FEATURE EXTRACTION, IMAGE SEGMENTATION, AND SURFACE FITTING: THE DEVELOPMENT OF A 3D SCENE RECONSTRUCTION SYSTEM

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Eric D. Lester

May 1998

DEDICATION

I would like to dedicate this thesis to my mother, Mrs. Martha Jane (Baird) Lester. Without her sacrifices I could have never reached this point in my life. She has done more for me than anyone can possibly imagine, without any recognition. I have learned a great deal in my time here at The University of Tennessee, but nothing more important than the values and principles she taught me since I was a young boy. I hope that someday I can repay her for all that she has done for me. Thanks Mama. I love you, Eric.

I would also like to mention in this dedication the new addition to our family, Alexander Jace Franklin. A.J. is my little nephew who has brought so much joy into my life over the past year and a half or so. His parents, Mr. and Mrs. Jason Franklin, have no idea what a great joy they have brought to our entire family by bringing him into this world. The greatest thing is that A.J. will be raised in a loving, Christian home. A.J., I love you more than I can say.

Finally, I want to mention my grandparents. My paternal grandparents are the late Mr. Henry Douglas Lester and the late Mrs. Florence Gertrude (Neville) Lester. My maternal grandparents are the late Mr. Eddie Lloyd Baird Sr. and Mrs. Sarah Elizabeth (Nave) Baird. I love you all very much.

ACKNOWLEDGMENTS

First and foremost, I would like to thank God Almighty for life itself. All that I have is due to His grace and I give all glory to him. Even though I have let him down many times in my life, He has always carried me through. I would like to thank my parents, Mr. and Mrs. Ray and Jane Lester, and my sister, Mrs. Lorie Marie (Lester) Franklin, for their relentless support of me in everything I do. Without my family, I could not have made it to where I am today. I would also like to thank all those who have helped me in my work here at the University of Tennessee: Dr. Mongi Abidi, Dr. Ross Whitaker, Dr. R.C. Gonzalez, Dr. M.O. Pace, Dr. Hamed Sari-Sarraf, and the late Dr. G.W. Hoffman. There are many people who have helped me a great deal in my work and life whom I have not mentioned. However, I want to thank them all, and they know who they are, for all they have done for me.

ABSTRACT

This thesis presents the development of a scene reconstruction system that takes a registered pair of range and amplitude images and produces 3D models of objects in a scene. The system incorporates a series of steps. Its begins by extracting features from both range and amplitude images to create feature maps, fusing these feature maps into a single edge map, segmenting that edge map, fitting surfaces to the objects in the scenes, and generating a three-dimensional model of the scene. The feature extraction method locates both step edges and crease edges from both amplitude and range images. Fusion in the form of Bernoulli's Rule of Combination is performed on the resulting feature maps. Image segmentation is performed using a morphological watershed algorithm. Finally 3D planes, spheres, and cylinders are fit to regions in the segmented scene by the minimization of least-squared error process. These geometric primitives are then displayed in 3D.

This work makes several contributions. First is the use of anisotropic diffusion as a preprocessing step to the extraction of features from range images. Also we show that the fusing of multiple feature maps from real data of complex scenes may be used as input to the watershed segmentation algorithm to obtain reasonable segmentation results. We have improved the watershed algorithm to handle fuzzy-valued inputs and to reduce over-segmentation by enforcing a minimum watershed depth. The result is a system that can start with data from a laser range finder and produce, with virtually no user interaction, 3D models of objects in the scene. Results shown are for the reconstruction of scene from synthetic data as well as real range data of actual scenes in our lab and at Oak Ridge National Labs.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Statement of the Problem

The purpose of this work is to develop a system that autonomously creates a three-dimensional model of a scene using a range-amplitude pair obtained from a single pose of a scanning laser range finder. This system is to be as robust as possible so that little or no user intervention is required in order to complete the processing. This thesis describes the development of such a system.

The first step in the 3D reconstruction process is the acquisition of data. There are three major types of data typically considered with the literature in the area of scene reconstruction. They are range, amplitude, and intensity data. Range data is the value returned from a laser range scanner representing the distance from the source of the laser to the nearest object along the line of sight. Amplitude data is the magnitude of the signal received from a laser range scanner [1]. Intensity data is the reflectance of an object in a scene received by a passive sensor such as a CCD array camera or standard video camera. Figure 1.1 shows these three types of images. Note that visually, an amplitude and intensity image do not differ greatly. However, the information contained in each image is different. The data returned in an amplitude image is dependent upon the reflectivity of the surface, the angle of incidence of the laser beam, and the texture of the surface. Therefore, areas of discontinuity in surface normals, or crease edges, may be detected from

the gradient magnitude of such data. However, intensity images are dependent upon the location and directions of various kind of light sources. Generally step edges are computed from the intensity image and from the range image.

a b c

Figure 1.1: Example of three major data types: (a) Range Image, (b) Amplitude Image, (c) Intensity Image

The range image and amplitude images in Figure 1.1 are both obtained from the Perceptron Laser Range scanner. The range image is that of an indoor hydrology lab at the University of Tennessee. The amplitude image is that of the inside of the foyer of Ferris Hall at the University of Tennessee. The intensity image is that of a small-scale model of a nuclear waste processing plant obtained using a fisheye lens mounted onto a standard CCD array sensor [2].

Noise presents one of the main challenges in analyzing these images. Figure 1.2 shows the type of noise present in range images obtained from the Perceptron Laser Range scanner. Both replacement and additive noise is shown to be present. We need to reduce this noise in order to obtain an image that is noiseless enough to begin extracting meaningful features.

Feature extraction is the process of locating an outstanding part, quality, or characteristic in a given image. One common feature to be extracted is an edge, therefore, edge

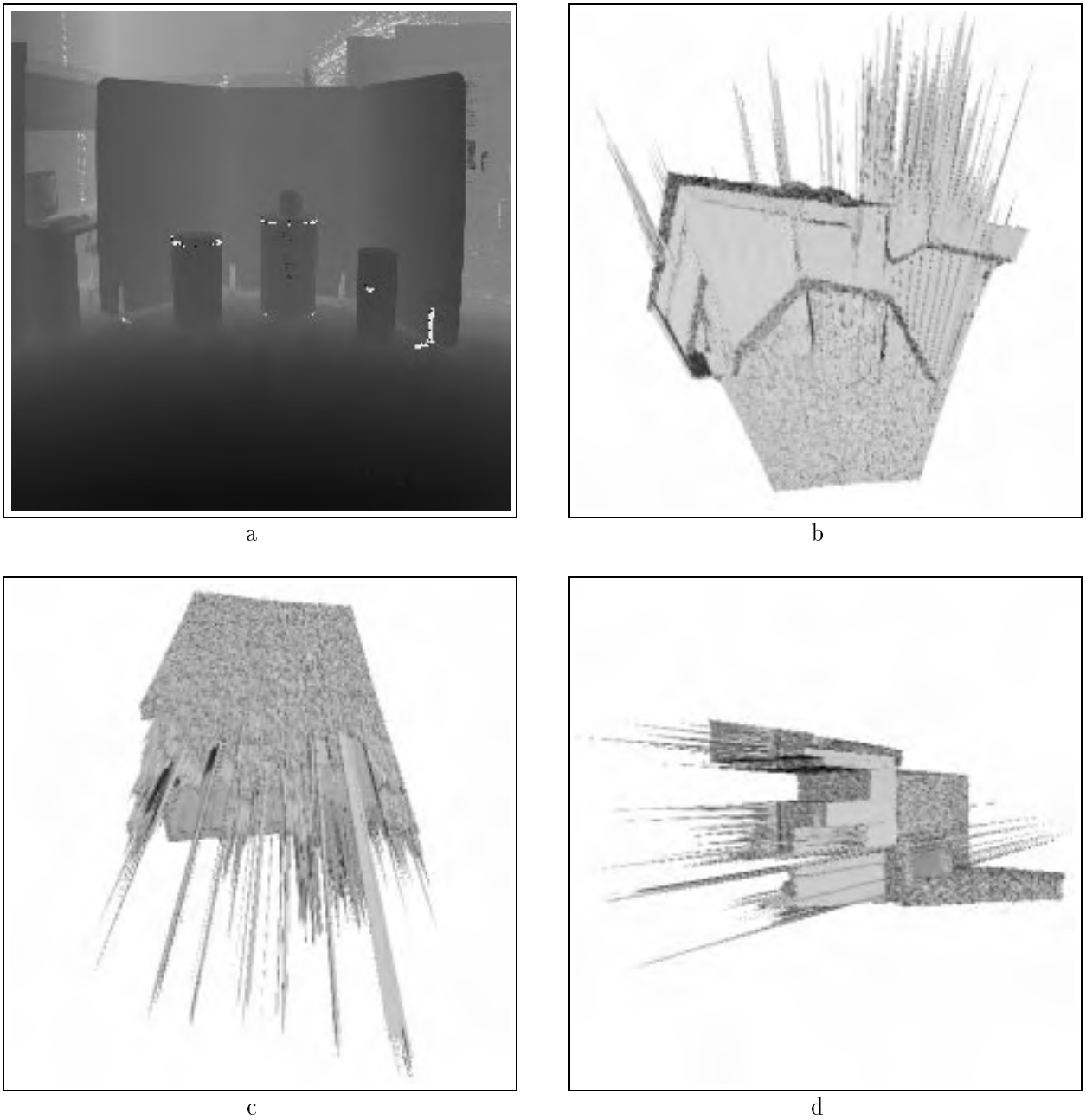


Figure 1.2: Range data with 3D surface plots showing noise. (a) range image, (b) top view (3D surface plot), (c) bottom view, (d) side view

detection is referred to frequently. After preprocessing edge detection is the next step in our surface reconstruction system. Other features include curvature and texture.

The next step in our system is image segmentation. Segmentation is the process of separating an image into distinct regions. Many researchers combine feature extraction and segmentation into a single step, however, in our research we separate the two operations into individual steps.

The final step in our reconstruction system is surface fitting. Surface fitting is the process of inferring those shape parameters that enable some geometric model to closely resemble the data. This step too is often combined in the literature with image segmentation. However, once again, our research makes a definite distinction between the two steps.

1.2 Review of Feature Extraction

Feature extraction is defined as locating those pixels in an image that have some distinctive characteristics. Typically that characteristic is some inhomogeneity in local image properties. Step edges, for instance, are inhomogeneities in intensity or range. Range images and amplitude images have different kinds of features that are meaningful; therefore, different approaches are used for each data type. There are two types of features we detect in images. They are crease edges and step edges. The classical works in edge detection are those of Canny [3] and Marr & Hildreth [4]. However, there are a variety of other methods used for feature extraction in intensity and amplitude images. Boyer and Sarker [5] give an assessment of the state of the art in edge detection in these two types of

images. They perform a fairly comprehensive evaluation and state that due to the research community's success in this area that "edge detection as a research topic should die with dignity." We agree with this premise, as it pertains to step edges in intensity images, but believe that "reliable" feature detection in range images still poses many questions.

In the area of feature extraction in range data, there are also a variety of methods implemented. We divide those methods into four major types. This is certainly not an exhaustive review, nor totally representative of all methods used for range image feature extraction. However, it does give a flavor for the type of work being done in this general research area. One method is that of regularization, or iterative methods. Using this approach, an image is modeled as a function f(x,y). A new function, the solution, f'(x,y), is an approximation with better noise properties. The *energy functional*, $E\langle f'(x,y)\rangle$, is a measure of both closeness to the original data and smoothness. This method is performed to reduce noise in an image while preserving the pertinent features such as edges. This regularization approach is typically an iterative method which involves several free parameters. These parameters control the level of smoothness, edge preservation, and the fidelity of the solution to the input data. This approach is seen in range image feature extraction, but may also be used in intensity or amplitude feature extraction and scene reconstruction. For instance, Richardson [6] develops a spatially variant hybrid stabilizer which is a combination of a first-order and second-order stabilizer capable of smoothing heavily in areas of low contrast and performing little or no smoothing in areas of high gradient, i.e. edges. Other examples of this type of work may be found in [7] [8] [9] [10] [11].

A second category is mathematical morphology. In this approach, morphological oper-

ators such as erosion, dilation, openings, and closings are used to detect and classify edge types in range data. Edges are detected by the residues obtained from openings and closings with various shaped morphological operators. Krishnapuram and Gupta [12] detect roof edges in range images by identifying the residues rendered by a flat, one-dimensional structuring element. For a positive roof edge, opening gives a nonzero response, and for a negative roof edge, closing renders a nonzero response. The basic morphological operators and their operations are given in [13] [14]. Other techniques based on this approach are given in [15] [16].

The third category is multi-scale feature extraction. Multi-scale feature extraction used neighborhood operators of various sizes for smoothing or polynomial approximation. In general, the larger the scale, the more the image is smoothed and edges are more likely to be spatially displaced from their actual location in the image. However, the smaller the scale, the more noise remains in the image and interferes with the feature extraction technique. Thus, by using multiple scales and combining them, a more complete analysis of the features present may be performed. Parvin and Medioni [17] propose an adaptive multi-scale method in which they select an appropriate scale which would be minimum in size while producing a maxima of the curvature at locations of the zero-crossing of the first and second derivatives. They perform this optimal selection in two steps. They begin by describing the original data points in an image as a continuous function and attempt to obtain a suitable surface fit to those points. The surface fit is obtained by applying a medium size kernel along a given direction. Whenever the error in the fit of their continuous function exceeds a certain threshold, the filter size is reduced until either the minimum scale is reached or the error reaches an acceptable level. Lastly, they

attempt to localize features by gradually applying smaller filters, or merge features by applying larger filters. The goal is to obtain an appropriate scale which is smooth enough for differentiation but does not introduce too much blurring. Other work in this area may be seen in [18] [19] [20].

The fourth category of range image feature extraction is differential geometry. Differential geometry in range images is locating and classifying geometric properties of the scene by means of estimating the derivatives of the digitized range data points and using these estimations to infer the geometry of the surfaces in the range image. Surface curvature, crease edges, and other surface characteristics such as critical points may be obtained through this method. A great deal of work has been done in this area by Besl. In [21] he lays out the mathematical methods by which Gaussian and mean curvature may be used to calculate the surface normal of a point in a range image. These steps are shown later in Chapter 2. Maas and Krekel [22] use differential geometry to compute the surface normal to a point in a range image. They then correct for errors in these normal directions, which are present at depth discontinuities, by predicting the depth values of the adjacent pixels and adjust the normal based on these adjacent depths. This is the approach which we have taken in our feature extraction techniques for range images. Other examples of work that rely on differential geometry may be seen in [23] [24]. Of course these four categories of feature detection are not always mutually exclusive. Techniques used in each category may be combined and even used with others. For instance, Burgiss's work [18], mentioned previously under multi-scale feature extraction, combines features extracted using differential geometry from multiple scales [25].

1.3 Review of Image Segmentation

Image segmentation may be defined as separating an image into homogeneous regions, or patches, each of which corresponds to a continuous portion of that image. There are a variety of methods used to segment both range and intensity images. Pal and Pal [26] give a somewhat comprehensive review of image segmentation techniques including intensity images. Our review concentrates primarily on range image segmentation.

Wani and Batchelor [27] state that, in general, three-dimensional image segmentation techniques may be categorized under two main classes: region growing methods, and edge-based methods. Bhandarkar and Siebert [28] state that most range image segmentation algorithms could be classified as either surface-based or edge-based. These two general classifications are typical of what we have seen in the literature. However, we classify segmentation algorithms into four general categories, which are similar to the classifications mentioned above. The four classifications we present are region growing based on surface types, model-based, hybrid approaches, and morphological watersheds.

The first is region growing or clustering based on surface types. Using this approach, the range image is analyzed by identifying properties of the data at each point. This may be the surface normal, surface curvature, planar fits, or other characteristics. Then regions are grown, starting with a seed point, or clustering is performed based on the similarity of the above mentioned characteristics at adjacent pixels. The result is a range image segmented into regions that contain the same properties. An example of this type of segmentation is Sabata et al [29], in which the scene is originally segmented into homogeneous regions by clustering surface normals to obtain an initial segmentation. Then these regions are

merged based on the similarity of their quadric fit parameters. Four individual methods implemented using this type of general approach may be found in [30]. Other methods include [31] [32] [33] [34] [35].

The second category is model-based segmentation. Model-based segmentation begins with a set of three-dimensional models and segments the range image based on the fit of the data to the given model. This approach relates closely to some of the work done in surface fitting, but it appears to be useful for segmentation. An example of this is shown in [36] where they begin with ten modeling primitives and recover the volumetric parts from range data by using a hierarchical segmentation by mapping boundary groups to faces, faces to aspects, and aspects to volumes. The aspects are the different views of the volumetric shapes and the surface shapes are defined by the sign minimum and maximum curvatures of the individual face. Other examples of this technique with various applications are given in [37] [38] [39] [40].

The third division of range image segmentation techniques is the set of hybrid approaches which combine both edge and surface properties to obtain a final segmentation. A classic example of this approach is given by Bhandarkar and Siebert [28] in which surface information in the form of planar fits to the pixel data is combined with roof and step edge parameters to obtain a final segmentation. A step called *vertex refinement* is done to correctly localize the edges. This provides a better segmentation when the surface fit information is combined with the edge information. Other hybrid approaches may be found in [27] [41].

The fourth and final approach is morphological watersheds. This approach takes as input a feature map created by the fusion of differing edge types and changes in the

surface normal, then performing a watershed algorithm on the final fused input feature map. Such an example is given by Baccar [42] in which a Gaussian weighted least squares technique is proposed to extract step and crease edges, then combine them using a special case of Dempster-Shafer fusion, Bernoulli's rule of combination, and segment the resulting, integer-valued feature map using a morphological watershed segmenting algorithm. This is the method which we have chosen for our surface reconstruction system because it is simple to implement, has relatively few free parameters, and offers a means of combining different types of image features. Other background material and examples of this type of segmentation may be found in [43] [44] [45] [42] [14] [46].

1.4 Review of Surface Fitting

The objective of surface fitting is to describe objects in a scene using mathematical models. There are a number of different methods for surface fitting. These methods may be divided into two separate categories: two-and-a-half-dimensional and three-dimensional. Two-and-a-half-dimensional surface fitting assumes that one 3D coordinate is a function of the other two coordinates, i.e. $z = f(x, y)$, and a surface is fit to the data in that form. Three-dimensional surface fitting assumes the data is a function of three independent variables, using either an implicit model

$$\mathbf{S} = \left\{ \left(\begin{array}{c} x \\ y \\ z \end{array} \right) \middle| f(x, y, z) = 0 \right\} \tag{1.1}$$

or a parametric model

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} f_1(u,v) \\ f_2(u,v) \\ f_3(u,v) \end{pmatrix} \tag{1.2}$$

The two-and-a-half-dimensional approach is limited because it assumes the surfaces and the scene can be described as functions of the horizontal and vertical indices. It models only a small subset of possible scenes and the description is not viewpoint invariant. However, since it does imply a depth measurement which is more than two-dimensional, we call it two-and-a-half-dimensional. Typically the solution to these types of fits are much simpler than in the three-dimensional case and lend themselves to closed-form solutions. They may be quadratics or other polynomials. Boyer et al [47] present such an approach based on the robust sequential estimator which is able to handle handle noise and remove outliers in the process of estimating the surface. Miller and Stewart [48] propose a method using unbiased scale estimates based on the least median of squares algorithm which is able to operate on data that contains large percentages of outliers. Their method can extract surfaces containing less than 50% of the total data points. One final method which we examined was the application of the fuzzy average to surface fitting [49]. These two-and-a-half-dimensional methods represent three separate methods of estimating the error between the proposed surface and the actual data points.

Three-dimensional surface fitting can be divided into two categories. The first is based on a single range image, or viewpoint of the scene. Taubin [50] proposes an improved method of surface fitting by using a better approximation of the Euclidean distance of the surface to the actual data. Ferrie et al [51] cover three approaches which are used in three-dimensional surface fitting and explain the use of differential geometry in volumetric modeling. Using this basic strategy, Kumar et al [52] use hyperquadrics to describe the range data in a given scene. DeCarlo and Metaxas [53] describe volumes in range data using a physics-based framework in the form of blending by allowing different geometric models combine to represent a single object in a unified model. Lastly, Keren et al [54] describe complex objects in a scene by fourth-degree polynomials and claim that they are mathematically easier to manipulate than superquadrics.

Another category of three-dimensional surface fitting strategies are those that combine or *fuse* two or more registered views of the same scene. In these approaches, several views of the same object or scene are registered and a volumetric model is fit to the combination of the views. Chen and Medioni [55] present a method that takes a volume composed of several views and fits a surface to the data by placing a model inside the data and allowing it to inflate out to the surface of the data like a balloon. Other methods are implemented to build complex models from range images based on volumetric occupancy grids and signed distance functions and are given in [56] [57] [58]. A good review of 3D surface fitting is given in [59].

1.5 Review of Scene Reconstruction Systems

We define scene reconstruction as inferring models of a scene from a set of noisy measurements (range or intensity). Scene reconstruction is used for a variety of purposes. Abidi et al [60] propose using scene reconstruction for robotic inspection and manipulation of objects in an area. Jung and Gupta [61] propose using scene reconstruction for path planning purposes. Sequeira et al [62] uses solely laser range sensing to model, or reconstruct a 3D environment for navigation and verification purposes. Walker [63] presents a method that uses the geometric properties of objects in a scene to aid in navigation and exploration.

As reviewed in the section on surface fitting, many systems utilize only one source of input data to reconstruct the scene. Other systems utilize multi-sensor or multi-view feedback for reconstruction. Sequeira, Goncalves, and Ribeiro [64] present an approach which combines various range images of a single scene from differing viewpoints to develop high-level descriptions of the scene. Turk and Levoy [65] combine multiple range images to create a zippered polygon mesh of an object in a scene so that the object may be rotated and viewed from any angle. Zhang and Wallace [66] present an approach similar to ours in that they combine range and intensity edge information to physically model a given scene.

Different methods may be used to accomplish the task of scene reconstruction. Bellon et al [67] propose what they call a hybrid approach to range image segmentation and reconstruction in which they use a clustering algorithm to segment and reconstruct a scene based on a single range image. Gokmen and Li [8] use refined regularization where

the smoothness of of the surface is regulated spatially over the entire image. Whitaker [68] proposes using level-set models to reconstruct a 3D scene using more than one point of view.

One of the most recent advances of scene reconstruction is the development of the Artisan system by The Robotics Institute at Carnegie Mellon University [69]. The Artisan system incorporates multiple range views and reconstructs the scene piece by piece, via a human operator, by matching CAD models to selected regions of interest, or objects. The results are then used by human operators to map out and execute remote robotic operations.

There are a variety of fields that warrant application-specific reconstruction systems. Waksman and Rosenfeld [70] propose a method in which geometric properties of trees may be recovered from one range image. The medical field is also an area of important application of 3D reconstruction. Menon and Acharya [71] propose representing DNA replication sites via finite element based deformable models to be used in biomedical image analysis. Ozturk et al [72] use structured light to develop a non-invasive 3D wound surface reconstruction system to assess the overall size and orientation of wounds.

1.6 Overview of Our Surface Reconstruction System

The system which we have developed is a series of steps. Figure 1.3 shows the steps which compose the system. The input to the system is a registered pair of range and amplitude images from a single pose of a given scene. We use median filtering and anisotropic diffusion to prepare the images for feature extraction. Then we compute the gradient

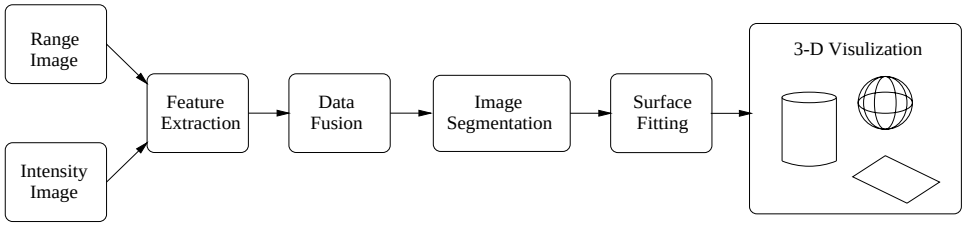


Figure 1.3: Overview of surface reconstruction system.

magnitude of the range image, the gradient magnitude of the amplitude image, and the gradient of the surface normal of the range image to generate three separate feature maps of the scene. These feature maps are meant to include both crease edges and step edges. These three feature maps are fused to produce a single fuzzy-valued edge map of the scene. This fuzzy-valued feature map is used as input to the watershed segmentation algorithm to produce a segmented scene excluding object boundaries. The segmentation algorithm is based on morphological watersheds and is able to accept fuzzy-valued edgemaps and produce a segmented scene. Each segment in the scene is labeled with an integer value. Finally each region in the scene of reasonable size is fit with a three-dimensional volume. The result is a three-dimensional reconstruction of the scene including the fitted volumetric models.

The remainder of the thesis will explain in detail each step in our surface reconstruction system. Chapter 2 will cover feature extraction and data fusion. Chapter 3 will cover image segmentation. Chapter 4 will cover surface fitting. Chapter 5 will give results and an evaluation of the entire system including 3D visualization. Finally in Chapter 6 we make some conclusions about our work and suggest future research to expound upon the work shown in this thesis.

CHAPTER 2

Feature Extraction

The goal of feature extraction is to locate points in the scene that lie along *boundaries*. By boundaries we mean sets of pixels that either separate objects from one another or represent changes in the surface geometry of an object. This work considers two basic types of boundaries; step edges and crease edges. Step edges are those edges that represent depth discontinuities, and crease edges are those that represent a crease, or a discontinuity in the surface normal, in a single object. From the range data, steps edges can be identified by the gradient magnitude and crease edges can be located by computing the gradient magnitude of the 3D surface normal. From the amplitude data, object boundaries coincide with high gradient magnitude when objects differ in color or texture. Crease edges can be located from the gradient magnitude of the amplitude image because of differing angles of incidence of the laser beam.

A short overview of our feature extraction methodology is as follows. We begin with some preprocessing to reduce the effects of noise. Given a registered pair of range and amplitude images of a scene from a single pose, we median filter the data to reduce replacement noise (outliers or misfires from the laser) and implement an edge preserving, spatially variant smoothing operator on both the range and amplitude data to reduce additive noise. Once the data has been filtered, we compute the gradient magnitude of the range, the gradient magnitude of the amplitude, and the gradient magnitude of the

surface normal to generate three distinct feature maps. These maps are fused to create a single fuzzy-valued edge-map which serves as input to the segmentation algorithm.

2.1 Image Acquisition

In this research we work with both real and synthetic data. Both data types are important to our work. Synthetic images are generated by a laser scanning simulation program using the Telegrip software package. This simulation, developed by Richard Collier [73], returns noise-free range data and is designed to mimic the spherical coordinate system of the Perceptron Laser Scanner.

In the synthetic data simulation, the spherical coordinate scanning system of the Perceptron Laser Scanner is modeled almost identically. The only difference is that the raw synthetic data returned contains no noise and does not need to be scaled and offset by Equation 2.1. Thus the raw data becomes the range data, and can be used in Equations 2.2 through 2.8 to compute the three dimensional coordinates of each data point. These synthetic images can also range in resolution from 256×256 on up to 1024×1024 and greater if so desired. However, we are only able to obtain a range image of the scene and not a registered pair including amplitude data.

