $p_1, p_2, \cdots, p_n$, and their SSD values on a subimage block. Sum of SSD for each image pair, $\{I_0, I_1\}$, $\{I_0, I_2\}$, $\cdots$, $\{I_0, I_n\}$, is evaluated and the pixel on the epipolar line in image I_n with the smallest value is considered as the corresponding point.

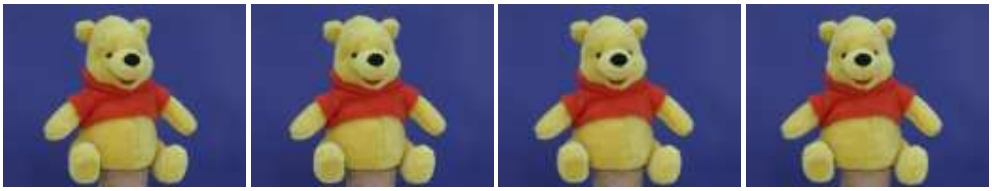
In rotational stereo, epipolar lines are derived first and used in stereo matching instead of *image rectification* [2]. This saves computation since the resolution of computed depth map is usually lower than the pixel resolution (because matching is done blockwise instead of pixelwise). For example, in our experiments, only a 320×240 depth map is created from 1280×960 intensity images. The computational savings is even greater when the number of images is increased for multiple base-angle approach.

2.6.1 Experimental Results

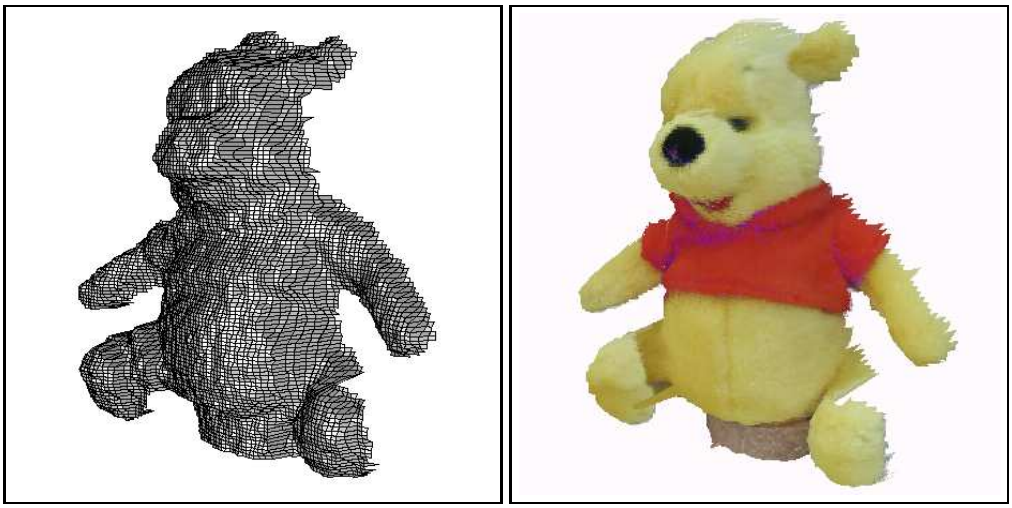
In this section, we present some results of multiple base-angle rotational stereo implemented on our SVIS-2 (Stonybrook VIsion System 2) camera system. A test object is placed at about 800 mm away in front of the camera (see Figure 2-12). A sequence of 4 images at 4 degree intervals (at 0, 4, 8, and 12 degrees) of rotation is recorded for multiple base-angle stereo matching. The foreground (object) region is separated from the blue background to restrict the stereo searching range. Alternately, the object region can be extracted by separating the dynamic and static scene with a stationary background.

To reduce both mismatches and computation time, a multi-resolution stereo matching method is used in a coarse-to-fine search strategy. First, a low resolution 80×60 depth map is created followed by 3×3 median filtering to minimize the depth error. It is then used to restrict the search range in computing a 160×120 resolution depth map. The resulting depth map is used to obtain a final high-resolution 320×240 depth map. Experiments show that this multi-resolution approach improves time efficiency by a factor of around 2.

Figure 2-21(a) shows the four acquired images of a toy object. The recovered 3D shape and the texture mapped 3D model are shown in Figure 2-21(b). It can be seen that part of the right leg near the body cannot be recovered very well due to occlusion. The results of a pumpkin object are shown in Figure 2-22 with four input images (Figure 2-22(a)) and two texture mapped 3D models (Figure 2-22(b)). All the results are direct output of our algorithm without any manual editing or smoothing. The computation times for the toy and pumpkin objects are 77.67 seconds and 27.61 seconds on a Pentium II 450 MHz PC, respectively. Figure 2-23 shows the



(a) Image taken at 0,4,8, and 12 degrees



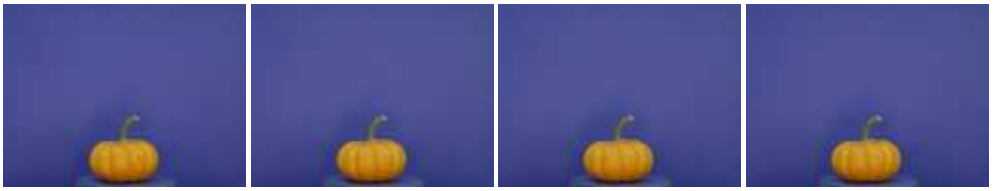
(b) Wireframe and textured 3D shapes

Figure 2-21: Input images and the output 3D shape of a toy object

acquired intensity images and texture mapped 3D shapes of another toy object with two different acquisition viewpoints.

2.7 Precision and Accuracy of Rotational Stereo

The precision of a stereo system is usually determined by the depth error due to one pixel error in stereo disparity. In rotational stereo, assuming the rotation axis is parallel to the image plane, perpendicular to the image scan line and intersects camera optical axis, then the stereo disparity is given by $d = bf/z$ for an object point at depth z, where b is the effective stereo baseline and f is the focal length.



(a) Image taken at 0,4,8, and 12 degrees

(b) Two views of textured 3D shapes

Figure 2-22: Input images and the output 3D shape of a pumpkin

The disparity difference at two depth z_1 and z_2 is given by

$$\Delta d = d_1 - d_2 = bf\left(\frac{1}{z_1} - \frac{1}{z_2}\right) \tag{2.21}$$

or

$$z_2 = \frac{bfz_1}{bf - z_1\Delta d}. \tag{2.22}$$

Thus, the depth difference Δz near the image center can be written as

$$\Delta z = \frac{z^2\Delta d}{2z_0 \sin\frac{\beta}{2} f - z\Delta d}. \tag{2.23}$$

where Δd is the disparity difference, z_0 is the distance to rotation axis, and β is the rotation angle.

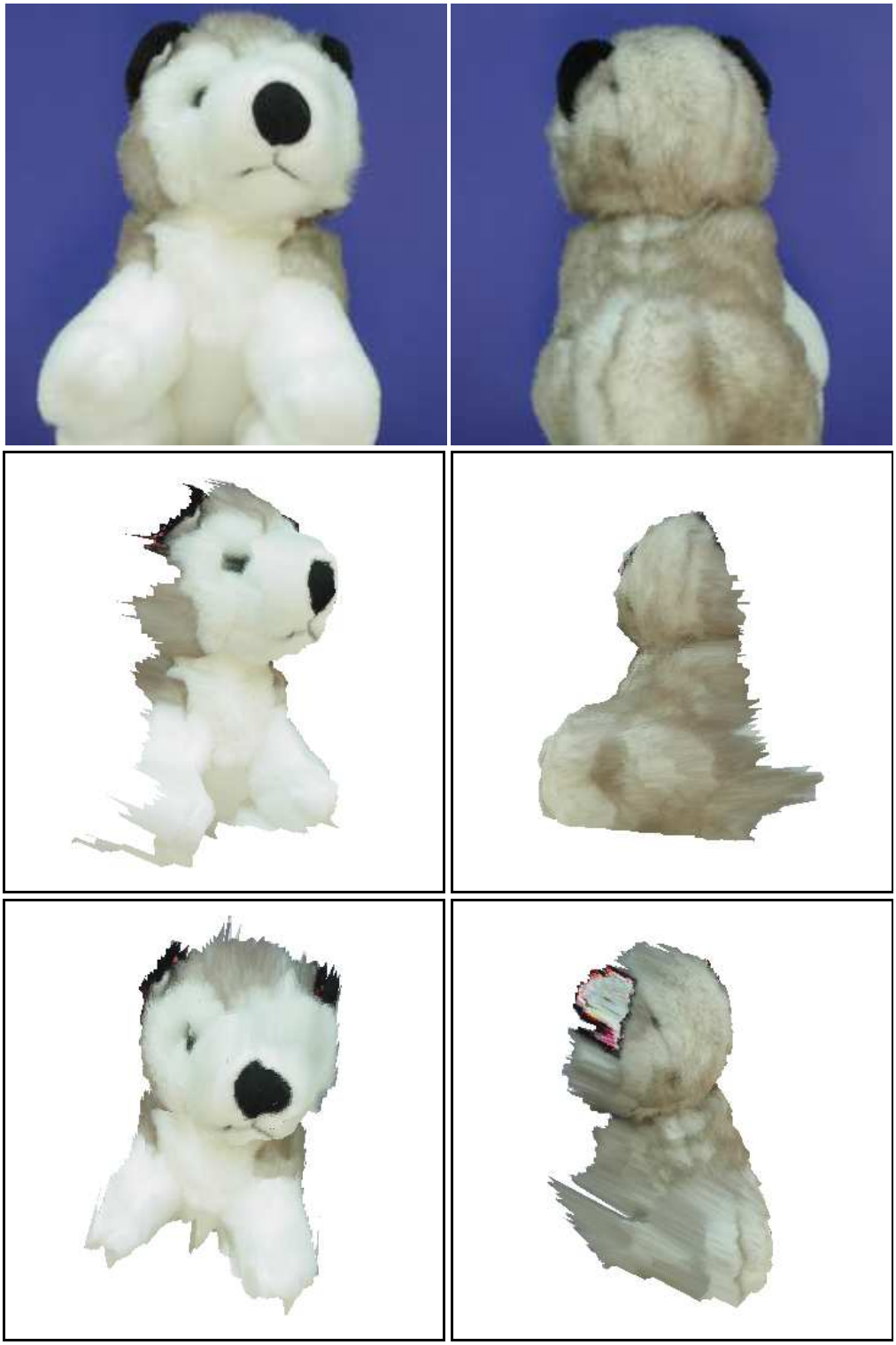


Figure 2-23: Experimental results for a toy object

Figure 2-24: Test objects with random pattern are used for planar fit

The parameters used in the experiments are as follows: the maximum base-angle is 12 degrees, the pixel size is 0.00552 mm, camera focal length is 19.35 mm and the distance to rotation axis is 833 mm. Thus, the maximum depth error for one pixel disparity is between 0.80 mm and 1.33 mm in the working range of 700 mm to 900 mm away from the camera.

A rudimentary error analysis of our rotational stereo system was done using 3 planar objects. Figure 2-24 shows the objects. The first and second objects were placed at about 750 mm and 800 mm away facing the camera. The third object was slanted (about 30 degrees from the image plane) at a range of 700 mm to 900 mm. For each object, a 320×240 depth map was computed, and on the 100×100 3D data points in the center, a plane was fitted. The results are shown in Table 2-2. The average absolute error and the RMS error are both less than 0.5 mm, and the maximum error is less than 2 mm in the working range.

2.8 Discussion

In this chapter, we present a digital vision system to acquire intensity and range images using shape from focus and rotational stereo. It takes approximately 5 minutes to recover a partial 3D model of an object. The processing time for 3D shape reconstruction is about 30 seconds (on a Pentium II 450 MHz PC), but the image acquisition time takes about 4 minutes. This is due to the slow data transfer rate of

Table 2-2: Errors of planar fitting (in mm).

distance	plane equation: $\hat{n} \cdot \mathbf{x} = d$	ave.	RMS	max.
750	$\hat{n} = (0.039, 0.002, -1)$, $d = 747.581$	0.35	0.44	1.77
800	$\hat{n} = (0.048, 0.001, -1)$, $d = 796.151$	0.29	0.36	1.47
700 -900	$\hat{n} = (0.661, 0.001, -1)$, $d = 789.871$	0.33	0.41	1.39

USB connection and the internal data writing. More advanced digital cameras (in terms of faster data transfer rate) should provide us a great improvement on data acquisition.

Our 3D shape reconstruction algorithm is based on the pinhole camera model. Lens distortion and other intrinsic parameters of the camera are not taken into account. As we will see in the next two chapters, integration of inaccurate partial 3D shapes gives unsatisfactory complete 3D model even with *perfect* registration. This should be improved by more careful camera calibration.

Theoretically the accuracy of rotational stereo highly depends on accurate calibration of rotation axis and accurate rotation angle which provides stereo image pair. Imperfect parameters of rotation axis and stereo rotation angle give incorrect epipolar line equations, which yield the wrong stereo matching results. One simple solution is to increase the vertical search range for stereo matching. However, this also increases the possibility of mismatches because of larger search area. To reduce stereo mismatches and improve the accuracy, a multiple base-angle approach with multi-resolution stereo matching is developed. Experimental results show that our rotational stereo system is able to extract depth with less than 0.5 mm error between 700 mm and 900 mm from the camera.

Chapter 3

Registration

3.1 Introduction

As discussed in Chapter 1, a typical 3D model reconstruction has to go through four main steps:

1. data acquisition,
2. **registration**,
3. surface integration, and
4. texture mapping.

Since a range image only samples the outer surface of an object which is visible from a given viewpoint, the acquisition of several range views is mandatory in order to cover the entire object. In addition, each range view is generally represented in their own camera (or sensor) coordinate system, the range data must be transformed from several different camera coordinate systems to a single coordinate system. This step is typically termed *registration* and has received sustained attention in the research community over the past several decades [14].

The goal of registration is to find the spatial transformation between the range images taken from an object at different viewpoints, so that the points found in different range image views that represent the same surface point are aligned. Depending on the registration techniques, it can be classified as *coarse registration* and *fine registration*. In coarse registration, a set of interframe registrations are determined to within a tolerance of a few degrees of rotation and a small amount of translation errors. Most 3D model acquisition systems, including our SVIS-2 camera system, employ a rotation stage upon which the object is placed for acquiring range data. Such systems do not need to solve a coarse registration problem.

Physical measurement gives a good approximation of coarse registration. Thus, our registration will mainly focus on fine registration.

Fine registration is to refine the transformation parameters estimated by coarse registration and provides a more accurate range data alignment. Given a set of coarse registered range data relating all the viewpoints, fine registration task aims to make small changes to the individual transformations to improve the overall quality of the registration. The quality criterion typically rewards a small distance between points in the corresponding surfaces of two overlapping views.

Since our data acquisition system acquires range images from different viewpoints by rotating the object in front of the camera, our registration problem is equivalent to finding the rotation axis of the rotation stage. The registration process consists of two steps. First, we find an initial estimated transformation by calibrating the rotation axis as described in Chapter 2, which is similar to a coarse registration process but provides more accurate results. Second, the estimated registration is further improved during surface integration. The overlapping parts of range images are used to refine the rotation axis and increase the accuracy of registration.

In the following sections, we first review some of the related work on registration. We then describe two registration algorithms to find the rotation axis and show some experimental results. A short discussion concludes the chapter.

3.2 Related Work

A popular method for refining a given registration is the iterative closest point (ICP) technique, first introduced by Besl and McKay [10]. It uses a nonlinear optimization procedure to further align the data sets from coarse registration. Given two data sets and a rough initial transformation, potential correspondences are developed as follows. For each point in the first data set, find its closest point in the second data set under the current transformation. Let P be the set of m points from the first data set $\{p_1, p_2, ..., p_m\}$ and Q be the set of n points from the second data set $\{q_1, q_2, ..., q_n\}$. An incremental transformation T_i is defined such that $Q_{i+1} = T_i \cdot Q_i$ for each iteration i, where $Q_0 = Q$. T_i reduces the registration error between P and the new point set Q_{i+1} by moving the points Q_i closer to P. Then Q_i is assigned to Q to initialize the system and update the next transformation T_{i+1}. The transformation usually involves a translation vector and a rotation matrix.

The ICP algorithm repeatedly computes the closest points between data sets and computes the transformation to register the data, until a minimum tolerance on a mean square distance metric between the surfaces is obtained. They show the ICP algorithm will monotonically converge to a minimum, but it is not necessarily the

global or the best minimum. In order to converge to the global minimum, the initial parameter estimate must be reasonably close to the true value to avoid converging to a non-optimal solution.

Chen and Medioni [16] also use an iterative refinement of initial coarse registrations between views to perform fine registration. A good initial estimate of the registration is assumed to be available from a knowledge about the data acquisition geometry. Instead of minimizing the distance between the closest points in the two data sets (such as general ICP algorithms), orientation information is used for minimization. Their technique refines a nominal transformation by iteratively computing incremental transformations that minimize the distance between the transformed points from the first view and the points from the second view. The minimized functional is expressed in terms of the distance between each *control point* in the first view and the tangent plane at an intersected point in the second view. This tends to not only minimize the distance between the two surfaces but also match their local shape characteristics. Because of the expense of computing intersections between lines and the discrete surface (generated from the tangent planes) they subsample the original data to obtain a set of control points. The control points are selected from this subsampling of the surface (regularized grid) that lie in smooth regions on the surface. In these areas of smoothness the surface normal can be calculated more reliably.

3.3 Registration with Global Resampling

In this section, we present a registration method to simultaneously register multiple range images using global resampling in the object-centered cylindrical coordinate system. As we have mentioned in the previous section, finding the registration transformation is equivalent to finding the rotation axis of the rotation stage in our system. We will search a small neighborhood of the estimated rotation axis obtained from system calibration to find the one with minimum error on the overlapping regions of the range images in a least squared sense.

The coordinate systems used in this dissertation for both registration and integration are shown in Figure 3-1. We use a left-handed camera coordinate system (X, Y, Z) as our world coordinate system. The local object coordinate system (r, y, θ) is a cylindrical coordinate system. The axis of the cylinder is aligned with the rotation axis of the object rotation stage and intersects the XZ-plane of the camera coordinate system at $(x_0, 0, z_0)$. The image coordinate system $(\hat{x}, \hat{y})$ is parallel with the XY-plane and has the origin at $(0, f, 0)$ in the camera coordinate system, where f is the focal length of the camera.

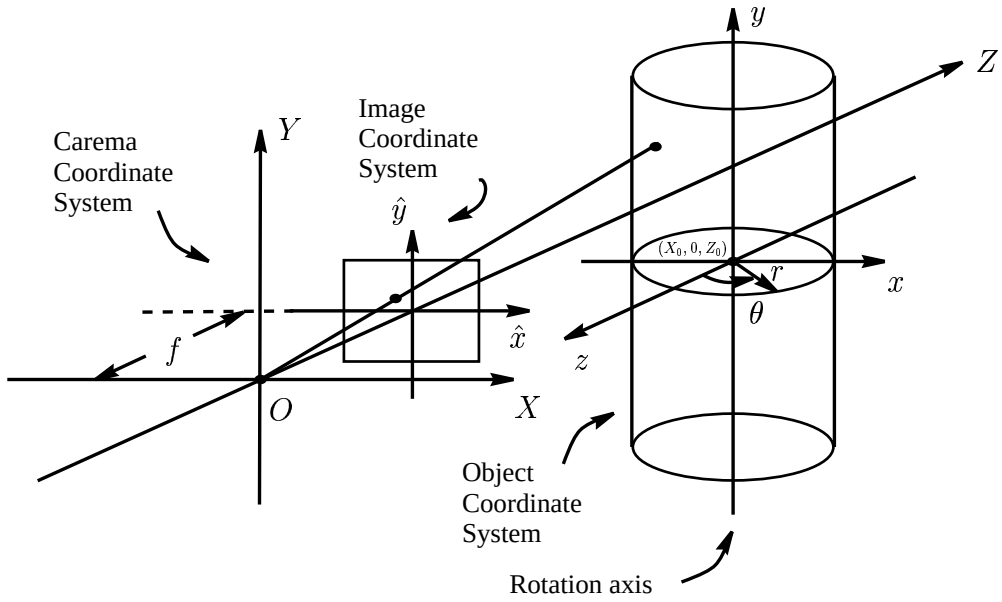


Figure 3-1: Coordinate systems used for registration and integration

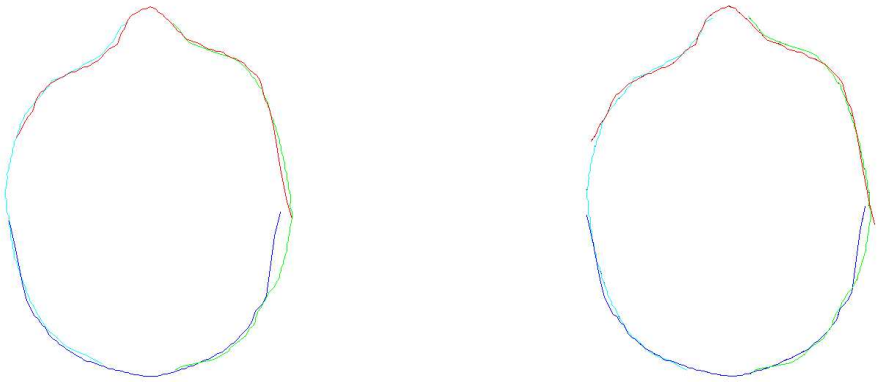
In our experimental setup, the calibrated rotation axis is almost perpendicular to the XZ-plane of the camera coordinate system. The unit vector of the rotation axis is found as $(4.123 \times 10^{-3}, 0.9997, -2.364 \times 10^{-2})$ in Section 2.3.2. Thus, the following assumptions are made to simplify the registration process:

1. The rotation axis of the rotation stage is perpendicular to the image scanlines and optical axis of the camera.

2. The object is *simple*– for each cross section of the object perpendicular to the rotation axis, there is a 1-1 correspondence between any point on the contour and the angle θ of the direction vector from the rotation center to that point. In other words, in a cylindrical coordinate system (r, θ, y) with its y-axis coinciding with the axis of rotation, the contour of the object's cross section at every y can be specified by a function of the form $r(\theta)$.

3. The rotation angles controlled by the step motor are accurate.

Assumption 1 is a direct result from the rotation axis calibration and simplifies the problem of finding the rotation axis to the problem of finding the rotation centers of the cross sections. Assumption 2 is to ensure that we can apply interpolation on



(a) Curves before registration (b) Curves after registration

Figure 3-2: Cross section curves of an object before and after registration

every data point in a cylindrical coordinate system. As for the third assumption, the accuracy of the rotation angle is $\pm 0.03\%$ from the manufacturer's data sheet (Arrick Robotics MD-2 Dual Stepper Motor System).

The process of finding the rotation center is given as follows. First, each range image represented in the camera coordinates (X, Y, Z) is converted to a representation in the cylindrical object coordinate system (r, θ, y) using the following relations:

$$\begin{aligned} \theta &= \tan^{-1} \frac{X - X_0}{Z_0 - Z} + \phi_i \\ r &= \sqrt{(X - X_0)^2 + (Z - Z_0)^2} \\ y &= Y \end{aligned} \tag{3.1}$$

where ϕ_i is the rotation angle of range image or viewpoint i with respect to the first viewpoint. In the experiments, the object is rotated by every 90 degrees for data acquisition and ϕ_i is equal to $i\pi/4$. At this point, only the data points that are part of the object are retained, and other points belonging to the background are eliminated by using a threshold on the Z coordinate of the points. Having retained only points on the object, all the points from different views are merged into one set that represent a discrete sampling of the visible surface of the object. Since the object is assumed to be simple so that it can be represented by a function of the

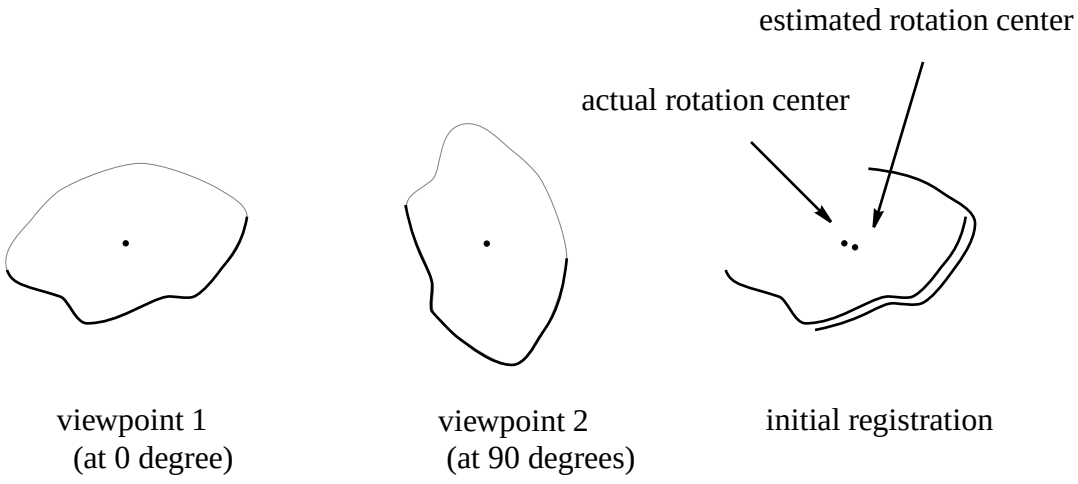


Figure 3-3: Registration of two curves from different viewpoints

form $r(\theta, y)$ in the cylindrical object coordinate system, the merged set of points can be thought of as a discrete sampling of this function.

Suppose that the object data points obtained from the i-th view is expressed as $r_i(\theta, y)$ in the cylindrical coordinate system. These discrete sample points are used to interpolate and uniformly resample the partial 3D shape of the object $r_i(\theta, y)$ for the i-th view in the (θ, y) space. For any two successive views, say i-th view and j-th view, assume that the object's surface overlaps partially, where $j = (i + 1)$ mod n and n is the number of viewpoints. In the overlapping part, assume that the resampling of r_i and r_j have been done for the same value of (θ, y). Let the position of the rotation axis (X_0, Z_0) be slightly different for different y. In this case, we can denote the position by $(X_0(y), Z_0(y))$. Now, we can improve the estimates for $(X_0(y), Z_0(y))$ by minimizing the squared distance between the same points in the successive views in the overlapping parts as follows:

$$\varepsilon(y) = \sum_{i=0}^{n} \sum_{\theta} \left(r_i(\theta, y) - r_j(\theta, y)\right)^2 \tag{3.2}$$

where $j = (i + 1)$ mod n. In the above equation, the summation over θ is done for those values of θ for which we have common resampled points on the overlapping surfaces from i-th and j-th views for $j = (i + 1)$ mod n. The error measure $\varepsilon(y)$ is computed for various values of $(X_0(y), Z_0(y))$ near the initial estimates (X_0, Z_0). In the experiments, this was done at 0.1 mm intervals in the range $[X_0 - 5, X_0 + 5]$ and $[Z_0 - 5, Z_0 + 5]$. The interval for resampling in the θ space was 1 degree.

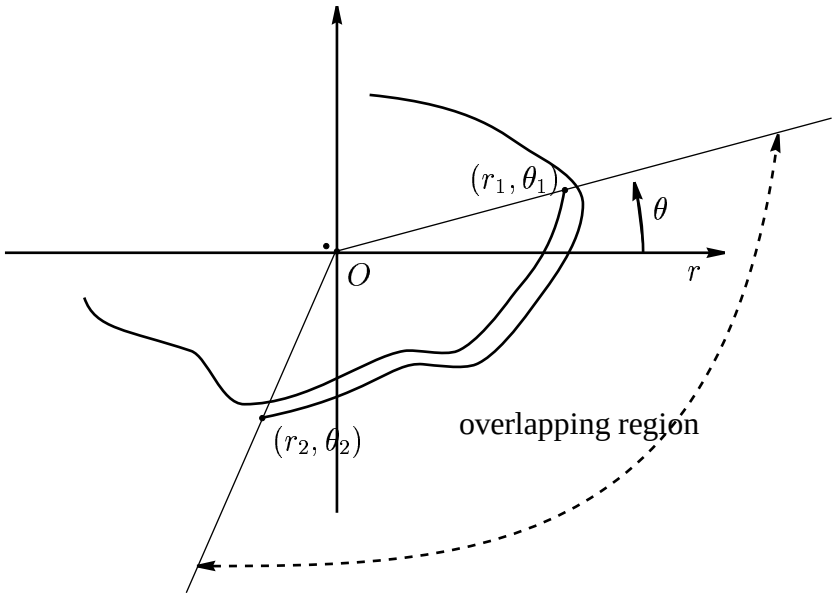


Figure 3-4: Overlapping region of two registered curves

The position where the error measure was a minimum was taken to be the correct position of the rotation axis.

Although Eq. (3.2) is used to minimize the error for two consecutive views, it can also be applied on the overlapped region of more than two views. The average of more overlapping range images should give us more accurate result. Figure 3-2 shows the cross sections of an object before and after registration. Four curves from different range images with only rotation axis calibration is shown in Figure 3-2(a). The initial rotation center is given as $(0.0, 833.0)$. After fine registration, the new rotation center with least squared error minimization is located at $(0.7, 830.1)$ as shown in Figure 3-2(b).

3.4 Iterative Registration

Registering range images in an iterative way is popular because of its ability of updating information after each iteration. Most of the iterative registration methods try to find the translation vector and rotation matrices by either matching the closest points from different views [10] or minimizing the distance from points in one view to planes in another view [17]. In this section, we propose a new approach

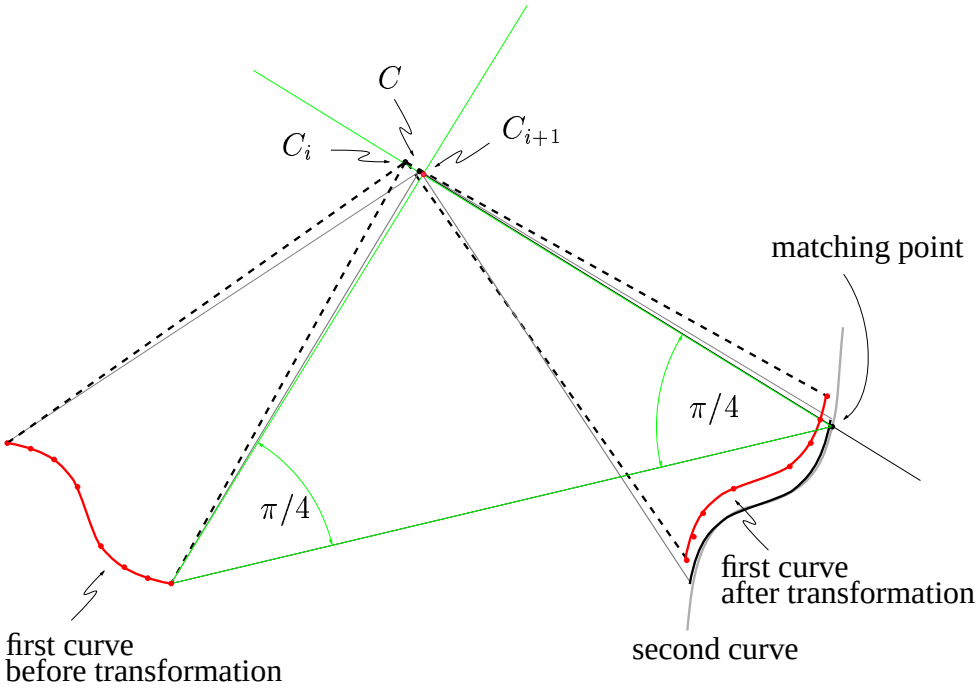


Figure 3-5: Finding the new rotation center

which directly computes the rotation axis after each iteration without data fitting or minimization. Assuming the rigid body motion of the object is known (e.g. the rotation angles in our case), selected control points before and after current transformations are used to find the next transformations. Those control points are chosen from the overlapping part of two range images with certain criteria. Since the unit vector of rotation axis is close to $(0, 1, 0)$, we will only consider the two-dimensional case (translation vector).

Two-dimensional registration is to find the rotation center of the rotation transformation. The goal is to partly match the curves on the same plane from two different views. Consider the registration model shown in Figure 3-3, two curves are observed from different viewpoints with 90 degrees rotation angle (viewpoint 1 and viewpoint 2). The curve from the second viewpoint is transformed back to the world coordinate system according to the initial or estimated rotation center as shown in the right figure. Then the overlapping part of these two curve is used to update the rotation center for the next rotation transformation. This process is repeated until the rotation center converges.

For each iteration, the next rotation center is found by three steps– (i) finding

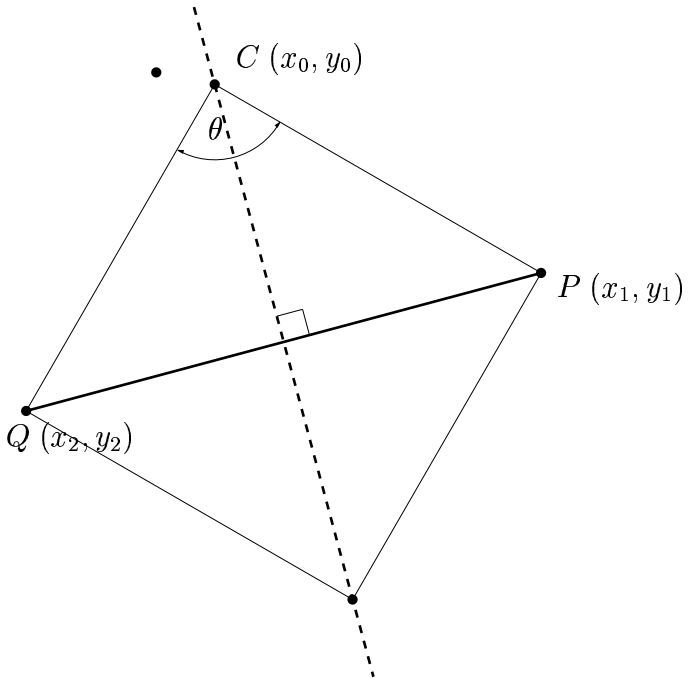
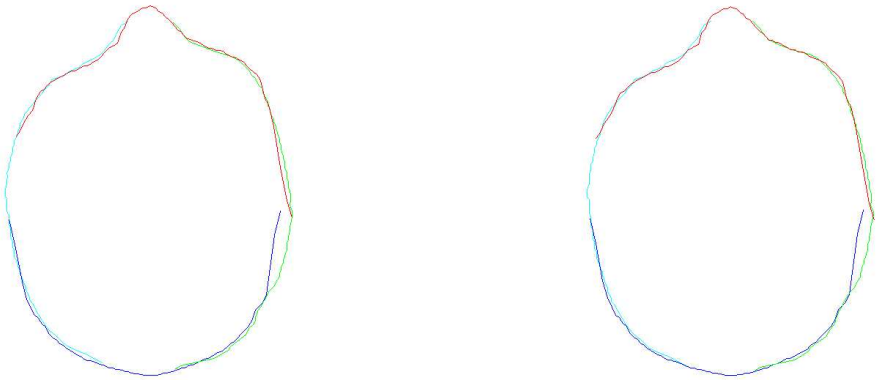


Figure 3-6: Calculation of the new rotation center

the overlapping part of two curves, (ii) finding the *matching point* on the second curve, (iii) using the matching point and the corresponding point on the first curve to compute the rotation center. To find the overlapping part of the registered curves, we first convert the data points to the polar coordinate system centered at the estimated rotation center (see Figure 3-4). The angles (θ values) are used to detect the overlap of these two curves. Let (r_1, θ_1) be the last point on the first curve and (r_2, θ_2) be the first point on the second curve, then the overlap can be specified by all the points (r, θ) with $\theta_2 \leq \theta \leq \theta_1$. Since the points from the second view will be used to update the rotation center, we only consider the overlapping part on the second curve. Here we assume that there exists a one-to-one mapping between r and θ on all data points and this is true for simple objects as we have previously defined. Generally coarse registration gives us the required accuracy on rotation center to find the overlapping region. In the implementation, to ensure that it is robust under the presence of noise, we only use half of the points on the central part of the overlap.

The second step is to choose proper points for computing the new rotation center. Since we assume the rotation angle is fixed (e.g. $\pi/4$ in this case), the rotation



(a) Curves before registration (b) Curves after registration

Figure 3-7: Cross section curves of an object before and after fine registration

center can be found if the points are known before and after rotation transformation. Let the overlapping data points on the second curve be P_1, P_2, ..., P_n, the current rotation center be denoted as C, and Q_1, Q_2, ..., Q_n be the projection of P_1, P_2, ..., P_n on the first curve along the lines $\overline{CP_1}$, $\overline{CP_2}$, ..., $\overline{CP_n}$. On the discrete sampled data, the projected point is given by the intersection of the projection line and the line segment of the data points on the first view. The *matching point* is defined as the projected point Q_i which satisfies

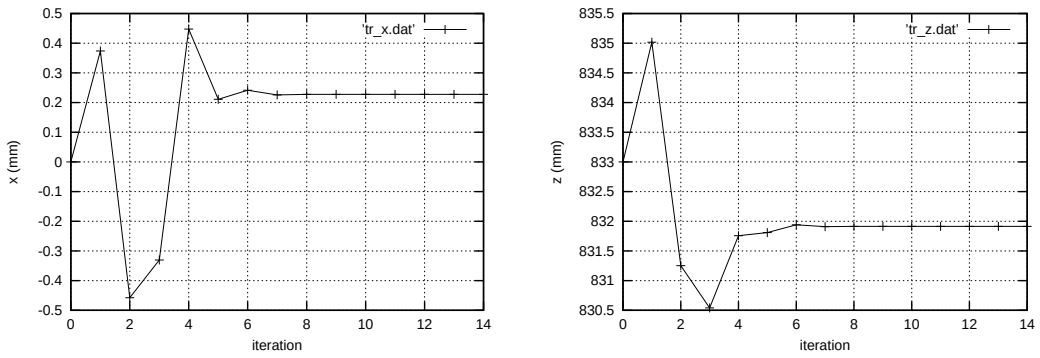
$$\overline{P_iQ_i} = \max_{1 \le j \le n} \overline{P_jQ_j} \tag{3.3}$$

That is, the matching point is a point on the first curve and farthest to the second curve under projection. For a given data point on the first curve, the line segment connected by two consecutive data points on the second curve to be intersected is found by testing all of the overlapping line segments on the second curve. The line segment is then used to find the distance in Eq. (3.3).

The matching point Q on the second curve and the corresponding data point P on the first curve before transformation are used to compute the new rotation center. Let (x_1, y_1) be the point before transformation and (x_2, y_2) be the matching point, then the rotation center (x_0, y_0) is given by

$$x_0 = (x_1 - y_2 + y_1 + x_2)/2 \tag{3.4}$$

$$y_0 = (y_1 + y_2 + x_2 - x_1)/2 \tag{3.5}$$



(a) x direction (b) z direction

Figure 3-8: Convergence of the translation vector

or

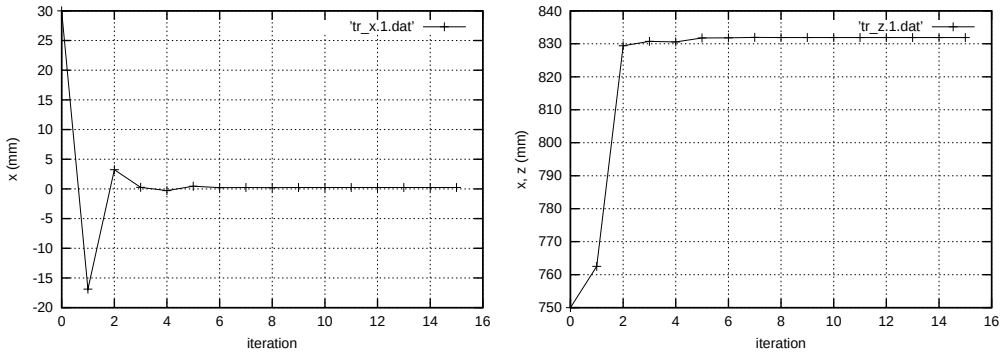
$$x_0 = (x_1 + y_2 - y_1 + x_2)/2 \tag{3.6}$$

$$y_0 = (y_1 + y_2 - x_2 + x_1)/2 \tag{3.7}$$

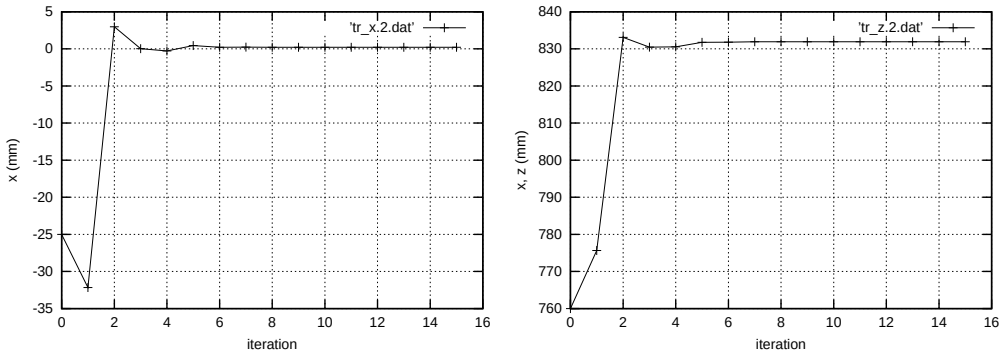
for a fixed rotation angle of $\pi/2$. There are two solutions located on both sides of $\overline{PQ}$. The ambiguity is solved by choosing the one on the same side of the previous rotation center (see Figure 3-6).

In the experiment, the cross sections of an object are used to find the rotation center. For any two range images from consecutive viewpoints, the rotation center is updated by taking the average of the rotation centers computed from the cross sections. Figure 3-7 shows the cross sections of an object before and after registration. Four curves from different range images before registration is shown in Figure 3-7(a). The initial rotation center is given as $(0.0, 833.0)$. After registration, the rotation center is moved to $(0.227, 831.914)$. This reduces the error between curves and the result is shown in Figure 3-7(b). Figure 3-8 shows the actual values of the translation vector converging with iterations. In the experiments, it shows that the convergence of translation vector is very fast (usually less than 20 iterations).