The real data is obtained using the Perceptron laser range scanner [1] [74] that we have in house. This laser scanner has a $55°$ field of view on the horizontal axis and a $45°$ field of view on the vertical axis. We work with images that vary in size from 256×256 pixels to 1024×1024; however, the Perceptron is capable of producing images larger in size. The scanning takes place using a spherical coordinate system.

The actual range values are obtained from the raw data by a scaling and offset given by

$$R(i,j) = 2.14\ S(i,j) + 2015, \tag{2.1}$$

where *R(i,j)* is the range value, in millimeters, at each pixel location and *S(i,j)* is the raw scanner value at each pixel location.

Both the azimuth and elevation angles, given in radians, at which a given pixel is scanned can be computed by the following equations:

Azimuth:

$$\theta(i) = \Delta_\theta i - \frac{1}{2130}w + \frac{1}{2130} \tag{2.2}$$

$$\Delta_\theta = \frac{1}{1065} \tag{2.3}$$

Elevation:

$$\phi(j) = \Delta_\phi j + \frac{1}{2632}h - \frac{1}{2632} \tag{2.4}$$

$$\Delta_\phi = \frac{1}{1316} \tag{2.5}$$

where i is the vertical image pixel index, j is the horizontal image pixel index, w is the width of the image (in pixels), and h is the height of the image (in pixels).

These angles can now be used to compute the Cartesian coordinates, relative to the laser origin, of a point in the scene. These coordinates can be found by the following equations:

$$X(i,j) = R(i,j)\cos(\theta(j)) \tag{2.6}$$

$$Y(i,j) = R(i,j)\cos(\theta(i))\sin(\phi(j)) \tag{2.7}$$

$$Z(i,j) = R(i,j)\cos(\theta(i))\cos(\phi(j)) \tag{2.8}$$

Using Equations 2.1 through 2.8, the true three dimensional location, relative to the laser origin, of each pixel in the image can be used for true three-dimensional feature extraction and surface fitting. The Perceptron also returns an amplitude image which is the amplitude of the laser signal returned. The amplitude value depends on the angle of incidence of the beam, the reflectance of the surface, and the distance to the surface. An example of a set of real registered range and amplitude pair of a scene obtained using this scanner is shown in Figure 2.1.

a b

Figure 2.1: Registered range and amplitude data obtained from Perceptron Laser Scanner: (a) Range Image, (b) Amplitude Image. (256×256)

2.2 Preprocessing

The range data returned from the Perceptron laser range scanner contains replacement and additive noise. Additive noise is small fluctuations added to the actual distance of the surfaces in the scene. Replacement noise points are data points in the image that are uncorrelated to the data. Replacement noise may be either extremely high in value or

extremely low in value relative to the actual value at that point. In the amplitude data, the replacement noise occurs when the laser does not receive a signal back, usually caused by a reflection on the surface. This type of replacement noise has a very low relative value, which we call *black noise*. In the range data, the replacement noise is both relatively high and low, which we call white and black noise respectively. The white noise in range data is caused by surfaces with high reflectivity and the black noise is caused by surfaces with low reflectivity. When black and white noise are present it is called salt and pepper noise.

We want to remove as much of this noise as possible; however, we must be careful not to over-smooth the data, eliminating valuable edge information. In order to remove the salt and pepper noise in the range and amplitude data respectively, we median filter both images. The neighborhood of the median filter mask typically ranges from 3×3 to 7×7. After this median filtering takes place, we implement a spatially variant smoothing operator called vector-valued anisotropic diffusion [75] on both the range and amplitude data in order to reduce additive noise while preserving the crease and step edge information.

The anisotropic diffusion that we implemented is an iterative process. At each iteration, the gradient magnitude of the range, amplitude, and surface normal are computed for every pixel. Each of these gradients is weighted, and the combination of that weight and the value of the gradients at each pixel determine the amount smoothing done on that pixel. At each pixel, the smoothing is inversely proportional to the three gradient magnitudes. Therefore, at areas of high gradient there is less smoothing and more smoothing at areas of low gradient. Finally, the result of anisotropic diffusion is a smoothed range-amplitude pair corresponding to the input range-amplitude pair. The mathematical

representation of anisotropic diffusion is as follows. Let the input image be $I(x, y)$ which is a two-dimensional function of x and y. Therefore, for a single image, the general form of the smoothing function is

$$\frac{\partial f(x, y, t)}{\partial t} = \nabla \cdot g(x, y, t)\, \nabla f(x, y, t). \tag{2.9}$$

Where g(x,y,t) = is the conductance function [76]. We incorporate multiple images in the anisotropic diffusion smoothing function. For multiple images, the smoothing function is

$$\frac{\partial \vec{f}(x, y, t)}{\partial t} = \nabla \cdot g(x, y, t)\, \nabla \vec{f}(x, y, t). \tag{2.10}$$

In general, $\vec{f}(x, y, t)$ may be composed of any number of images. For our work, $\vec{f}(x, y, t)$ is given by

$$\vec{f} = \begin{bmatrix} r \\ a \\ n_x \\ n_y \\ n_z \end{bmatrix} \tag{2.11}$$

Where r is the range image, a is the amplitude image, n_x is the x component of the surface normal, n_y is the y component of the surface normal, and n_z is the z component of the surface normal. The conductance term we use is

$$g = \exp\left[-\frac{(\nabla r)^2}{k_r^2} - \frac{(\nabla a)^2}{k_a^2} - \frac{(\nabla n_x)^2}{k_{nx}^2} - \frac{(\nabla n_y)^2}{k_{ny}^2} - \frac{(\nabla n_z)^2}{k_{nz}^2}\right] \tag{2.12}$$

This equation for the conductance term which we use introduces some free parameters which will be discussed later in Chapter 5. However we can show these free parameters in vector form as $\vec{K}$ by

$$\vec{K} = \begin{bmatrix} k_r \\ k_a \\ k_{nx} \\ k_{ny} \\ k_{nz} \end{bmatrix}. \tag{2.13}$$

We can see from Equation 2.12 that the amount of smoothing performed on each of the images in Equation 2.11 is inversely proportional to value of the gradient of these images and their respective conductance parameters. We implement this form of anisotropic diffusion iteratively using finite forward differences. At each iteration, the conductance is recomputed from the smoothed images created in the previous iteration. The final result from this smoothing which we use for feature extraction is a smoothed range and amplitude image.

This form of anisotropic diffusion is similar to the approach that Richardson [6] uses via regularization where a hybrid stabilizer is used that smoothes varying amounts depending upon the gradient of the data. Figure 2.2 and 2.3 shows the results, before and after, of anisotropic diffusion on a range-amplitude image pair obtained from the Perceptron Laser Range Scanner.

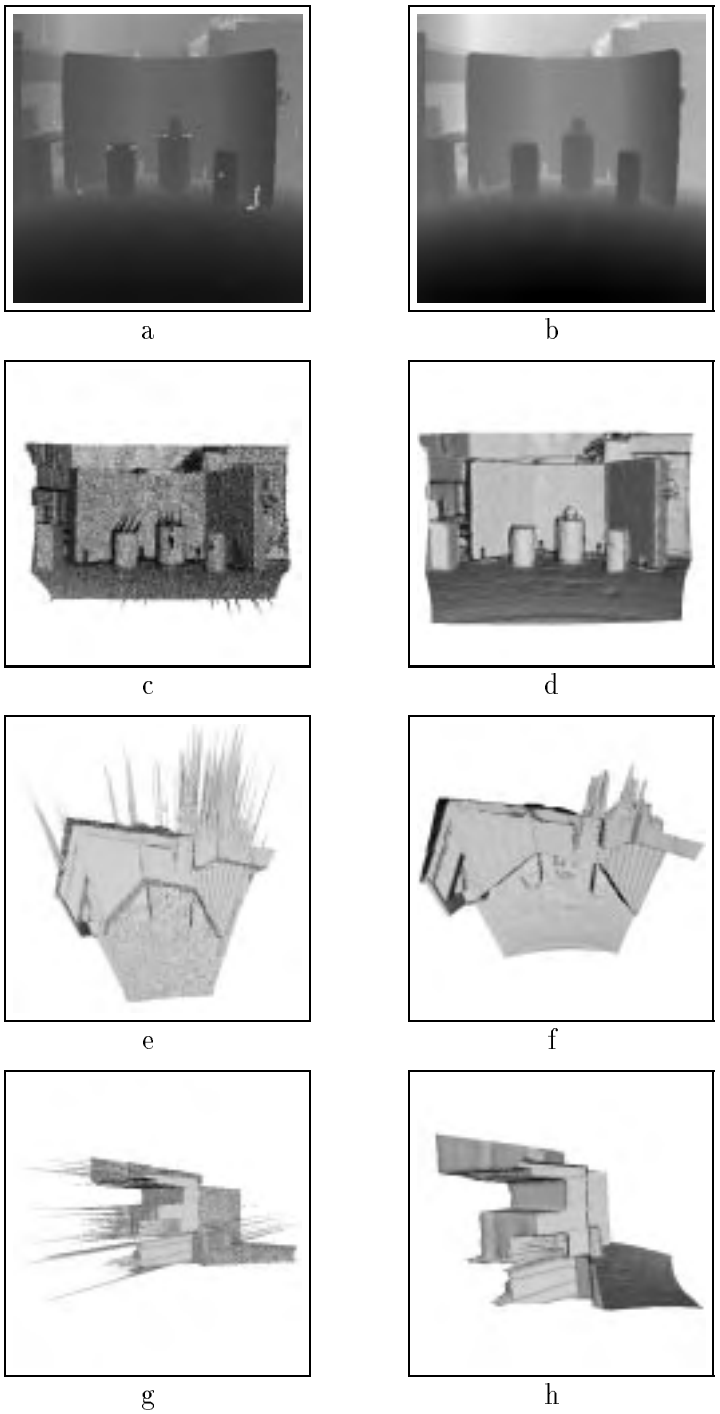


Figure 2.2: Results of median filtering and anisotropic diffusion on range data: before smoothing: left column & after smoothing: right column.

a b

Figure 2.3: Result of median filtering and anisotropic diffusion on real amplitude data (256×256) obtained from Perceptron Laser Scanner: (a) range Image, (b) amplitude Image.

2.3 Location of step edges

We use the gradient magnitude of both the range and amplitude data to locate the step edges and crease edges, respectively, in the scene. The simple formulation for the gradient magnitude of amplitude and range data is

$$g(\phi,\theta) = \sqrt{\left[\frac{\partial f(\theta,\phi)}{\partial \theta}\right]^2 + \left[\frac{\partial f(\theta,\phi)}{\partial \phi}\right]^2}, \tag{2.14}$$

where $g(\phi,\theta)$ is the output gradient magnitude and $f(\theta,\phi)$ is the input range or amplitude data. These derivatives are computed using *centralized differences*. The kernels for these derivatives in the θ and ϕ directions, K_θ and K_ϕ are are given in Equations 2.15 and 2.16, respectively.

$$K_\theta = [\ -0.5 \quad 0 \quad 0.5\ ], \tag{2.15}$$

$$K_\phi = \begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix}, \tag{2.16}$$

Therefore the approximations of the partial derivatives are

$$\frac{\partial f}{\partial \theta} \approx \Delta_\theta K_{theta} \otimes R(i,j) \tag{2.17}$$

and

$$\frac{\partial f}{\partial \phi} \approx \Delta_\phi K_{phi} \otimes R(i,j) \tag{2.18}$$

where $''\otimes''$ is a discrete convolution.

2.4 Location of crease edges

We can locate the crease edges in the image from the range data by identifying changes in the surface normal. The following is a general procedure for calculating the normal to a surface $z = f(x, y)$. Similar procedures can be seen in [21].

Consider the special case where the projection onto the scanner coordinate system is orthographic. A point on the surface is $z = f(x, y)$. The partial derivatives, or the vectors

tangent to the surface, needed to find the surface normal are

$$\vec{T}_{(x)} = \begin{bmatrix} 1 \\ 0 \\ f_x \end{bmatrix}, \tag{2.19}$$

and

$$\vec{T}_{(y)} = \begin{bmatrix} 0 \\ 1 \\ f_y \end{bmatrix}, \tag{2.20}$$

where $f_x = \frac{\partial z}{\partial x}$ and $f_y = \frac{\partial z}{\partial y}$.

The unnormalized surface normal is the cross product of $\vec{T}_{(x)}$ with $\vec{T}_{(y)}$;

$$\vec{S} = \begin{bmatrix} -f_x \\ -f_y \\ 1 \end{bmatrix}. \tag{2.21}$$

Then the magnitude of the surface normal is found and used for normalization. That normalized surface normal is

$$\vec{N} = \frac{\vec{S}}{\|\vec{S}\|}, \tag{2.22}$$

To locate the gradient of this surface normal, we can take the partial derivative of $\vec{N}$ with respect to x and to y. Each will produce a three-dimensional vector. This gradient,

G, is the magnitude of both vectors summed together, i.e.

$$G = \left\| \frac{\partial \vec{N}}{\partial x} \right\|^2 + \left\| \frac{\partial \vec{N}}{\partial y} \right\|^2 \tag{2.23}$$

where

Taking this approach assumes that the horizontal and vertical pixel indices of the image are equal to the x and y three-dimensional laser coordinates, respectively. However, this is not true with the range data produced by the Perceptron Laser Scanner in Equations 2.1 - 2.8. Therefore, we have developed an approach that considers the true three-dimensional location of each pixel in computing the surface normal and its gradient. The following three sections explain methods we have implemented for locating crease edges. The first is a complete three-dimensional method which we implemented that considers the true 3D laser coordinates of the scene, but is too mathematically intensive for practical application. It also introduces errors in the calculations. The second method is a two-and-a-half-dimensional approach using the Second Fundamental Form as developed by Besl [21] to compute the gradient of the surface normal. The final method is a hybrid method we developed which considers the three-dimensional properties of the range surface but is less mathematically intensive than the first, complete three-dimensional method we developed.

2.4.1 Full 3D Method

We begin by computing the true x, y, and z coordinates for each pixel in the image by Equations 2.1 through 2.8 so that each pixel in the image represents a three-dimensional

position vector:

$$\vec{p}(\theta,\phi) = \begin{bmatrix} x(\theta,\phi) \\ y(\theta,\phi) \\ z(\theta,\phi) \end{bmatrix} \tag{2.24}$$

Where θ and ϕ are functions of the horizontal and vertical image indices, i and j, respectively. To compute the surface normal, the partial derivatives are given by:

$$\vec{D}_{p\theta} = \begin{bmatrix} \frac{\partial x}{\partial \theta} \\ \frac{\partial y}{\partial \theta} \\ \frac{\partial z}{\partial \theta} \end{bmatrix} \tag{2.25}$$

and

$$\vec{D}_{p\phi} = \begin{bmatrix} \frac{\partial x}{\partial \phi} \\ \frac{\partial y}{\partial \phi} \\ \frac{\partial z}{\partial \phi} \end{bmatrix} \tag{2.26}$$

Similar to Equation 2.21, the unnormalized surface normal, $\vec{S}$ is computed by taking the cross product of $\vec{D}_{p\theta}$ with $\vec{D}_{p\phi}$, given by

$$\vec{S} = \begin{bmatrix} \left(\frac{\partial y}{\partial \theta}\frac{\partial z}{\partial \phi} - \frac{\partial z}{\partial \theta}\frac{\partial y}{\partial \phi}\right) \\ \left(\frac{\partial z}{\partial \theta}\frac{\partial x}{\partial \phi} - \frac{\partial x}{\partial \theta}\frac{\partial z}{\partial \phi}\right) \\ \left(\frac{\partial x}{\partial \theta}\frac{\partial y}{\partial \phi} - \frac{\partial y}{\partial \theta}\frac{\partial x}{\partial \phi}\right) \end{bmatrix} \tag{2.27}$$

This result is normalized, $\vec{N} = \frac{\vec{S}}{\|\vec{S}\|}$, in the same manner and the gradient of this surface normal follows the procedure given in Equations 2.21 through 2.23. This result now takes

into account the true three dimensional location of each point in the image.

2.4.2 Shape Matrix

The second method is the use of the Shape Matrix to compute curvature of a surface [21]. The Shape Matrix, which we will call β is derived from the First Fundamental Form, **I**, and the Second Fundamental Form, **II**. The general formula for the Shape Matrix is given by

$$\beta = \mathbf{II} \cdot \mathbf{I}^{-1}. \tag{2.28}$$

The Gaussian curvature, or H is

$$H = \frac{1}{2}\ \mathrm{Tr}[\beta]. \tag{2.29}$$

Assuming a point in the image is $z = f(x, y)$, H may be expanded to the form

$$H = \frac{1}{2}\frac{(1+f_y^2)f_{xx} + (1+f_x^2)f_{yy} - 2f_x f_y f_{xy}}{(1+f_x^2+f_y^2)^{\frac{3}{2}}}. \tag{2.30}$$

and the mean curvature, K is

$$K = det[\beta], \tag{2.31}$$

or

$$K = \frac{f_{xx}f_{yy} - f_{xy}^2}{(1+f_x^2+f_y^2)^2}, \tag{2.32}$$

where: $f_x = \frac{\partial z}{\partial x}$, $f_y = \frac{\partial z}{\partial y}$, $f_{xx} = \frac{\partial^2 z}{\partial x^2}$, $f_{yy} = \frac{\partial^2 z}{\partial y^2}$, and $f_{xy} = \frac{\partial^2 z}{\partial x \partial y}$.

The norm of the Shape Matrix is $H^2 - K$.

2.4.3 Hybrid Method

The final method which we have implemented takes into consideration the true three dimensional coordinates of a given point in the data, yet is not as computationally intensive as the *Full 3D Method*. Given a function $f(\theta, \phi)$ and Equations 2.19 and 2.20:

$$f_x = -\left[\frac{\partial z}{\partial \theta}\right]\left[\frac{\partial \theta}{\partial x}\right] = \frac{K_\theta \otimes z}{K_\theta \otimes x} \tag{2.33}$$

and

$$f_y = -\left[\frac{\partial z}{\partial \phi}\right]\left[\frac{\partial \phi}{\partial y}\right] = \frac{K_\phi \otimes z}{K_\phi \otimes y} \tag{2.34}$$

Given that the x, y, and z values are derived from Equations 4.54 through 2.8, the formulation for the gradient magnitude of the surface normal is as follows:

$$N_{(x)} = \frac{f_x}{\sqrt{f_x^2 + f_y^2 + 1}} \tag{2.35}$$

$$N_{(y)} = \frac{f_y}{\sqrt{f_x^2 + f_y^2 + 1}} \tag{2.36}$$

$$N_{(z)} = \frac{1}{\sqrt{f_x^2 + f_y^2 + 1}}. \tag{2.37}$$

Where the surface normal can be expressed as

$$\vec{N} = \begin{bmatrix} N_{(x)} \\ N_{(y)} \\ N_{(z)} \end{bmatrix}. \tag{2.38}$$

Where the parentheses in the subscript are to distinguish this notation from partial derivatives. The gradient magnitude of the surface normal, GSN, is

$$||\nabla \vec{N}|| = \left[\frac{\partial N_{(x)}}{\partial \theta}\right]^2 + \left[\frac{\partial N_{(x)}}{\partial \phi}\right]^2 + \left[\frac{\partial N_{(y)}}{\partial \theta}\right]^2 + \left[\frac{\partial N_{(y)}}{\partial \phi}\right]^2 + \left[\frac{\partial N_{(z)}}{\partial \theta}\right]^2 + \left[\frac{\partial N_{(z)}}{\partial \phi}\right]^2. \tag{2.39}$$

Where the gradient is taken with respect to the image coordinates.

2.4.4 Fuzzy Remapping of Features

The resulting values of Equations 2.14 and 2.39 do not have an upper bound. That is, the values at a given pixel may vary from 0 to the maximum gradient for the given image. This maximum gradient will be different for almost every image. Therefore, to fuse any two or more of these feature maps together, we remap the these feature maps to fuzzy values between 0.0 and 1.0. This remapping is done using the following procedure.

Let f_i be the initial feature map obtained from either Equation 2.14 or 2.39. Then let $\langle f \rangle$ be the mean value of all the pixels in f_i. Finally, let p be the remapping percentage used to scaled the image to fuzzy range values varying from 0.0 to 1.0. The resulting feature map, f_o is

$$f_o = 1 - e^{\frac{-f_i^2}{2k^2}}. \tag{2.40}$$

Where

$$k = \frac{p \cdot \langle f \rangle}{100} \tag{2.41}$$

Therefore, the resulting feature map, f_o, is always fuzzy valued between 0.0 and 1.0. The remapping percentage may be altered to allow more features to be remapped to higher fuzzy values. The higher the remapping percentage, low values of f_i are mapped closer to 0.0 than when using a lower remapping percentage. A discussion of this free parameter p is given in Chapter 6.

2.5 Image Fusion

Information about a scene is contained in both the range and amplitude data. In order to gain the most comprehensive segmentation of the scene, we would like to incorporate all the information we possibly can. Therefore, fusion is performed using the amplitude gradient, range gradient, and surface normal gradient where possible.

2.5.1 Methods of image fusion

Because these edge-maps are actually fuzzy feature maps, and not binary, we have examined some fuzzy set unions and intersections given in Equations 2.42 through 2.46.

Fuzzy Intersection $a \cap b$:

Dubois & Prade [77]

$$\frac{a * b}{\max(a, b, \alpha)} \tag{2.42}$$

Dombi [77]

$$\frac{1}{1+[(\frac{1}{a}-1)^{\lambda}+(\frac{1}{b}-1)^{\lambda}]^{\frac{1}{\lambda}}} \tag{2.43}$$

Fuzzy Union $a \cup b$:

Dubois & Prade [77]

$$\frac{a+b-a*b-\min(a,b,1-\alpha)}{\max(1-a,1-b,\alpha)} \tag{2.44}$$

Dombi [77]

$$\frac{1}{1+[(\frac{1}{a}-1)^{-\lambda}+(\frac{1}{b}-1)^{-\lambda}]^{\frac{-1}{\lambda}}} \tag{2.45}$$

where $\alpha \in (0,1)$ and $\lambda \in (0,\infty)$.

Bernoulli's rule of combination [78]

$$1-(1-a)*(1-b) \tag{2.46}$$

Given we are fusing data that may not be redundant, we have chosen to implement fuzzy unions to combine the data. As shown in Figure 2.4 the results from the three methods of taking fuzzy unions are extremely similar. Because of this and the fact that Bernoulli's rule of combination has no free parameters and can be easily extended to accept three or more inputs, we have chosen to implement it. The overall scheme of the data fusion algorithm is shown in Figure 2.5.

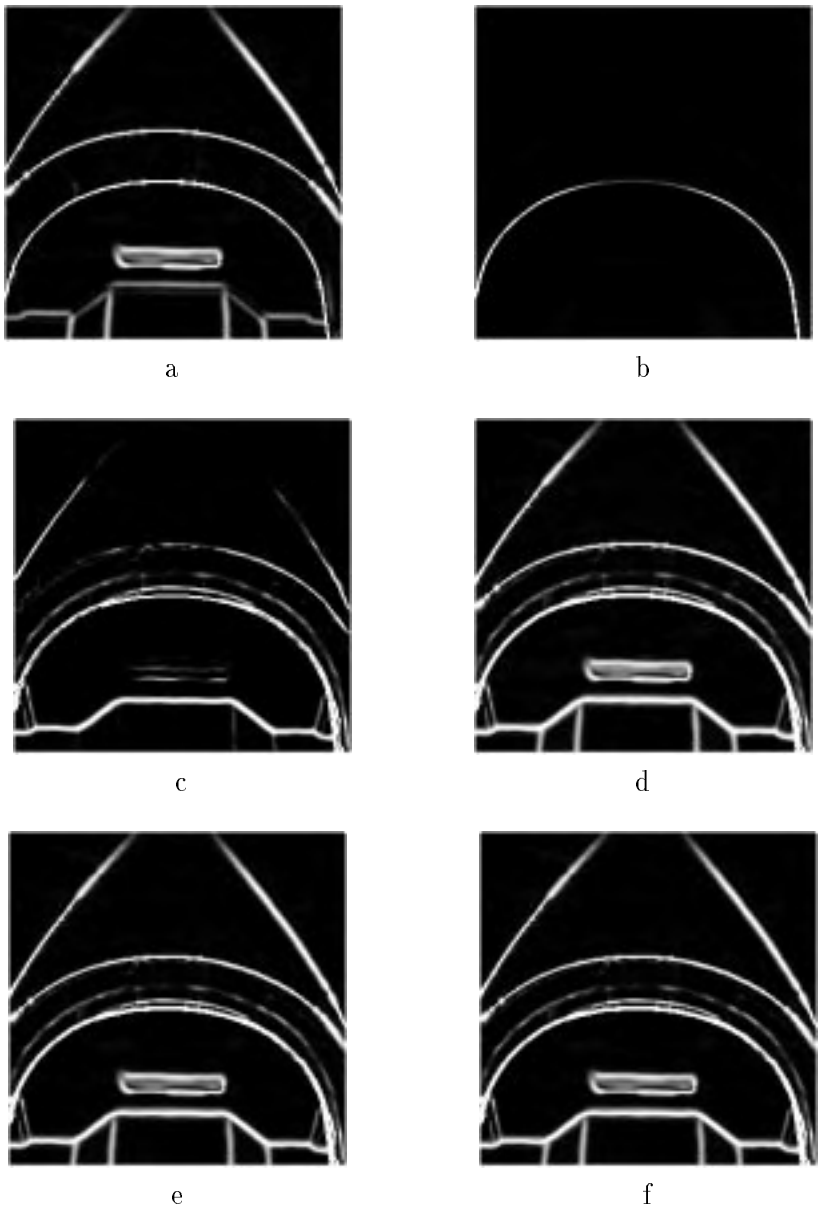


Figure 2.4: Comparison of fzzy union data fusion methods: (a) gradient magnitude of amplitude data [input 1], (b) gradient magnitude of range data [input 2], (c) gradient magnitude of surface normal [input 3], (c) Bernoulli's Rule of Combination of (a),(b), and (c), (d) Dubois & Prade Union of (a),(b), and (c), (e) Dombi Union of (a),(b), and (c).