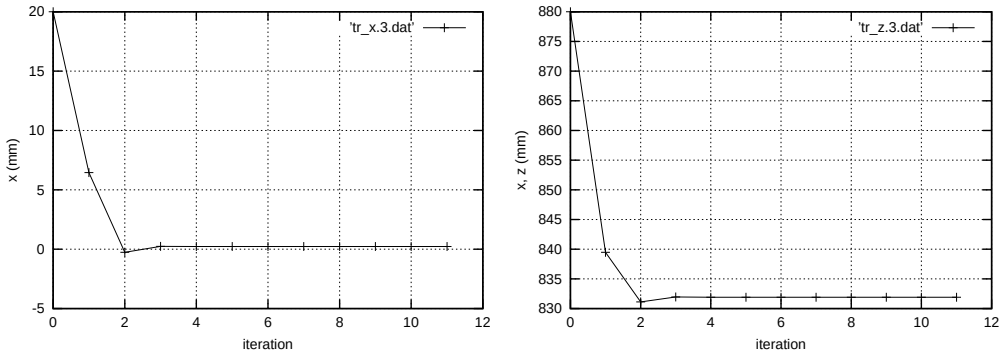
This iterative registration method is also very robust under poor initial values. In Figure 3-9, we show some results of the convergence of translation vector with different starting points. The initial rotation centers are given as $(30.0, 750.0)$, $(-25.0, 760.0)$ and $(20.0, 880.0)$. After several iterations, they all converge to the same value $(0.227, 831.914)$. The cross sections before and after registration for the initial value $(20.0, 880.0)$ are shown in Figure 3-10. Figure 3-10(b) is the same as 3-2(b).



(a) Convergence of translation with vector starting point at $(30.0, 750.0)$

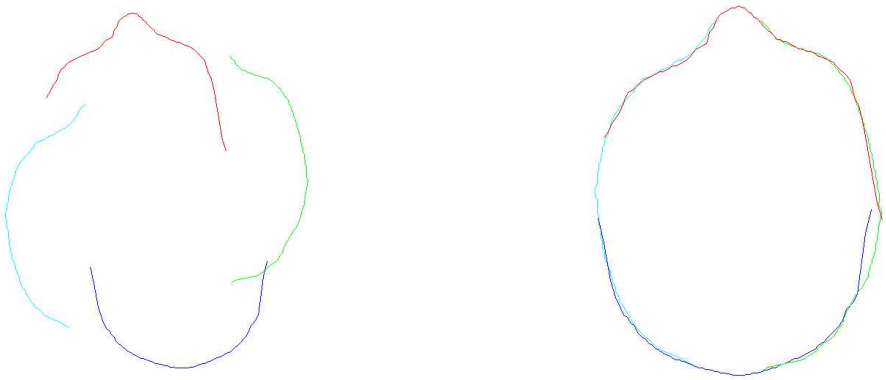


(b) Convergence of translation with vector starting point at $(-25.0, 760.0)$



(c) Convergence of translation with vector starting point at $(20.0, 880.0)$

Figure 3-9: Convergence of the translation vector with a poor starting point



(a) Curves before fi ne registration (b) Curves after fi ne registration

Figure 3-10: Registration with a poor initial rotation center

3.5 Discussion

In this chapter we describe two methods for data registration. The first algorithm searches for the best rotation center in a given region by minimizing the error between range images. Sum of differences on the uniformly resampled points is used to calculate the error. The second algorithm updates the rotation center iteratively by direct computation using the current matching points. Without searching and minimization, it converges very fast even with poor initial estimates. The first method usually takes more time to obtain a high precision result compared to the second method due to the small search interval. However, it will not give local minima which may occur in the second method.

As we can see on the registered curve (e.g. Figure 3-10), they do not perfectly match with each other. This causes problems because smooth data points are preferred for surface integration. To overcome this problem, we can take the weighted average on the overlapping parts of the range images [25], which will be discussed in the next chapter.

Chapter 4

Surface Integration

4.1 Introduction

Once the registration process has been completed, the data contained in each view can then be transformed into a single object coordinate system for integration. Depending on the type of sensor that captures the data and the method of registration, the points on the surface can be in one of the two basic forms– *unstructured data set* and *structured data set*. In the first case, the range data form a point cloud without spatial connectivity information. In the second case, the data are structured via relationships (such as image-plane connectivity) known a priori in each view. The method of integration depends on the form of the input data set (structured or unstructured) and the final representation required by the application. They are usually classified as *structured* and *unstructured* integration methods.

Unstructured integration algorithms create a polygonal surface from arbitrary collection of 3D points. In this case, the integration is performed by collecting together all the data points from multiple scans and presenting them to the polygonal reconstruction procedure. Structured integration algorithms make use of information about how the range data are obtained, such as adjacency information between points within one range image. Our range data obtained in Chapter 2 falls into the second category. The range images from different viewpoints are represented as a rectangular grid. The goal of our integration algorithm is to combine the partly overlapping data sets to a complete non-redundant 3D data set without any loss of detail in the original raw data.

In the following sections, we first review some related work on surface reconstruction. We then describe our surface integration algorithms and show some experimental results. We also evaluate the accuracy of our algorithms. A short discussion concludes the chapter.

4.2 Related Work

Surface reconstruction from range data has been extensively investigated over the past several decades. The methods in the literature that are able to create seamless meshes from 3D range data set can be divided into two groups, one is *volumetric approach* and the other one is *surface approach*. The volumetric approaches [39, 5, 68, 21, 37, 91] first store the 3D data points into a volumetric data structure, either a voxel grid or an octree. Each voxel contain eight vertices and by evaluating a field function at those vertices it is possible to extract a level surface of the field function. The field function is often chosen as a signed distance function. The triangular mesh is then created using an iso-surface extraction algorithm such as *marching cubes* algorithm [48]. The surface approaches [85, 70, 75, 76, 58] create an initial set of triangulated meshes from the original 3D range images. These meshes are then merged together to create the final complete 3D model.

Volumetric approaches work on both structured and unstructured input data. Hoppe *et al.* [39] introduce an algorithm to construct 3D surface models from a cloud of unorganized points without spatial connectivity. They determine an approximate tangent plane at each data point using least squares on k nearest neighbors, and then take the signed distance to the nearest point's tangent plane as the distance function in 3D space. The distance function is then interpolated and polygonalized by the marching cubes algorithm. In two subsequent steps [40, 41], the constructed mesh is optimized (i.e., the number of triangles is reduced while the distance of the mesh from the data points is kept small) and then a piecewise smooth subdivision surface is built on it.

Curless and Levoy [21] propose a similar approach to Hoppe's algorithm with a few differences. They integrate distance estimates at each voxel instead of searching for the closest point from a voxel's center to determine the signed distance. The range images are taken separately and scanned along the line of sight to each of them. The integration is done on the signed distance to the point for each voxel the line passes through. The final signed distance estimate is a weighted average of all the estimates from different range images. The marching cube algorithm is then used to approximate the zero set of the distance function by a set of connected triangles using the values of the distance function at the voxel vertices.

Pulli *et. al.* [62] propose a simpler and attractive method for volumetric approach. They classify the voxels in the volumetric grid for each range image as either outside, inside or on the surface of the object by using geometric properties about the viewing angle. Their algorithm recursively subdivides each voxel classified to be on the surface into eight smaller voxels and the process is stored in an octree data structure. By some simple rules the classifications of the voxels from the different range images are combined into one common classification. Those

rules are also used to remove the outliers. When a predefined level of refinement is reached, a triangular mesh is then constructed from those voxels.

Reed *et al.* [66] develop a volume-based method using a constructive solid geometry (CSG) technique to form a solid model of the object from multiple registered range views. They create a solid for each view by sweeping the range data from the object away from the scanner filling in the space self occluded by the object. The surfaces of the solid are labeled as being either visible to the sensor or occluded by the object view. This labeling technique is used for view planning since it provides the information about which portions of the current viewspace are not covered by the previous views. Once all the visible portions of the object have been covered by one or more views, the view solid models are intersected to form a complete model.

Turk and Levoy [85] propose a polygonal method that fits a triangular mesh to each range image. They employ an incremental algorithm that updates a reconstruction by eroding redundant data, followed by "zippering" along the remaining boundaries. The final "consensus" step reintroduces the original geometry to establish final vertex positions. Their method utilizes the structure in each range image but can have bad behavior in areas of high curvature.

Soucy and Laurendeau [75, 74] describe a method that builds surface descriptions from multiple registered range images using Venn diagrams. They decompose range images into "canonical views", which are areas common to a unique subset of range images. A common reference plane is defined for each canonical view and the data points are projected onto it. Sets of common points are found using neighborhood and visibility tests. A Delaunay triangulation is made on each reference plane and they are combined to form the complete model by re-parameterization.

4.3 Integration with Global Resampling

The first integration algorithm we developed for surface reconstruction is based on (r, θ, y) space representation. It works under the *simple object* assumption as described in Section 3.3– for each cross section of the object perpendicular to the rotation axis, there is a 1-1 correspondence between any point on the contour and the angle θ of the direction vector from the rotation center to that point. Linear interpolation is used to resample the data points to create a rectangular grid on the object's surface. The complete 3D model is then described by vertices and quadrilaterals provided by the rectangular grid.

Similar to the registration process described in Section 3.3, the coordinate systems used for integration are shown in Figure 3-1. Any data point (x_i, y_i, z_i) of the ith range image is first converted to a representation in the cylindrical object

coordinate system (r, θ, y) by

$$\begin{aligned} \theta &= \phi_i + \tan^{-1} \frac{x_i - X_0}{Z_0 - z_i} \\ r &= \sqrt{(x_i - X_0)^2 + (z_i - Z_0)^2} \\ y &= y_i \end{aligned} \tag{4.1}$$

where ϕ_i is the rotation angle of range image or viewpoint i with respect to the first viewpoint and (X_0, Z_0) is the rotation center. At this point, only the data points that are part of the object are retained, and other points belonging to the background are eliminated. Having retained only points on the object, all the points from different views are merged into one set and represent a discrete sampling of the visible surface of the object. In the object coordinate system, this merged set of points represent a discrete sampling of the visible surface of the object. Since visible surface is assumed to be simple so that it can be represented by a function of the form $r(\theta, y)$ in the cylindrical object coordinate system, the merged set of points can be thought of as a discrete sampling of this function. Given these discrete samples, we obtain a complete 3D model of the object as follows.

The discrete sample points are used to interpolate and uniformly resample the object's surface in the (θ, y) space. In the experiments, the resolution of range images is 27×27 and the object's surface is uniformly resampled by 27 points in the y space, and 120 points (3 degree intervals) in the θ space. The resulting rectangular sampling grid in the (θ, y) space is used to define a set of vertices (corresponding to sample points) and quadrilaterals (corresponding to rectangles in the grid) that give a piecewise approximation of the object's 3D shape. In order to render the 3D shape, the coordinates of the vertices are computed in the Cartesian object coordinate system from their cylindrical coordinates as follows:

$$\begin{aligned} x &= r \sin\theta \\ y &= y \\ z &= r \cos\theta \end{aligned} \tag{4.2}$$

In our experiments, we used a simple separable linear interpolation scheme for resampling. First the 27×27 depth-map obtained for each of the four views of the object was resampled vertically at 27 points along the Y-axis. Then the points on the object were represented in the cylindrical object coordinate system. After this, for each value of y, the surface was resampled at 3 degree intervals (120 points) using a simple linear interpolation scheme.

The results of a cylinder object and a box object, both with different geometric shapes pasted on their outer surfaces, are shown in Figure 4-1 and Figure 4-2

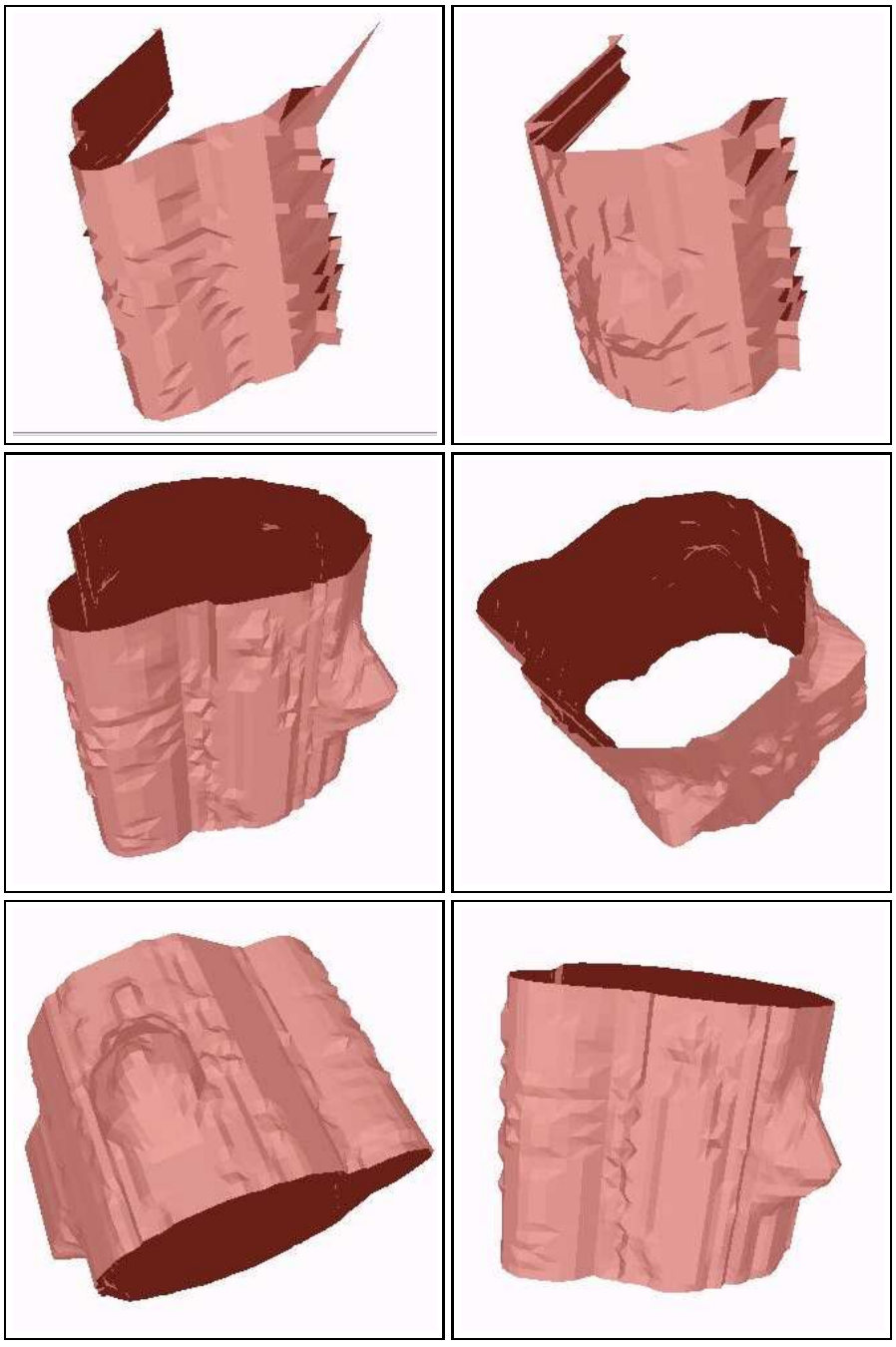


Figure 4-1: Partial 3D shapes and the complete 3D model of a cylinder object

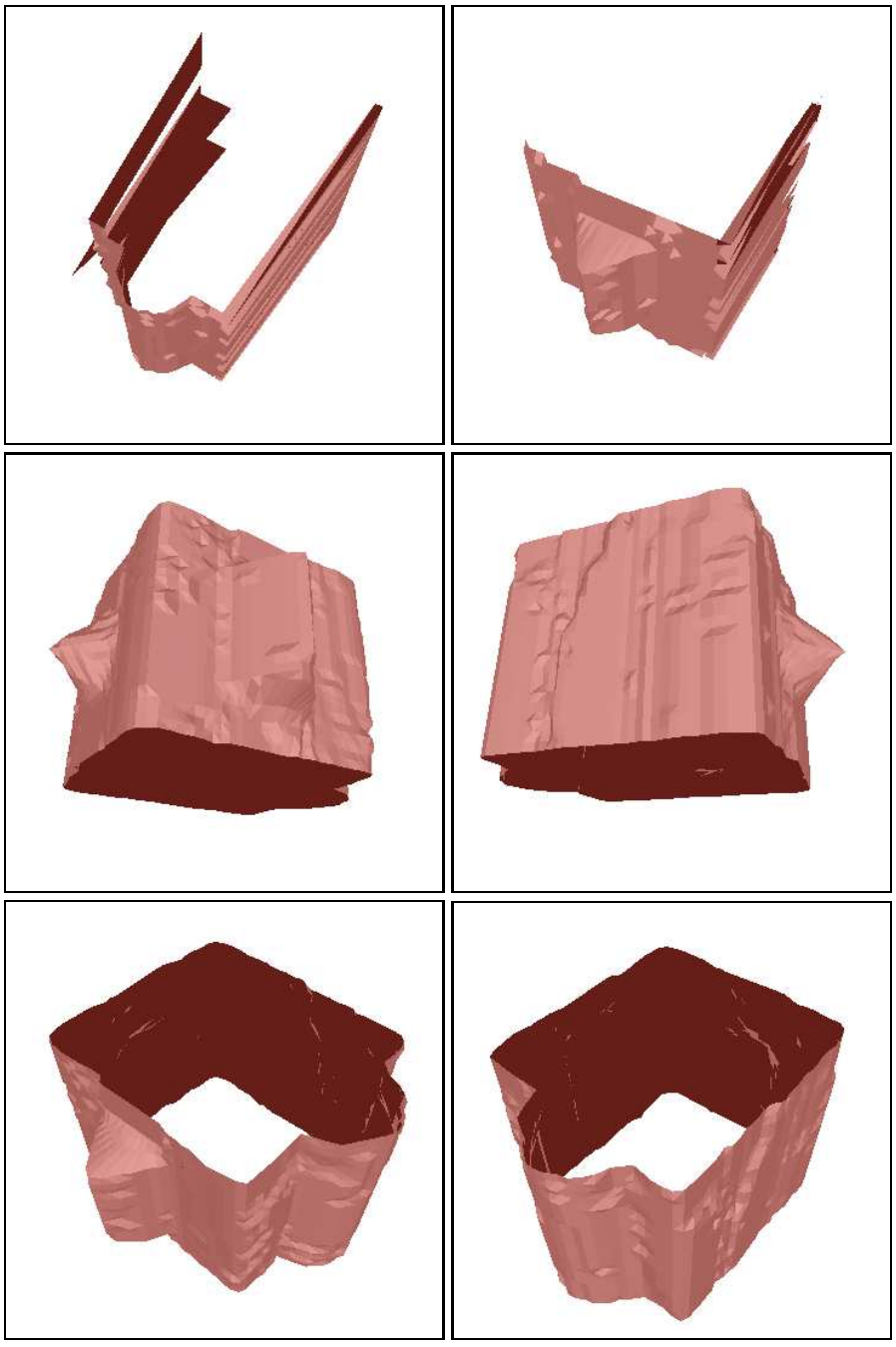


Figure 4-2: Partial 3D shapes and the complete 3D model of a box object

respectively. Two figures on the top are the partial 3D shape from different viewpoints. These range data are acquired using our previous vision system SVIS with Olympus Deltis VC-1000 digital camera. The resolution of 3D shapes are 27×27 recovered from 640×480 intensity images using conventional parallel stereo. The figures on the bottom are the reconstructed 3D model viewed from different directions.

4.4 Region-of-Construction Algorithm

Our input range data acquired in Chapter 2 is given on a regular grid of points with spatial connectivity for each partial 3D model. Therefore, we develop a specialized triangulation method depending on the acquisition viewpoints. The basic idea is to *stitch* the *Regions-of-Construction* of different viewpoints at their boundaries to create a complete 3D model. This algorithm takes advantage of the known topology of each range image and does not involve any spatial search of the data points. The mesh triangulation is done using indices of data points on a grid network. As a result, it is much faster than the general algorithms using our input data.

Given two registered range images from adjacent viewpoints, the overlap between them is illustrated in Figure 4-3. In order to directly stitch these two range images to create a single mesh representation, the redundant data on the overlapping part has to be removed. To select and remove the redundant points, it is reasonable to begin from the edges of the range images since the data points are usually less accurate when they are away from the center of a range image (due to imperfect projection model and lens distortion, etc.). In this algorithm, the overlapping part of range images from two consecutive viewpoints is divided into two regions and each region is assigned to one of the range images for construction. Region-of-Construction for each range image is created and used to obtain a non-overlapping data set for surface integration.