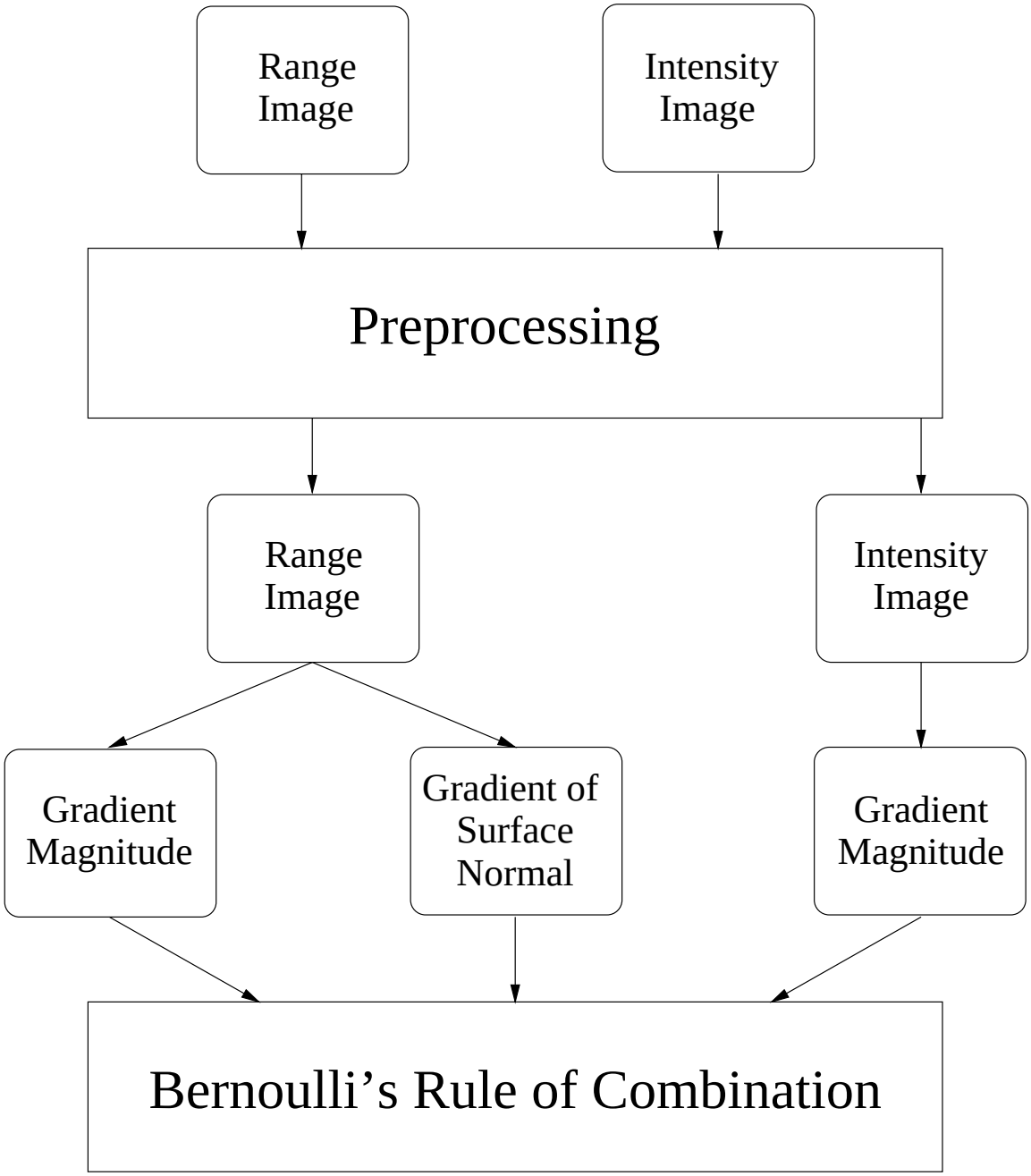


Figure 2.5: Flowchart of fusion algorithm

CHAPTER 3

Image Segmentation

3.1 Introduction

In order to isolate specific objects, to which we will fit surfaces, we must segment the input data. We do this using a segmentation method based on morphological watersheds. This concept of watersheds and catchment basins are discussed in [42]. In this section we will begin by giving a rough overview of the segmentation algorithm based on morphological watersheds, introducing some terms that will be used later in the section. Then we will describe each of the steps individually and show some intermediate results, obtained from a synthetic image, to help show how our algorithm works. Our segmentation algorithm on Baccar's method [42], but it has several important differences.

The principles underlying morphological watersheds is that a symbolic drop of water, placed on a pixel of high gradient, will drain down to a regional minimum. A regional minimum is the lowest gradient value in a given region. Each pixel that lies in the descending path of this drop of water will be associated with that same regional minimum forming catchment basins which will ultimately identify distinct regions having different labels. Each local minimum has an associated segment defined by its catchment basin. The watershed algorithm will operate on fuzzy feature maps, similar to Rosin et al. [79], discussed previously in Chapter 2. High values are those that are more likely to be region boundaries. Basically we use gravity to allow the *water* to drain down from areas of high

gradient. The basic principle of our watershed algorithm is shown in the one-dimensional case in Figure 3.1.

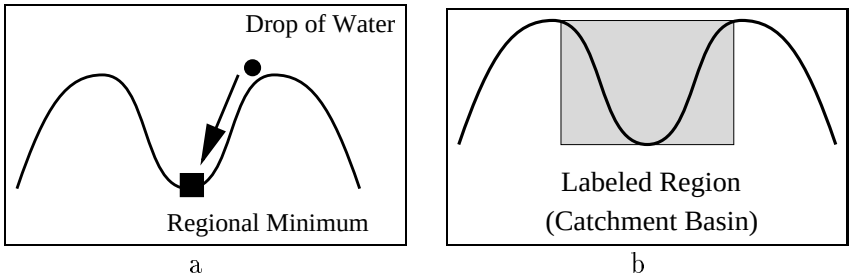


Figure 3.1: Forming of catchment basins (1D): (a) flow of water from high gradient to regional minimum, (b) catchment basin formed by watershed algorithm.

3.2 Details of the Watershed Segmentation Algorithm

The steps of the watershed algorithm are the following.

1. Create an *image boundary* around the input feature map.
2. Threshold small fluctuations on the low end of the input feature map.
3. Locate and label all singe pixel regional minima.
4. Locate and mark all flat regions.
5. Trace down all pixels, which are not members of flat regions, to their local minima.
6. Trace all remaining unlabeled flat regions to their respective regional minima.
7. Mark flat regions that are regional maxima.
8. Mark boundary pixels between regions.

9. Threshold the initially segmented image according to a minimum watershed segment depth to create a binary image with any broken watershed boundary appendages removed.

10. Sequentially label all individual regions bounded by the marked edges in the binary image to create a segmented image.

The following paragraph explains each of these steps in more detail.

We begin by accepting as input a fuzzy-valued feature map of a scene and build a *image boundary* around the borders of the image. The image boundary is set so that no *water* can flow outside the bounds of the image indices. The setting of the height of the image boundary is shown in Figure 3.2. The fuzzy-valued feature map ranges in values from 0.0 to 1.0, therefore the height of the image boundary may be set anywhere above 1.0. We set the image boundary to 2.0.

The second step is to threshold any small fluctuations present on the low end of the fuzzy-valued feature map. The lower end being between 0 and 10% of the maximum value. The feature map is scaled between 0.0 and 1.0. Therefore, any pixel with a value between 0.0 and 0.09 is set to 0.09. The value 0.09 is a free parameter that can be altered. However, we have found this to be an effective threshold value. The reason we perform this thresholding is to speed up the processing in the later stages of the algorithm by eliminating those catchment basins that are too shallow to represent meaningful segments. A 1D example of this initial thresholding is shown in Figure 3.3. There are two types of minima present in a fuzzy-valued feature map: single-pixel minima and multiple-pixel, flat minima. Single-pixel minima are pixels that are lower in value than all of their 4-

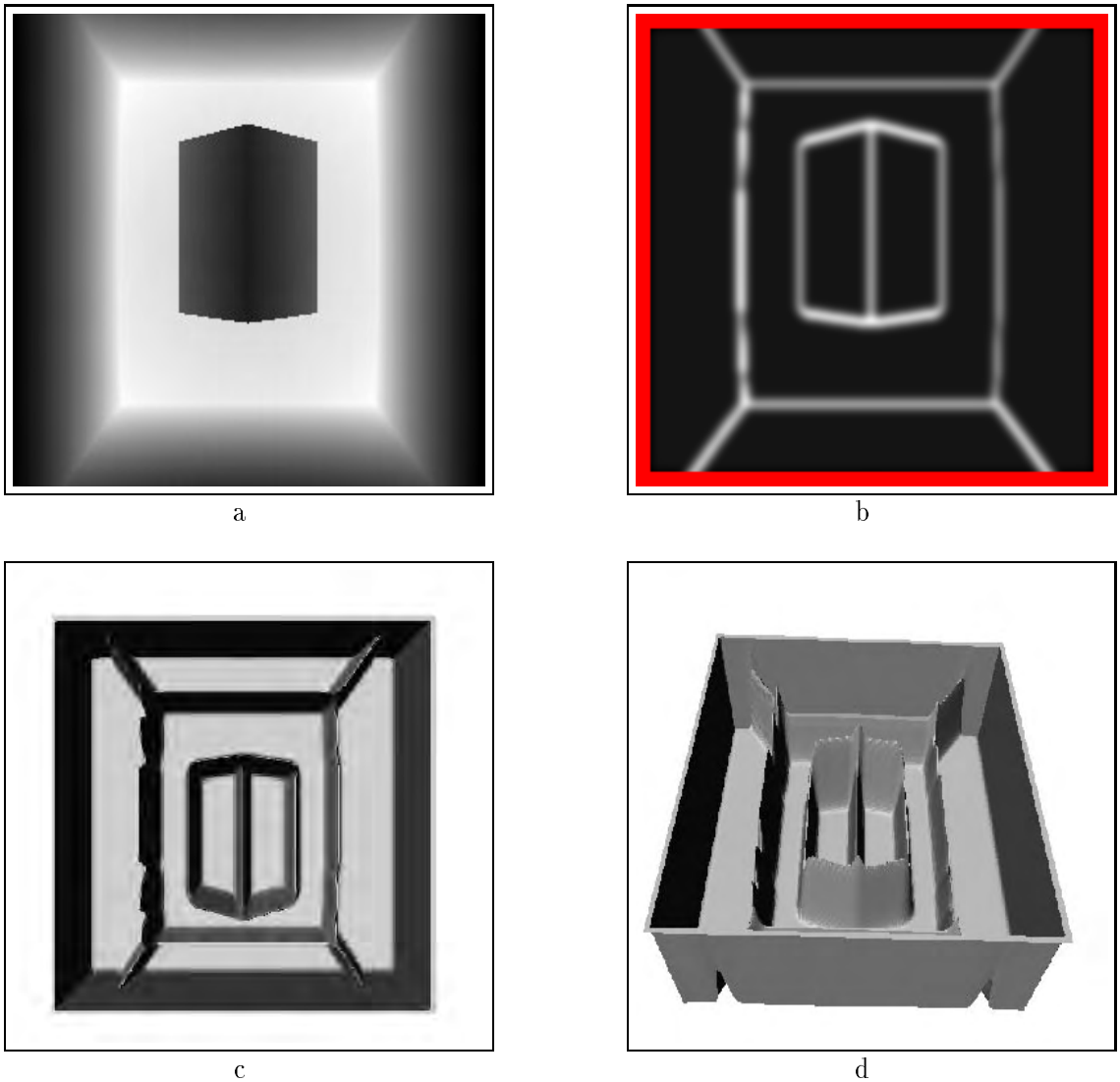


Figure 3.2: Input to watershed algorithm: (a) original range image, (b) watershed input with image boundary built around border (image boundary shown in red), (c) top view of image boundary built around image, (d) tilted view of image boundary built around image.

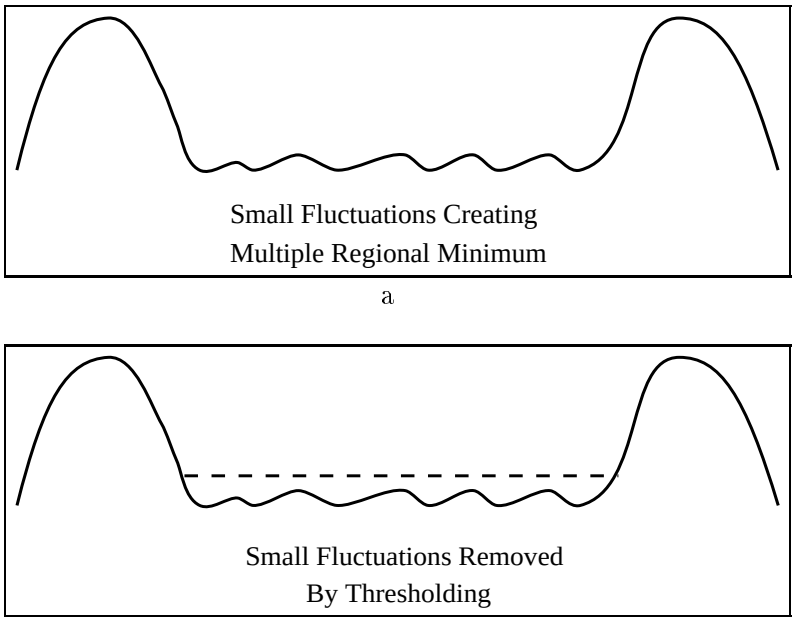


Figure 3.3: Thresholding of small fluctuations in feature maps: (a) original feature map (1D) with small fluctuations, (b) resulting feature map with small fluctuations thresholded.

neighbors. Multiple-pixel, flat minima are those regions that are flat but are surrounded by pixels that are all higher valued-pixels.

Therefore, the third step in the segmentation algorithm is to label all the single-pixel regional minima. Once the single pixel regional minima are located, they are labeled with a unique integer value beginning with 1 and increasing up to the total number of minima located.

There are three distinct types of flat regions present in a feature map. They are the flat basin, the flat plateau, and the flat maximum. The flat basin is a minimum, the flat plateau is an intermediate region whose boundaries flow down to a regional minimum, but is not a flat maximum. Finally, a flat maximum is a region of high gradient whose boundaries are greater than all their surrounding pixels. These flat maxima generally represent features in an object or scene which are sometimes more than one pixel wide.

Figure 3.4 shows a 1D example of the three types of flat regions present in a feature map. The fourth step is to identify all these flat regions, regardless of which of the three types they are, give them a distinct label, flag them as a flat region, and locate the lowest pixel surrounding the boundary of the region. The fifth step is to trace down all pixels, which

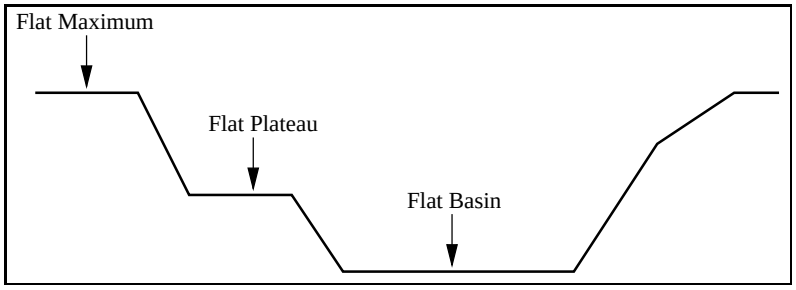


Figure 3.4: Example of three flat region types (1D).

are not members of flat regions, to their local minima. This tracing is done by following the path of steepest descent of the four neighbors of each visited pixel. Each subsequent pixel visited is marked and given the label of either the single-pixel regional minimum, flat basin, or flat plateau that it ultimately reaches. This step terminates (as it must), when it reaches a state where all four neighbors are greater than or equal to the present pixel value.

The sixth step is to trace all remaining unlabeled flat regions to their respective regional minima. This is done by beginning with the lowest pixel surrounding the boundary of the region, which has already been identified, and following its path of steepest descent to a regional minimum. Following this step, all pixels in the image are labeled with a distinct region label.

One of the purposes of this segmentation algorithm is to give a *conservative estimate* of the regions in the scene. Therefore, the seventh and eighth steps are to flag the regional

maximum flat regions and mark the boundary pixels between regions, respectively. These two types of pixels generally lie on the borders between objects in a scene and we choose to ignore them later in the surface fitting algorithm. This is because the surface fitting algorithm can be disrupted by border pixels, resulting from the mixed-pixel effect, which do not lie on the 3D surface that is to be fit to the surface, or segment. Therefore the result of the eighth step is a binary image with the region boundary pixels and the image boundary marked as 0 and the remainder of the pixels as 1. We call this image the boundary image.

Once all the segments in the region have been marked, the *depth* of each region, or watershed, is computed. This is done be computing the difference between the largest and smallest valued pixel in that watershed. Then, in the ninth step, the watersheds are thresholded according to their depths, as shown in Figure 3.5. The boundaries of watersheds, which are too shallow as specified by a preset threshold, are broken. This thresholding results in a binary image in the same form as that resulting from step eight of the algorithm. The appendages of this binary image, resulting from the boundary breaking, are then removed.

The tenth and last step in the segmentation process is the final region labeling step. We begin with a final binary image with the boundaries of the regions marked. We take this image and sequentially label all the individual regions bounded by the marked edges with a unique integer value. The final result of this step is shown in Figure 3.6 including the original image, the input to the watershed, the final binary image, and the final segmented image including the image boundary around the border of the image. Finally, we point out some of the differences between the algorithm we have implemented and the

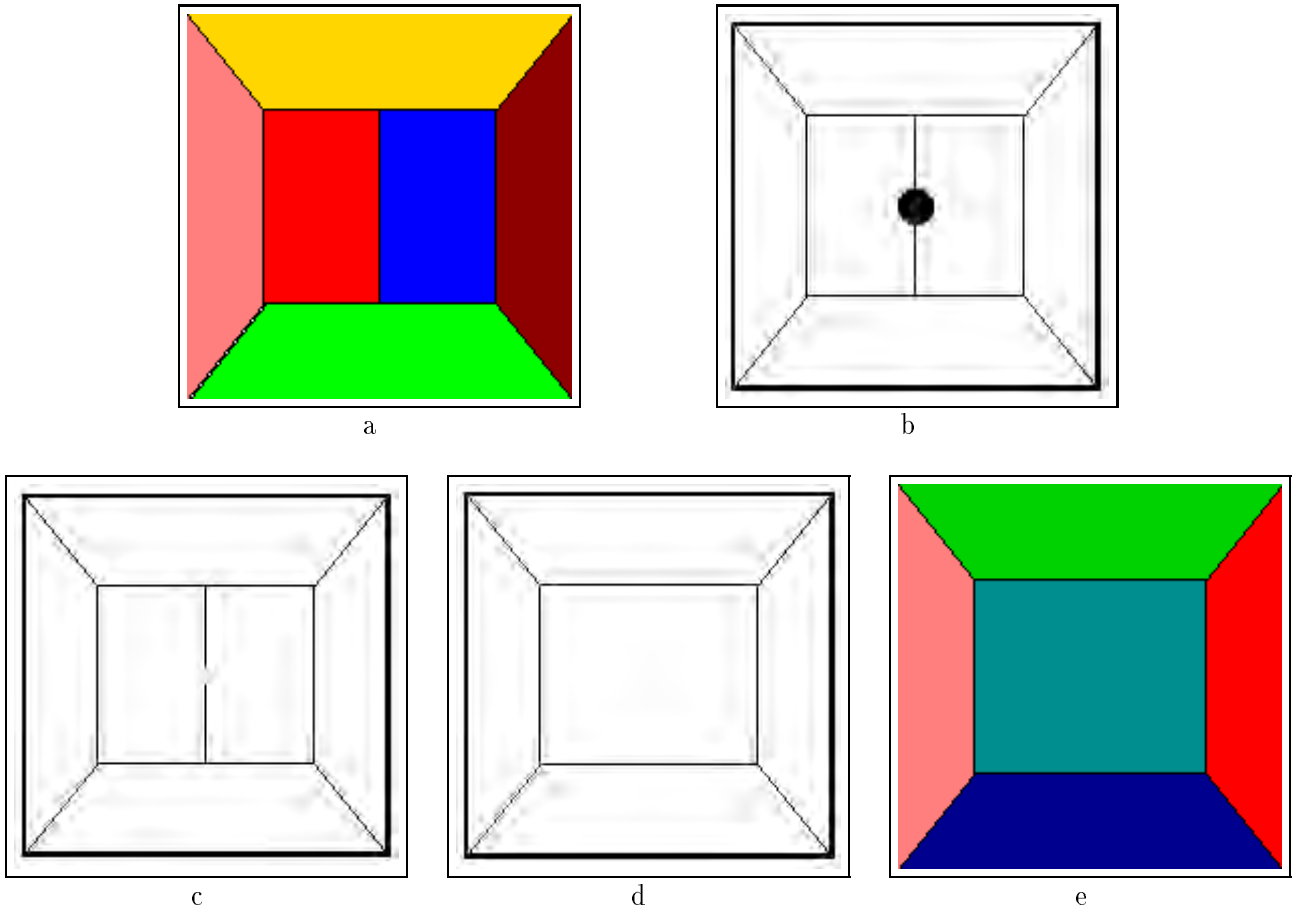


Figure 3.5: Example of watershed depth thresholding: (a) originally segmentation (synthetic) before watershed depth thresholding, (b) boundaries of the segments marked with filled circle flagging boundary pixels which are lower than the watershed depth threshold, (c) boundary of regions thresholded, or broken, (d) result of pruning done on (c), (e) final segmentation after watershed depth thresholding.

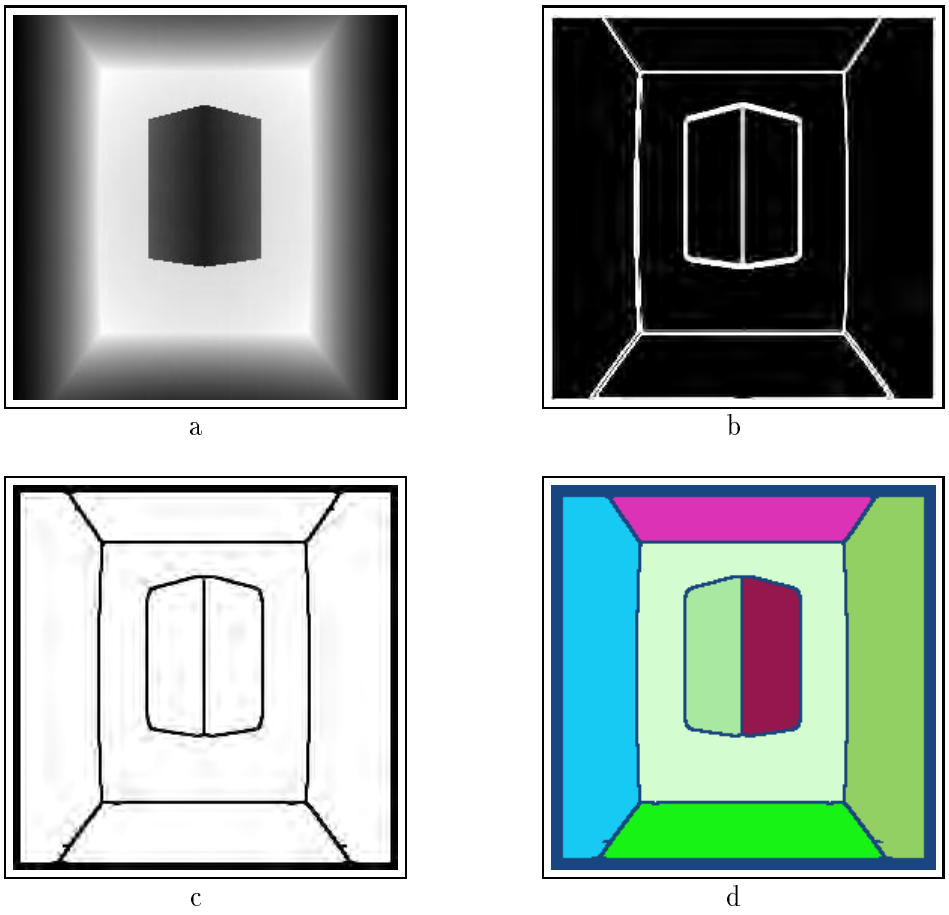


Figure 3.6: Final watershed results: (a) original range image, (b) input to watershed algorithm [fusion of gradient magnitude of range with gradient magnitude of surface normal], (c) final binary image of marked boundaries, (d) final image segmentation with edge regions and image boundary marked separate from all distinct regions.

algorithm proposed by Baccar [42]. Our algorithm accepts fuzzy-valued inputs, Baccar's algorithm accepts integer-valued input edge-maps. The two types of inputs accepted by each algorithm gives rise to a fundamental difference in the two algorithms. By accepting only integer-valued inputs, Baccar's algorithm builds catchment basins from the bottom up increasing the *water level* in integer increments. Our algorithm develops catchment basins and regions by allowing *water* to flow down to regional minima forming the catchment basins from the top down. We also use the minimum watershed depth as a saliency measure.

CHAPTER 4

Surface Fitting

Up to this point we have a segmented scene containing the three-dimensional location, relative to the laser origin, of each image pixel in each region. The next step is to fit surfaces to those image segments. We have chosen to fit 3D planes, spheres, and cylinders to those patches, because these three geometric primitives are fairly easy to deal with mathematically and are able to accurately describe a variety of the objects in the scenes with which we are concerned. Also, the techniques we are developing can be generalized to other types of geometric primitives. Each of these three geometric primitives have certain parameters which determines their shapes and positions in 3D. These parameters must be chosen initially to minimize the overall error.

Given a segment in an image including 3D Cartesian coordinates of its pixels, our strategy is to autonomously determine if the segment is a plane, sphere, or cylinder, compute an initial estimate of its parameters, and refine these parameters by minimizing the mean-squared error. This resulting minimization problem is non-linear and we have chosen to solve it using a variation of the Levenberg-Marquardt (L-M) method. Initialization of the parameters is crucial for the minimization process to converge to an acceptable solution. Once a fitted surface is computed by minimizing this mean squared error, we create a 3D model of the surface and combine it with all the models of the fitted surfaces to yield a reconstruction of the original scene.

4.1 Segment Classification

Given a segment in the original scene we must autonomously distinguish between planes, spheres, and cylinders. To distinguish between the three types of primitives we first implement a second-order polynomial fit to the data, construct the Shape Matrix according to the fit, and analyze the eigenvalues of the Shape Matrix to determine which of the three geometric primitives, if any, best models the segment.

To accomplish this we first assume a two-and-a-half-dimensional approach to the data, i.e. $z = f(x, y)$ and perform a least-squares fit of a second order polynomial to the data. We call this original data $\mathbf{X}$ and mathematically define it by the following equations.