We first give the definition of Region-of-Construction for a range image. For a set of registered range images, Region-of-Construction is defined as a part of a range image and does not overlap Regions-of-Construction of any other range images. The two-dimensional case of Region-of-Construction on a cross section of an object is illustrated in Figure 4-4. The object is assumed to be completely described by n viewpoints. For each viewing direction, the left and right *line-of-sight* is defined as the line determined by the viewpoint and the leftmost and rightmost data point, respectively. The equiangular line passes the intersection of right (left) line-of-sight of $i-1$ ($i+1$) range image and left (right) line-of-sight of i range image is defined as the left (right) *line-of-division* for range image i. The 2-D

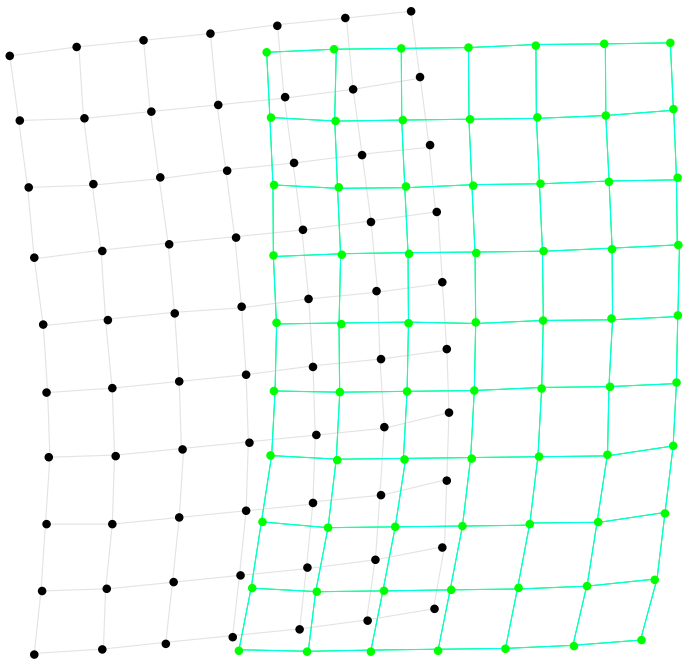


Figure 4-3: Overlap of two range images from adjacent viewpoints

Region-of-Construction for each cross section of an object is then bounded by the left and right line-of-division. For each viewpoint, Region-of-Construction is made by all of the 2-D cases on the cross sections.

The way we define and create Region-of-Construction ensures that the central part of a range image is always retained for surface reconstruction and the data points near the edges of the range image are removed. Since Region-of-Construction is obtained from the 2-D cases of each cross section, generally the perspective projection of each row of the range images cannot be directly used. Resampling along the vertical direction of range images can be carried out first. However, the error due to perspective projection in our experimental setup on SVIS-2 camera system can be neglected in most real objects because the depth of field is relatively small compared to the object's distance. The lines-of-division for each row of range images are computed on the vertical projection on the cross section of the object.

For each Region-of-Construction, the triangular mesh is created as follows. We start at the upper left corner of the range image, find the points which belong to the Region-of-Construction, mark them as *valid points* and establish the connection

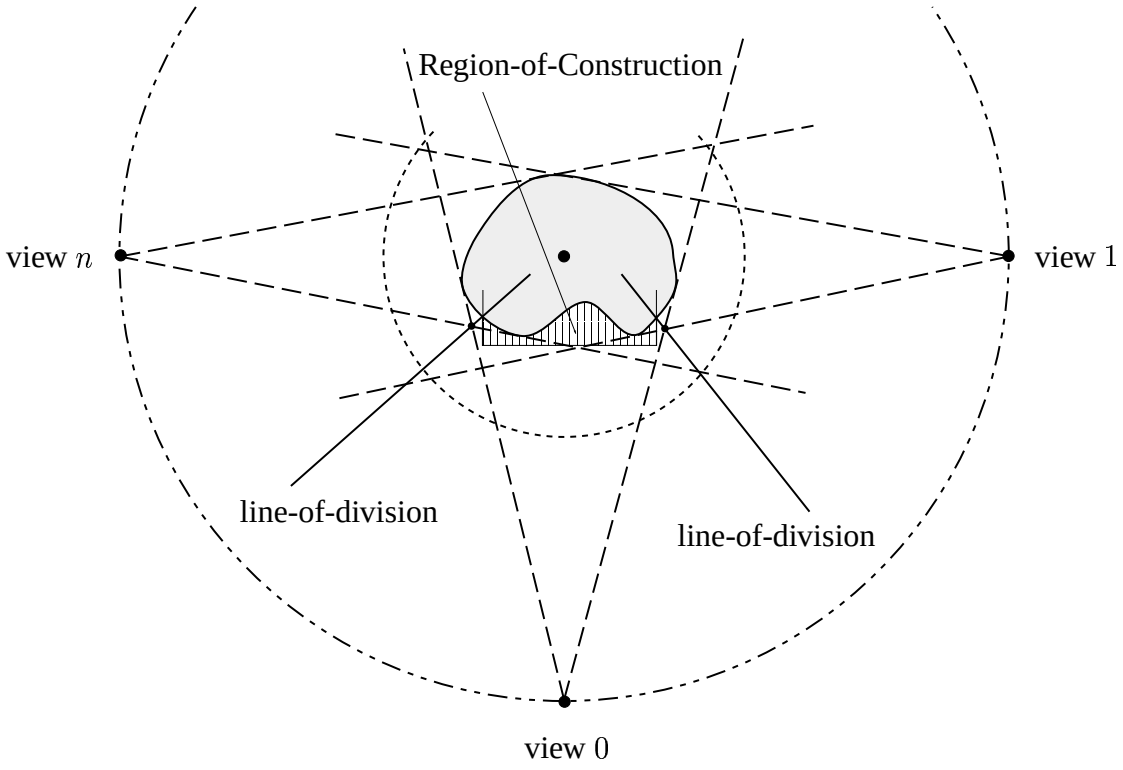


Figure 4-4: Region-of-Construction on a cross section of an object

[70]. For each valid point, take it as the upper left corner vertex of a polygon connection. There exist five possible tessellations for triangles or quadrilaterals as shown in Figure 4-5.

For the first case the column index of the first valid point in the ith row is smaller than the column index of the first valid point in the jth row, where $j = i+1$. Let p and q be the column index of the first valid point in the ith and jth row, respectively. Then we have $p < q$ for these two adjacent rows. The triangulation in this case is done by connecting the vertex indices $(j, q) - (i, p) - (i, p + 1)$, $(j, q) - (i, p + 1) - (i, p + 2), \cdots$, until $p = q - 1$. For the second case the column index of the first valid point in the ith row is larger than the one in the jth row, where $j = i + 1$. Let p and q have the same definitions as in the first case, then the triangles are created by connecting the vertex indices $(i, p) - (j, q) - (j, q + 1)$, $(i, p) - (j, q + 1) - (j, q + 2), \cdots$, until $q = p - 1$. For the third case the column index of the last valid point in the ith row is smaller than the column index of the last valid point in the jth row, where $j = i + 1$. Let p and q be the column

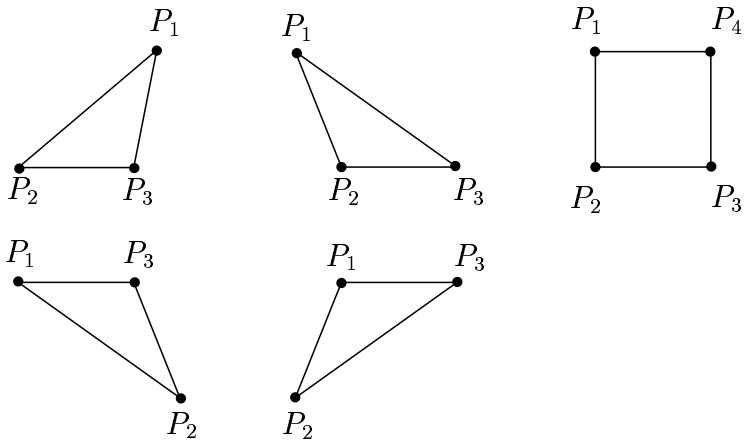


Figure 4-5: Five possible tessellations within a Region-of-Construction

index of the last valid point in the ith and jth row, respectively. Then we have $p < q$ for these two adjacent rows. In this case one or more triangles are created by connecting the indices $(i,p) - (j,p) - (j,p+1)$, $(i,p) - (j,p+1) - (j,p+2)$, $\cdots$, $(i,p) - (j,q-1) - (j,q)$ for any two adjacent rows. For the fourth case the column index of the last valid point in the ith row is larger than the one in the jth row, where $j = i+1$. Let p and q be defined as before, then the triangles are created by connecting the indices $(i,p)-(j,q)-(i,p+1)$, $(i,p+1)-(j,q)-(i,p+2)$, $\cdots$, $(i,p-1)-(j,q)-(i,p)$. In these four cases one or more triangles are created near the boundaries of Region-of-Construction depending on the column index difference between any two consecutive rows.

Finally we consider the case that four valid points adjacent to each other with one difference for both column and row indices. For the valid points in the ith and $i+1$th row with column index j and $j+1$, the mesh connection is given by $(i,j) - (i+1,j) - (i+1,j+1) - (i,j+1)$. In this case, lots of quadrilaterals in the central part of the range image are produced and the number of quadrilaterals depends on the difference between the last and first column index of two consecutive rows. Those quadrilaterals are further divided into triangles by connecting the shorter diagonals (see Figure 4-6). The resulting triangles are indexed by either $(i,j) - (i+1,j) - (i+1,j+1)$, $(i,j) - (i+1,j+1) - (i,j+1)$ or $(i,j) - (i+1,j) - (i,j+1)$, $(i,j+1) - (i+1,j) - (i+1,j+1)$.

Figure 4-7 shows an example of Regions-of-Construction created for four range images. The original range images are shown as shaded images including the background region. Region-of-Construction for each range image is indicated

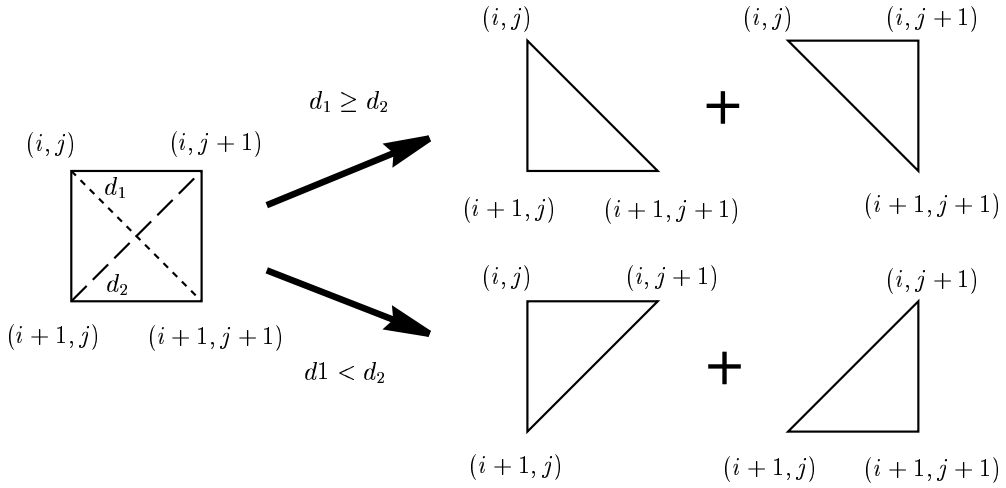


Figure 4-6: Quadrilateral are further broken into triangles with shorter diagonals.

by wireframe region. It can be seen that each Region-of-Construction is part of the original range image and do not overlap each other.

After the meshes for Regions-of-Construction are created for each view, the partial 3D models are *stitched* together by connecting the last valid point of the current range image to the first valid point of the next range image in the same row (see Figure 4-8). The resulting quadrilaterals on the boundaries of two range images are also broken into triangles with shorter diagonals.

In this algorithm the mesh of an object is created by combining four meshes from different viewpoints using only indices of the range images. It is not only fast (in a sense that no spatial search involved) but also facilitating the texture mapping process as we will see in the next chapter. Although individual meshes are preferable for texture mapping, a single mesh can also be created for wireframe or shaded 3D model. To create Region-of-Construction for each range image, we assume that the overlap only happens on two consecutive viewpoints. However, it can be extended to the cases that more than two range images overlap at the same time by considering only two adjacent range images.

Figure 4-9 shows the integrated and shaded 3D wireframe model of a toy object from several viewpoints. The top four figures are the low resolution model which contains 2,111 vertices and 4,147 triangles, whereas the bottom figures show the high resolution model which contains 8,328 vertices and 16,511 triangles. Figure 4-10 shows the integrated results of a bottle and a cylinder object. Figure 4-11 shows the complete 3D model of a head object. It contains 27,692 vertices and

Figure 4-7: Example of Regions-of-Construction of four range images of a real object. Only parts of the range images are stitched together to create a complete 3D model.

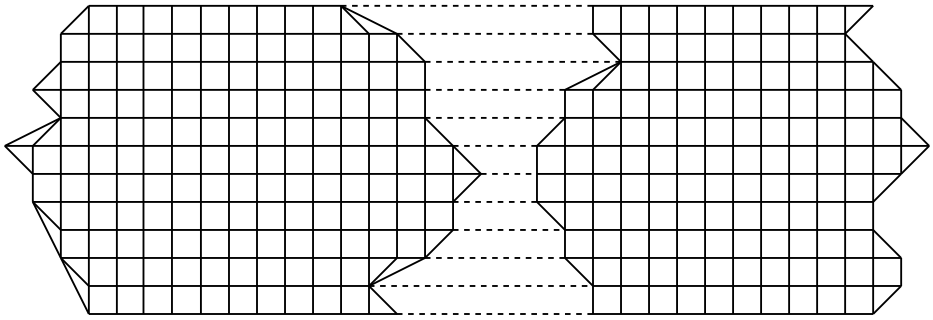


Figure 4-8: *Stitching* of two range images

55,111 triangles. In this example, the overall shape has been captured very well except the top area which cannot be seen by the camera. Figure 4-12 shows the images taken from different acquisition viewpoints of the test objects.

4.4.1 Complex Object with Holes

In this section we consider more complicated objects with holes. It is a relatively difficult problem in surface integration because simply projecting and merging range images on cylindrical or spherical grids will fail in this case [16, 61, 88]. Since the Region-of-Construction algorithm provides only non-overlapping surfaces, complex objects with holes can be reconstructed by selecting proper viewpoints that include the holes. We assume that if a hole can be observed from one viewpoint, it can also be observed from the *opposite* viewpoint. This generally holds for most real objects. Under this assumption, a 3D model is first constructed using the algorithm described in the previous section without considering its hole. Then the boundary points of the hole which belongs to two opposite viewpoints are connected using the following algorithm. (Again, the mesh is created using the indices of range images and does not involve spatial search.)

First, we connect the top and bottom rows of one range image to the top and bottom rows of the other range image from the opposite viewpoint respectively. The row indices can be different. The column indices of the top and bottom rows are used to create a triangle/quadrilateral mesh similar to the previous section. Without loss of generality, we consider the top rows i_1 for one range image with the hole and the top row i_2 for the other range image with the same hole. Let p_1, q_1, and p_2, q_2 be the first and last vertices for the first and second range image respectively. Then triangles are created for differences of p_1, p_1 and q_1, q_1, and quadrilaterals are created for the central parts with same column indices. For both side boundaries of

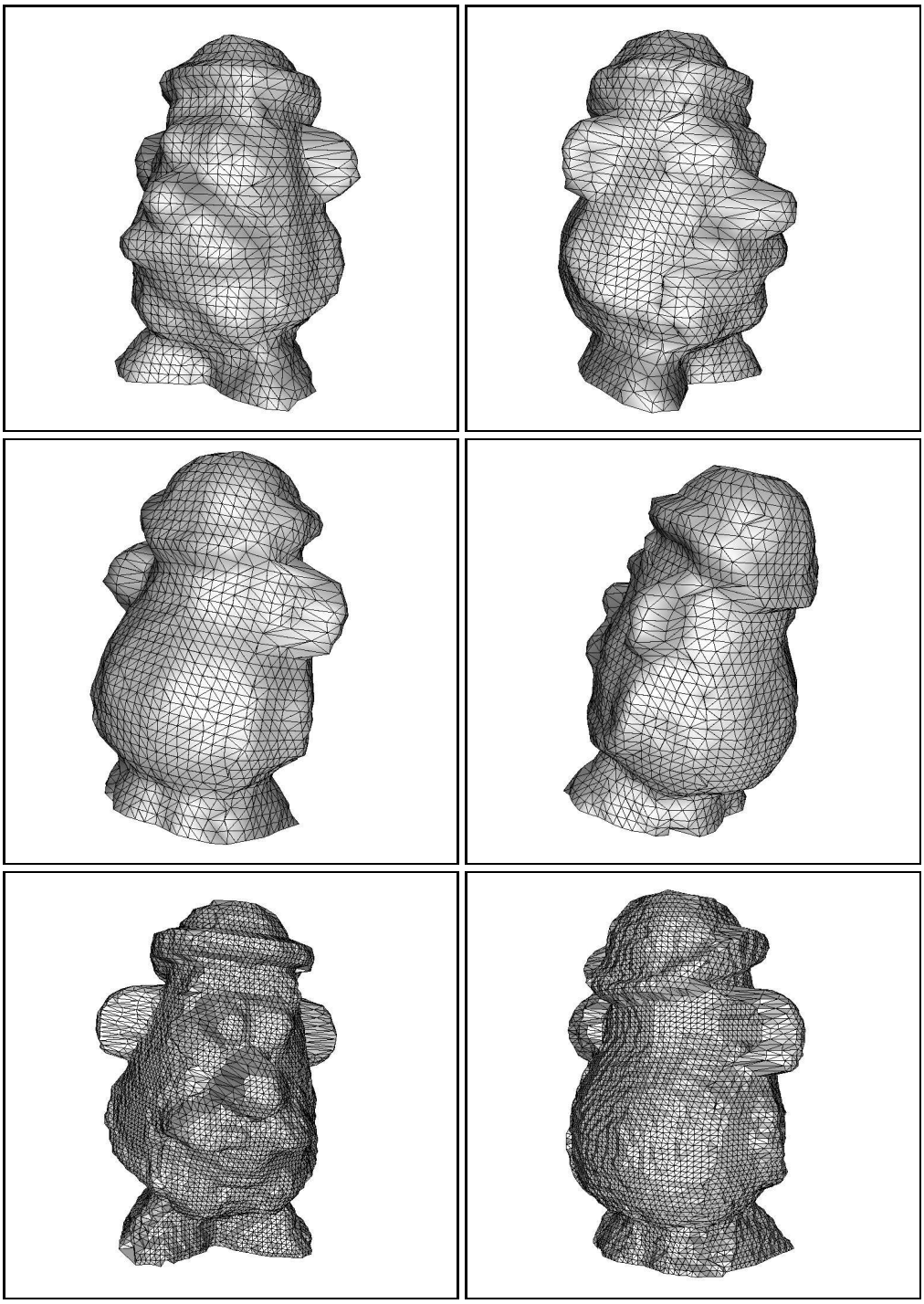


Figure 4-9: Wireframe model of a toy object

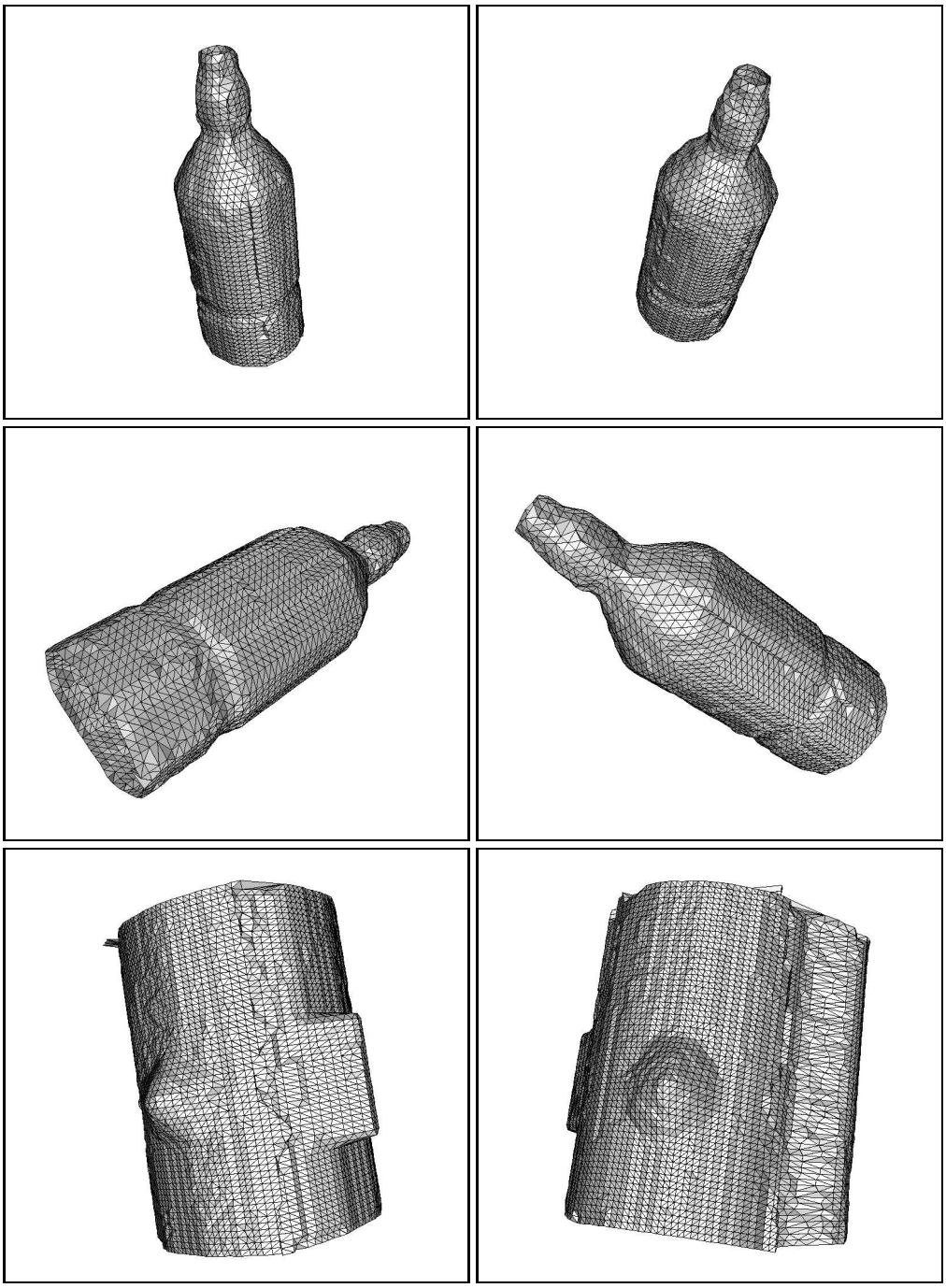


Figure 4-10: Wireframe models of a bottle and a cylinder object

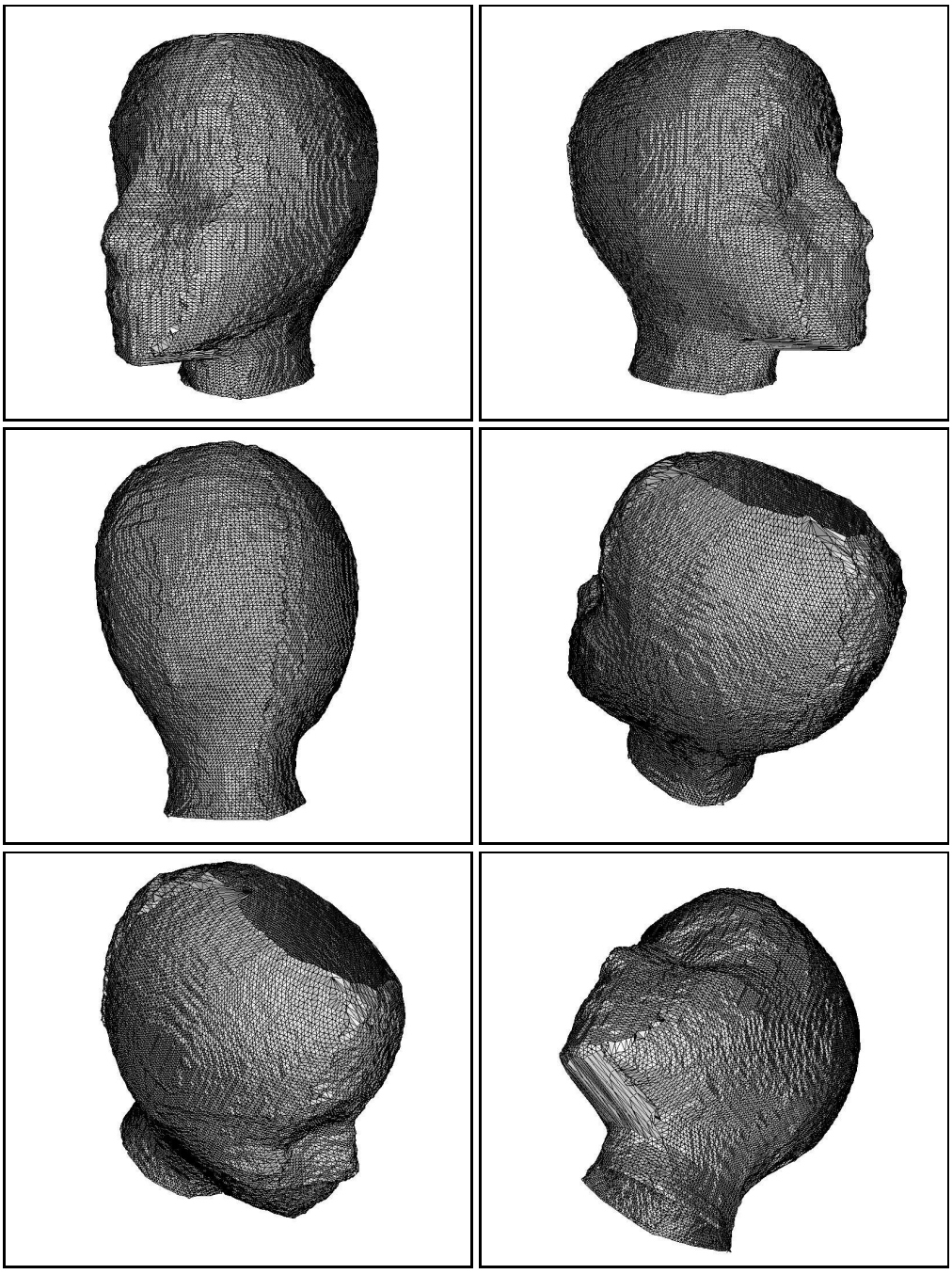


Figure 4-11: Wireframe model of a head object

Figure 4-12: Different acquisition viewpoints of test objects

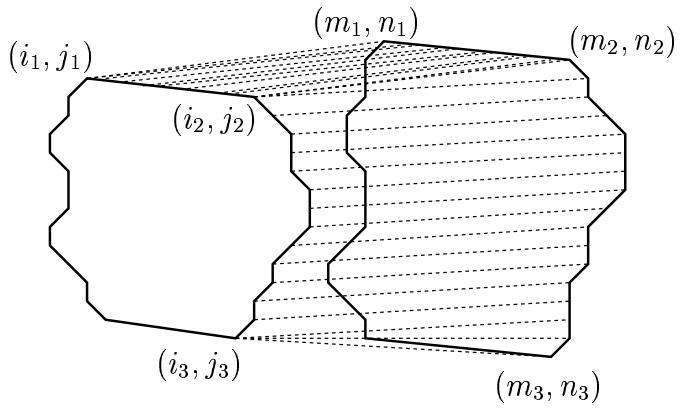


Figure 4-13: The index differences between (m, n) and (i, j) are used to generate quadrilaterals or triangles.

the hole, two triangle/quadrilateral meshes are created with variable row index and the fixed column index in a row (one belongs to object points and next to a background point). Figure 4-13 illustrates this *hole-creating* process. The quadrilaterals are further broken into triangles with short diagonals.

In some cases the observed hole does not appear in any Region-of-Construction defined in the previous section but it gives depth discontinuities in one of the range images (see Figure 4-14). To reconstruct the hole in this case, a *principal view* is first defined as the viewpoint containing the hole and then used for the hole-creating process. The lines-of-division of principal view are shifted towards the edges of the range image such that the new Regions-of-Construction can completely cover the hole. In Figure 4-15(a), a hole is partly covered by this Region-of-Construction. The principal view is hence selected and a larger Region-of-Construction is created to enclose this hole (see Figure 4-15(b).

4.4.2 Integration of Multiple Objects

In this section, we describe an algorithm to deal with surface integration of multiple objects. To create the 3D model of multiple objects in a scene, we assume the objects are separable in the vertical direction. Also, the separation of objects can be detected from at least two viewpoints (with opposite directions). This generally holds for most objects, possibly with some manual rearrangements. The range images for different viewpoints are first segmented for each object in the scene to create a range data set for each object. The 3D model for each object is then constructed using the integration algorithms described in the previous sections. Finally

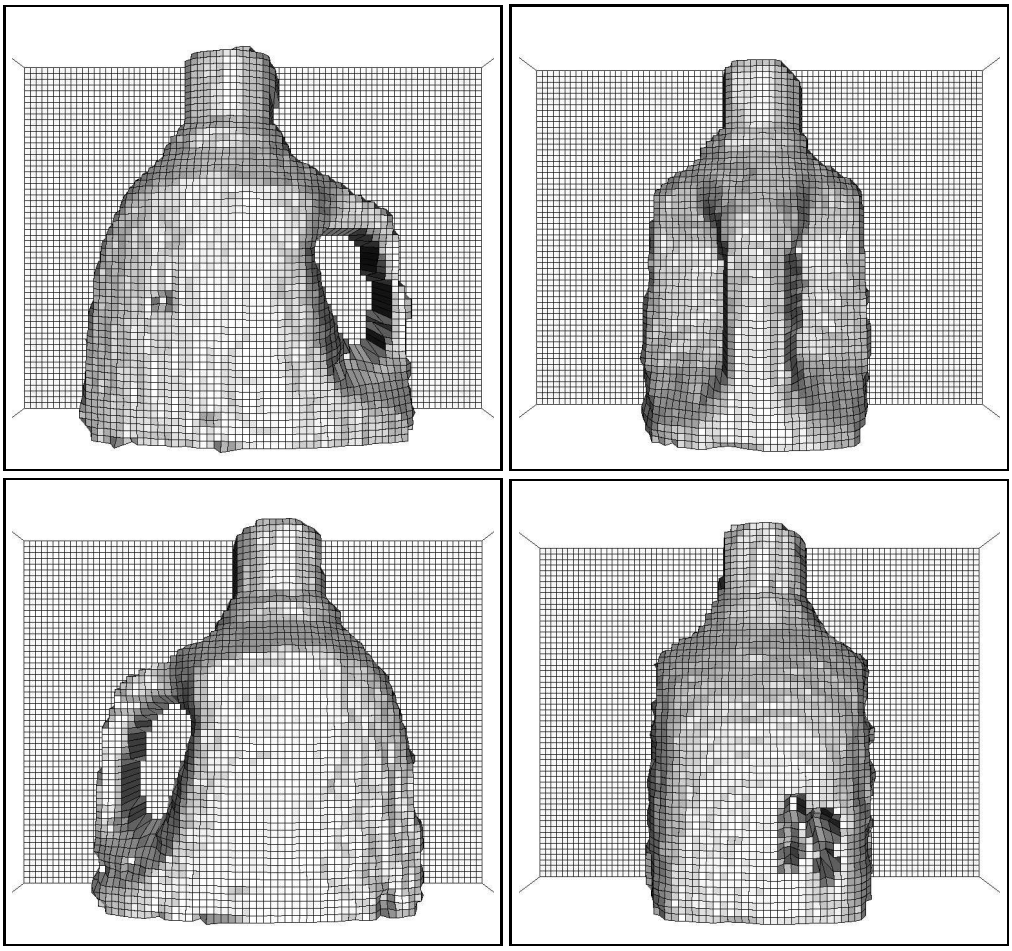
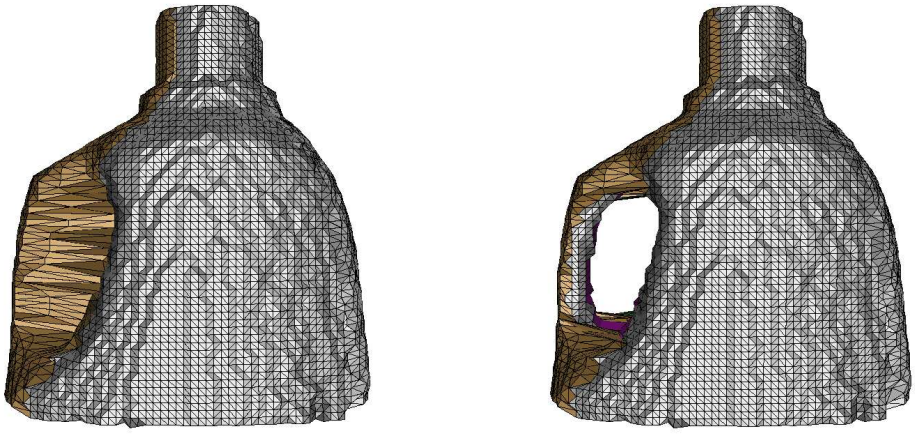


Figure 4-14: Object with a hole can be seen from two opposite viewpoints.



(a) 3D model without hole-creating (b) Principal view enclosing the hole

Figure 4-15: Extended Region-of-Construction in the principal view

the 3D models are combined together to form a single representation.

For each range image, the segmentation masks for the objects are obtained by considering the rotation geometry of the scene. A depth threshold is used to separate different objects in case of self-occlusion. In the experiment, four range images are obtained for every 90 degrees of rotation angle. The objects are placed on the rotation stage such that their separation can be detected for all viewpoints. The reconstructed focused images are shown in Figure 4-16. The top two figures show two partial 3D shapes from different viewpoints. The final 3D models are shown in the bottom figures for several viewing directions.

Due to perspective projection, the constructed 3D model looks distorted if the depth difference of the scene is too large. It shows that there are some missing parts in the 3D models, especially near the top and bottom of the objects. In Figure 4-17, the discontinuities and missing parts can be seen near the boundaries where we stitch partial shapes together. However, the data points on the reconstructed model are exactly the same as those from range image data– no information has been lost during surface integration.

4.4.3 Accuracy Analysis

To analyze the accuracy of a given reconstructed 3D model, one way is to compute the error of the final mesh relative to the original 3D range data [69]. This

Figure 4-16: Reconstructed focused images of multiple-object scene

type of accuracy analysis assumes that the acquired 3D data is accurate and focuses on minimizing the integration error on given range images. Usually it is done by the researchers who work mainly on integration and take range data from other sources. Another type of analysis on the accuracy of a complete 3D model is to measure the error between the reconstructed model and the real object. This method is more reasonable since the goal of reconstruction is to obtain a 3D model as faithful to the original object as possible. However, it is very difficult (if possible) to get the *accurate* 3D data of arbitrary object. Our accuracy analysis belongs to the second type because we provide the complete system for 3D model reconstruction which includes acquisition of range data. We are interested not only in the accuracy of the integration process from given 3D data but also in the overall accuracy of our vision system.

Since Region-of-Construction algorithm merges the original 3D data without any modification (only overlapping parts on range images are removed), the accuracy of our reconstruction depends on the accuracy of the range images and data registration. The accuracy of range images is determined by the depth maps ob-

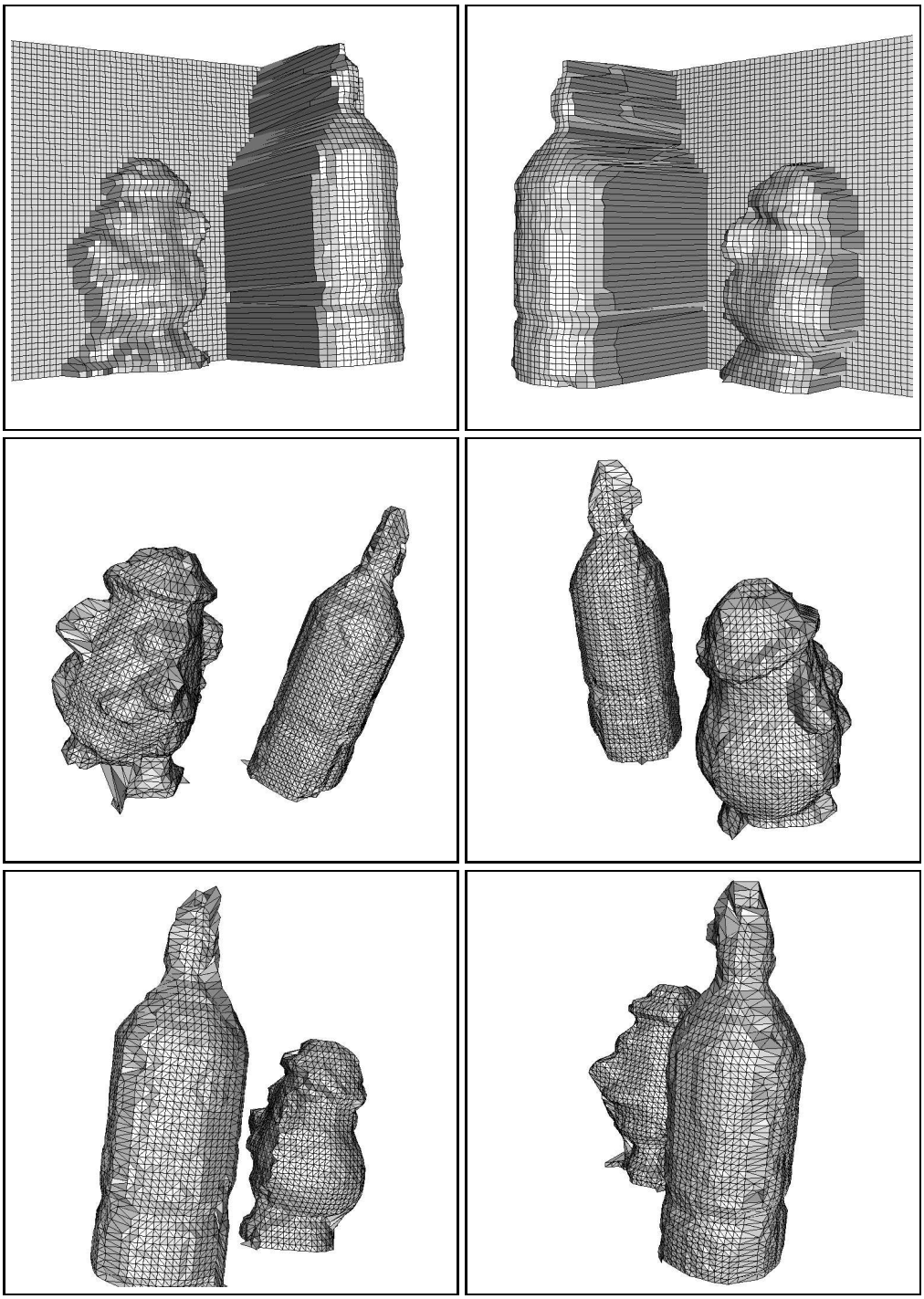


Figure 4-17: Partial and complete 3D models of multiple objects

tained from rotational stereo and inverse projection to the 3D space. As mentioned in Chapter 2, quantization error is always present in depth map because of stereo matching on digital images. To restore the 3D points for range images, we applied perspective projection model on depth maps. This can give inaccurate results especially near the edges of the range images due to camera lens distortion. The accuracy of data registration mainly depends on the rotation axis calibration and its refinement using overlapping parts of range images.

To measure the overall accuracy of our system, we use a cylinder as test object. The same procedure and algorithms are applied to obtain the complete 3D model. The acquired 3D data set is then fitted to the *perfect* cylinder from our physical measurement to calculate the average error of each data point acquired by our system. The calculation is given as follows.

Let d be the *actual* radius of the cylinder by physical measurement, $\mathbf{p}_1$, $\mathbf{p}_2$, $\cdots$, $\mathbf{p}_n$ be the sampled data points, and d_1, d_2, $\cdots$, d_n be the distances from the data points to the line that passes through the center of the cylinder. The average error for the 3D data set is then computed as $\frac{1}{n}\sum |d_i - d|^2$, where d_i is the distance from the data point $\mathbf{p}_i$ to the axis of the cylinder with line equation L_{eq} and denoted as $d_i = d(\mathbf{p}_i, L_{eq})$. Since the line equation L_{eq} is unknown in the camera coordinate system, it is approximated by minimizing the sum of distances from each data point to the line, i.e., $\min \sum d(\mathbf{p}_i, L_{eq})$. In other words, we are finding the best fitting cylinder for the acquired data set.

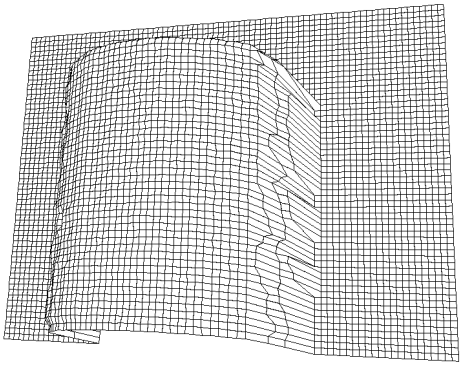
The test object and the reconstructed 3D model used for accuracy analysis is shown in Figure 4-18. The scattered 3D data points of one range image and the complete 3D model are shown in Figure 4-18(b) and 4-18(d). The measured diameter of the cylinder is 155 mm. By fitting the 3D points to the perfect cylinder, the average error for each data point is calculated as 0.44 mm. Therefore, the overall accuracy of the reconstruction is about 0.28% (or 1 part in 200).

4.5 Slice-by-Slice Algorithm

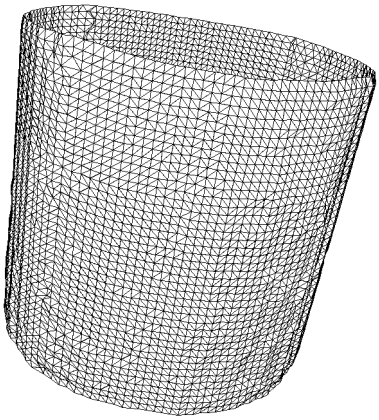
Surface reconstruction from polygonal contours on parallel slices in 3D is a well studied problem due to its number of applications such as computer tomography (CT) or magnetic resonance imaging (MRI) scan techniques [6, 12]. The input to the surface reconstruction problem is a series of parallel slices, each representing a cross section of the solid to be modeled. As illustrated in Figure 4-19, there are three major problems in such surface reconstruction [54]:

1. the correspondence problem– which contours should be connected by the surface?