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \\ y_1 & y_2 & \dots & y_n \\ z_1 & z_2 & \dots & z_n \end{bmatrix} = \begin{bmatrix} \vec{X_1}, & \vec{X_2}, & \dots, & \vec{X_n} \end{bmatrix} \tag{4.1}$$

Where n is the number of data points in the segment. The result of this fit yields the coefficients from which we construct First and Second Fundamental Forms. The second-order polynomial we compute is given by

$$z = C + C_x x + C_y y + C_{xx} x^2 + C_{xy} x y + C_{yy} y^2. \tag{4.2}$$

The coefficients may be represented in vector form as

$$\vec{C} = \begin{bmatrix} C_x \\ C_y \\ C_{xx} \\ C_{yy} \\ C_{xy} \end{bmatrix}. \tag{4.3}$$

This linear least-squares fit is achieved by a singular value decomposition [80]. We classify the surface based on the differential properties at the middle of the patch. The derivatives of $f(x, y)$ are

$$f_x = 2C_{xx}\mathcal{M}[\vec{x_i}] + C_{xy} * \mathcal{M}[\vec{y_i}] + C_x \tag{4.4}$$

$$f_y = 2C_{yy}\mathcal{M}[\vec{y_i}] + C_{xy}\mathcal{M}[\vec{x_i}] + C_y \tag{4.5}$$

$$f_{xx} = 2C_{xx} \tag{4.6}$$

$$f_{yy} = 2C_{yy} \tag{4.7}$$

$$f_{xy} = C_{xy} \tag{4.8}$$

Where $\mathcal{M}[\cdot]$ is the median value of a set of data. From the previous five equations, we construct the First Fundamental Form, **I**, and the Second Fundamental Form, **II**, given

by the following two equations (as in [21].

$$\mathbf{I} = \begin{bmatrix} (1 + f_x^2) & (f_x * f_y) \\ (f_x * f_y) & (1 + f_y^2) \end{bmatrix} \tag{4.9}$$

$$\mathbf{II} = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{xy} & f_{yy} \end{bmatrix} * \frac{1}{\sqrt{(1 + f_x^2 + f_y^2)}} \tag{4.10}$$

Now we construct the Shape Matrix β from $\mathbf{I}$ and $\mathbf{II}$ by the following equation.

$$\beta = \mathbf{I}^{-1}\mathbf{II} \tag{4.11}$$

Finally we compute the eigenvalues of β to determine whether the segment is a plane, cylinder, or sphere. The eigenvalues of β are the *principle curvatures* of the surface. The principle curvatures are invariant expressions of the surface curvature at that point.

For a plane, both of the eigenvalues of β will be close to, or identically 0. Therefore, we set a threshold for the eigenvalues, which we call c, close to zero. If both of the eigenvalues are less than or equal to c, then we classify the segment as a plane. For a cylinder, one of the eigenvalues of β will be close to, or identically 0, and the other eigenvalue will be much greater than zero. Therefore, if one of the eigenvalues is less than c or equal to c, and the other is greater than c, we classify the segment as a cylinder. For a sphere, both of the eigenvalues of β will be much greater than zero and almost equal to each other. Therefore, if both of the eigenvalues are much greater than c and are relatively close to each other, we classify the segment as a sphere. Given that e_1 and e_2 are the eigenvalues of β, the classification rules are:

- If $((e_1 \leq c)\ \&\ (e_2 \gg c)) \parallel ((e_2 \leq c)\ \&\ (e_1 \gg c)) \Rightarrow$ CYLINDER,
- If $(e_1 \& e_2) \leq c \Rightarrow$ PLANE,
- If $((e_1\ \&\ e_2) \gg c)\ \&\ (e_1 \simeq e_2) \Rightarrow$ SPHERE.

Once the segment type is determined, the initial rotation matrix and initial parameters for each segment may be computed.

4.2 Computation of Initial Rotation

For the surfaces we fit, the minimization process takes place in what we call the *model space* to reduce the number of parameters in the minimization process. This model space is a 3D Cartesian coordinate system where the *special axis* is the z axis. For each geometric shape, there will be certain requirements associated with this special axis that the data must meet. We will discuss these properties for each of the shapes. We construct an initial rotation matrix, $\mathbf{R}$, to transform the original data into such a model space. Thus, the data in the model space is represented by

$$\mathbf{X}' = \mathbf{R}\mathbf{X}. \tag{4.12}$$

This initial rotation matrix is computed once and is not included in the minimization process. If we minimize the squared error of the surface fitting without an initial rotation matrix and allow the rotation angles to be minimized in the fitting process, the minimization process will have difficulty near singularities in the derivatives in the rotation matrix. Therefore, only small rotation angle may be included as free parameters and have

the minimization process converge to a suitable solution. Transforming the data to the model space ensures that only small adjustments to the angles will be needed to achieve the optimal fit. In the following three sections, we will explain how we compute $\mathbf{R}$ for each type of surface.

4.2.1 Initial Rotation for Cylinders

Once a segment is classified as a cylinder, we construct an initial rotation matrix, $\mathbf{R}$, that rotates the data so that the rulings of the cylinder in the model space are close to, or exactly parallel to the z axis. For a cylinder, we compute $\mathbf{R}$ in the following manner.

We already have computed the Shape Matrix, β, based on the initial second-order polynomial fit to the original data. We have also computed the eigenvalues and eigenvectors of β. Let $\mathbf{V}$ be the eigenvectors of β and define γ by

$$\gamma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ f_x & f_y \end{bmatrix}. \tag{4.13}$$

Where γ is matrix that takes us from the parameterized (x, y) space to the associated tangent vectors in 3D. Then a 3×2 matrix we will call $\mathbf{B}$ is computed by the following equation.

$$\mathbf{B} = [\gamma] * [\mathbf{V}] \tag{4.14}$$

We normalize the two columns of $\mathbf{B}$ by their respective magnitudes. Then we take the cross product of the normalized columns of $\mathbf{B}$ to create a create the third column of $\mathbf{B}$,

now a 3×3 rotation matrix, from which we will construct the initial rotation matrix $\mathbf{R}$.

The smallest eigenvalue of β is associated with the eigenvector that points in the direction of the major axis along which the cylinder in the original data space lies. Therefore, we set the third column of $\mathbf{R}$ equal to the column of $\mathbf{B}$ that is associated with the eigenvector of the smallest eigenvalue of $\mathbf{V}$. Then, the second column of $\mathbf{R}$ is set equal to the column of $\mathbf{B}$ associated with the eigenvector of the largest eigenvalue of $\mathbf{V}$. Finally the first column of $\mathbf{R}$ is set equal to the third column of $\mathbf{B}$. Now the data has been rotated into the model space, so that $\mathbf{X}'$ is a cylinder whose rulings lie almost or exactly parallel to the z axis in the model space.

4.2.2 Initial Rotation for Planes

If the segment is classified as a plane, we construct the initial rotation matrix, $\mathbf{R}$, to rotate the data into the model space so that the plane lies close to, or exactly perpendicular to the z axis. For a plane, we construct $\mathbf{R}$ in the following manner.

We compute the covariance matrix, $\mathbf{C}$, of $\mathbf{X}$. Then we compute the eigenvalues and eigenvectors of the 3×3 matrix $\mathbf{C}$. We know that, for a plane, the eigenvector associated with the smallest eigenvalue of $\mathbf{C}$ will point in the direction normal to the plane in the original data space. Therefore we set the third column of $\mathbf{R}$ equal to the eigenvector associated with the smallest eigenvalue of $\mathbf{C}$. The second column of $\mathbf{R}$ is set equal to the eigenvector associated with the largest eigenvalue of $\mathbf{C}$. The first column of $\mathbf{R}$ is set equal to the remaining eigenvalue of $\mathbf{C}$. Now the original data can be rotated into the model space so that $\mathbf{X}'$ is a plane that lies close to, or exactly perpendicular to the z axis.

4.2.3 Initial Rotation for Spheres

If the segment is determined to be a sphere, then we set the initial rotation matrix, $\mathbf{R}$, equal to the Identity matrix, $\mathbf{I}$. This is because spheres are rotionally invariant. Therefore, in the case of a sphere, the data in the model space, $\mathbf{X}'$, is equal to the original data, $\mathbf{X}$. Therefore,

$$\mathbf{X}^{'} = \mathbf{X}. \tag{4.15}$$

4.3 Computation of Initial Parameters

At this point we have an initial rotation matrix, obtained through heuristic methods, that rotates the original data into the model space. However, this method involves no error minimization, and is simply an approximation of the correct rotation. Therefore, in order to minimize the error of the solution, we must introduce perturbation angles, which are relatively small, into the minimization process. The models we are using are rotationally symmetric about the z axis in the model space. Therefore, these small perturbation rotations in the model space must take place around the x and y axes. These rotation matrices are denoted by $\mathbf{R}_\theta$ and $\mathbf{R}_\phi$, respectively. With θ being the perturbation rotation angle around the x axis and ϕ being the perturbation rotation angle around the y axis. These perturbation rotations are given by the following equations [13].

$$\mathbf{R}_\theta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) \\ 0 & -\sin(\theta) & \cos(\theta) \end{bmatrix} \tag{4.16}$$

$$\mathbf{R}_\phi = \begin{bmatrix} \cos(\phi) & 0 & -\sin(\phi) \\ 0 & 1 & 0 \\ \sin(\phi) & 0 & \cos(\phi) \end{bmatrix}. \tag{4.17}$$

Now these perturbation rotations may be included in the minimization process to minimize the error of the surface fit. We also need the derivatives of these perturbation rotations in later calculations. These derivatives are given by

$$\frac{\partial \mathbf{R}_\theta}{\partial \theta} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\sin(\theta) & \cos(\theta) \\ 0 & -\cos(\theta) & -\sin(\theta) \end{bmatrix} \tag{4.18}$$

$$\frac{\partial \mathbf{R}_\phi}{\partial \phi} = \begin{bmatrix} -\sin(\phi) & 0 & -\cos(\phi) \\ 0 & 0 & 0 \\ \cos(\phi) & 0 & -\sin(\phi) \end{bmatrix}. \tag{4.19}$$

The following sections will explain in more detail the remaining parameters needed for each specific surface.

4.3.1 Initialization of Parameters for Cylinders

There are five parameters included in the minimization process for a cylinder. The first two are the perturbation rotation angles. We assume that the initial rotation places the data in the model space, for a cylinder, so that the rulings of the cylinder are relatively

close to parallel to the z axis. Therefore, we set the perturbation rotation angles initially to 0, knowing that these angles should not be much greater than zero for the correct solution. The third parameter is the the initial radius of the cylinder, C_r. We assume that the cylinder is circular, and is initially close to , or perfectly symmetric about the z axis in the model space. Therefore, we approximate the initial radius by the following equation.

$$C_r = \frac{\max_i(x_i^{'}) - \min_i(x_i^{'})}{2} \qquad \text{for } i = 1, 2, \ \ldots \ , n \tag{4.20}$$

The last two parameters are the center of the cylinder along the x axis and the y axis. These center locations are denoted by $x_o^{'}$ and $y_o^{'}$ respectively and are approximated by the following equations.

$$x_o^{'} = \mathcal{M}[x_i^{'}] \tag{4.21}$$

$$y_o^{'} = \mathcal{M}[y_i^{'}] \tag{4.22}$$

4.3.2 Initialization of Parameters for Planes

For a plane there are only three parameters used in the minimization of the error of the fit. The first two are the perturbation rotation angles. As with the case of a cylinder, these two angles are initially set to 0 for the same basic reasons presented for a cylinder. Then the last parameter to be set is the initial value for z in the model space, which we call $z_o^{'}$. Because the initial rotation positions the data close to perpendicular to the z axis in the model space, we can approximate $z_o^{'}$ by the following equation.

$$z_o^{'} = \mathcal{M}[z_i^{'}] \qquad \text{for } i = 1, 2, \ \ldots \ , n \tag{4.23}$$

4.3.3 Initialization of Parameters for Spheres

Because spheres are rotationally invariant, we do not need an initial rotation matrix $\mathbf{R}$, therefore we do not need any perturbation rotation matrices and angles. There are four parameters to be determined for a sphere. They are the radius of the sphere, S_r, and the center of the sphere $(x_o^{'}, y_o^{'}, z_o^{'})$. These parameters are approximated using the following equations.

$$S_r = \frac{\max_i[x_i^{'}] - \min_i[x_i^{'}]}{2} \qquad \text{for } i = 1, 2, \ldots, n \tag{4.24}$$

$$x_o^{'} = \mathcal{M}[x_i'] \qquad \text{for } i = 1, 2, \ldots, n \tag{4.25}$$

$$y_o^{'} = \mathcal{M}[y_i'] \qquad \text{for } i = 1, 2, \ldots, n \tag{4.26}$$

$$z_o^{'} = \max_i[z_i'] \qquad \text{for } i = 1, 2, \ldots, n \tag{4.27}$$

The segment type has been identified and the associated parameters have been initialized. The minimization process refines these parameters to fit the data by minimizing the squared error. This nonlinear minimization is carried out using variation of the L-M method [80].

4.4 Levenberg-Marquardt Minimization Method

To minimize the error in our surface fitting algorithm, we will implement a variation of the Levenberg Marquardt minimization method. The L-M method set forth in [80] begin

in the following manner. The model to be fitted is

$$y = y(x; \vec{a}). \tag{4.28}$$

The merit function χ^2 is:

$$\chi^2(\vec{a}) = \sum_{i=1}^{n} \left[\frac{y_i - y(x_i; \vec{a})}{\sigma_i} \right]^2 \tag{4.29}$$

In this formulation, we know that at the minimum of χ^2, the gradient of this merit function with respect to the parameters $\vec{a}$ will be zero. This gradient is given

$$\frac{\partial \chi^2}{\partial a_k} = -2 \sum_{i=1}^{N} \frac{[y_i - y(x_i; \vec{a})]}{\sigma_i^2} \frac{\partial y(x_i; \vec{a})}{\partial a_k} \qquad k = 1, 2, ..., M. \tag{4.30}$$

Another partial derivative is taken to yield

$$\frac{\partial^2 \chi^2}{\partial a_k \partial a_l} = 2 \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \left[\frac{\partial y(x_i; \vec{a})]}{\partial a_k} \frac{\partial y(x_i; \vec{a})}{\partial a_l} - [y_i - y(x_i; \vec{a})] \frac{\partial^2 y(x_i; \vec{a})}{\partial a_l \partial a_k} \right]. \tag{4.31}$$

The components are scaled and used in the form

$$\beta_k \equiv -\frac{1}{2} \frac{\partial \chi^2}{\partial a_k} \tag{4.32}$$

and

$$\alpha_{kl} \equiv \frac{1}{2} \frac{\partial^2 \chi^2}{\partial a_k \partial a_l}. \tag{4.33}$$

Which, according to [80], now gives a set of linear equations to solve that will minimize the merit function with respect to the given parameter set $\vec{a}$. That set of linear equations

is

$$\sum_{l=1}^{M} \alpha_{kl} \delta a_l = \beta_k, \tag{4.34}$$

where

$$\delta a_l = C \times \beta_l. \tag{4.35}$$

Where C is some constant value. In this formulation, the second derivative is included. However, the inclusion of this term can introduce instabilities if the model does not fit well or is affected by outlier points. Therefore the formulation for the Hessian matrix, as recommended in [80], is reduced to

$$\alpha_{kl} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \left[\frac{\partial y(x_i; \vec{a})}{\partial a_k} \frac{\partial y(x_i; \vec{a})}{\partial a_l} \right]. \tag{4.36}$$

Marquardt developed a formulation of this minimization process that varies smoothly between the inverse-Hessian method and the steepest descent method. He set the scale of the constant term in equation 4.35 by defining δa_l as follows:

$$\delta a_l = \frac{1}{\lambda \alpha_{ll}} \beta_l \tag{4.37}$$

$$\lambda \alpha_{ll} \delta a_l = \beta_l \tag{4.38}$$

He then extended the Hessian matrix by scaling the diagonal of that matrix as follows:

$$\alpha'_{jk} = \begin{cases} \alpha_{jk}(1 + \lambda) & \text{for } j = k \\ \alpha_{jk} & \text{for } j \neq k \end{cases} \tag{4.39}$$

Thus the system of equations is redefined with the newly scaled Hessian matrix:

$$\sum_{l=1}^{M} \alpha_{kl}^{'} \delta a_l = \beta_k \tag{4.40}$$

Given the new system of equations, here are the steps of the method we used as recommended by Marquardt:

- Compute $\chi^2(\vec{a})$
- Pick a modest value for λ ($\lambda = 0.001$)
- (*) Solve the linear equations for $\delta(\vec{a})$ and evaluate $\chi^2(\vec{a}) + \delta(\vec{a})$
- If $\chi^2(\vec{a}) + \delta\vec{a}) \geq \chi^2(\vec{a})$, increase λ by a factor of 10 and go back to (*)
- If $\chi^2(\vec{a}) + \delta\vec{a}) \leq \chi^2(\vec{a})$, decrease λ by a factor of 10, update the trial solution $\vec{a} \leftarrow \vec{a} + \delta\vec{a}$, and go back to (*)

The way we implement this method varies slightly from the exact form set forth previously. In our situation, the merit function χ^2 is the Euclidean distance D_i from a given point in the data to the *nearest point* on the surface which we are fitting. Therefore we define the distance to the model by

$$\chi^2 = \frac{D_i^2}{\sigma_i^2} \tag{4.41}$$

In Equation 4.41, as with all equations including σ_i, we let $\sigma_i = 1$. However, that variable may be changed to represent a confidence of how true the point is to the actual data. A low confidence would be given to a noisy or outlying point, and a high confidence would be given to points true to the actual scene. This is an area for future study.

We have also implemented an outlier removal step in this process. This step occurs immediately after the initial parameter and residual calculation. Here we compute the average error per pixel and remove any pixels from the data set which have an error of three times the average error per pixel.

4.4.1 Volumetric Models

We fit volumetric models by minimizing the squared distance from the data to the surface model. That is, the error function term we is the distance from the data point to a point on the fitted surface. This distance is a function of a the three-dimensional coordinates of the data in the model space, $\mathbf{X}'$ and the set of free parameters for each surface, $\vec{a}$. Therefore, the function to be minimized is

$$\sum D^2(\mathbf{X}', \vec{a}). \tag{4.42}$$

In the model space, the error minimization takes place with the updated approximation of the fitted data at a given iteration to be

$$\mathbf{X}'' = \mathbf{R}_\delta(\theta, \phi)\mathbf{X}' - \vec{T}. \tag{4.43}$$

Where $\mathbf{R}_\delta(\theta, \phi)$ is

$$\mathbf{R}_\delta(\theta, \phi) = (\mathbf{R}_\theta \mathbf{R}_\phi)^{-1} \tag{4.44}$$

and

$$\vec{T} = \begin{bmatrix} T_{(x)} \\ T_{(y)} \\ T_{(z)} \end{bmatrix}. \tag{4.45}$$

In general, double prime refers to coordinates in the model space with perturbation and translation.

As shown in the previous section on the L-M method, there are derivatives of the minimized function to be taken with respect to the free parameters, $\vec{a}$. These derivatives will be given for each of the three surface types in the following three subsections. From these derivatives and the distance, D_i, from $\vec{X}_i$ to the nearest point on the model, we compute the merit function, D^2 by

$$D^2 = \sum_{i=1}^{n} D_i^2 \qquad \text{for } i = 1, 2, \ldots, n. \tag{4.46}$$

These subsections will also explain other details of the functions to be minimized and how the final surface parameters are evaluated so that surface models can be rendered in a 3D graphics system.

Fitting of Planes

If (x_i, y_i, z_i) is a point on a plane that passes through the point (x_o, y_o, z_o), with normal vector

$\vec{A} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix}$, the following equation must be satisfied:

$$A_1(x_i - x_o) + A_2(y_i - y_o) + A_3(z_i - z_o) = 0. \qquad \text{for } i = 1, 2, \ldots, n \tag{4.47}$$

Because the data in the model space lies close to or exactly perpendicular to the z axis, we let $\vec{A} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$. Therefore, the parameters to be used in the minimization of a fitted plane are

$$\vec{a} = \begin{bmatrix} z_o' & \theta & \phi \end{bmatrix}^T \tag{4.48}$$

and the distance, D_i, from a given data point to a point on the fitted plane is given by:

$$D_i = z_i'' - z_o' \qquad \text{for } i = 1, 2, \ldots, n. \tag{4.49}$$

Therefore, the translation

$$\vec{T} = \begin{bmatrix} 0 \\ 0 \\ z_o' \end{bmatrix}. \tag{4.50}$$

The derivatives of the distance, D_i, with respect to the three parameters are given by the following three equations.

$$\frac{\partial D_i}{\partial z_o'} = -1 \tag{4.51}$$

$$\frac{\partial D_i}{\partial \theta} = -\cos(\phi)\sin(\theta)z_i^{'} + \cos(\phi)\cos(\theta)y_i^{'} \tag{4.52}$$

$$\frac{\partial D_i}{\partial \phi} = -\cos(\phi)x_i^{'} - \sin(\phi)\cos(theta)z_i^{'} - \sin(\phi)\sin(\alpha)y_i^{'} \tag{4.53}$$

Once the parameters are minimized and a solution for a fitted plane are computed, the data must be transformed from the model space back into the original data space. One of the parameters yielded in the output of the minimization process is the final z_o value for the fitted plane. Therefore, we replace all the $z_i^{'}$ values with the final z_o value to force the data in the model space to lie parallel to the fitted plane in the model space. This new set of data is denoted by $\mathbf{X}_f^{'}$, and is expressed by

$$\mathbf{X}_f^{'} = \begin{bmatrix} x_1^{'} & x_2^{'} & \dots & x_n^{'} \\ y_1^{'} & y_2^{'} & \dots & y_n^{'} \\ z_o & z_o & \dots & z_o \end{bmatrix} \tag{4.54}$$

Using the final values computed for θ and ϕ, we can rotate the fitted plane from the model space to yield the fitted plane in the original data $\mathbf{X_f}$ space by the following equation.

$$\mathbf{X}_f = \mathbf{R}^{-1}\mathbf{R}_\delta{}^{-1}\mathbf{X}_f^{'} \tag{4.55}$$

Now the fitted plane in the original space, $\mathbf{X}_f$, may be rendered in a 3D graphics system.

Fitting of Cylinders

If (x_i, y_i, z_i) is a point on a circular cylinder whose rulings are parallel to the z axis, centered at and whose radius is C_r, the following equation must be satisfied:

$$(x_i - x_o)^2 + (y_i - y_o)^2 = C_r^2. \tag{4.56}$$

Therefore the parameters to be used in the minimization of a fitted cylinder are

$$\vec{a} = \begin{bmatrix} x_o^{'} & y_o^{'} & C_r & \theta & \phi \end{bmatrix}^T \tag{4.57}$$

and the distance, D_i, from a given data point to the nearest point on the fitted cylinder is given by:

$$D_i = \sqrt{d_i} - C_r. \tag{4.58}$$

Where

$$d_i = (x_i^{''})^2 + (y_i^{''})^2 \tag{4.59}$$

and the translation

$$\vec{T} = \begin{bmatrix} x_o^{'} \\ y_o^{'} \\ 0 \end{bmatrix}. \tag{4.60}$$

The derivatives of the distance, D_i, with respect to the parameters, $\vec{a}$ are given by the following equations.

$$\frac{\partial D_i}{\partial x_o^{'}} = \frac{1}{2\sqrt{d_i}} \frac{\partial d_i}{\partial x_o^{'}} = \frac{-x_i^{''}}{\sqrt{d_i}} \tag{4.61}$$

$$\frac{\partial D_i}{\partial y_o^{'}} = \frac{1}{2\sqrt{d_i}} \frac{\partial d_i}{\partial y_o^{'}} = \frac{-y_i^{''}}{\sqrt{d_i}} \tag{4.62}$$

$$\frac{\partial D_i}{\partial C_r} = -1 \tag{4.63}$$

$$\frac{\partial D_i}{\partial \theta} = \frac{1}{2\sqrt{d_i}} \frac{\partial d_i}{\partial \theta} = \frac{1}{\sqrt{d_i}} \begin{bmatrix} x_i^{''} \\ y_i^{''} \\ 0 \end{bmatrix}^T \frac{\partial \mathbf{R}_\theta}{\partial \theta} \mathbf{R}_\phi \begin{bmatrix} x_i^{'} \\ y_i^{'} \\ z_i^{'} \end{bmatrix}^T \tag{4.64}$$

$$\frac{\partial D_i}{\partial \phi} = \frac{1}{2\sqrt{d_i}} \frac{\partial d_i}{\partial \phi} = \frac{1}{\sqrt{d_i}} \begin{bmatrix} x_i^{''} \\ y_i^{''} \\ 0 \end{bmatrix}^T \mathbf{R}_\theta \frac{\partial \mathbf{R}_\phi}{\partial \phi} \begin{bmatrix} x_i^{'} \\ y_i^{'} \\ z_i^{'} \end{bmatrix}^T \tag{4.65}$$

Once the parameters are minimized and a solution for a fitted cylinder are computed, the center of the cylinder in the model space $(x_o^{'}, y_o^{'}, z_o^{'})$ must be found. $(x_o^{'}, y_o^{'})$ have already been computed in the minimization process. $z_o^{'}$ is computed by the following equation.

$$z_o^{'} = \mathcal{M}[z_i^{'}] \qquad \text{for } i = 1, 2, \ \ldots \ , n \tag{4.66}$$

The height of the cylinder C_h must also be computed. We approximate the height by the following equation.

$$C_h = \max_i(z_i^{'}) - \min_i(z_i^{'}) \qquad \text{for } i = 1, 2, \ \ldots \ , n \tag{4.67}$$

The center of the cylinder in the model space may be expressed in vector form as $\vec{C_c'}$ by

$$\vec{C_c'} = \begin{bmatrix} x_o' & y_o' & z_o' \end{bmatrix}^T . \tag{4.68}$$

To obtain the center of the cylinder in the original data space, $\vec{C_c}$, we use the following equation.

$$\vec{C_c} = \mathbf{R}^{-1}\mathbf{R}_\delta^{-1}\vec{C_c'} \tag{4.69}$$

Now all the size, translation, and rotation parameters for the cylinder in the original space have been computed to create a model of the cylinder for 3D visualization.

Fitting of Spheres

If (x_i, y_i, z_i) is a point on a sphere whose center is located at (x_o, y_o, z_o) and whose radius is S_r, the following equation must be satisfied.

$$(x_i - x_o)^2 + (y_i - y_o)^2 + (z_i - z_o)^2 = S_r^2 \tag{4.70}$$

Because spheres are rotationally invariant, $\mathbf{X}'' = \mathbf{X} - \vec{T}$. Therefore, the parameters to be used in the minimization of a fitted sphere are:

$$\vec{a} = \begin{bmatrix} x_o' & y_o' & z_o' & S_r \end{bmatrix}^T \tag{4.71}$$

and the distance, D_i, from a given data point to the nearest point on the fitted cylinder is given by:

$$D_i = \sqrt{d_i} - S_r. \tag{4.72}$$

Where

$$d_i = (x_i^{''})^2 + (y_i^{''})^2 + (y_i^{''})^2 \tag{4.73}$$

and the translation

$$\vec{T} = \begin{bmatrix} x_o^{'} \\ y_o^{'} \\ z_o^{'} \end{bmatrix}. \tag{4.74}$$

The derivatives of the distance, D_i, with respect to the parameters, $\vec{a}$ are given by the following equations.

$$\frac{\partial D_i}{\partial x_o^{'}} = \frac{1}{2\sqrt{d_i}} \frac{\partial d_i}{\partial x_o^{'}} = \frac{-x_i^{''}}{\sqrt{d_i}} \tag{4.75}$$

$$\frac{\partial D_i}{\partial y_o^{'}} = \frac{1}{2\sqrt{d_i}} \frac{\partial d_i}{\partial y_o^{'}} = \frac{-y_i^{''}}{\sqrt{d_i}} \tag{4.76}$$

$$\frac{\partial D_i}{\partial z_o^{'}} = \frac{1}{2\sqrt{d_i}} \frac{\partial d_i}{\partial z_o^{'}} = \frac{-z_i^{''}}{\sqrt{d_i}} \tag{4.77}$$

$$\frac{\partial D_i}{\partial S_r} = -1 \tag{4.78}$$

Once the parameters are minimized and a solution for the fitted sphere are computed, the center and radius of the sphere, computed in the minimization process are used to create a model of the surface for 3D visualization.