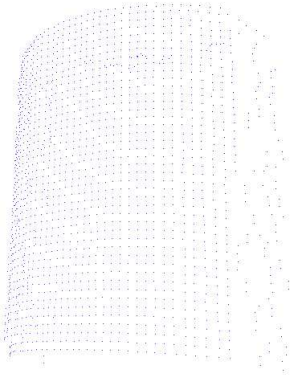
(a) Test object used for accuracy analysis



(b) Partial 3D shape of the front view



(c) Reconstructed 3D model



(d) Data points of the 3D model

Figure 4-18: Measurement of overall accuracy of 3D model reconstruction

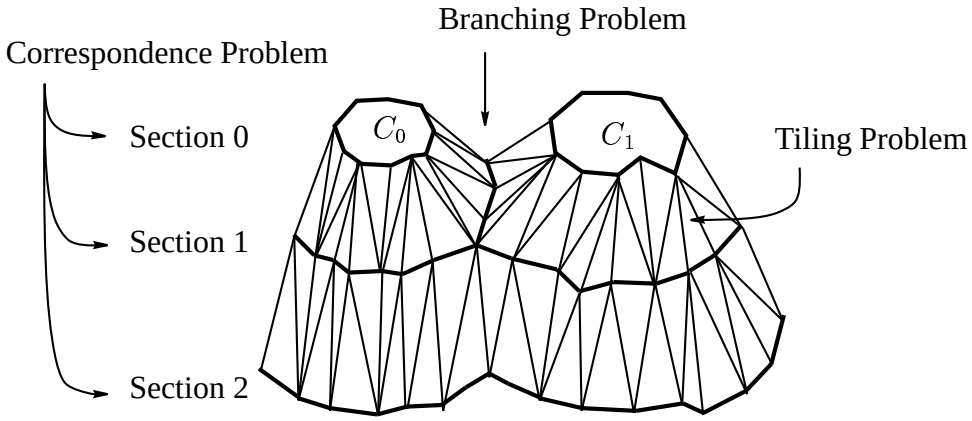


Figure 4-19: Surface from contours

2. the tiling problem– how should the contours be connected?

3. the branching problem– what do we do when there are branches in the surface?

The *correspondence problem* has to be solved by determining which of the contours from each of the cross sections should be connected together. The *branching problem* has to be solved by determining how to connect the two contours in section 1 to the one contour in section 2, and to each other. Since we only construct relatively simple objects with range image input data, these two problems are not considered here.

In this algorithm we focus on solving the *tiling problem* by choosing a "best" surface connection between two adjacent contours. As we mentioned earlier, the format of our input data is range image, which is different from those used for general surface reconstruction algorithm from a series of contours. Usually the 3D input from CT scan does not have redundant data on the overlapping parts between different scans. Therefore, we have to *integrate* the data points which are on the same cross section of the object but from different range images to obtain the contours for surface reconstruction.

Compared with our previous algorithm, this surface integration algorithm uses all of the available data points of an object from different viewpoints. After registration, the data points are treated as unorganized points within each row. First, the data points from all range images are sorted to create a sequence of points for any fixed row. Then, the nearest three points in every two consecutive rows (two points in one row and one point in the other row) are used to create the triangular mesh (see Figure 4-20). Since we assume that there is only one single connected

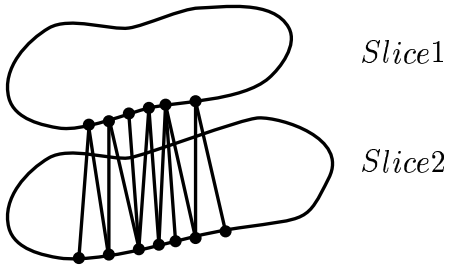


Figure 4-20: Slice-by-slice algorithm

contour for each cross section of the object, the data points are sorted according to their angle with respect to a reference point inside the object. The reference point has to satisfy the condition that there exists a 1-1 mapping between the data points and their corresponding angles. For a *simple* object defined in Chapter 3, this point can be the rotation center where the rotation axis intersects the cross section of the object. Currently the reference point is chosen as the centroid of a contour and the contours need not be convex.

Due to perspective projection, the same row of different range images does not correspond to the same cross section of the object. To construct the contour for each cross section, we make slices at equal distances along the vertical direction and assign the data points to the closest slice. The contour is then made by the data points of one slice without considering the y values.

Some integration results of 3D model are shown in Figure 4-21. This slice-by-slice algorithm provides a denser 3D model compared to the algorithm described in the next section. However, it is very sensitive to the input data. If the data set is not perfect (due to registration error, stereo mismatch or noise), the mesh connecting data points between range images back and forth will produce a non-smooth surface on the overlapping part.

4.6 Discussion

In this chapter we develop three algorithms for surface integration. The integration algorithm using global resampling provides a uniformly resampled grid on the object's surface. Resolution of the final mesh can be easily adjusted by increasing the number of sampling points on the object-centered cylindrical coordinate system. To deal with high curvature regions, second or higher order interpolations can be carried out instead of linear interpolation. The major drawback is that it can

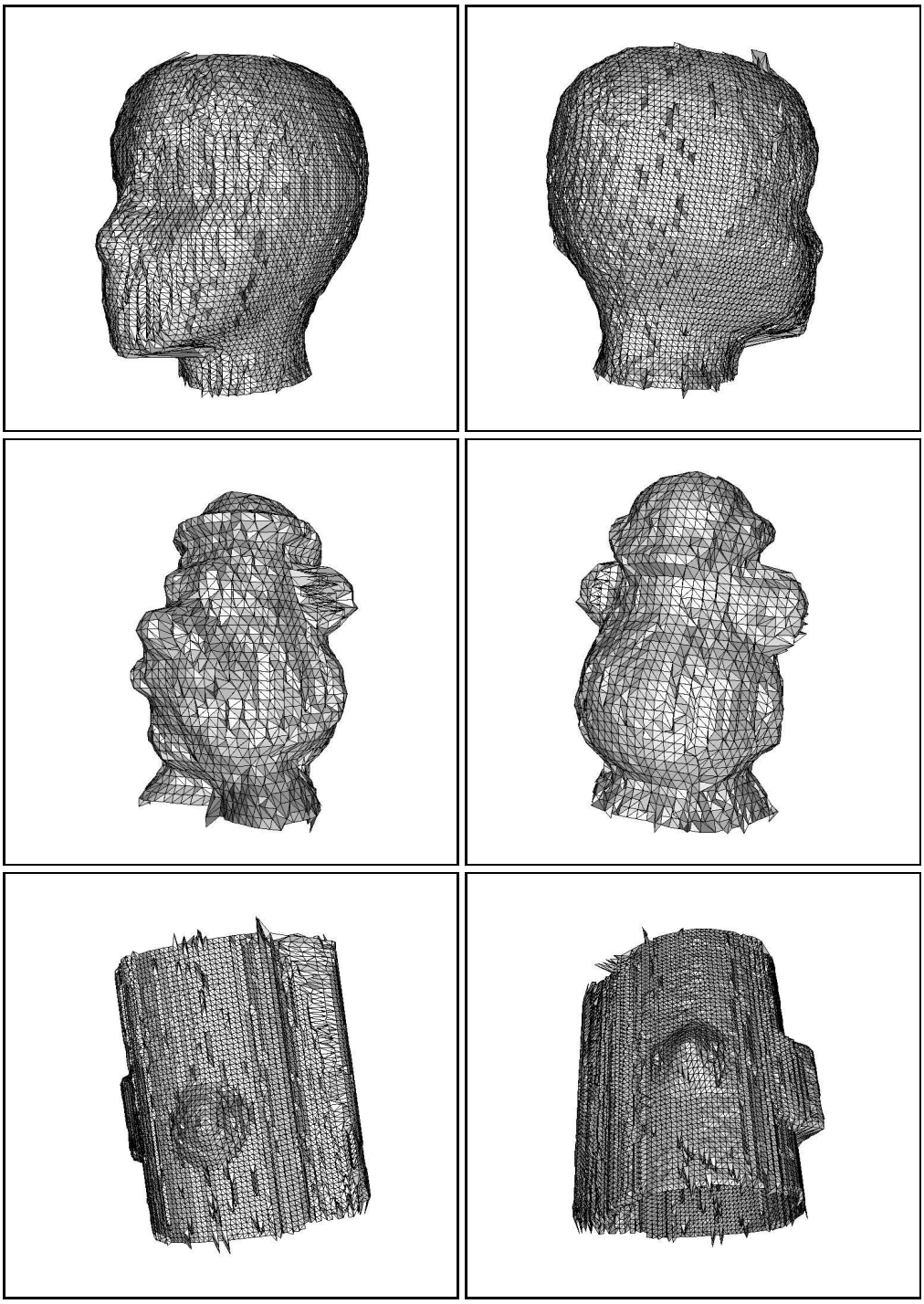


Figure 4-21: Integration results of slice-by-slice algorithm

only be used to construct objects with *star convex* [1] cross sections.

Region-of-Construction algorithm directly stitches non-overlapping regions of range images with raw input data. No approximation on the data points is made for creating 3D models. Mesh simplification based on this raw data is thus more reliable. The registration error is limited to the boundaries of stitches between range images without propagation. The resulting gap (or discontinuity) can be further smoothed by taking average on the neighborhood of the boundaries. As we shall see in the next chapter, texture mapping is more efficient using this integration algorithm because texture information is directly mapped onto each region of construction.

Slice-by-slice integration algorithm uses all available data points to construct a dense 3D model. Because of overlap of range images, sorting algorithm should be carried out for each slice. It is sensitive to registration error and noise, and post-processing should be applied to get a smooth surface.

Accuracy of a reconstructed 3D model is usually difficult to evaluate because of absence of the ground truth. In this dissertation, we analyze the accuracy using a cylinder object with known dimensions. For objects with arbitrary shape, range data acquired from more sophisticated techniques such CMM (Coordinate Measuring Machine) can be used to compare with our results.

[1] A Region R is *star convex* (or *star-shaped*) if there is a point O in R such that, for any point P in R, the line segment OP is contained in R.

Chapter 5

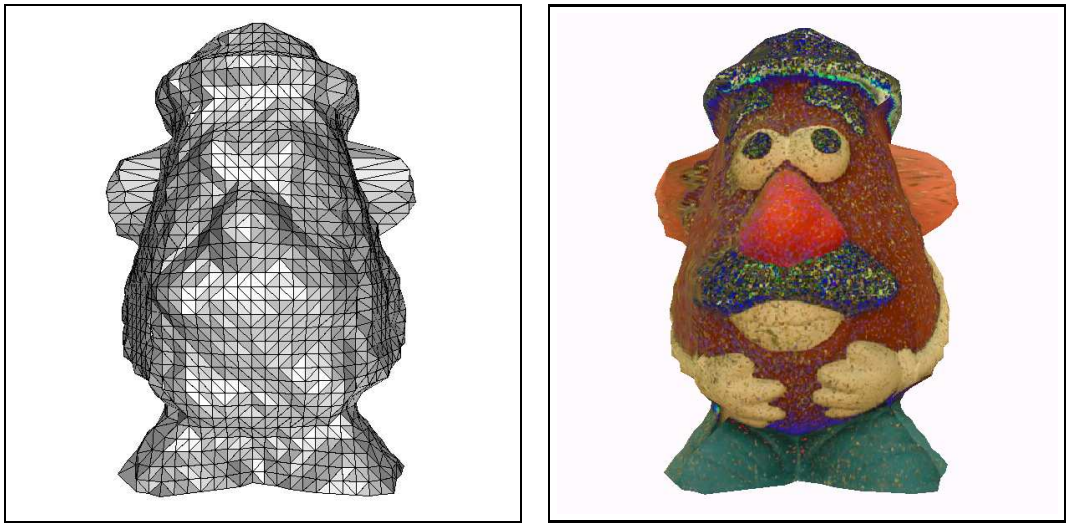
Texture Mapping

5.1 Introduction

Texture mapping was one of the first developments towards making images of three-dimensional objects more interesting and apparently more complex [89]. Since one of the purpose of 3D model reconstruction is to display on a computer monitor, rendering of the reconstructed model represented by only geometric information is not enough. Information about the surface texture has to be added to improve the appearance of the 3D model. As shown in Figure 5-1, by adding the texture information to the wireframe model of a toy object (Figure 5-1(a)), a more realistic 3D model can be obtained (Figure 5-1(b)).

One major advantage of using passive camera systems (instead of say laser range scanners) to acquire 3D model is that the recorded images are used not only for measuring the 3D shape but also for providing the texture information for the objects. Although some 3D range sensors [9] provide intensity or color value associated with each 3D data point, the information is usually not sufficient for realistic texture mapping on top of the triangulation. Some researchers combine the range and image sensing techniques to create the photo-realistic models [77]. One major drawback of their approach is that the information provided by the range and image sensors has to be properly registered. This requires the knowledge of the internal camera parameters (effective focal length, principal point and distortion parameters) and the relative position and orientation between the centers of projection of the camera and the range sensor.

In our approach, the color images used for texture mapping are obtained from shape from focus with several different focus positions (typically 4 different focus steps). The texture image is therefore focused almost everywhere and provides a better textured 3D model compared to those models that use a single image for texture mapping. The focused images constructed for four different viewpoints of a toy object are shown in Figure 5-2. Having modeled the 3D shape of an object by a



(a) Wireframe model of the toy object (b) Textured model of the toy object

Figure 5-1: Photo-realistic 3D model is created by adding the surface texture to the wireframe model.

set of vertices and polygons, the image texture of the object can be modeled in two ways. One is to specify the image texture of each polygon by the projected pattern on one of the focused images. The surface normals of polygons are used to decide the best viewing direction. The other way is to provide the whole (individual) image for each region of construction. The texture mapping is then done for each partial 3D shape using the same image. In the following sections, we will first describe the details for these two mapping methods followed by some experimental results. A short discussion concludes the chapter.

5.2 Texturing on Surface Patches

Texture mapping is an image synthesis technique in which a texture image is mapped onto a surface of a 3D model followed by projection onto the screen for display. Figure 5-3 shows the overall process from the texture domain (u, v) to screen space $(\hat{x}, \hat{y})$. The first transformation, sometimes known as *surface parameterization*, takes the two-dimensional texture pattern and 'glues' it on the object. That is, each point of the texture array is attached to a point of the polygonal surface by a mapping function. The second transformation is called *viewing projection*, which is the standard object to screen space mapping. Surface parameterization is part of

(a) Focused image from viewpoint 1

(b) Focused image from viewpoint 2

(c) Focused image from viewpoint 3

(d) Focused image from viewpoint 4

Figure 5-2: Focused image constructed by SFF and used for texture mapping. Viewpoints 1, 2, 3 and 4 are corresponding to object rotation angles of 0, 90, 180 and 270 degrees.

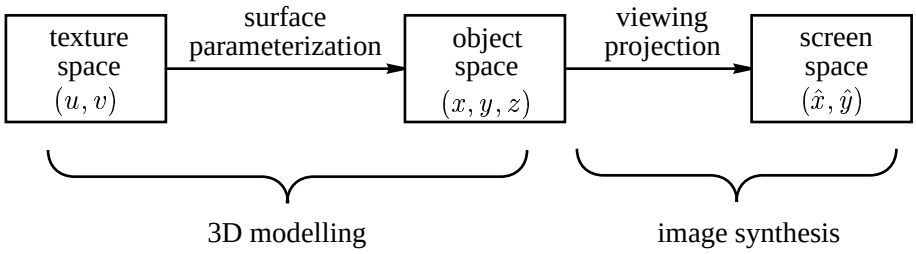


Figure 5-3: An overall texture mapping consists of a surface parameterization followed by the normal geometric transformations.

3D model reconstruction, while viewing projection is part of image synthesis and will not be further discussed here.

Since our 3D models are represented by triangular polygons, surface parameterization of polygonal surfaces is computed for texture mapping. The parameterization of a triangular polygon is performed by specifying the related texture space coordinates (u_i, v_i) of each triangle vertex (x_i, y_i, z_i), where $i = 1, 2, 3$. To obtain the (u, v) coordinate of a given vertex (x, y, z), the perspective projection $(\hat{x}, \hat{y})$ on the image plane of the data point is first calculated by

$$\hat{x} = \frac{x \cdot f}{z} \qquad \text{and} \qquad \hat{y} = \frac{y \cdot f}{z} \tag{5.1}$$

where f is the camera focal length. Then the (u, v) coordinate is given by

$$u = \frac{\hat{x} + w/2}{w} \qquad \text{and} \qquad v = \frac{\hat{y} + h/2}{h} \tag{5.2}$$

where w and h are the image width and height, respectively. Since texture mapping in this method is done on each individual polygon separately, only a rectangular subimage enclosing the projected triangle on the image plane is extracted for the mapping. As illustrated in Figure 5-4, *surface uv* is calculated by

$$u_i = \frac{x_i}{\max\limits_{j=1,2,3} x_j - \min\limits_{j=1,2,3} x_j} \qquad \text{and} \qquad v_i = \frac{y_i}{\max\limits_{j=1,2,3} y_j - \min\limits_{j=1,2,3} y_j} \tag{5.3}$$

where $i = 1, 2, 3$. Once the (u, v) coordinates of the vertices of a polygon are obtained, the texture mapping can be automatically handled by graphics rendering software.

For a given polygon of a complete 3D model, texture mapping can be modeled in two ways. One is to project the polygon onto the focused image and extract an

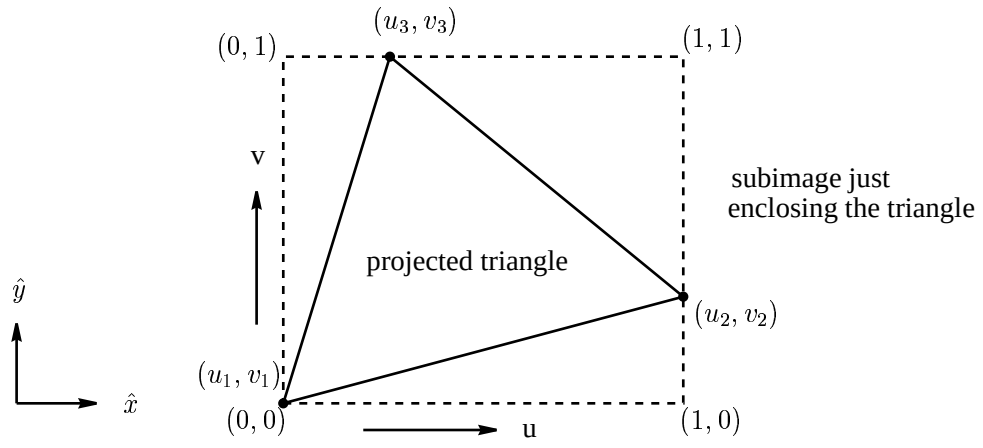
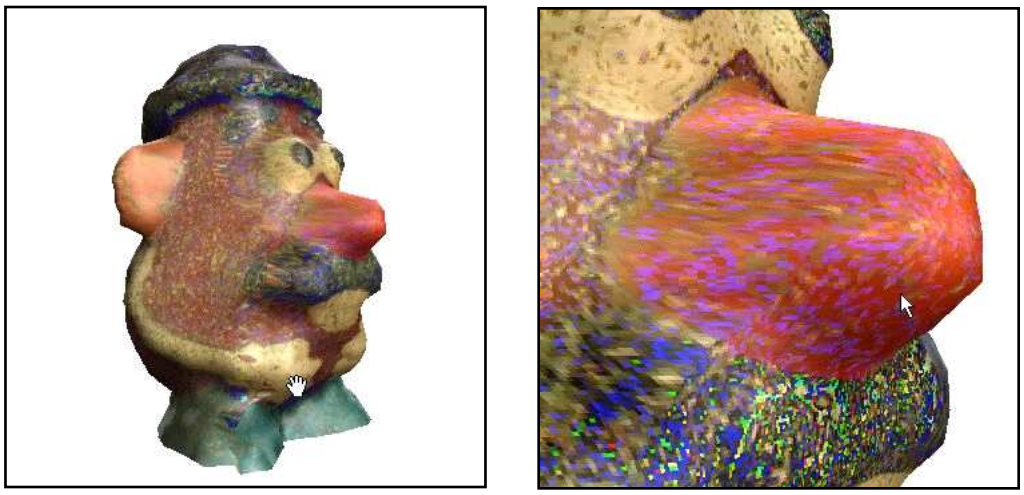


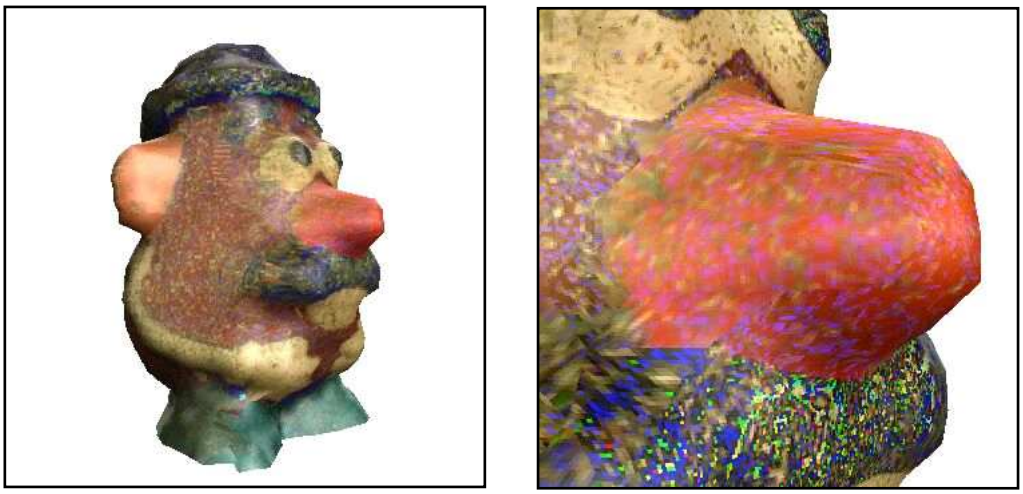
Figure 5-4: Texture mapping on a triangular polygon using the enclosing subimage

enclosing subimage according to its original viewing direction of data acquisition. It is straightforward and requires less computation. However, if the surface normal to the polygon is almost perpendicular to the direction of view, the polygon will project onto a very small region and the image texture will be distorted due to coarse sampling. The other way is to extract the texture from the focused image of "best" acquisition viewpoint. The best viewpoint is chosen with the following criteria: (i) the polygon is visible from the viewpoint, and (ii) the inner product of the surface normal of the polygon and the direction of view is a maximum. Using this mapping method we have less texture distortion on each polygon but more adjacent polygons mapped from different images.

Figure 5-5 shows the results of texture mapping on surface patches. In Figure 5-5(a), the texture is mapped from the color image where the vertices of range image are acquired. Texture mapping in Figure 5-5(b) is done by mapping the texture according to the best viewpoint. As we can see in the magnified image of the object's nose, image distortion in the latter case is reduced compared to the former case because large image size is used for the mapping. Due to the non-ideal illumination conditions and error in 3D shape, there exists texture distortion at the boundaries of polygons which are textured from different images. A local texture filter can be applied to smooth this texture discontinuity [56].



(a) Texture mapping from original viewpoint



(b) Texture mapping from the “best” viewpoint

Figure 5-5: Texture mapping on surface patches

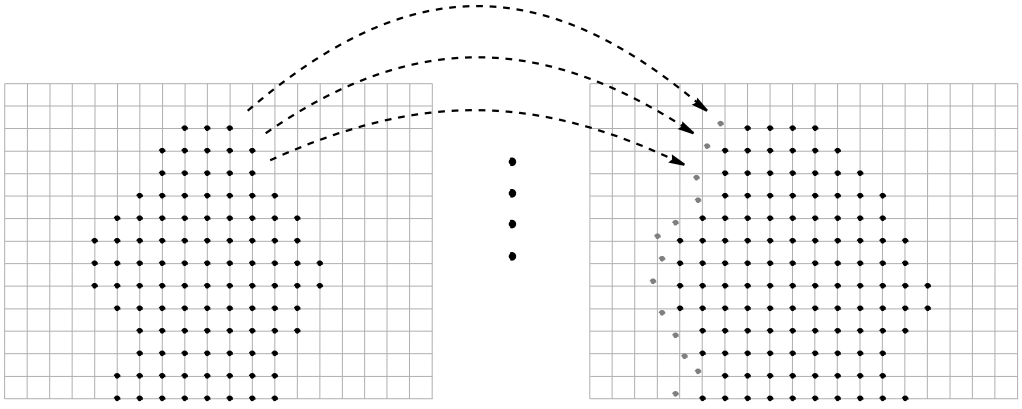


Figure 5-6: Texture mapping on a boundary strip. Boundary points of Region-of-Construction on image 0 are projected onto image 1 to calculate surface uv for texture map.

5.3 Texturing on Regions-of-Construction

In contrast to the previous approach (texturing on each surface patch), texture information in this method is mapped on the object's surface according to the regions of construction of its geometric model. As defined in Chapter 4, Region-of-Construction of a viewpoint is part of the corresponding range image of the object, and therefore part of the corresponding focused image. Since the complete 3D model is created by stitching all Regions-of-Construction, the textured 3D model can be obtained by combining all textured Regions-of-Construction of the object. That is, non-overlapped and texture mapped partial 3D models are created first and then stitched together.

To obtain a texture mapped Region-of-Construction, the texture space coordinate (u, v) of each vertex is computed first. Since a one-to-one correspondence can be easily found between range and focused images, the calculation of surface uv is straightforward for index (i, j) of a range image and given by

$$u = \frac{j}{w-1} \qquad \text{and} \qquad v = 1 - \frac{i}{h-1} \tag{5.4}$$

where w and h are the width and height of range images, respectively. Although surface uv is calculated for each data point of the range images, only those points inside the regions of construction are used to extract the texture map.

Figure 5-7: Extracted subimages using bounding boxes which enclose Regions-of-Construction.

To stitch textured Regions-of-Construction, the texture map on the boundary strip connecting two consecutive Regions-of-Construction has to be obtained separately. This is because the boundary strips are not covered by any Regions-of-Construction. To obtain the texture map, each boundary strip is assigned to one focused image. The vertices of the boundary strip on the adjacent Region-of-Construction (range image) are projected onto the focused image to calculate their surface uv. The surface uv is then used to extract the texture map on the focused image. As illustrated in Figure 5-6, the 3D points on the right boundary of range image 0 are projected onto range image 1 and its corresponding focused image according to the viewing geometry. The surface uv is calculated on each projected point and used to extract the texture map from focused image 1 for the boundary strip between Region-of-Construction of range image 0 and 1.

In this approach, the number of boundaries on the texture map of a complete 3D model depends on the number of focused images used. In our experiment, 4 focused images from the corresponding viewpoint are used and 4 boundaries appear on the textured 3D model. Because the illumination conditions of different images are not exactly the same, the texture discontinuity across the boundary is magnified. This can be eliminated by applying a local texture filter as previously mentioned.

Since the whole areas of images including the background regions are used for texture mapping in this method, the size of 3D model usually becomes very large. The model size is reduced in two ways. First, a bounding box is applied on the image to extract a rectangular region just enclosing the foreground region

(or Region-of-Construction) and discard the background region. The reduction of image size depends on the percentage of the foreground region in a scene. For the toy object, the file size reduces up to 80%. Figure 5-7 shows the images used for texture mapping extracted from Figure 5-2 with bounding boxes enclosing the Regions-of-Construction. Second, we down-sample the original 1280×960 image to size of 640×480. This reduces another 75% and the quality of the resulting 3D model is usually not degraded from our visual perception.

5.4 Shading

Realistic display of an object is obtained by generating perspective projections and applying natural lighting effects to the visible surfaces. An *illumination model*, or *shading model* is used to calculate the intensity of light that we should see at a given point on the surface of an object. A *surface-rendering algorithm* uses the intensity calculations from a shading model to determine the light intensity for all projected pixel positions for the various surfaces in a scene [33].

The shading information of an object is provided by the normal vector of each data point. The vertex normal is obtained by averaging the surface normals of all polygons sharing that vertex. Instead of using data points of the complete 3D model, we calculate the surface normals from the range images to keep the original shading information. The resulting vertex normals on the boundaries of two consecutive partial shapes are different from the ones obtained from the complete 3D model and can be used to check the smoothness of surface integration.

5.5 Experimental Results

In this section, we show some experimental results of textured 3D model and the overall acquisition time for creating a complete 3D model. The results of five different objects– a foam head, a toy “Mr. Potato Head”, a cylinder, a bottle and a detergent box are shown in Figure 5-8 – 5-12 with several viewing directions. As we mentioned in Chapter 2, acquiring all of the images required for 3D model reconstruction takes about 15 minutes. The total execution time and processing time for each stage are shown in Table 5-1 with several object (in seconds, on a Pentium II 450 MHz PC).

Table 5-1: Overall execution times for complete 3D model reconstruction.

Object	Data Acquisition	Integration	Texturing	Total
Head	5.68 sec.	8.05 sec.	21.10 sec.	34.83 sec.
Toy	5.52 sec.	5.18 sec.	18.39 sec.	28.99 sec.
Cylinder	5.59 sec.	9.37 sec.	35.06 sec.	49.92 sec.
Detergent	3.03 sec.	9.87 sec.	37.02 sec.	49.92 sec.
Bottle	5.48 sec.	5.51 sec.	19.63 sec.	30.62 sec.

5.6 Discussion

In this chapter, we described two methods to map the color images constructed by SFF onto the object's surface. Texture mapping on each surface patch of the 3D model extracts only the useful information (subimage blocks) from the original image. Texturing on the surface patch with the best viewpoint image gives less distortion on the image texture. The major drawback of this approach is that the real-time rendering is slower. Texture mapping on each region of construction is efficient but performs poorly when surface normals are away from the corresponding viewpoint. The size of a 3D model is reduced by extracting the subimage on a bounding box which just encloses the region of construction.

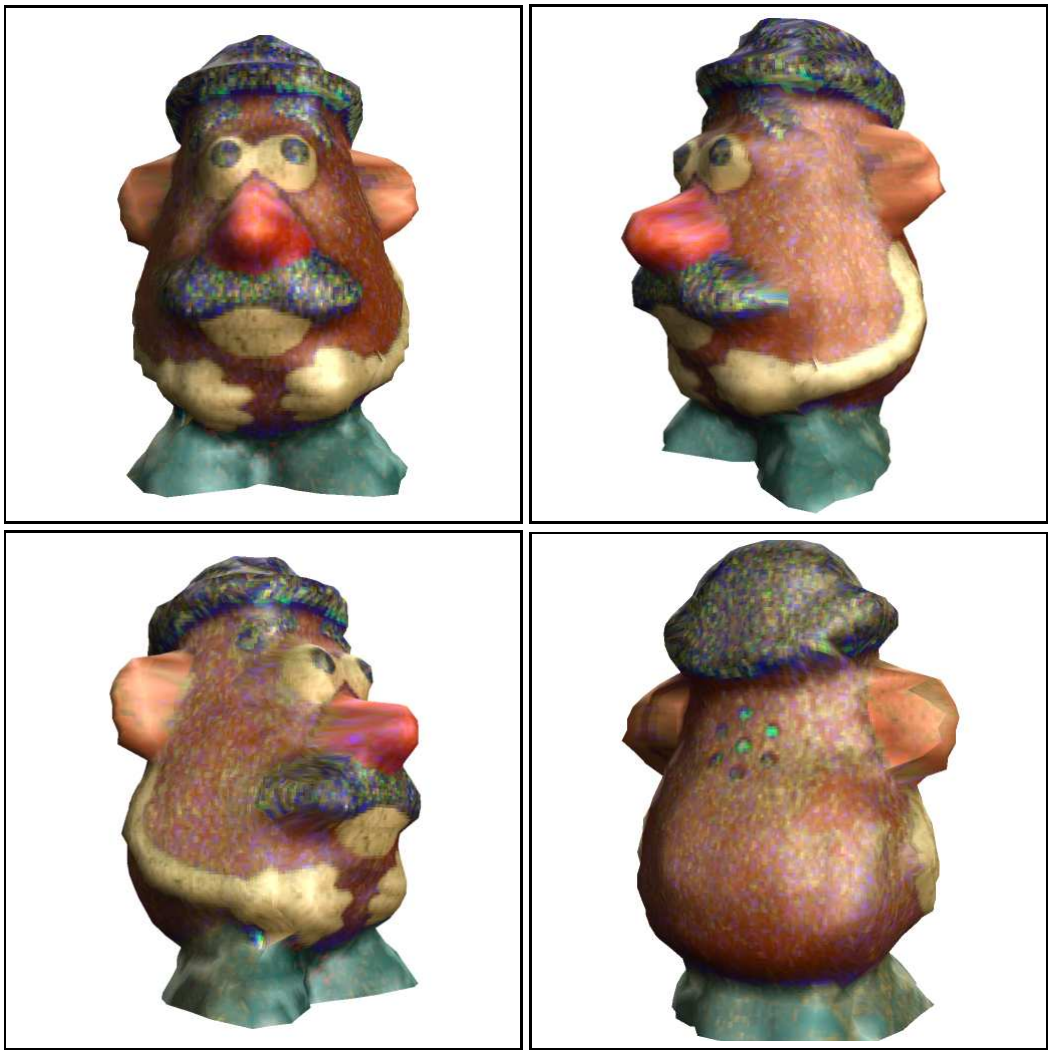


Figure 5-8: Complete 3D model with texture mapping– toy

Figure 5-9: Complete 3D model with texture mapping– head

Figure 5-10: Complete 3D model with texture mapping– cylinder

Figure 5-11: Complete 3D model with texture mapping– bottle

Figure 5-12: Complete 3D model with texture mapping– detergent container

Chapter 6

Conclusion

6.1 Summary

In this dissertation, we have presented a complete system for automatic 3D model reconstruction. This digital vision system is comprehensive in that it includes all stages– data acquisition, registration, surface integration, and texture mapping, to create a photo-realistic 3D model. The complete 3D model is constructed by merging multiple range images with texture information from different viewpoints. The system can be implemented with low-cost equipments and constructs complete 3D models in under 20 minutes.

In data acquisition stage, range and focused intensity images are recovered by rotational stereo and shape from focus. For each viewpoint, two sequences of images with different focus positions are recorded before and after a stereo rotation angle. A focused image pair and the corresponding rough depth maps are constructed from the image sequences by shape from focus. Rotational stereo model is used to compute the epipolar geometry of the image pair instead of image rectification. Stereo matching is done along the epipolar lines with the restrictions provided by rough depth map to obtain an accurate depth map. Partial 3D shapes from different viewpoints are obtained by rotating the object with known rotation angles.

The partial 3D models are then registered to a common coordinate frame with necessary alignment for surface integration. This includes rotation axis calibration and registration. The calibration estimates the translation and unit vectors of the rotation axis. The estimated rotation axis is then refined using the overlapping regions of range images. We describe two registration algorithms, one is based on global resampling of range data set, and the other one directly computes and updates the rotation axis iteratively.

A surface integration algorithm, Region-of-Construction algorithm, is developed to remove the redundant data from range images and stitch the resulting non-

overlapping regions to create a seamless 3D model. It takes advantage of known topology of range images and thus requires less computation compared to general integration algorithms. This algorithm also has the ability to construct complex objects with a hole by adding a hole-creating process. We also develop two other integration algorithms. One is based on (r, θ, y) representation by resampling the data points. Registered range images are resampled by linear interpolation in both y and θ directions on a cylindrical coordinate system. The other one is slice-by-slice algorithm, which first turns range images to slices and combines the slices to a complete 3D model.

The final photo-realistic 3D model is created by mapping the texture information onto the complete surface model. For fast rendering, the mapping is done on each region of construction and combining them together. We also implement a method to map the texture on each surface patch of the complete 3D model.

6.2 Future Work

The future work of this research can be extended on all four stages of 3D model reconstruction as well as other improvements. Possible related research is discussed below.

Data Acquisition

Currently the 3D shape recovery is based on a pinhole camera model. By taking other camera parameters such as lens distortion into account, the resulting 3D model should be more precise. Furthermore, 3D shape recovery from depth map can be adjusted by considering all viewpoints simultaneously instead of simple inverse perspective projection model.

Registration

Registration in this research is to find the translation and unit vectors of the rotation axis. To create a complete 3D model including the top and bottom, an object has to be rotated along a horizontal direction either manually or automatically. In this case, a more sophisticated registration method should be implemented to obtain those transformation matrices. For higher resolution range images, high accuracy of registration becomes very important. More input range images with larger overlapping regions could be used to obtain more accurate registration results.

Surface Integration

Surface integration is a research topic in computational geometry. Most researchers focus on turning unorganized points to a mesh model. Since our input range image contain the information about the spatial connectivity, we should investigate how to use this information to efficiently construct an accurate model. We can also investigate how to convert the range images to point cloud or slice data, and apply the existing integration algorithms to create a 3D model. This conversion usually requires very accurate registration of the range images.

Texture Mapping

Texture mapping in this research assigns to each surface its own texture map with its acquisition viewpoint. In case the surface normal is almost perpendicular to the viewpoint, the image texture will be displayed with large distortion for the viewing directions close to the surface normal. Future research can focus on how to render the textured 3D model consistently for different viewing directions. Another research topic is *shading inversion*– given the surface normal (assuming uniform illumination from all directions), how can we correct (or invert) for the brightness change caused by shading effects.

Surface Simplification

With more and more high precision range data acquisition techniques available, the reconstructed 3D model contains more vertex information than required by applications. These models are usually large, and difficult for real-time rendering. Thus, surface simplification is important in the further research [74, 40]. It should be able to reduce both the data size and the error on the data point (caused by noise or incorrect registration, etc.) at the same time.

View Planning

The model reconstruction process we have described in this dissertation performs no planning of sensor viewpoints. Equal rotation angles are used to acquire partial 3D models between views. How to select the minimum number of views to best cover an object surface is an important research topic [66, 59]. This can speed up the data acquisition process, especially when a large number of viewpoints has to be taken to completely cover an object.

Appendix A

Quaternion and Rotation

A common problem in robotics and computer vision is solving for rigid-body motions consisting of a rotation and translation in three-dimensional space. A number of techniques have been developed to deal with this problem and *quaternion* have been found to be the most efficient representation. In this Appendix, we will define the quaternion, state some of its essential properties and use it to represent and algebraically manipulate rotations.

A.1 Quaternion

A quaternion can be thought of as a vector with four components. Alternately, it can be thought of as a higher-order complex number with one real part and three imaginary parts, and can be written as

$$\hat{q} = s + ia + jb + kc \tag{A.1}$$

where the coefficients a, b, and c in the imaginary terms are real numbers, and parameter s is a real number called the *scalar part*. Parameters i, j, k are defined with the properties

$$i^2 = j^2 = k^2 = -1, ij = -ji = k \tag{A.2}$$

From the properties, it follows that

$$jk = -kj = i, ki = -ik = j \tag{A.3}$$

Scalar multiplication is defined in analogy with the corresponding operations for vectors and complex numbers. That is, each of the four components of the quaternion is multiplied by the scalar value. For a scalar α, the scalar multiplication is given by

$$\alpha\hat{q} = \alpha s + i\alpha a + j\alpha b + k\alpha c \tag{A.4}$$

Similarly, quaternion addition is defined as

$$\hat{q}_1 + \hat{q}_2 = (s_1 + s_2) + i(a_1 + a_2) + j(b_1 + b_2) + k(c_1 + c_2) \tag{A.5}$$

Multiplication of two quaternions is carried out using the operations in Eqs. A.2 and A.3. It is associative but not commutative, as can easily be checked.

An ordered-pair notation for a quaternion is also formed in analogy with complex number notation:

$$\hat{q} = (s, \mathbf{v}) \tag{A.6}$$

where **v** is the vector (a, b, c). In this notation, quaternion addition is expressed as

$$\hat{q}_1 + \hat{q}_2 = (s_1 + s_2, \mathbf{v_1} + \mathbf{v_2}) \tag{A.7}$$

Quaternion multiplication can then be expressed in terms of vector dot and cross products as

$$\hat{q}_1\hat{q}_2 = (s_1 s_2 - \mathbf{v_1} \cdot \mathbf{v_2}, s_1\mathbf{v_2} + s_2\mathbf{v_1} + \mathbf{v_1} \times \mathbf{v_2}) \tag{A.8}$$

The norm of a quaternion is given by

$$|\hat{q}| = (\hat{q}\hat{q})^{-\frac{1}{2}} = (s^2 + \mathbf{v} \cdot \mathbf{v})^{-\frac{1}{2}} \tag{A.9}$$

and the inverse of a quaternion is

$$\hat{q}^{-1} = \frac{1}{|\hat{q}|^2}(s, -\mathbf{v}) \tag{A.10}$$

so that

$$\hat{q}\hat{q}^{-1} = \hat{q}^{-1}\hat{q} = (1, 0). \tag{A.11}$$

The conjugate $\hat{q}^*$ of a quaternion $\hat{q}$ is defined by negating the vector component:

$$\hat{q}^* = (s, -\mathbf{v}) \tag{A.12}$$

It can be easily checked that the conjugate of a unit quaternion with $|\hat{q}| = 1$ is the inverse of the unit quaternion. As a result, for a unit quaternion,

$$\hat{q}^*\hat{q} = \hat{q}\hat{q}^* = 1. \tag{A.13}$$

A.2 Rotation using Quaternion

Rotations about a specific axis in three-dimensional space can be conveniently represented by unit quaternions (Figure A-1). The unit quaternion $\hat{q}$ corresponding to a rotation angle θ about the unit vector $\mathbf{u}$ is given by $(s, \mathbf{v})$ where

$$s = \cos\frac{\theta}{2}, \quad \mathbf{v} = \mathbf{u}\sin\frac{\theta}{2} \tag{A.14}$$

Any point position $\mathbf{P}$ to be rotated by this quaternion can be represented in quaternion notation as

$$\mathbf{P} = (0, \mathbf{p}) \tag{A.15}$$

with the coordinates of the point as the vector part $\mathbf{p} = (x, y, z)$. The rotation of the point is then carried out with the quaternion operation

$$\mathbf{P}' = \hat{q}\mathbf{P}\hat{q}^{-1}. \tag{A.16}$$

This transformation produces the new quaternion with scalar part equal to 0:

$$\mathbf{P}' = (0, \mathbf{p}') \tag{A.17}$$

and the vector part is calculated with dot and cross products as

$$\mathbf{p}' = s^2\mathbf{p} + \mathbf{v}(\mathbf{p}\cdot\mathbf{v}) + 2s(\mathbf{v}\times\mathbf{p}) + \mathbf{v}\times(\mathbf{v}\times\mathbf{p}) \tag{A.18}$$

Parameters s and $\mathbf{v}$ have the rotation values given in Eqs. A.14.