4.5 Results of Surface Fitting

In order to test the surface fitting algorithm we have developed, implementation is performed on both real and synthetic data sets. In this chapter we will show results on synthetic data, both with and without simulated noise. Results from real data will be shown in Chapter 5. First an example of noise-free synthetic data is presented. Then two noisy synthetic scenes will be reconstructed. The first noisy scene contains a cylinder inside a room, the second contains a sphere inside a room of the same size. Both of the noisy scenes contain walls, or planes. Therefore surface fitting on planes, spheres, and cylinders in the presence of noise will be demonstrated in these results.

4.5.1 Noise-Free Data

The scene shown in Figure 4.1 has a resolution of 256 x 256 pixels. It contains 11 different segments. Four of the segments include the side and back walls and the floor of a room. One segment is a large cylinder of radius 20, another is a sphere of radius 10, and the final five segments are a series of steps parallel to the back wall. The units for all the range images and results are millimeters.

The qualitative results of the surface fitting of this range data is shown is Figure 4.1. The 3D Inventor renderings of the superposition of the fitted surfaces onto the range data show the original range data in a semi-transparent color so that the fitted surfaces can be seen more clearly. The quantitative results of the surface fitting are given in Table 4.1. For each segment in the image, an identifying description of the segment, the type of surface fitted to it, the total number of pixels in that segment, the rms error per pixel,

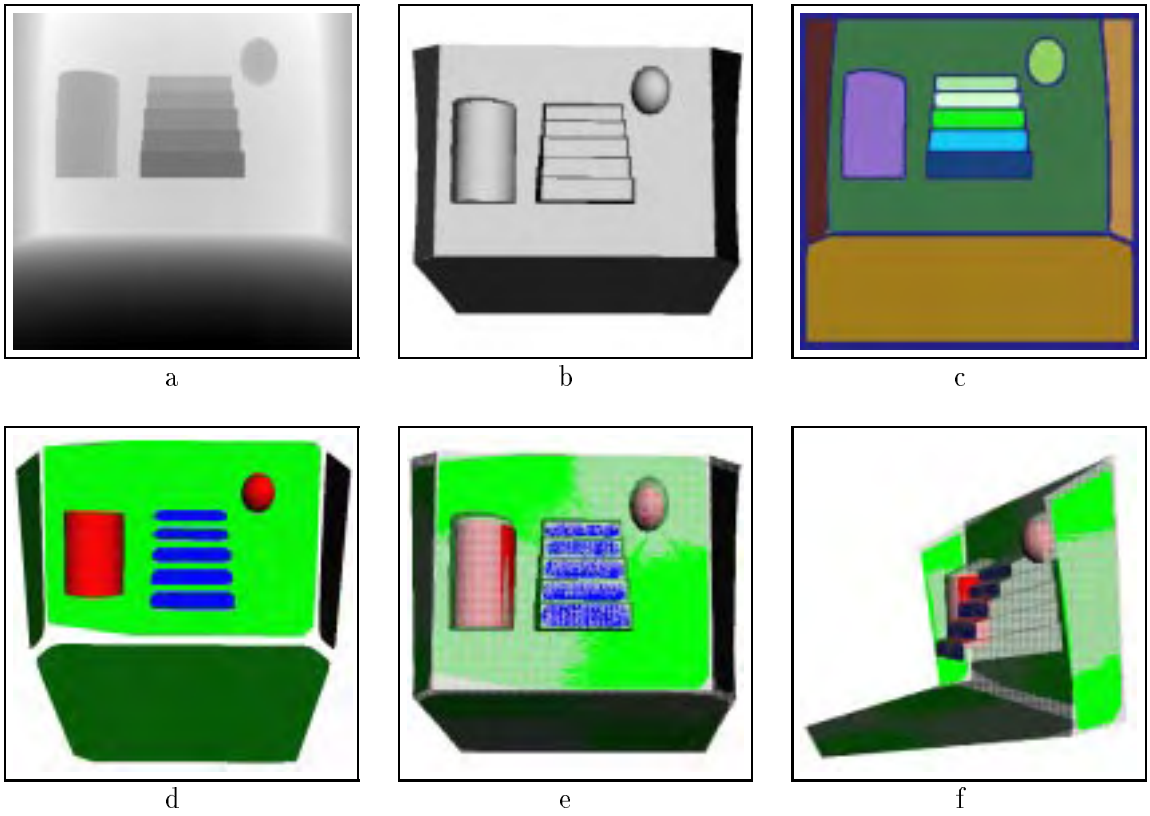


Figure 4.1: Results of system implementation on noiseless synthetic data: (a) Original range image, (b) 3D rendering of original range image, (c) Watershed segmentation of original range image, (d) 3D rendering of fitted surfaces, (e) 3D rendering of superposition of fitted surfaces onto range data (front view), (f) 3D rendering of superposition of fitted surfaces onto range data (side view).

Table 4.1: Results of surface fitting for noise-less synthetic data

Description	Surface	# of Pixels	RMS Error/Pixel (mm)	Iterations
Left Wall	Plane	2748	3.8×10^{-3}	1
Floor	Plane	19531	3.8×10^{-3}	1
Back Wall	Plane	22714	$2.3 \times 10-3$	1
Large Cylinder	Cylinder	3498	0.36	19
1st Step (Bottom)	Plane	1380	4.1×10^{-4}	1
2nd Step	Plane	1080	3.7×10^{-4}	1
3rd Step	Plane	960	4.0×10^{-4}	1
4th Step	Plane	696	5.3×10^{-4}	1
5th Step (Top)	Plane	605	5.9×10^{-4}	1
Small Sphere	Sphere	707	0.44	10
Right Wall	Plane	2924	3.8×10^{-3}	1

and the number of iterations taken to converge to a final answer is given. Not shown in the table, but of importance, is the radii of the fitted cylinder and sphere compared to the actual radii. The fitted radius computed for the cylinder is 19.9404 mm yielding an absolute error/radius of .48%. The fitted radius computed for the sphere is 11.1650 mm yielding an absolute error/radius of 11.65%.

4.5.2 Noisy Data

For the noisy data we used, shown in Figure 4.2, the image contains a cylinder having a radius of 5 and a height of 30. The qualitative results of segmentation and surface fitting are shown in Figure 4.2. The quantitative results of the surface fitting are given in Table 4.2. This image has been corrupted with replacement and additive noise, showing how the segmentation and surface fitting steps of the system react in the present of noise. Both Figure 4.2 and Table 4.2 show the surface fitting results for the planes and the cylinder present in the range image. The additive noise in the image has a standard deviation of 0.75, or 3% of the radius of the cylinder and 9% of the pixels in the image are corrupted

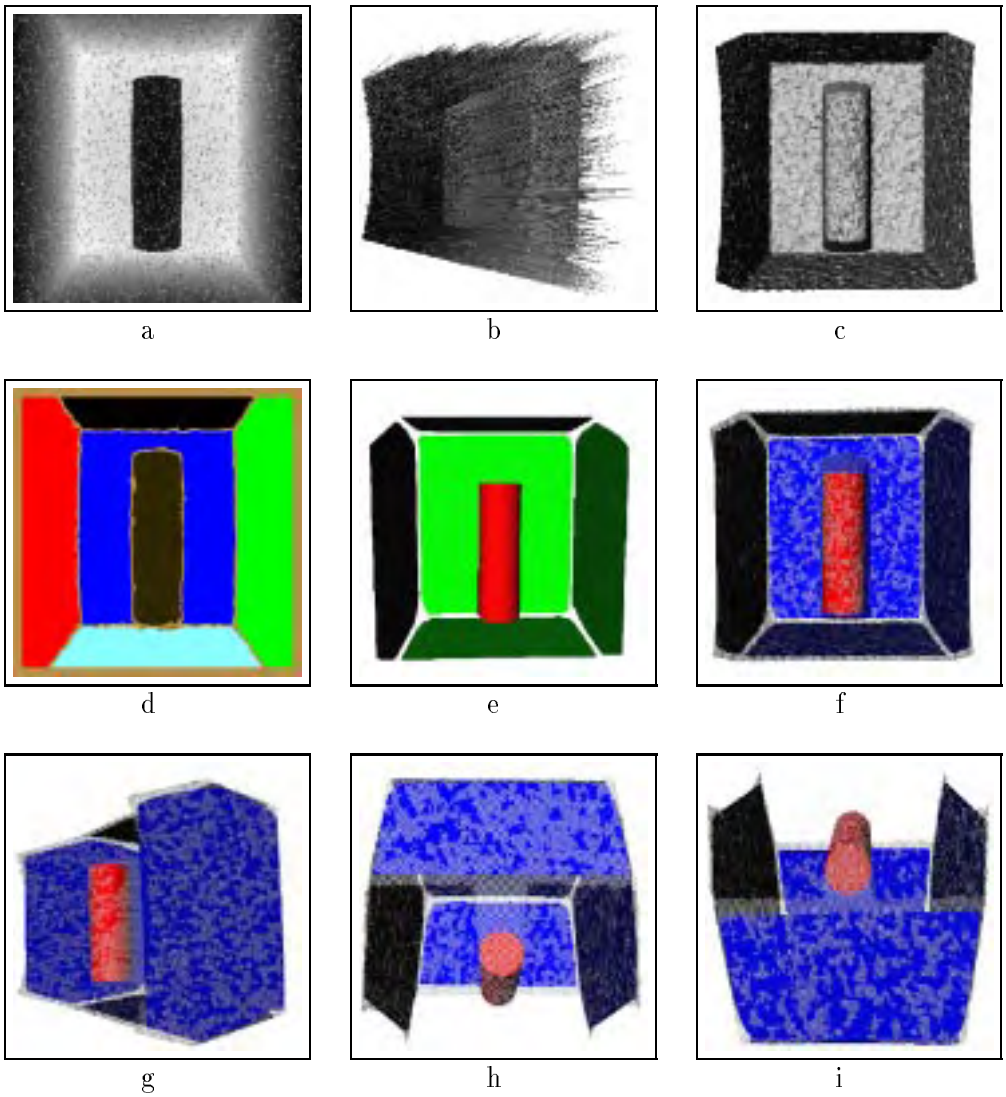


Figure 4.2: Results of surface fitting on noisy cylinder image: (a) original range image, (b) 3D rendering of original range image, (c) 3D rendering of median filtered range image, (d) Watershed segmentation of range image, (e) 3D rendering of fitted surfaces, (f) 3D rendering of superposition of fitted surfaces onto (c) (front view), (g) tilted side view of (f), (h) top view of (f), (i) bottom view of (f).

Table 4.2: Results of surface fitting for cylinder and planes with noise

Description	Surface	# Pixels	Err./Pixel (mm)	It.	Radius	% Err. (Radius)
Left Wall	plane	10959	0.078	3	NA	NA
Floor	plane	5685	0.049	17	NA	NA
Ceiling	plane	4371	0.051	3	NA	NA
Back Wall	plane	15963	0.15	3	NA	NA
Right Wall	plane	11185	0.081	3	NA	NA
Cylinder	cylinder	6442	0.157	11	5.8224	16.448

with replacement noise. This noise may be seen in Figure 4.2b. One result of importance not given in Table 4.2 is the computed height of the cylinder in the surface fitting step compared to the actual cylinder height. The computed height is 30.6367 and the actual height is 30. This computation yields and absolute error of 2.122%. The second noisy image, shown in Figure 4.3 tested contains a sphere having a radius of 10. The qualitative results of segmentation and surface fitting are shown in Figure 4.3. The quantitative results of the surface fitting are given in Table 4.3. This image has been corrupted with replacement and additive noise, showing how the segmentation and surface fitting steps of the system react in the present of noise. The additive noise in the image has a standard deviation of 0.75, or 1.5% of the radius of the sphere and 9% of the pixels in the image are corrupted with replacement noise.

Table 4.3: Results of surface fitting for sphere and planes with noise

Description	Surface	# Pixels	Err./Pixel (mm)	It.	Radius	% Err. (Radius)
Left Wall	plane	10955	0.078	3	NA	NA
Floor	plane	5737	0.049	17	NA	NA
Ceiling	plane	4345	0.051	3	NA	NA
Back Wall	plane	17566	0.15	3	NA	NA
Right Wall	plane	11169	0.081	3	NA	NA
Sphere	sphere	4926	0.24	10	11.7161	17.161

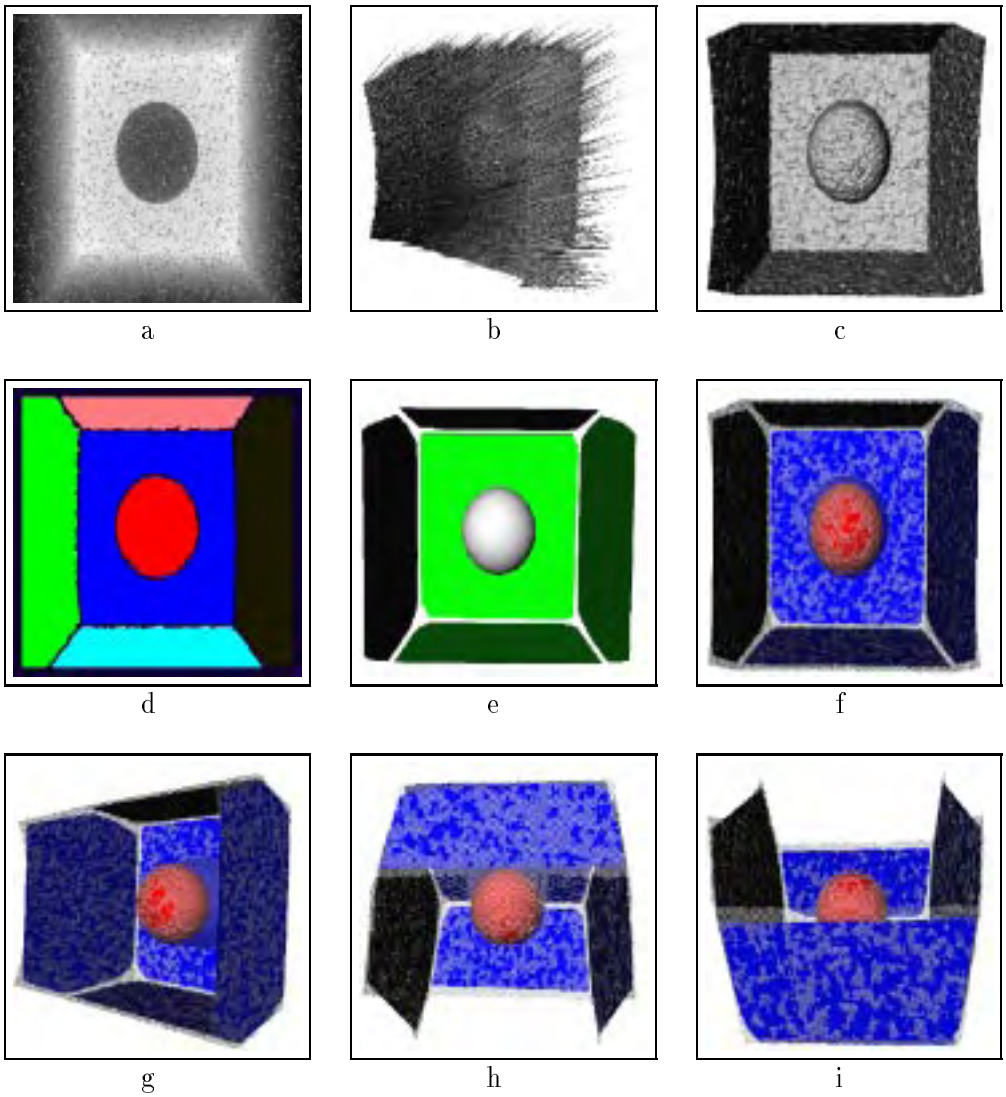


Figure 4.3: Results of surface fitting on noisy sphere image: (a) original range image, (b) 3D rendering of original range image, (c) 3D rendering of median filtered range image, (d) Watershed segmentation of range image, (e) 3D rendering of fitted surfaces, (f) 3D rendering of superposition of fitted surfaces onto (c) (front view), (g) tilted side view of (f), (h) top view of (f), (i) bottom view of (f).

CHAPTER 5

Results and Analysis

In this chapter we present results obtained from our surface reconstruction system. The system takes as input the original range-amplitude pairs and produces a 3D visualization of the surface fitting. The results shown are obtained using real data from the Perceptron Laser Range scanner of three different scenes. Each step in the surface reconstruction system contains a number of free parameters. We will begin by discussing these parameters in Section 5.1. Then, in Section 5.2, we show the results for the three scenes. Section 5.2 shows the images of original data, the preprocessed data, the results of feature extraction, segmentation, and finally the 3D visualization of the fitted surfaces. Those results show qualitatively how well each step in the surface reconstruction performs on the data. This section also discusses the parameters used in the processing steps and any errors in the final surface reconstructions. Errors are noted and their possible causes analyzed.

5.1 Free Parameters

Each step in the system possesses some free parameters which are given in Table 5.1. In this table we state the step of the system that contains the free parameter, the variable name by which we refer to the parameter, the range of values for the parameter as used in the results obtained in this section, and a subjective *sensitivity measure*. This measure of sensitivity is either *Low*, *Medium*, or *High*. Where *Low* means that the parameter is

Table 5.1: Free parameters in our surface reconstruction system.

Processing Step	Parameter (Variable Name)	Range of Values	Sensitivity
Median filtering	n	3 - 7	Medium
Anisotropic Diffusion	k_r	0.1 - 3.0	Medium
Anisotropic Diffusion	k_a	0.1 - 20.0	Medium
Anisotropic Diffusion	k_n	1.0 - 10.0	Medium
Anisotropic Diffusion	i	3 - 5	Low
Feature Extraction	n_n	3 - 5	Low
Feature Extraction	p_r	200 - 300	Low
Feature Extraction	p_a	200 - 300	Low
Feature Extraction	n_n	50 - 200	Medium
Watershed Segmentation	d	0.2 - 0.3	Medium
Surface Fitting	c	0.0001 - 0.02	High

highly robust and could be easily set and used for a variety of images, and *High* means that the parameter is highly sensitive may need to be changed from image to image. The first parameter is the neighborhood of the median filter used in the preprocessing step to eliminate replacement noise, or n. This neighborhood size is fairly robust. It has a Medium sensitivity measure. Although we obtained reasonable results for n between 3 and 5, a larger neighborhood such as 9 or 11 may be used to eliminate large patches of outliers which are present in real range data. However, this would tend to oversmooth the data and reduce the change in the surface normal at crease edges: the very features we are trying to detect. Therefore, for most images we set n equal to 5. A neighborhood of 5 may not eliminate larger patches of replacement noise, but it does remove individual outliers and patches large enough to affect the segmentation while retaining the crease edges in the range data.

The next three parameters are the weights for the gradients in the anisotropic diffusion step given by k_r, k_a, and k_n. The gradient of the range is weighted by k_r, the amplitude by k_a, and the surface normal by k_n. These values are expressed in units of the average of

their associated features [81]. Each of these three parameters have a Medium sensitivity measure. The values we have used most often for these parameters are 3.0, 5.0, and 10.0 respectively. However, changing these parameters depending on the noise levels in the image could help further reduce the noise that interferes with feature extraction and segmentation. The fifth parameter is the number of iterations used in anisotropic diffusion, denoted by i. This parameter has a Low sensitivity measure because for all images we have tested, the results obtained using between 3 and 5 iterations were basically the same. For all the images shown in this chapter the number of iterations was 5.

The sixth parameter, used in the feature extraction step, is the neighborhood of the median filter used to filter the surface normals at each pixel, denoted by n_n. This parameter has a Low sensitivity measure. For all images shown in this chapter, we set the n_n to 5. The next three parameters are the scaling percentages used to remap the gradient magnitude of the range data, the amplitude data, and the surface normals to fuzzy feature maps that contain values between 0 and 1. These parameters are denoted by p_r, p_a, and p_n respectively. The first two remapping percentages have a Low sensitivity measure. For all the images shown we have set both p_r and p_a to 200, which are expressed in units as an average percentage of their respective features. The remapping percentage for the surface normal has a Medium sensitivity measure. We have found that the gradient of the surface normal tends to be slightly more sensitive to the remapping percentage, therefore it may need to be altered for images containing different levels of noise to obtain better results.

The tenth parameter is the minimum watershed depth threshold used in the segmentation step, denoted by d. This parameter has a Medium sensitivity measure. There are basically three values which we use for the d: 0.2, 0.25, and 0.3, operating on input feature

maps remapped to fuzzy values between 0.0 and 1.0. Depending upon the input to the segmentation algorithm, d may need to be altered in order to reduce oversegmentation, or to allow more regions to be segmented distinctly from adjacent regions. For most of our testing we have set the d to 0.25 and have obtained good results.

The final parameter listed in Table 5.1 is the eigenvalue initial threshold, denoted by c. This parameter has a High sensitivity measure. The value of this threshold is used to initially determine if the segment is a plane, cylinder, or sphere. The absolute values of the eigenvalues, $\|e_i\|$ of the Shape Matrix obtained from the second-order polynomial fit to the initial data are thresholded according to c. If both $\|e_i\|$ are greater than c, the segment is determined to be a sphere. If both $\|e_i\|$ are less than c, the segment is determined to be a plane. Otherwise the segment is determined to be a cylinder. The sensitivity of c seems to be proportional to the noise level in the range data. We arrived at a reasonable range for the values of this parameter through experimentation.

5.2 Results on Real Data

We have selected three scenes, scanned using the Perceptron Laser Range Scanner, on which to test our surface reconstruction system. Each of the range-amplitude image pairs have a resolution of 1024×1024. The first set of results we present for our surface reconstruction is obtained using a scan of an empty room. The majority of the image is the vertical wall, therefore we will refer to this image as the *Wall Image*. Figure 5.1 shows the original range image, the preprocessed range data, and the an image containing the noise removed by the preprocessing step. Figure 5.2 shows the results of preprocessing

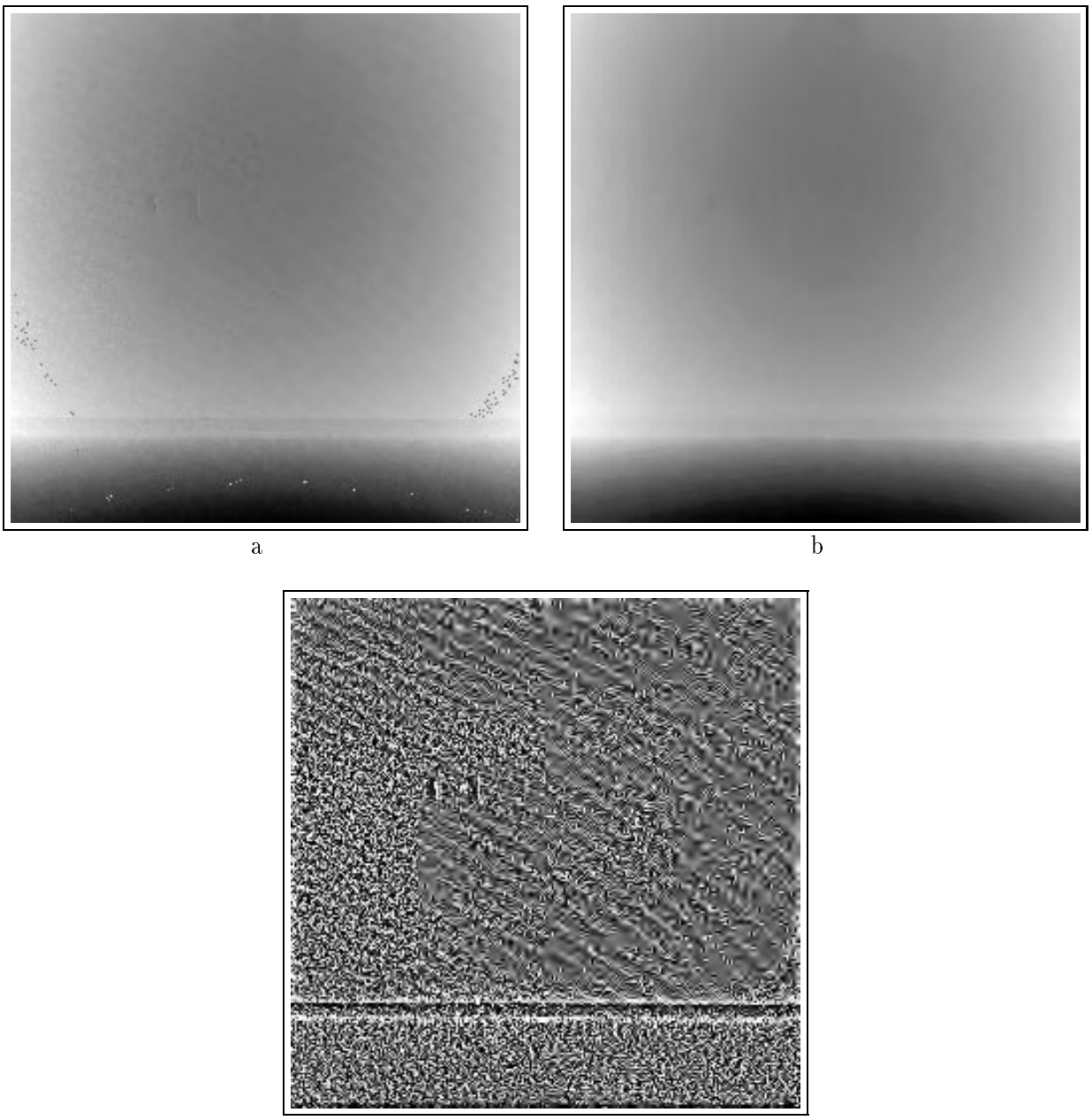
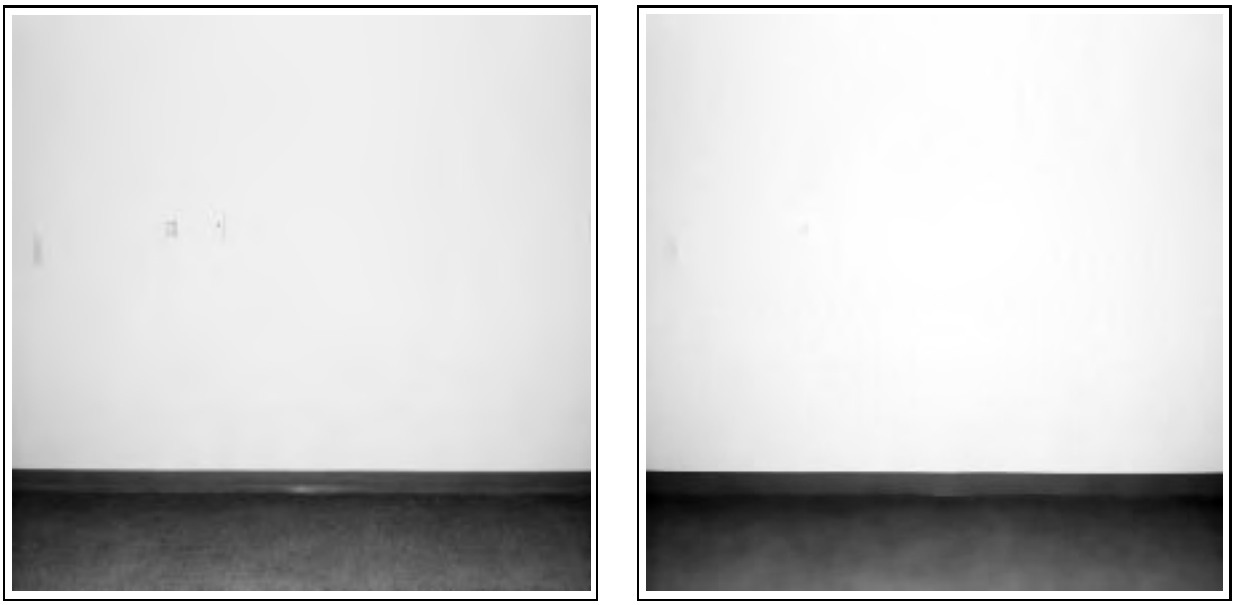
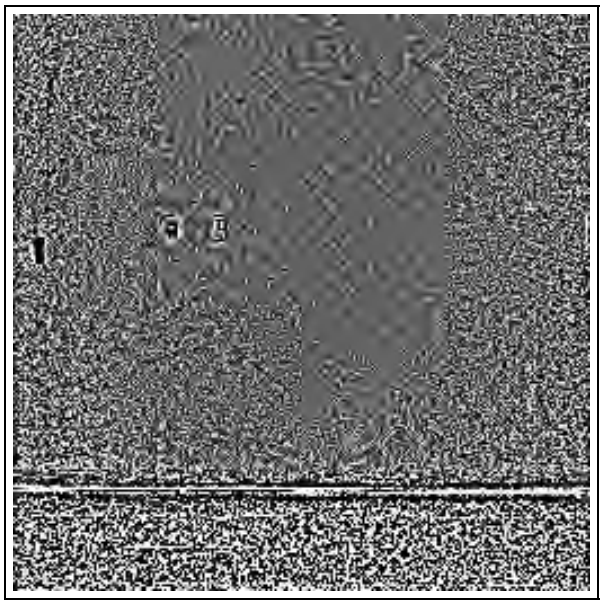