Transformation A.16 is equivalent to rotation about an axis that passes through the coordinate origin. This is the same as the sequence of rotation transformations that aligns the rotation axis with the z axis, rotates about z, and then returns the rotation axis to its original position.

Using the definition for quaternion multiplication, and designating the components of the vector part of $\hat{q}$ as $\mathbf{v} = (a, b, c)$, we can evaluate the terms in Eq. A.18 to obtain the elements for the composite rotation matrix in a 3 by 3 form as

$$\mathbf{R}(\theta) = \begin{bmatrix} 1-2b^2-2c^2 & 2ab-2sc & 2ac+2sb \\ 2ab+2sc & 1-2a^2-2c^2 & 2bc-2sa \\ 2ac-2sb & 2bc+2sa & 1-2a^2-2b^2 \end{bmatrix} \tag{A.19}$$

where a, b, c and s are given in Eqs. A.14. Notice that when the rotation angle is zero, then the rotation matrix in A.19 becomes a 3×3 identity matrix, i.e., $\mathbf{R}(0) = \mathbf{I}$.

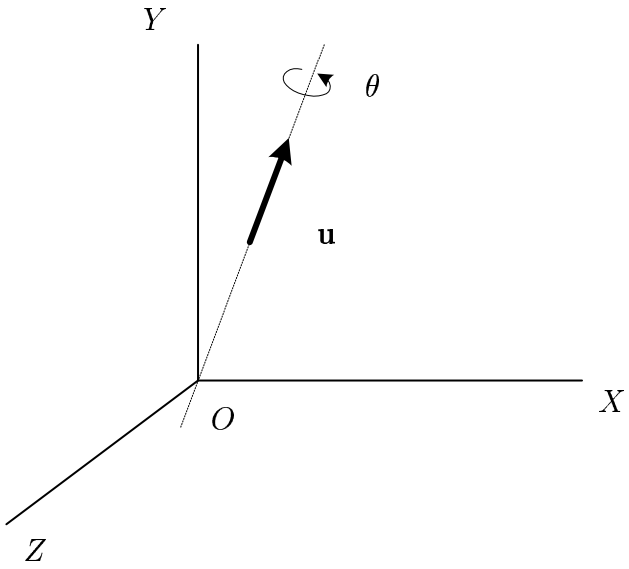


Figure A-1: Unit quaternion parameters θ and **u** for rotation about a specified axis

A.3 Concluding Remark

The unit quaternion representation can often simplify problems in which we have to deal with the attitude of an object in space. Quaternions are also useful in a number of other computer graphics procedures, including three-dimensional rotation calculations. They require less storage space than 4 by 4 matrices, and it is simpler to write quaternion procedures for transformation sequences. This is particularly important in animations that require complicated motion sequences and motion interpolations between two given positions of an object.

Bibliography

[1] E. Angel, *Interactive Computer Graphics: A Top-Down Approach with OpenGL*, Addison Wesley, 1999.

[2] N. Ayache and C. Hansen, "Rectification of Images for Binocular and Trinocular Stereovision," *Proceedings of International Conference on Pattern Recognition*, pp. 11-16, 1988.

[3] N. Amenta, M. Bern and M. Kamvysselis, "A New Voronoi-Based Surface Reconstruction Algorithm," *Proceedings of SIGGRAPH '98*, pp. 415-422, July 1998.

[4] S. T. Barnard and M. A. Fischler, "Computational Stereo," *ACM Computing Surveys*, 14(4), pp. 553-572, December 1982.

[5] C. Bajaj, F. Bernardini, and G. Xu, "Automatic Reconstruction of Surfaces and Scalar Fields from 3D Scans," *Proceedings of SIGGRAPH '95*, pp. 109-118, August 1995.

[6] C. J. Bajaj, E. J. Coyle, and K. Lin, "Arbitrary Topology Shape Reconstruction from Planar Cross Sections," *Graphical Models and Imaging Processing*, 58(6), pp. 524-543, 1996.

[7] R. Bergevin, M. Soucy, H. Gagnon, and D. Laurendeau, "Towards a General Multi-View Registration Technique," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 5, May 1996.

[8] P. J. Besl and R. C. Jain, "Three-Dimensional Object Recognition," *ACM Computing Surveys*, 17(1), pp. 75-145, March 1985.

[9] P. Besl, "Active Optical Range Imaging Sensors," *Advances in Machine Vision*, pp. 1-63, Springer-Verlag, 1989.

[10] P. J. Besl and N. McKay, "A Method of Registration of 3D Shape," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 2, 1992.

[11] G. Blais and M. D. Levine, "Registering Multiview Range Data to Create 3D Computer Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 8, August 1995.

[12] J-D. Boissonnat, "Shape Reconstruction from Planar Cross Sections," *Computer Vision, Graphics, and Image Processing*, 44(1) pp. 1-29, 1988.

[13] J-D. Boissannat and B. Geiger, "Three Dimensional Reconstruction of Complex Shapes Based on the Delaunay Triangulation," Report 1697, INRIA Sophia-Antipolis, Valbonne, France, 1992.

[14] L. G. Brown, "A Survey of Image Registration Techniques," *ACM Computing Surveys*, 24(4), pp. 325-376, December 1992.

[15] G. Champleboux, S. Lavallee, R. Szeliski, and L. Brunie, "From Accurate Range Imaging Sensor Calibration to Accurate Model-Based 3-D Object Localization," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 153-158, June 1992.

[16] Y. Chen and G. Medioni, "Object Modeling by Registration of Multiple Range Images," *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 2724-2729, April 1991.

[17] Y. Chen and G. Medioni, "Object Modeling by Registration of Multiple Range Images," *Image and Vision Computing*, Vol. 10, No. 3, pp. 145-155, 1992.

[18] C. S Chen, Y. P. Hung, and J. B. Chung, "A Fast Automatic Method for Registration of Partially-Overlapping Range Images," *Proceedings of IEEE International Conference on Computer Vision*, pp. 242-248, January 1998.

[19] S. D. Cochran and G. Medioni, "3-D Surface Description from Binocular Stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 10, 1992.

[20] G. Cross and A. Zisserman, "Surface Reconstruction from Multiple Views Using Apparent Contours and Surface Texture," *Confluence of Computer Vision and Computer Graphics*, pp. 25-47, Kluwer, 2000.

[21] B. Curless and M. Levoy, "A Volumetric Method for Building Complex Models from Range Images," *Proceedings of SIGGRAPH '96*, pp. 303-312, August 1996.

[22] U. R. Dhond, J. K. Aggarwal, "Structure from Stereo - A Review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 6, pp. 1489-1510, December 1989.

[23] C. Dorai, G. Wang, A. K. Jain, and C. Mercer, "From Images to Models: Automatic 3D Object Model Construction from Multiple Views," *Proceedings of International Conference on Pattern Recognition*, pp. 770-774, 1996.

[24] C. Dorai, J. Weng and A. K. Jain, "Optimal Registration of Object Views Using Range Data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 10, pp. 1131-1138, March 1997.

[25] C. Dorai, G. Wang, A. K. Jain, and C. Mercer, "Registration and Integration of Multiple Object Views of 3D Model Construction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 1, January 1998.

[26] D. W. Eggert, A. W. Fitzgibbon, and R. B. Fisher, "Simultaneous Registration of Multiple Range Views For Use In Reverse Engineering," *Proceedings of International Conference on Pattern Recognition*, pp. 243-247, 1996.

[27] P. Eisert, E. Steinbach, and B. Girod, "Automatic Reconstruction of Stationary 3-D Objects from Multiple Uncalibrated Cameras Views," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, No. 2, March 2000.

[28] O. Faugeras, *Three-Dimensional Computer Vision*, The MIT Press, 1993.

[29] R. Fisher, A. Fitzgibbon, A. Gionis, M. Wright, and D. Eggert, "A Hand-Held Optical Surface Scanner for Environment Modeling and Virtual Reality," Tech. Rep. DAI, No. 778, University of Edinburgh, December 1995.

[30] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice*, Addison Wesley, 1992.

[31] H. Gagnon, M. Soucy, R. Bergevin, and D. Laurendeau, "Registration of Multiple Range Views for Automatic 3-D Model Building," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 581-586, June 1994.

[32] G. Häusler and S. Karbacher, "Reconstruction of Smoothed Polyhedral Surfaces from Multiple Range Images," *3D Image Analysis and Synthesis '97*, pp. 191-198, 1997

[33] D. Hearn and M. P. Baker, *Computer Graphics*, Prentice Hall, 1997.

[34] R.M. Haralick and L.G. Shapiro, *Computer and Robot Vision*, Addison Wesley, 1992.

[35] P. S. Heckbert, “Survey of Texture Mapping,” *IEEE Computer Graphics and Applications*, 6(11):56-67, November 1986.

[36] P. S. Heckbert, “Fundamentals of Texture Mapping and Image Warping,” Dept. of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720, 1989.

[37] A. Hilton, J. Stoddart, J. Illingworth, and T. Windeatt, “Reliable Surface Reconstruction from Multiple Range Images,” *Proceedings of European Conference on Computer Vision*, Vol. 1, pp. 117-126, April 1996 .

[38] A. Hilton, A. J. Stoddart, J. Illingworth, and T. Windeatt, “Marching Triangles: Range Image Fusion for Complex Object Modelling,” *IEEE International Conference on Image Processing*, Vol. 2, pp. 381-384, 1996.

[39] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, “Surface Reconstruction from Unorganized Points”, *Proceedings of SIGGRAPH ’92*, pp. 71-78, July 1992.

[40] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, “Mesh Optimization,” *Proceedings of SIGGRAPH ’93*, pp. 19-26, August 1993.

[41] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schwitzer, and W. Stuetzle, “Piecewise Smooth Surface Reconstruction,” *Proceedings of SIGGRAPH ’94*, pp. 295-302, 1994.

[42] B. K. P. Horn, *Robot Vision*, McGraw-Hill Book Company, 1986.

[43] T. Kanade and M. Okutomi, “A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 9, pp. 920-932, September 1994.

[44] S. B. Kang, J. A. Webb, C.L. Zitnick, and T. Kanade, ”A Multibaseline Stereo System with Active Illumination and Real-time Image Acquisition,” *Proceedings of International Conference on Computer Vision*, pp. 88-93, Cambridge, MA, June 1995.

[45] V. Koivunen and J.-M. Vezien, “Multiple Representation Approach to Geometric Model Construction from Range Data,” *Proceedings of the 2nd CAD-based Vision Workshop*, pp. 132-139, February 1994.

[46] E. Krotkov, “Focusing,” *International Journal of Computer Vision*, 1, pp. 223-237, 1987.

[47] B. U. Lee, C. M. Kim, and R. H. Park, “An Orientation Reliability Matrix for the Iterative Closest Point Algorithm,” *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 10, pp.1205-1208, October 2000.

[48] W. E. Lorenen and H. E. Cline, “Marching Cubes: A High Resolution 3D Surface Reconstruction Algorithm,” *Proceedings of SIGGRAPH ’87*, pp. 163-169, July 1987.

[49] H. Y. Lin and M. Subbarao, “Three-dimensional Model Acquisition using Rotational Stereo and Image Focus Analysis,” *Machine Vision and Three-Dimensional Imaging Systems for Inspection and Metrology, Proceedings of SPIE*, Vol. 4189, pp. 201-210, November 2000.

[50] H. Y. Lin and M. Subbarao, “A Vision System for Fast 3D Model Reconstruction,” *Proceedings of Computer Vision and Pattern Recognition*, pp. 663-668, Kauai, HI, December 2001.

[51] H. Y. Lin, M. Subbarao, and S. Y. Park, “Complete 3-D Model Reconstruction from Multiple Views,” *Machine Vision and Three-Dimensional Imaging Systems for Inspection and Metrology II, Proceedings of SPIE*, Vol. 4567, October 2001.

[52] H. Y. Lin and M. Subbarao, “Multiple Base-angle Rotational Stereo for Accurate 3D Model Reconstruction,” *submitted to International Conference on Image Processing*, Rochester, NY, September 2002.

[53] J. Luck, C. Little, and W. Hoff, “Registration of Range Data Using a Hybrid Simulated Annealing and Iterative Closest Point Algorithm,” *Proceedings of IEEE International Conference on Robotics and Automation*, San Francisco, CA, April, 2000.

[54] D. Meyers and S. Skinner, “Surfaces from Contours,” *ACM Transactions on Graphics*, Vol. 11, No. 3, pp. 228-258, July 1992.

[55] W. Niem, H. Broszio, “Mapping Texture from Multiple Camera Views onto 3D-Object Models for Computer Animation,” *Proceedings of the International Workshop on Stereoscopic and Three Dimensional Imaging*, Santorini, Greece, September 1995.

[56] W. Niem, “Automatic reconstruction of 3d objects using a mobile camera,” *Image and Vision Computing*, Vol. 17, No. 2, pp. 125-134, February 1999.

[57] M. Okutomi, T. Kanade, “A Multiple-Baseline Stereo,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 4, pp. 353-363, April 1993.

[58] R. Pito, “Mesh Integration Based on Co-Measurements,” *Proceedings of IEEE International Conference on Image Processing*, pp. 397-400, 1996.

[59] R. Pito, “A Sensor Based Solution to the Next Best View Problem,” *Proceedings of International Conference on Pattern Recognition*, pp. 941-945, Vienna, Austria, August 1996.

[60] R. Pito, “A Solution to the Next Best View Problem for Automated Surface Acquisition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 10, pp. 1016-1030, October 1999.

[61] M. Potmesil, “Generating Models of Solid Objects by Matching 3D Surface Segments,” *Proceedings of the 8th IJCAI*, pp. 1089-1093, August, 1983.

[62] K. Pulli, T. Duchamp, H. Hoppe, J. McDonald, L. Shapiro, and W. Stuetzle, “Robust Meshes from Multiple Range Maps”, *Proceedings of International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pp. 205-211, May 1997.

[63] K. Pulli, M. Cohen, T. Duchamp, H. Hoppe, L. Shapiro, and W. Stuetzle, “View-based Rendering: Visualizing Real Objects from Scanned Range and Color Data,” *Proceedings of 8th Eurographics Workshop on Rendering*, St. Etienne, France, June 1997.

[64] K. Pulli, H. Abi-Rached, T. Duchamp, L. G. Shapiro, and W. Stuetzle, “Acquisition and Visualization of Colored 3D Objects,” *Proceedings of International Conference on Pattern Recognition*, pp. 11-15, Brisbane, Australia, August 1998.

[65] K. Pulli, “Multiview Registration for Large Data Sets,” *Proceedings of Second International Conference on 3D Digita l Imaging and Modeling*, pp. 160-168, Ottawa, Canada, October 1999.

[66] M. Reed and P. K. Allen, “3D Modeling from Range Imagery: An Incremental Method with a Planning Component,” *Image and Vision Computing*, Vol. 17, No. 1, pp. 99-111, February 1999.

[67] B. Ross, “A Practical Stereo Vision System,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 148-153, New York, June 1993.

[68] G. Roth and E. Wibowoo, “A Fast Algorithm for Making Mesh Models from Multi-view Range Data,” *Proceedings of the DND/CSA Robotics and Knowledge Based Systems Workshop*, pp. 339-356, St. Hubert, Quebec, October 1995.

[69] G. Roth and E. Wibowoo, “An Efficient Volumetric Method for Building Closed Triangular Meshes from 3-D Image and Point Data,” *Graphics Interface 97*, pp. 173-180, Kelowna, B.C., Canada, May 1997.

[70] M. Rutishauser, M. Stricker, and M. Trobina, “Merging range images of arbitrarily shaped objects”, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 573-580, June 1994.

[71] Y. Sato, M. D. Wheeler, and K. Ikeuchi, “Object Shape and Refectance Modeling from Observation,” *Proceedings of SIGGRAPH ’97*, pp. 379-387, August 1997.

[72] I. Söderkvist, “Introductory Overview of Surface Reconstruction Methods,” *Lulea University of Technology, Dept. of Mathematics, Reseach Report 10*, 1999.

[73] M. Soucy and D. Laurendeau, “Generating non-redundant surface representations of 3D objects using multiple range views,” *Proceedings of International Conference on Pattern Recognition*, pp. 198-200, Atlantic City, New Jersey, June 1990.

[74] M. Soucy and D. Laurendeau, “Multi-Resolution Surface Modeling from Multiple Range Views,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 348-353, June 1992.

[75] M. Soucy and D. Laurendeau, “A General Surface Approach to the Integration of a Set of Range Views,” *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 4, pp. 344-358, April 1995.

[76] M. Soucy and D. Laurendeau, ”A Dynamic Integration Algorithm to Model Surfaces from Muiltiple Range Views,” *Machine Vision and Applications*, Vol. 8, No.1, pp. 53-62, 1995.

[77] I. Stamos and P. K. Allen, "3-D Model Construction Using Range and Image Data," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, SC, June 2000.

[78] A. J. Stoddart, A. Hilton, "Registration of Multiple Point Sets," *Proceedings of International Conference on Pattern Recognition*, pp. 40-44, Vienna, Austria, 1996.

[79] H. Shum, K. Ikeuchi, and R. Reddy, "Principal Component Analysis with Missing Data and Its Application to Object Modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 9, pp. 854-867, September, 1995.

[80] H. Shum, M. Hebert, K. Ikeuchi, and R. Reddy, "An Integral Approach to Free-Form Object Modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 19, No. 12, December 1997.

[81] M. Subbarao, T. S. Choi, and A. Nikzad, "Focusing Techniques", *Journal of Optical Engineering*, pp. 2824-2836, November 1993.

[82] M. Subbarao and T. S. Choi, "Accurate Recovery of Three-Dimensional Shape from Image Focus," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 266-274, March 1995.

[83] M. Subbarao, T. Yuan, and J. K. Tyan, "Integration of Defocus and Focus Aalysis with Stereo for 3-D Shape Recovery," *Proceedings of SPIE*, Vol. 3204, pp. 11-23, October 1997.

[84] R. Szeliski "Shape from Rotation," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 625-630, Maui, HI, June 1991.

[85] G. Turk and M. Levoy, "Zippered Polygon Meshes from Range Images", *Proceedings of SIGGRAPH '94*, pp. 311-318, July 1994.

[86] E. Trucco and A. Verri, *Introductory Techniques for 3D Computer Vision*, Prentice Hall, 1998.

[87] J. K. Tyan, *Analysis and Application of Autofocusing and Three-Dimensional Shape Recovery Techniques based on Image Focus and Defocus*, Ph.D. Thesis, Dept. of Electrical Engineering, State University of New York, Stony Brook, Dec. 1997.

[88] B. C. Vemuri and J. K. Aggarwak, “3D Model Construction from Multiple Views Using Range and Intensity Data,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 435-438, June 1986.

[89] A. Watt, *3D Computer Graphics*, Addison Wesley, 1999.

[90] S. Weik. “Registration of 3-D Partial Surface Models Using Luminance and Depth Information,” *Proceedings of International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pp. 93-100, May 1997.

[91] M. Wheeler, Y. Sato, and K. Ikeuchi, “Consensus Surfaces for Modeling 3D Objects from Multiple Range Images,” *Proceedings of IEEE International Conference on Computer Vision*, pp. 917-924, January 1998.

[92] R. Yang and P. K. Allen, “Registering, Integrating, and Building CAD Models from Range Data,” *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 3115-3120, Leuven, Belgium, May 1998.

[93] T. Yuan, and M. Subbarao, “Integration of Multiple-Baseline Color Stereo Vision with Focus and Defocus Analysis for 3-D Shape Measurement”, *Proceedings of SPIE*, Vol. 3520, pp. 44-51, November 1998.

[94] T. Yuan, H. Y. Lin, M. Subbarao and X. Qin, “A Digital Vision System for Three-Dimensional Model Acquisition,” *Vision Geometry IX, Proceedings of SPIE*, Vol. 4117, pp. 70-80, San Diego, July 2000.

[95] Z. Zhang, “Iterative Point Matching for Registration of Free-form Curves and Surfaces,” Technical Report, INRIA, Sophia-Antipolis, 1992, Research Report 1658.

[96] C.L. Zitnick and T. Kanade, “A Cooperative Algorithm for Stereo Matching and Occlusion Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 7, pp. 675-684, July 2000.