Figure 5.1: Results of preprocessing on *Wall Image* (range): (a) original range image, (b) preprocessed range image, (c) subtraction of (b) from (a), i.e. noise removed



a b

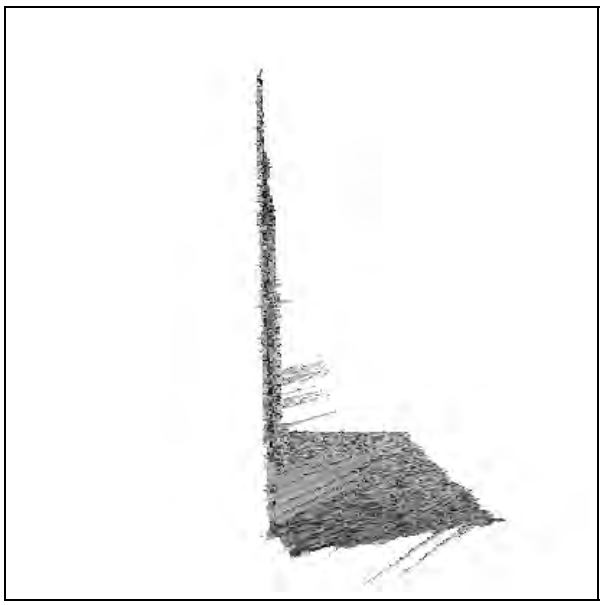


c

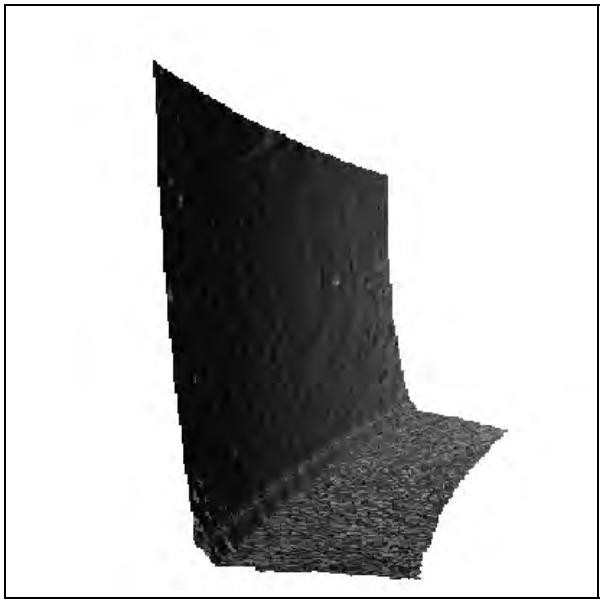
Figure 5.2: Results of preprocessing on *Wall Image* (amplitude): (a) original amplitude image, (b) preprocessed amplitude image, (c) subtraction of (b) from (a), i.e. noise removed

on the amplitude image. In the preprocessing step, the neighborhood for median filtering on the images is 7. The weights in the anisotropic diffusion are 1.0, 1.0, and 10.0 for k_r, k_a, and k_n, respectively. A better method of presenting the results of noise removal from range data is by a 3D plot of the data, before and after. Figure 5.3 shows just that. Figure 5.4 shows the results of the feature extraction step including the gradient magnitude of the range, the amplitude, and the surface normal. Figure 5.4d shows the fusion of the first three images using Bernoulli's Rule of Combination. The remapping percentages, p_r, p_a, and p_n, for these first three images are all 200. The result of the watershed segmentation is shown in Figure 5.5a. This result was obtained using a watershed depth threshold of 0.2. There are three distinct region shown in red, green, and blue to which we attempted to fit surfaces. The top region is the vertical wall. The middle segment is the rounded floor molding at the base of the wall. The bottom segment is the floor of the room, which is perpendicular to the vertical wall. The qualitative results of the surface fitting are shown in Figure 5.5 b,c, & d. For the vertical wall, the average rms error per pixel is equal to $7.29mm$ which is approximately 0.22% rms error per pixel relative to the average distance from the surface to the laser origin. The surface fitting took three iterations to converge to this solution. For the horizontal floor, the average rms error per pixel is equal to $4.54mm$ which is approximately 0.48% squared error per pixel. The surface fitting algorithm took three iterations to converge to this solution. The middle segment was determined to be an concave cylinder by the initial eigenvalue analysis of the Shape Matrix, therefore it was discarded and a surface was not fit to this segment. This system is only designed to fit surfaces to planes, convex cylinders, and spheres.

The second scene considered contains three large vertical panels with three different



a



b

Figure 5.3: 3D comparison of preprocessing of *Wall Image*: (a) 3D rendering of original range data, (b) 3D rendering of preprocessed range data.

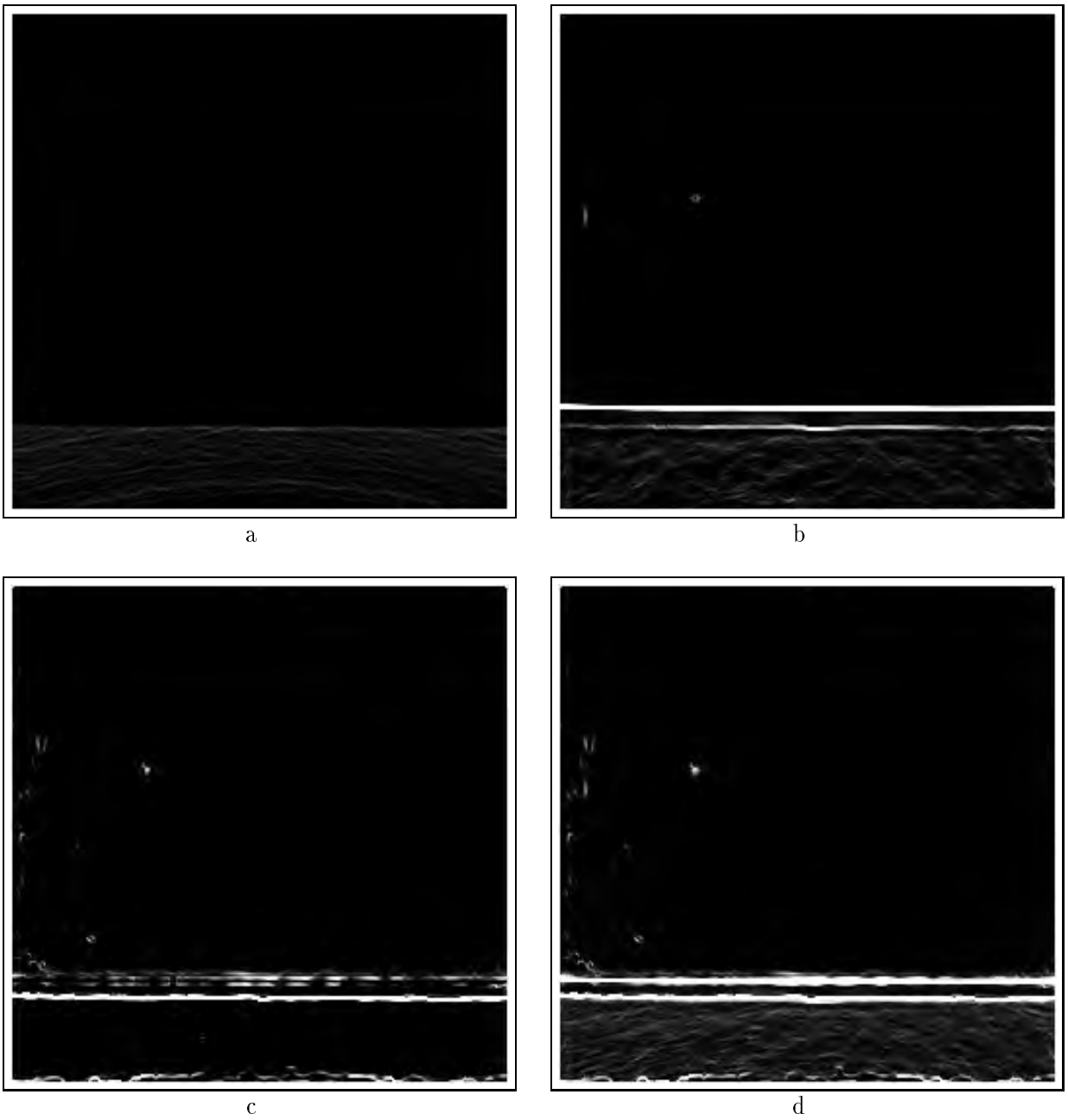


Figure 5.4: Results of feature extraction of *Wall Image*: (a) gradient magnitude of range image, (b) gradient magnitude of amplitude image, (c) gradient magnitude of surface normal of range image, (d) fusion of (a),(b), & (c).

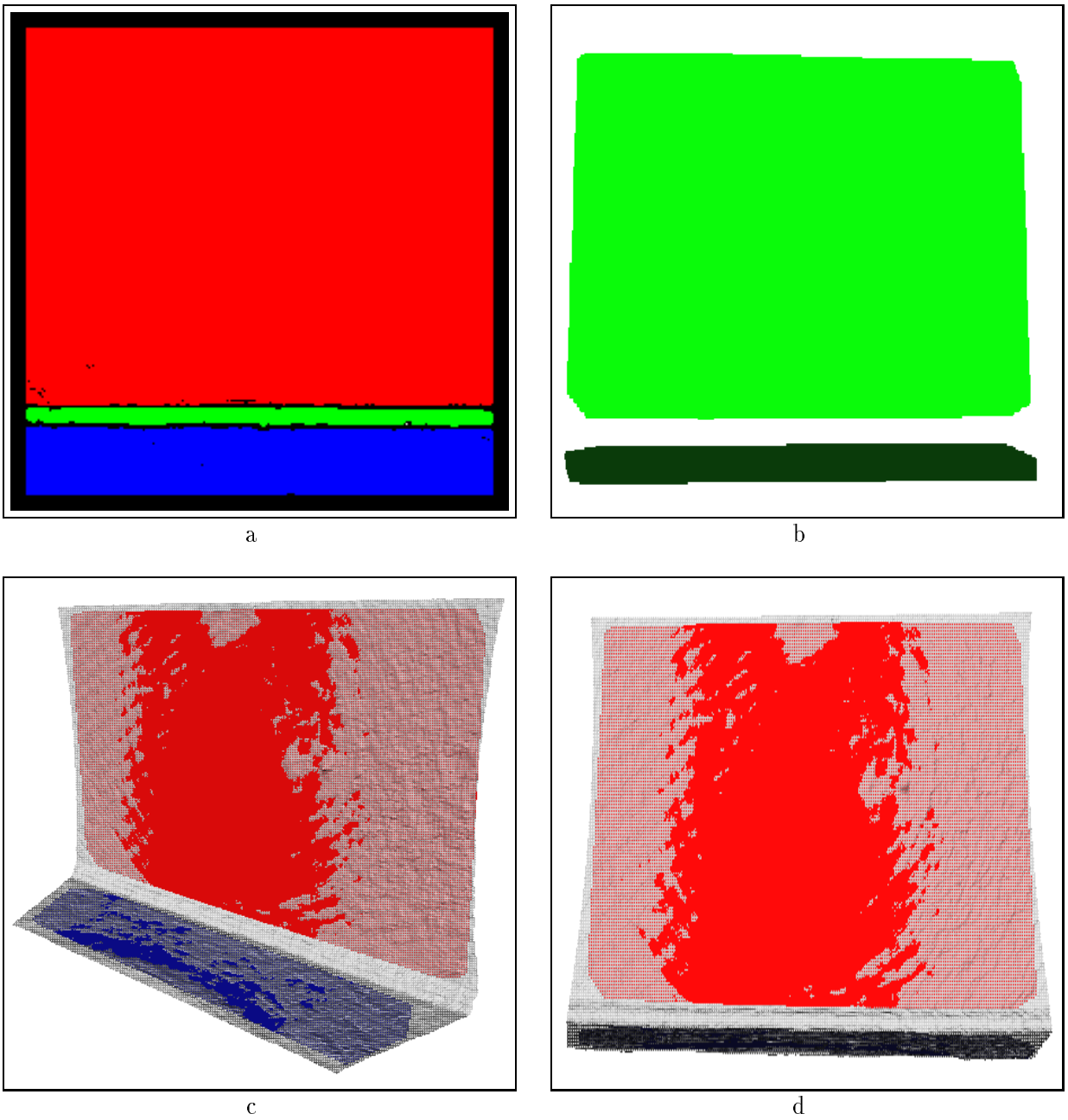
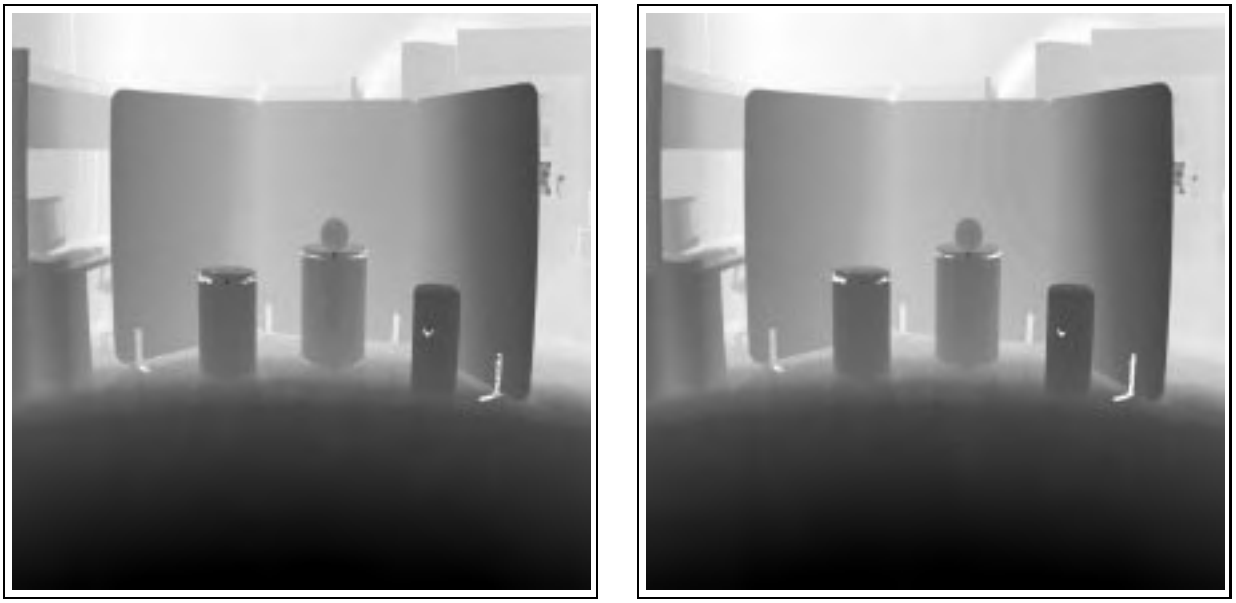


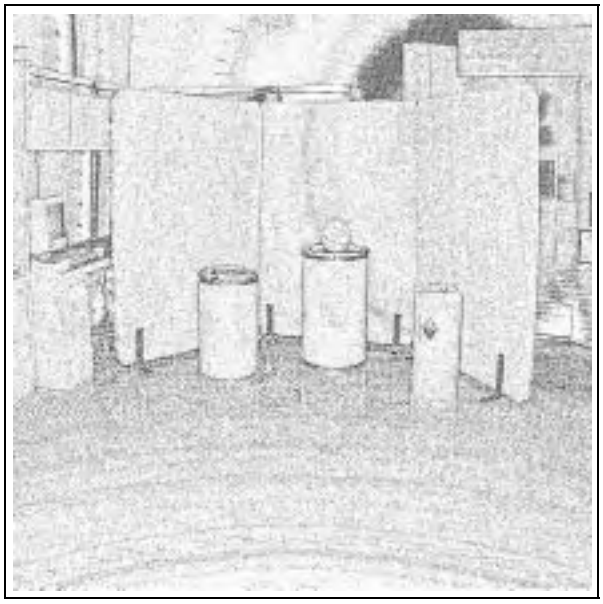
Figure 5.5: Results of segmentation and surface fitting on *Wall Image*: (a) segmentation of Figure 5.4 d, (b) 3D rendering of fitted surfaces, (c) 3D rendering of superimposed fitted surfaces (tilted view), (d) 3D rendering of superimposed fitted surfaces (front view).

sized barrels and a basketball placed on top of the center barrel. These seven objects are the main objective of the surface reconstruction system for this scene. The sphere we fit to is a regulation-sized basketball, therefore we will refer to this scene as the *Basketball Image*. Figure 5.6 shows the original range image, the preprocessed range data, and the an image containing the noise removed by the preprocessing step. Figure 5.7 shows the results of preprocessing on the amplitude image. In the preprocessing step, the neighborhood for median filtering on the images is 5. The weights in the anisotropic diffusion are 3.0, 5.0, and 10.0 for k_r, k_a, and k_n, respectively. Figure 5.8 shows the 3D plotting of the original range data and the resulting preprocessed range data. Figure 5.9 shows the results of the feature extraction step including the gradient magnitude of the range, the amplitude, and the surface normal. Figure 5.9d shows the fusion of the first three images using Bernoulli's Rule of Combination. The remapping percentages for these first two images are 200 and 100 for the gradient of the surface normal. Figure 5.10 shows the result of fusing all three feature maps and the segmentation of that fusion. It also shows the fusion of the gradient of the range with the gradient of the surface normal and the resulting segmentation of that fusion. We show this comparison of fusion inputs and segmentation to demonstrate that the gradient of the amplitude, for this scene, introduces features that cause oversegmentation of the scene. This oversegmentation can be scene most in the small barrel on the right side of the image. We could use the oversegmented image, shown in Figure 5.10c, for surface fitting. However, we have chosen to use Figure 5.10d as the input to the surface fitting step because the barrels, or cylinders, are not individually broken into separate regions as with the oversegmented data.

For both of these segmentations, the minimum watershed depth is set to 0.25. This

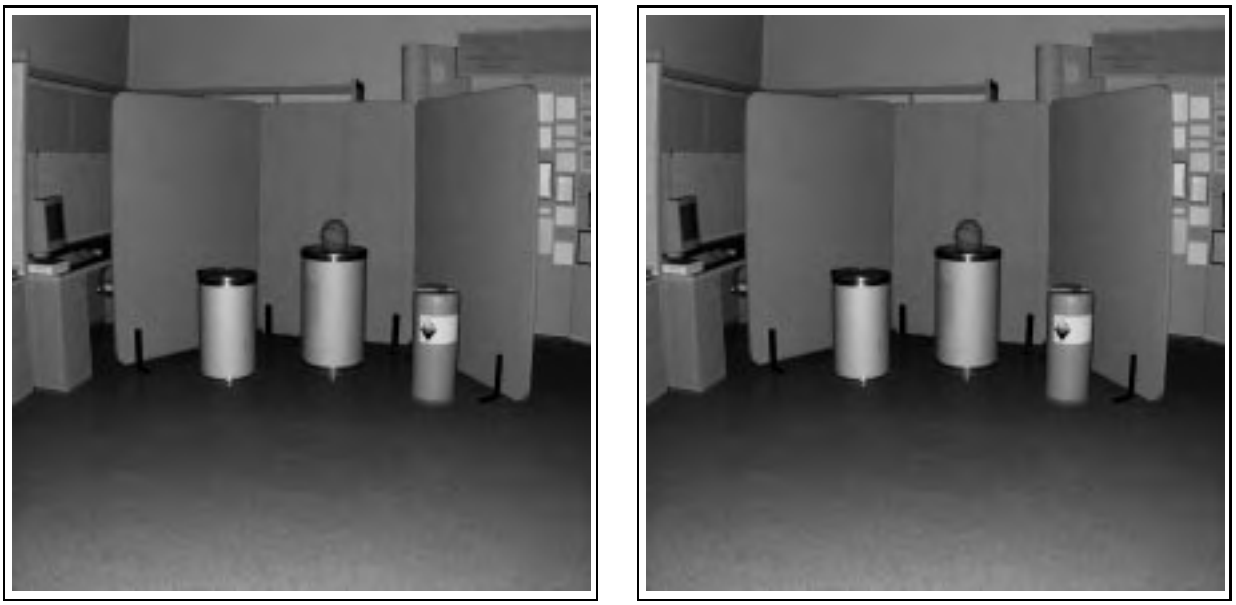


a b

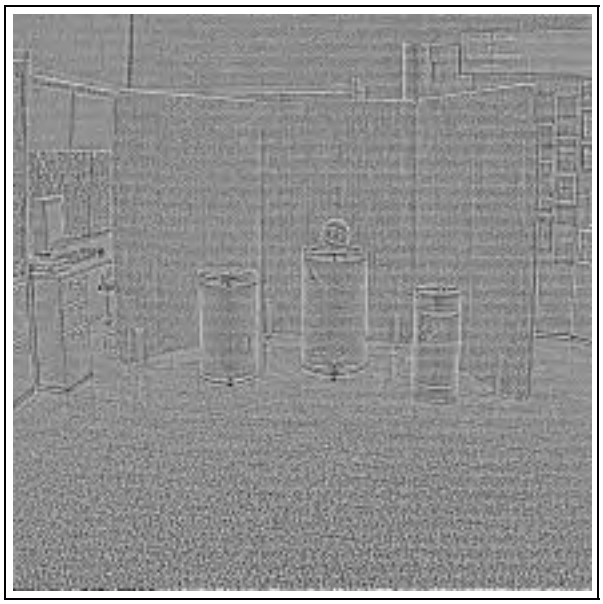


c

Figure 5.6: Results of preprocessing on *Basketball Image* (range): (a) original range image, (b) preprocessed range image, (c) subtraction of (b) from (a), i.e. noise removed.

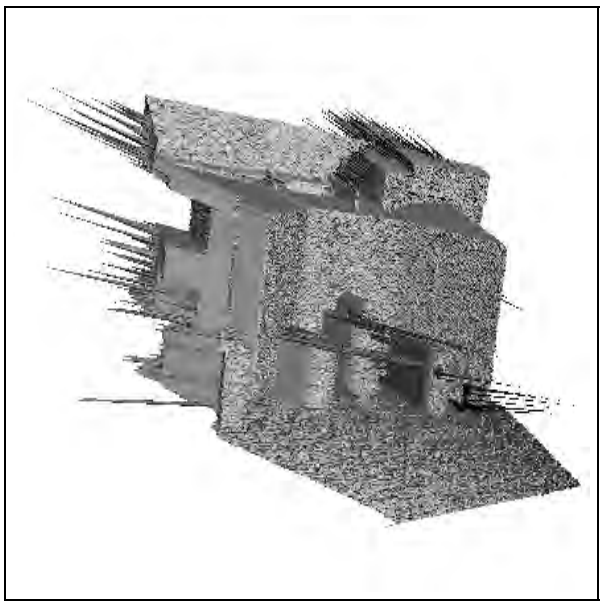


a b

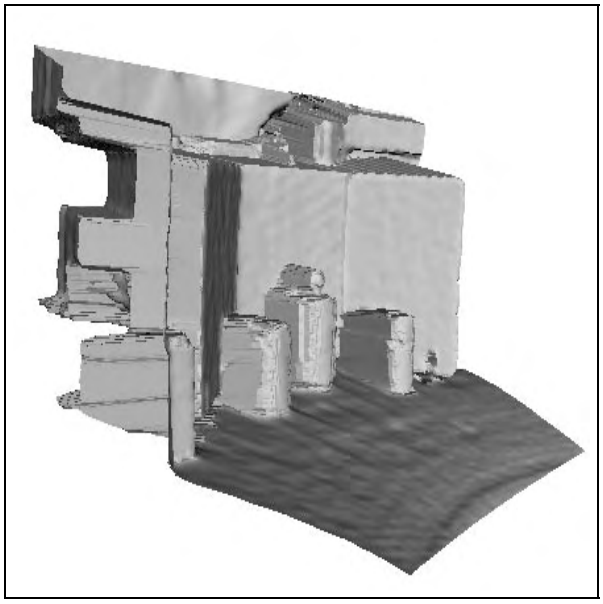


c

Figure 5.7: Results of preprocessing on *Basketball Image* (amplitude): (a) original amplitude image, (b) preprocessed amplitude image, (c) subtraction of (b) form (a), i.e. noise removed.

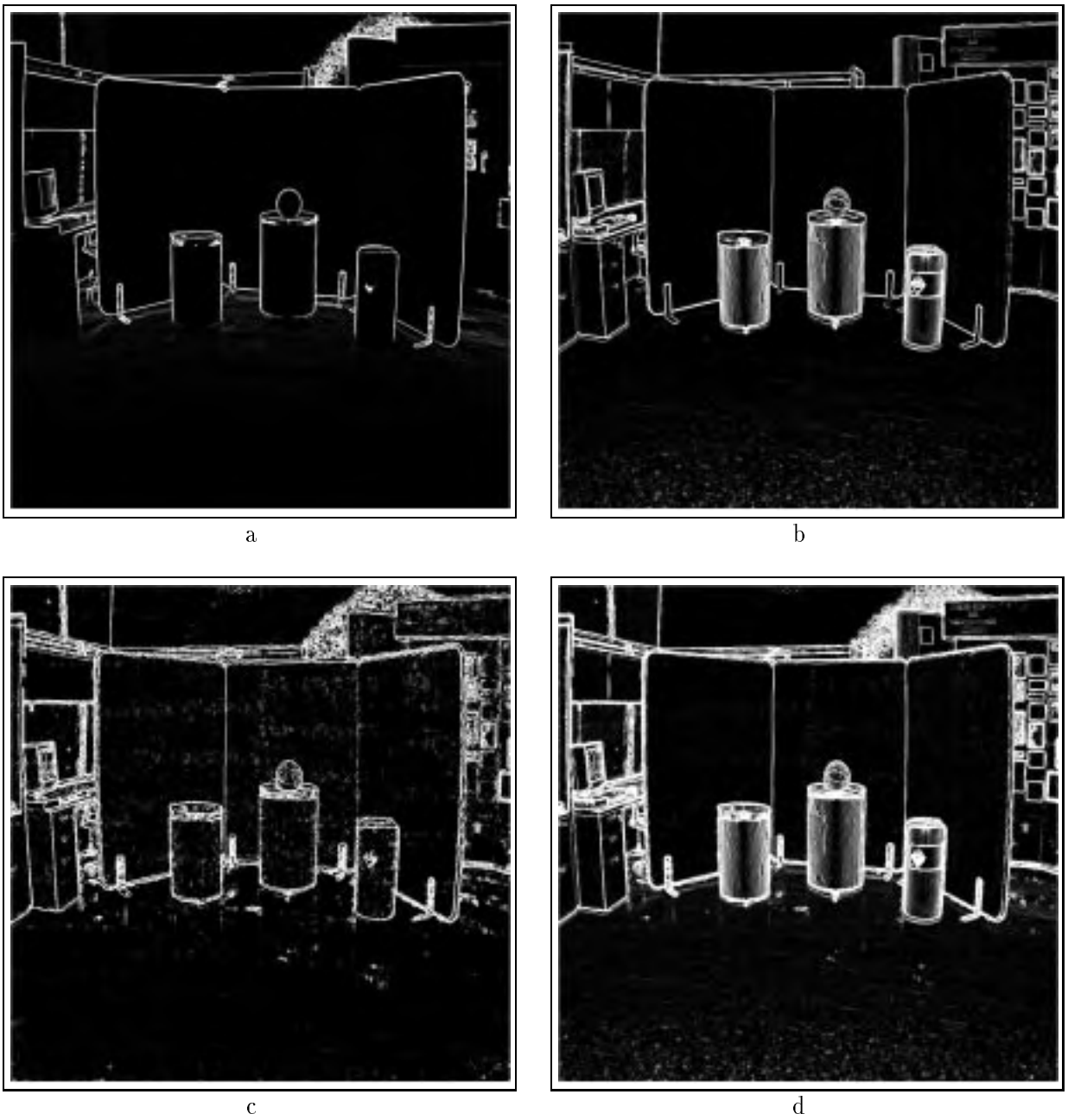


a



b

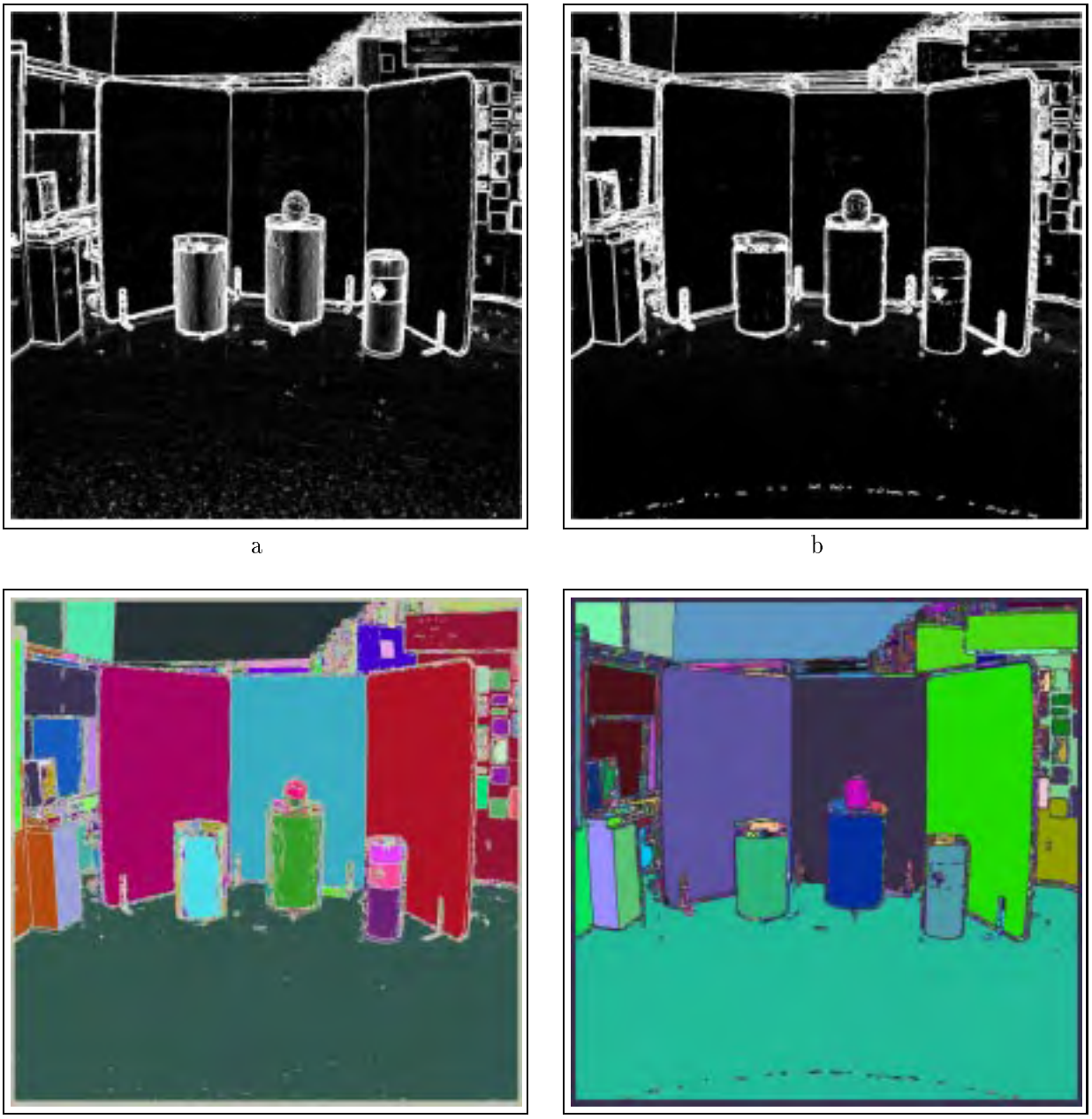
Figure 5.8: 3D comparison of preprocessing of *Basketball Image* (range): (a) 3D rendering of original range data, (b) 3D rendering of preprocessed range data.



a b

c d

Figure 5.9: Results of feature extraction of *Basketball Image*: (a) gradient magnitude of range image, (b) gradient magnitude amplitude image, (c) gradient magnitude of surface normal of range image, (d) fusion of (a),(b), & (c).



a b

c d

Figure 5.10: Comparison of fusion and segmentation of features obtained from *Basketball Image*: (a) fusion result shown in Figure 5.9 d, (b) fusion of Figure 5.9 a & c, (c) segmentation of (a), (d) segmentation of (b).

Table 5.2: Quantitative results of surface fitting on backdrop planes in *Basketball Image*.

Segment Description	Error/pixel (mm)	% Error/pixel (mm)
Left Vertical Backdrop Plane	62.59	1.2
Center Backdrop Plane	81.43	1.5
Right Vertical Backdrop Plane	31.69	0.7

Table 5.3: Quantitative results of surface fitting on cylinders and sphere in *Basketball Image (units in mm)*.

Segment Description	Error/ pixel	Actual Radius	Fitted Radius	% Error/ radius	Actual Height	Fitted Height	% Error/ height
Left Barrel	62.46	241.3	223.1	7.5	609.6	589.5	3.3
Center Barrel	46.92	279.4	265.03	5.1	723.9	682.25	5.75
Right Barrel	293.45	177.8	124.75	29.8	660.4	602.43	8.8
Basketball	80.2103	114.3	105.95	7.3	NA	NA	NA

brings up an important point. While the amplitude data is useful for smoothing (anisotropic diffusion), it might not be good for segmentation. The range data is a semi-transparent gray color so that the fitted surfaces may be viewed more easily. Table 5.2 gives the quantitative results for the surface fitting of the three main vertical background planes. Table 5.3 gives the quantitative results for the surface fitting of the three barrels and the basketball. The qualitative results of the surface fitting are shown in Figure 5.11. Figure 5.11 b,c, & d shows differing views of the fitted results superimposed onto the smoothed range data. The surface fitting results for the three main planes are within what we consider acceptable errors. The error per pixel is the rms error per pixel in units of mm. The percentage error is the rms error per pixel expressed as a percentage of the average distance from the plane to the laser origin. The left and middle barrel along with the basketball also yield acceptable surface fitting results. However the right barrel yields unacceptable surface fitting results in terms of the average rms error per pixel and the % error of the radius. We believe these high errors are due to the initialization of the

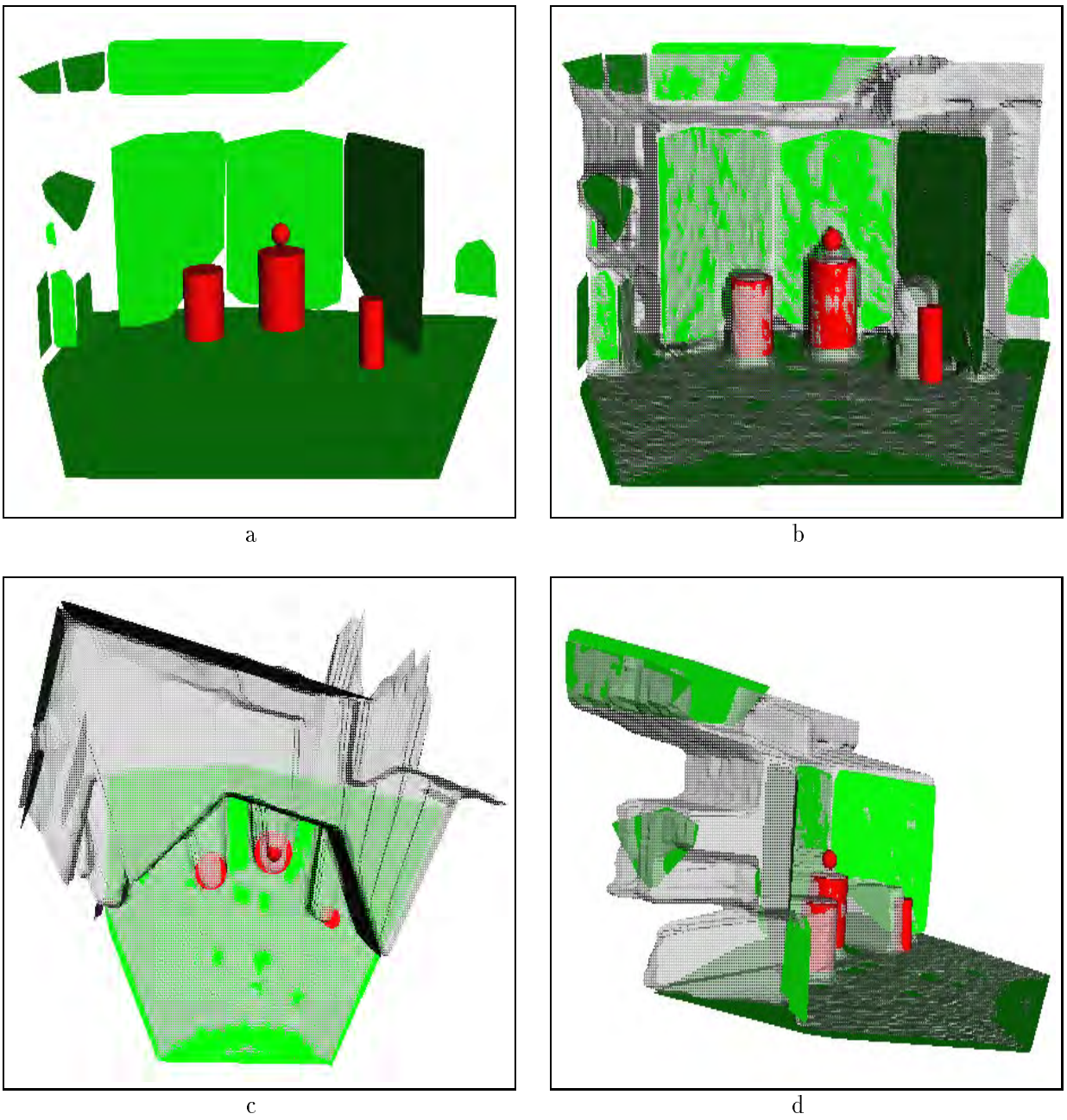
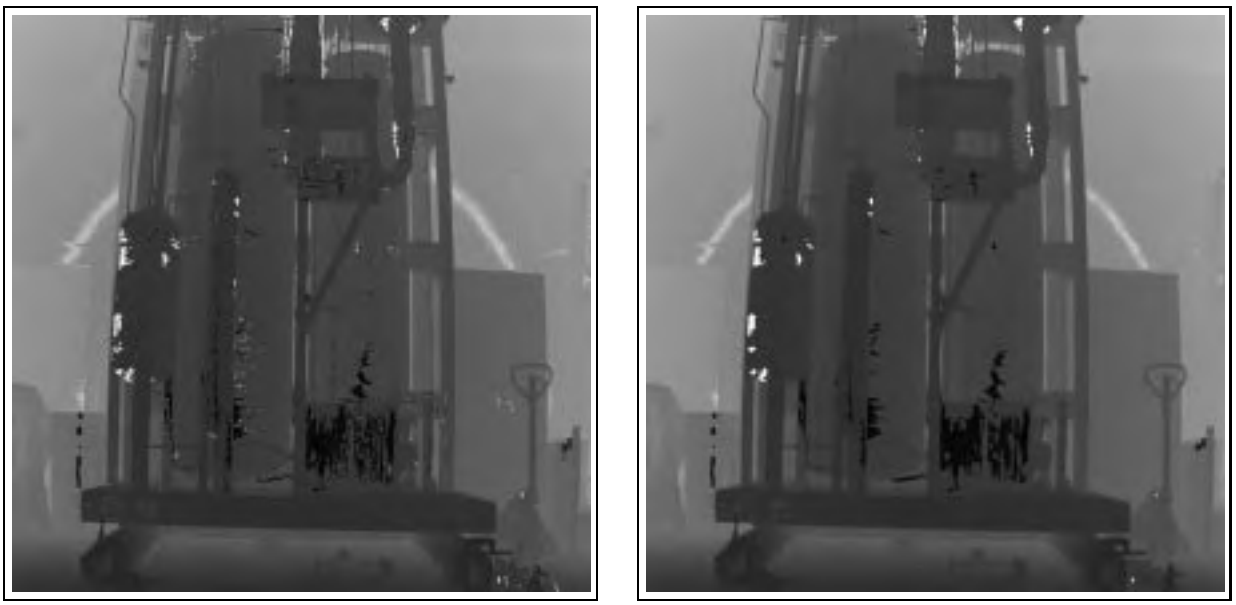


Figure 5.11: Results of surface fitting on *Basketball Image*: (a) 3D rendering of fitted surfaces, (b) 3D rendering of superimposed fitted surfaces (front view), (c) 3D rendering of superimposed fitted surfaces (top view), (d) 3D rendering of superimposed fitted surfaces (side view).

cylinder. The initialization of all the surfaces is crucial for the surface fitting to converge to a correct solution.

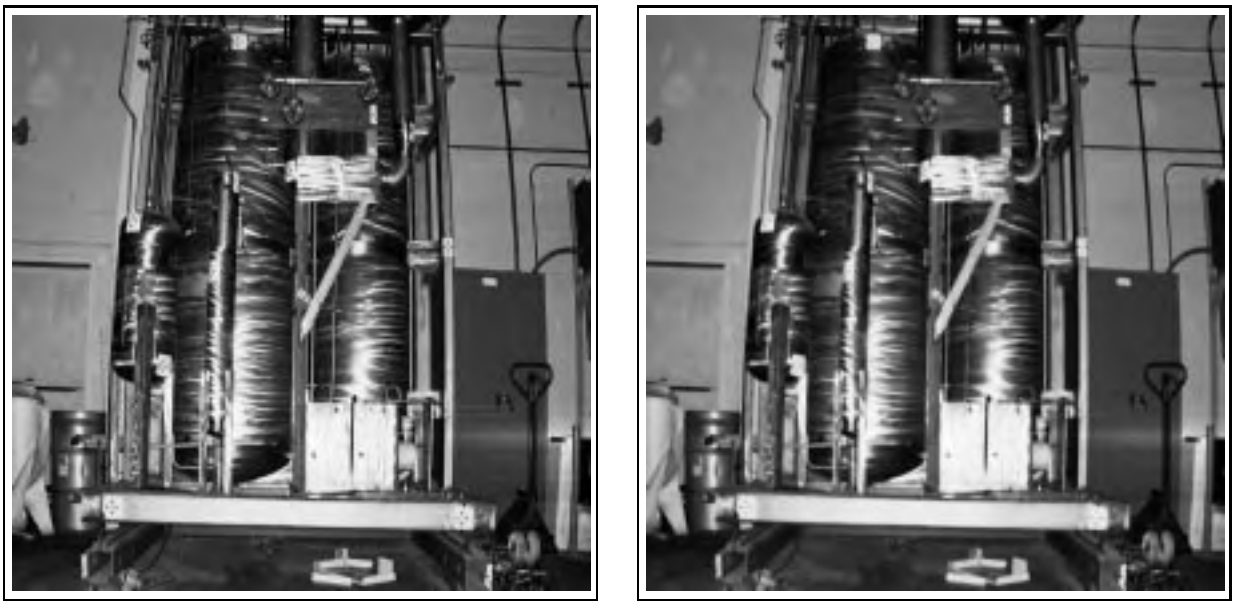
The final set of results we present for our surface reconstruction is a scan of an acid fractionator located inside the *Highbay* facility at Oak Ridge National Laboratories. Therefore we refer to this scene as the *Highbay Image*. This image presents the greatest challenge because of the high reflectivity of some of the surfaces in the scene, the high level of noise in both the range and amplitude images, and the relatively high complexity of the scene in terms of the number of different objects. The qualitative results, are shown in Figures 5.12 - 5.16. Figure 5.12 shows the original range image, the preprocessed range data, and an image containing the noise removed by the preprocessing step. Figure 5.13 shows the results of preprocessing on the amplitude image. In the preprocessing step, the neighborhood for median filtering on the images is 5. The weights in the anisotropic diffusion are 3.0, 20.0, and 1.0 for k_r, k_a, and k_n, respectively. We set k_a to a relatively high value of 20 to try to eliminate the effects of the high reflectance some of the objects in the amplitude image. Figure 5.14 shows the 3D plotting of the original range data and the resulting preprocessed range data. Figure 5.15 shows the results of the feature extraction step including the gradient magnitude of the range, the amplitude, and the surface normal. Figure 5.15d shows the fusion of the first three images using Bernoulli's Rule of Combination. The remapping percentages for these first two images are 200 and 100 for the gradient of the surface normal. The result of the watershed segmentation is shown in Figure 5.16a. This result was obtained using a watershed depth threshold of 0.2.
We have attempted to fit surfaces to four regions in this scene. These surfaces encompass three planes and one cylinder. These surfaces are shown in Figure 5.16. For the three



a b

c

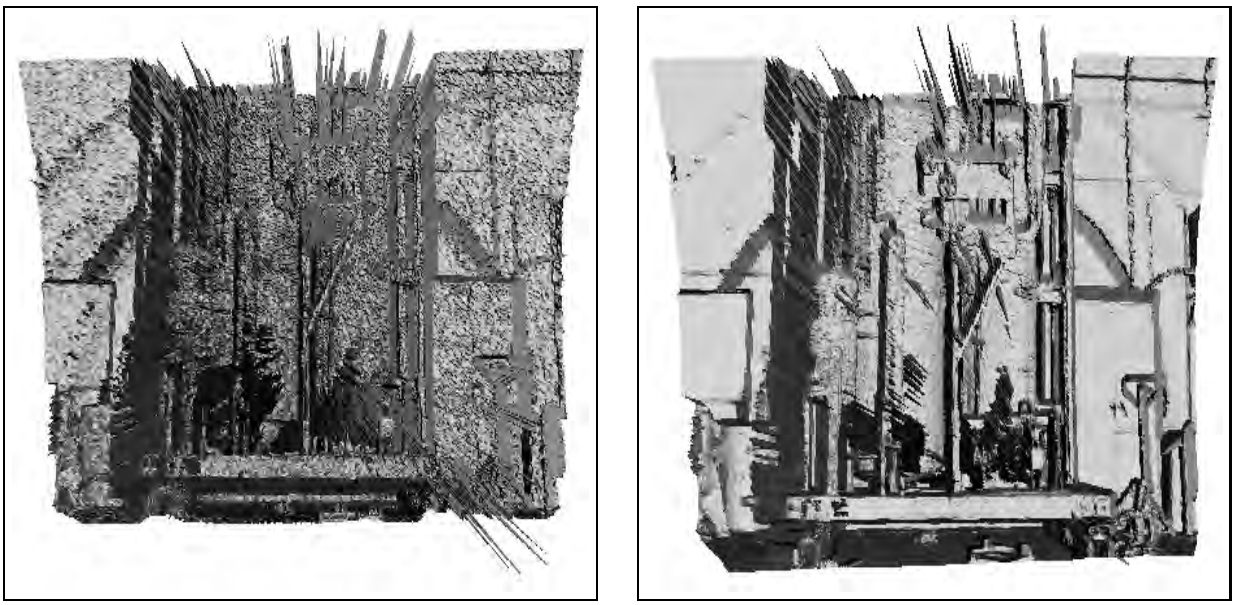
Figure 5.12: Results of preprocessing on *Highbay Image* (range): (a) original range image, (b) preprocessed range image, (c) subtraction of (b) from (a), i.e. noise removed.



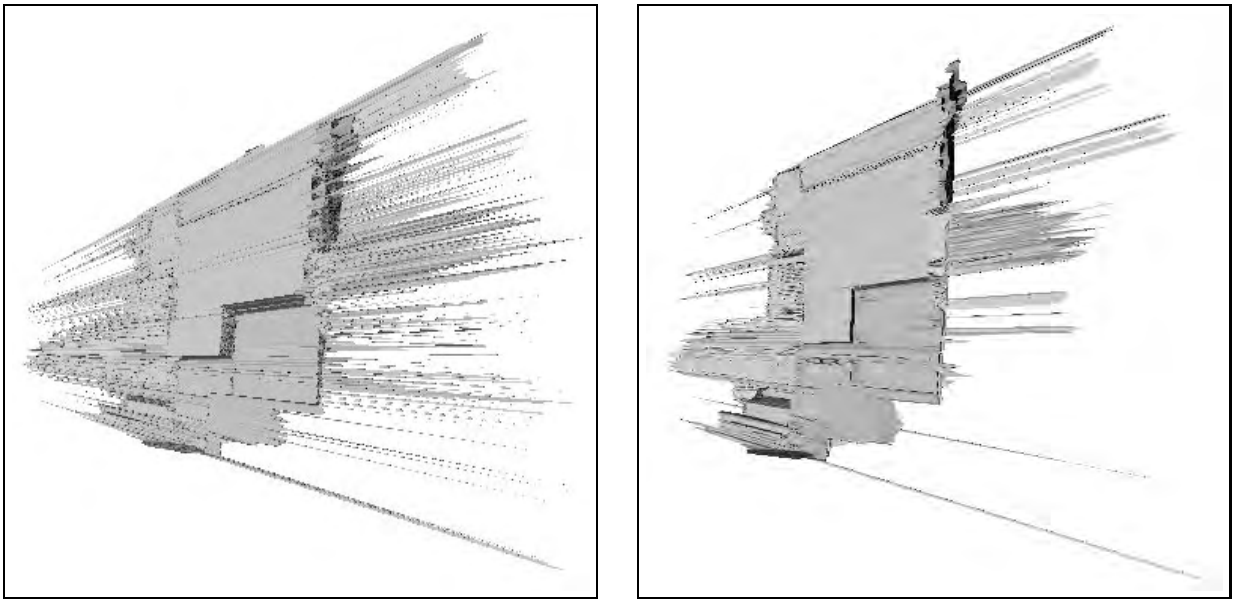
a b

c

Figure 5.13: Results of preprocessing on *Highbay Image* (amplitude): (a) original amplitude image, (b) preprocessed amplitude image, (c) subtraction of (b) from (a), i.e. noise removed.

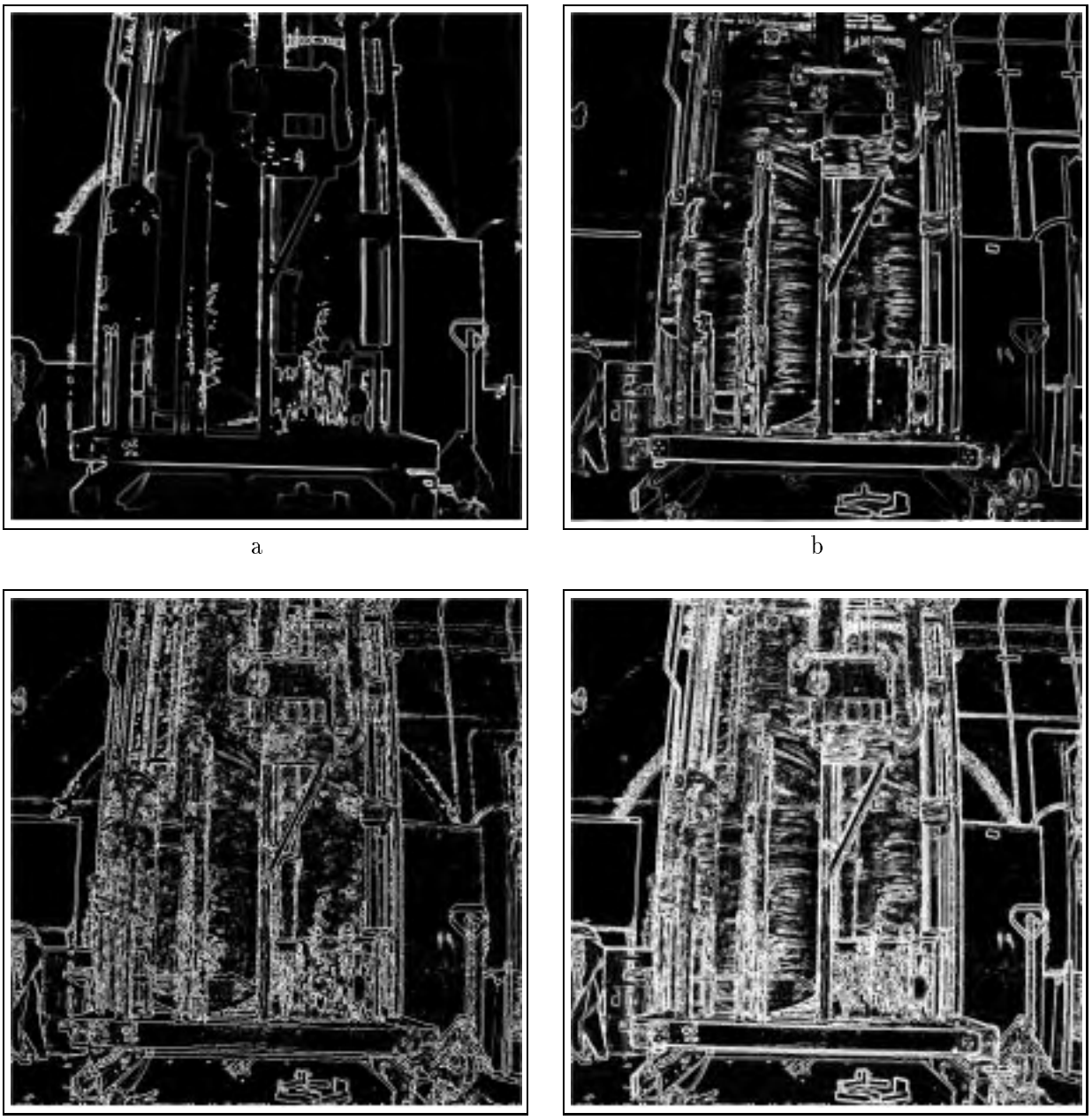


a b



a b

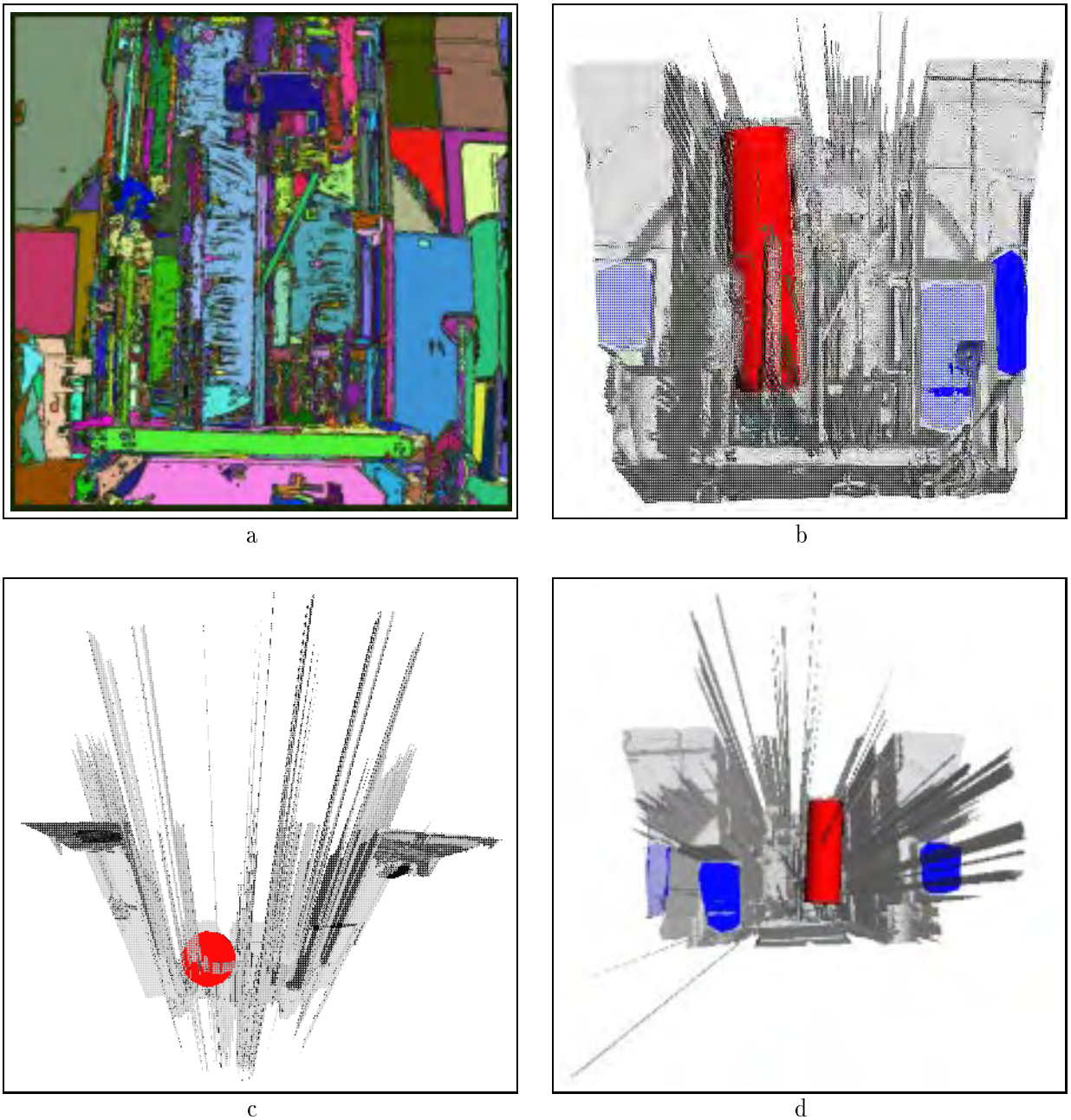
Figure 5.14: 3D comparison of preprocessing of *Highbay Image* (range): (left column) original data, (right column) preprocessed data.



a b

c d

Figure 5.15: Results of feature extraction of *Highbay Image*: (a) gradient magnitude of range image, (b) gradient magnitude amplitude image, (c) gradient magnitude of surface normal of range image, (d) fusion of (a),(b), & (c).



a b c d

Figure 5.16: Results of segmentation of surface fitting of *Highbay Image*: (a) segmentation of Figure 5.15 d, (b) 3D rendering of superimposed fitted surfaces (front view), (c) 3D rendering of superimposed fitted surfaces (top view), (d) 3D rendering of superimposed fitted surfaces (back view).

planes fitted, the plane on the far right in the 3D rendered plot yielded an average rms error per pixel of $20.1mm$, or 0.32% of the average distance from the plane to the laser origin. The remaining two planes yielded an average rms error per pixel of $6.59mm$ or approximately 0.15%. However, the cylinder we fit a surface to yielded an average squared error per pixel of $52.79mm$. This is an extremely high error, but when the fitted cylinder is superimposed onto the median filtered data, it can be seen that the cylinder radius and height are close to the actual radius and height. (We are not given the actual height and radius of the cylinder, therefore, we are not able to quantify the exact error of the surface fitting for the cylinder.) Therefore, we can conclude that the initialization of the cylinder produced such a high error per pixel. This error can be seen in the offset of the fitted cylinder from the actual cylinder position. This scene is extremely difficult to fit surfaces to, and this is shown by the results of the fitting which we have done. However, we have shown that it is possible to fit surfaces to this scene.

CHAPTER 6

Conclusions and Future Work

We have developed a scene reconstruction system that is viable for automatically reconstructing parts of, or entire scenes of both real and synthetic range data with convex cylinders, planes, and spheres. There is no user interaction required to operate the system other than initially setting certain parameters and selecting the segments to fit surfaces to. This almost completely satisfies our initial development goal which was to make this system as robust as possible and totally autonomous. Although the system contains a number of free parameters, these parameters are robust enough to function with very little knowledge of the inputs and a very small amount of effort. The only free parameter that we have seen to be highly sensitive is the initial eigenvalue analysis threshold, c. The remainder of this chapter will be dedicated to setting forth possible solutions for the correction of any errors present in this system so that future development of this system might yield a totally autonomous, extremely robust system.

Our contributions are the following.

- Implementing anisotropic diffusion for aiding in feature extraction from range and amplitude data.
- Fusion of crease edges and step edges obtained from amplitude and range data.
- Introduced an improved watershed algorithm with the watershed depth as the saliency.

- Development of a complete scene reconstruction system operating on real data.

The first step of our system that can be altered to yield better results is the extraction of the Cartesian coordinates of the range data obtained from the Perceptron Laser Range Scanner. The calibration for the scanner we use is found in Equations 2.1 - 2.8. However, recent recalibration of the Perceptron scanner we use has shown that these equations have changed since the initial calibration used to compute these equations. Therefore, it is crucial to know the correct calibration of a scanner in order for this surface reconstruction system to yield correct results on data obtained from that scanner.

The next step that may be improved is the anisotropic diffusion performed on the data. The surface normal values we have used in the conductance term of the anisotropic smoothing are an approximation of the surface normal of the image. The surface normal may be exactly computed using the same Hybrid Method set forth in Section 2.4.3 to obtain better preservation of the crease edges in the range image.

The feature extraction step including Bernoulli's Rule of Combination for combining the feature maps has proved to be robust. We believe that this step of the surface reconstruction system need not be altered except to eliminate any variance that is present in the exponential scaling percentage for the gradient magnitudes computed. However, alterations *could* eliminate these from the list of *free parameters* thus making the entire system even more robust.

In the area of image segmentation, the watershed algorithm handles fuzzy inputs well without producing high levels of oversegmentation. However, a further step of region merging should be examined in order to cut down on the oversegmentation that is present. This merging may be based on region size and similarity in surface characteristics of

adjacent segments. Merging in this manner should help to compensate for false edges which have been shown to be present in the inputs to the segmentation algorithm caused by noise in the original images. Not only may region merging be considered to improve the segmentation algorithm, but an advanced method of regional maxima to remove outliers in the segments should also be studied. The objective of our segmentation algorithm is to yield a conservative estimate of the contiguous, distinct regions in the scene. Although we have addressed the problem of region boundaries introducing outliers into a segment, blurred edge pixels still show up in some segments in the form of outliers. It may be necessary to create a more conservative estimate of the regions by allowing fewer pixels of high gradient (surrounding a distinct region) to flow into the catchment basins that ultimately will define the region. As for the selection of the segments to be used as input to the surface fitting step, autonomous identification and selection of the n^{th} largest regions may be used to automatically input segments into the surface fitting algorithm. Also, segments containing a number of pixels less than a certain threshold may be eliminated as possible candidates for surface fitting.

The surface fitting method we have developed produces acceptable results. Further improvements in this area include more comprehensive handling of outliers in the segments which may be present due to noise in the image or inexact segmentation. This may be done by removing pixels from the data set which yield squared errors much greater than the average squared error per pixel at later iterations other than in the initial computation of the residual. Outliers may also be reduced or eliminated by assigning a confidence value to all the data points in the range image. This confidence value is represented by σ_i in the merit function of Equation 4.29 given in Section 4.4. To correctly assign these

confidence values, the noise in the range data needs to be modeled so that erroneous pixels may be initially identified. The final improvement that could be studied is a methodical computation of the initial eigenvalue analysis threshold, c. We have assigned a range of values to this threshold through heuristic methods. However, this parameter changes proportionally with the amount of noise in the range data contained in the segment. Therefore, a method of setting this threshold may incorporate the modeling of the noise present in the original range data. Finally, additions to the types of geometric primitives may be added to the set we have already. Not only may geometric primitives be added to this set, but surfaces such as deformable models and superquadrics may also be added.

In conclusion, we have obtained the goal set forth for this surface reconstruction system. We have also set forth, in this chapter, a number of improvements and additions to the system that could be used to surpass the goals previously set forth, making this surface reconstruction system totally autonomous and even more robust.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] R. Pito, "Characterization, calibration and use of the Perceptron laser range finder in a controlled enviroment," Tech. Rep. MS-CIS-95-05, Department of Computer and Information Science, University of Pennsylvania, 1995.

[2] K. Johnson, "Development of a versatile wide-angle lens characterization strategy for use in the onmister stereo vision system," Master's thesis, The University of Tennessee, Knoxville, 1997.

[3] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **8**(6), pp. 679–698, 1986.

[4] D. Marr and E. Hildreth, "Theory of edge detection," in *Proc. R. Soc. Lond.*, pp. 187–217, 1980.

[5] K. Boyer and S. Sarkar, "Assessing the state of the art in edge detection: 1992," in *SPIE Applications of Artificial Intelligence X: Machine Vision and Robotics*, pp. 352–362, 1992.

[6] C. Richardson, "A regularized data fusion approach to multisensor surface reconstruction and edge detection applications," Master's thesis, The University of Tennessee, Knoxville, 1996.

[7] M. A. R. Salinas, C. Richardson and R. Gonzalez, "Robotic sensory data fusion: A regularized color edge detection approach," in *38th Symposium on Circuits and Systems. Proceedings*, Aug 1995.

[8] M. Gokmen and C.-C. Li, "Edge detection and surface reconstruction using refined regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**, May 1993.

[9] A. Tikhonov and V. Arsenin, *Solutions of Ill-Posed Problems*, Winston ans Sons, 1977.

[10] M. Abidi, *Fusion of Multi-Dimensional Data Using Regularization*, ch. 10. Academic Press, Inc.: Harcourt Brace Jovanovich, 1992.

[11] W. Snyder, Y.-S. Han, G. Bilbro, R. Whitaker, and S. Pizer, "Image relaxation: restoration and feature extraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence* , pp. 620–624, June 1995.

[12] R. Krishnapuram and S. Gupta, "Morphological methods for detection and classification of edges in range images," *Journal of Mathematical Imaging and Vision* **2**, pp. 351–375, December 1992.

[13] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, 1992.

[14] L. Gee and M. Abidi, "Segmentation of range images using morphological operations: Review and examples," in *SPIE Proceedings: Intelligent Robots and Computer Vision XIV: Algorithms, Techniques, Active Vision, and Materials Handling*, 1995.

[15] T. Esselman and J. Verly, "Feature extraction from range imagery using mathematical morphology," in *Visual Communications and Image Processing II, Proc. Soc. Photo-Opt. Instrum. Eng.*, vol. 845, pp. 233–240, 1987.

[16] J. Lee, R. Haralick, and L. Shapiro, "Morphological edge detection," *IEEE Journal of Robotics and Automation* **Ra-3**, pp. 142–156, 1987.

[17] B. Parvin and G. Medioni, "Adaptive multiscale feature extraction from range data," *Computer Vision, Graphics, and Image Processing* **45**, pp. 346–356, March 1989.

[18] S. Burgiss, R. Whitaker, and M. Abidi, "Range image segmentation through pattern analysis of multi-scale difference information," in *SPIE: Intelligent Robots and Computer Vision XVI: Algorithms, Techniques, Active Vision, and Materials Handling*, pp. 374–381, 1997.

[19] S. Mallat and S. Zhong, "Characterization of signals from multiscale edges," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**, July 1992.

[20] M. D. Heath, S. Sarkar, T. Sanochi, and K. W. Bowyer, "A robust visual method and assessing the relative performance of edge-detection algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**, pp. 1338–1359, December 1997.

[21] P. J. Besl, *Surfaces In Range Image Understanding*, Springer-Verlag, 1988.

[22] H. Maas and P. Krekel, "Automatic geometric feature extraction from depth images of a structured environment," in *Progress in image analysis and processing III: proceedings of the 7th International Conference on Image Analysis and Processing*, pp. 166–173, 1993.

[23] G. Medioni and R. Nevatia, "Description of 3-d surfaces using curvature properties," in *Proceedings of the Image Understanding Workshop*, pp. 291–299, 1984.

[24] M. Brady, J. Ponce, A. Yuille, and H. Asada, "Describing surfaces," in *Proceedings 2nd International Symposium on Robotics Research*, 1985.

[25] S. Burgiss, "Range image segmentation through pattern analysis of difference information from the multi-scale wavelet transform," Master's thesis, The University of Tennessee, Knoxville, 1998.

[26] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition* , 1993.

[27] M. Wani and B. Batchelor, "Edge-region-bases segmentation of range images," *IEEE Transactions of Pattern Analysis and Machine Intelligence* , 1994.

[28] S. M. Bhandarkar and A. Siebert, "Integrating edge and surface information for range image segmentation," *Pattern Recognition* **25**(9), pp. 947–962, 1993.

[29] B. Sabata, F. Arman, and J. Aggarwal, "Segmentation of 3d range images using pyramidal data structures," *CVGIP: Image Understanding* **53**, pp. 373–387, May 1993.

[30] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldof, K. Boyer, D. W. Eggert, A. Fitzgibbon, and R. B. Fisher, "An experimental comparison of range image segmentation algorithms," *IEEE Transactions of Pattern Analysis and Machine Intelligence* , 1996.

[31] P. J. Besl and R. C. Jain, "Segmentation through variable-order surface fitting," *IEEE Transactions of Pattern Analysis and Machine Intelligence* , March 1988.

[32] G. Taubin, "Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**, pp. 1115–1138, November 1991.

[33] J. Mukerjee, P. Das, and B. Chatterji, "Segmentation of range images," *Pattern Recognition* **25**(2), pp. 1141–1156, 1992.

[34] T. I. C. Young Soo LIM and K. H. PARK, "Range image segmentation based on 2d quadratic function approximation," *Pattern Recognition Letters* , pp. 699–708, October 1990.

[35] R. W. Taylor, M. Savini, and A. P. Reeves, "Fast segmentation of range imagery into planar regions," *Computer Vision, Graphics, and Image Processing* **45**, pp. 42–60, January 1989.

[36] S. J. Dickinson, D. Metaxas, and A. Pentland, "The role of model-based segmentation in the recovery of volumetric parts from range data," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**, pp. 259–267, 1997.

[37] D. Terzopolous and D. Metaxas, "Dynamic 3d models with local and global deformations: Deformable superquadrics," *IEEE Transactions of Pattern Analysis and Machine Intelligence* , pp. 703–714, 1991.

[38] D. Terzopolous and D. Metaxas, "Dynamic deformation of solid primitives with constraints," *IEEE Transactions of Pattern Analysis and Machine Intelligence* , pp. 703–714, June 1993.

[39] D. Metaxas and D. Terzopolous, "Shape and nonrigid motion estimation through physics-based synthesis," *IEEE Transactions of Pattern Analysis and Machine Intelligence* , June 1993.

[40] I. Cohen, L. D. Cohen, and N. Ayache, "Using deformable surfaces to segment 3-d images and infer differential structures," *CVGIP: Image Understanding* **56**, pp. 242–263, September 1992.

[41] N. Yokoya and M. D. Levine, "Range image segmentation based on differential geometry: A hybrid approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11**, pp. 643–649, June 1989.

[42] M. Baccar, "Surface characterization using a gaussian weighted least squares technique towards segmentation of range images," Master's thesis, The University of Tennessee, Knoxville, 1994.

[43] J. Serra, *Image Anaylsis and Mathematical Morphology Volume 1*, Academic Press, Inc., 1982.

[44] E. Dougherty, ed., *Mathematical Morphology in Image Processing*, Marcel Dekker, Inc., 1993.

[45] J. Russ, *The Image Processing Handbook*, CRC Press, 1995.

[46] R. Gonzalez, M. Baccar, and M. Abidi, "Segmentation of range images via data fusion and morphological watersheds," in *Proceedings of the 8th Scandinavian Conference on Image Analysis*, vol. 1, 1993.

[47] K. L. Boyer, M. J. Mirza, and G. Ganguly, "The robust sequential estimator: A general approach and its application to surface organization in range data," *IEEE Transactions on Pattern Analysis and Machine Intelligence* , pp. 987–1001, October 1994.

[48] J. V. Miller and C. V. Stewart, "Muse: Robust surface fitting using unbiased scale estimates," *IEEE Computer Society Conference on Pattern Recognition an Image Processing, Proceedings* , pp. 300–306, 1996.

[49] L. S. Gao and H. Kawarada, "Applications of fuzzy average to curve and surface fitting," in *Proceedings of 1995 IEEE International Conference on Fuzzy Systems. The International Joint Conference of the Fourth IEEE Internation Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium*, pp. 971–978, 1995.

[50] G. Taubin, "An improved algorithm for algebraic curve and surface fitting," in *Proceedings Fourth Internation Conference on Computer Vision*, pp. 658–665, 1993.

[51] F. P. Ferrie, J. Lagarde, and P. Whaite, "Darboux frames, snakes, and super-quadrics: Geometry from the bottom up," *IEEE Transactions on Pattern Analysis and Machine Intelligence* , pp. 771–784, August 1993.

[52] S. Kumar, S. Han, D. Goldgof, and K. Bowyer, "On recovering hyperquadrics from range data," *IEEE Transactions on Pattern Analysis and Machine Intelligence* , pp. 1079–1083, November 1995.

[53] D. DeCarlo and D. Metaxas, "Shape evolution with structural and topological changes using blending," Tech. Rep. MS-CIS-97-12, Department of Computer & Information Science: University of Pennsylvania, Philadelphia, PA, 1997.

[54] D. Keren, D. Cooper, and J. Subrahmonia, "Describing complicated objects by implicit polynomials," *IEEE Transactions on Pattern Analysis and Machine Intelligence* , pp. 38–53, January 1994.

[55] Y. Chen and G. Medioni, "Fitting a surface to 3-d points using an inflating balloon model," in *Proceedings of the IEEE CAD-Based Vision Workshop*, pp. 266 – 281, 1994.

[56] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Computer Graphics Proceedings. SIGGRAPH '96*, pp. 303–312, 1996.

[57] A. Elfes and L. Matthies, "Sensor integration for robot navigation: combining sonar and range data in a grid-based representation," in *Proceedings of the 26th IEEE Conference on Decision and Control*, pp. 1802–1807, 1987.

[58] D. Elsner, "Volumetric modeling through fusion of multiple range images with confidence estimate," Master's thesis, The University of Tennessee, Knoxville, 1997.

[59] R. M. Bolle and B. C. Vemuri, "On three-dimensional surface reconstruction methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**, January 1991.

[60] M. A. Abidi, R. O. Eason, and R. C. Gonzalez, "Autonomous robotic inspection and manipulation using multisensor feedback," *Computer* , April 1991.

[61] D. Jung and K. K. Gupta, "Spatial occupancy recovery from a multiple-view range imaging system for path planning applications," in *1995 Internation Conference on Systems, Man and Cybernetics. Intelligent Systems for the 21st Century*, pp. 472–477, 1995.

[62] V. Sequeira, J. G. Goncalves, and M. I. Ribeiro, "3d environment modelling using laser range sensing," *Robotics and Autonomous Systems* **16**, pp. 82–91, 1995.

[63] E. Walker, "Exploiting geometric relationships for object modeling and recognition," in *SPIE: Intelligent Robots and Computer Vision IX: Neural, Biological, and 3-D Methods*, pp. 353–363, 1990.

[64] V. Sequeira, J. G. M. Goncalves, and M. I. Ribeiro, "High-level surface descriptions from composite range images," in *Proceedings Internation Symposium on Computer Vision*, pp. 163–168, 1995.

[65] G. Turk and M. Levoy, "Zippered polygon meshes from range images," in *Computer Graphics Proceedings. Annual Conference Series 1994. SIGGRAPH 94 Conference Proceedings*, 1994.

[66] G. Zhang and A. Wallace, "Physical modeling and combination of range and intensity edge data," *CVGIP: Image Understanding* **58**, pp. 191–220, September 1993.

[67] O. R. P. Bellon, J. E. C. Castanho, and C. L. Tozzi, "A hybrid approach for solving the range image segmentation/reconstruction problem," in *1995 IEEE International Conference on Systems, Man and Cybernetics. Intelligent Systems for the 21st Century*, 1995.

[68] R. T. Whitaker, "A level-set approach to 3d reconstruction from range data," Tech. Rep. EE-96-07-01, Imaging, Robotics, and Intelligent Systems Laboratory: Department of Electrical Engineering: The University of Tennessee, Knoxville, 1996.
To appear in International Journal of Computer Vision.

[69] A. E. Johnson, R. Hoffman, J. Osborn, and M. Hebert, "A system for semi-automatic modeling of complex environments," in *International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pp. 213–220, May 1997.

[70] A. Waksman and A. Rosenfeld, "Sparse, opaque three-dimensional texture 1. absorbant patterns," *CVGIP: Image Understanding* **57**, May 1993.

[71] R. P. Menon and R. S. Acharya, "Three dimensional surface representation of dna replication sites using finte element based deformable models," in *International Society for Optical Engineers*, pp. 516–522, 1996.

[72] C. Oxturk, S. Dubin, M. E. Schafer, W.-Y. Shi, and M.-C. Chou, "A new structured light method for 3-d wound measurement," in *Proceedings of the IEEE Twenty-Second Annual Northeast Bioengnieering Conference*, pp. 70–71, 1996.

[73] R. W. Collier, "Construction and characterization of a range scanning system utilizing a point laser rangefinder," Master's thesis, The University of Tennessee, Knoxville, 1998.

[74] I. S. Kweon, R. Hoffman, and E. Krotkov, "Experimental characterization of the perceptron laser rangefinder," Tech. Rep. CMU-RI-TR-91-1, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1991.

[75] R. T. Whitaker, "Geometry-limited diffusion in the characterization of geometric patches in images," *CVGIP: Image Understanding* **57**, pp. 111–120, January 1993.

[76] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. Pattern Analysis and Machine Intelligence* **12**, pp. 429–439, 1990.

[77] G. J. Klir and T. A. Folger, *Fuzzy Sets, Uncertainty, and Information*, Prentice Hall, 1988.

[78] G. Shafer, *A Mathematical Theory of Evidence*, Princeton University, 1976.

[79] P. Rosin, A. Colchester, and D. Hawkes, "Early image representation using regions defined by maximum gradient profiles between singular points." Scheduled to appear in Pattern Recognition.

[80] W. H. Press, S. A. Teukolsky, W. T. Vettering, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing: Second Edition*, Cambridge University Press, 1992.

[81] R. T. Whitaker, "Sparse-field volumetric deformable models for 3d reconstruction," Tech. Rep. EE-96-07-01, Imaging, Robotics, and Intelligent Systems Laboratory: Department of Electrical Engineering: The University of Tennessee, Knoxville, 1996.

VITA

Eric Douglas Lester was born in Nashville, TN in 1973. He lived in Norene, TN and graduated high school from Friendship Christian School in May 1991. Later that year he began work toward his BSEE at the University of Tennessee-Knoxville. He completed the degree in December 1995 with a specialization in image processing. Mr. Lester enrolled in the MSEE program at the University of Tennessee-Knoxville in 1996 where he is currently working as a Graduate Research Assistant in the Imaging, Robotics and Intelligent Systems Laboratory under the supervision of Dr. R. T. Whitaker and Dr. M. A. Abidi. Eric will graduate in May 1998 with a specialization in image processing and robotic vision.