ABSTRACT

Title of thesis: A SYSTEM FOR 3D SHAPE ESTIMATION AND TEXTURE EXTRACTION VIA STRUCTURED LIGHT

Richard John Miller III, Master of Science, 2010

Thesis directed by: Dr. Rama Chellappa
Department of Electrical and Computer Engineering

Shape estimation is a crucial problem in the fields of computer vision, robotics and engineering. This thesis explores a shape from structured light (SFSL) approach using a pyramidal laser projector, and the application of texture extraction. The specific SFSL system is chosen for its hardware simplicity, and efficient software. The shape estimation system is capable of estimating the 3D shape of both static and dynamic objects by relying on a fixed pattern. In order to eliminate the need for precision hardware alignment and to remove human error, novel calibration schemes were developed. In addition, selecting appropriate system geometry reduces the typical correspondence problem to that of a labeling problem. Simulations and experiments verify the effectiveness of the built system. Finally, we perform texture extraction by interpolating and resampling sparse range estimates, and subsequently flattening the 3D triangulated graph into a 2D triangulated graph via graph and manifold methods.

A SYSTEM FOR 3D SHAPE ESTIMATION AND TEXTURE EXTRACTION VIA STRUCTURED LIGHT

by

Richard John Miller III

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2010

Preface

I would like thank the people who supported this thesis, as it would not have been accomplished without their help. First, my advisor Dr. Chellappa for his support during my time at UMD, Ascendant Engineering Solutions (AES) in Austin, and for the committee members Dr. Wu and Dr. Davis. I would also like to thank my office-mates Ashish, Vishal and Garrett for all their time spent whiteboarding with me during this process.

The work contained in this thesis originated as work on a proof-of-concept device for my employer AES. The intent of the system was to quickly determine the 3D shape of a person's face, arms, hands etc. in an unobtrusive way. A guiding design principal was to use minimal hardware and efficient software so as to implement future version on embedded electronics in a small form factor (such as a handheld device). The implemented proof-of-concept system is limited to sampling small objects and surfaces. While the original intent of this work focused on scanning smooth surfaces, much effort has been made to generalize the approach for other objects.

Dedication

To my wife, parents, and sister, whose love and support made this work possible.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years, cameras have proliferated into almost every aspect of modern life, effortlessly capturing snapshots and video. The sciences of computer vision, digital signal processing, and image processing allowed for this aggressive expansion of technology, yet there remain numerous problems yet to be solved. One of the cornerstone problems in the field of computer vision is 3D shape estimation, i.e. given an image or set of images of some scene, attempt to recover models for the objects in that scene. Since images are 2D projections of 3D objects, this is typically a very difficult problem, as an entire dimension has vanished. This thesis explores a structured light approach to solve this problem, along with a specific application termed texture estimation. The goal of the structured light approach taken in this thesis is to estimate the global shape of some surface (whether it be stationary or dynamic) in a fast and efficient manner. Such a system is highly scalable, allowing for the miniaturization of the technology. While the primary focus of this thesis is 3D shape estimation, the problem of texture estimation is also analyzed in conjunction with the structured light approach. This is a natural pairing of problems, since 3D shape estimation requires surface interpolation schemes, denoising, and shape analysis, all of which apply directly to the problem of texture extraction. This chapter details the motivation and outline for the shape and texture estimation system described throughout this thesis.

1.1 System Motivation

The need for shape and texture estimates occurs in many engineering practices, such as pattern recognition, machine vision for part inspection, medical imaging, and many others. In a part inspection setting, one may wish to measure feature distances on various shaped parts as a quality metric. Traditional approaches, include using complicated and expensive telecentric lenses or line scanners that preserve object level distances when projected onto the camera sensor plane. If a handheld tool could perform the same function, this would increase the flexibility of the manufacturing line by coping with odd shaped parts, or complicated configurations. To do so, the shape estimation system would need to be capable of scanning a surface in a quick fashion, without any complicated calibration or large projectors. In book archiving, expensive industrial scanners must be used in order for optical character recognition (OCR) algorithms to function properly (as these scanners physically flatten the book's pages, or scan in a sophisticated manner). A much cheaper and flexible solution would utilize an ordinary off-the-shelf camera, operated by the user in a handheld fashion. If such a camera were used to image a book's pages, the shape distortion would need to be accounted for in order for the OCR algorithms to work. By first estimating the shape of the book with a shape estimation system, the book archiving process is simplified into that of taking a single picture of a page. The texture extraction system would then generate an image of the page as if it were imaged via the industrial scanner. In medical imaging, a doctor may want to compare a skin abnormality against a database to diagnose a disease. This would involve running image-processing algorithms on images of skin in order to analyze them in addition to studying the topography of the problem area. Using a shape and texture estimation system would allow the doctor to scan the target area quickly, yielding a 3D model of the area as well as a large flattened image (which could be compared to the database, independent of the skin topography). The guiding principles in the design of this shape and texture estimation system are simple hardware requirements (i.e. little to no calibrated hardware or precision alignment), scalability (i.e. the solution can be implemented for a hand held device or a lab-oriented machine), and modularity so that the various hardware and software components are easily interchangeable and upgradeable. By not requiring calibrated, precision hardware, we open up the possibility to miniaturize the technology. This scalability will enable approaches, such as the one described in this thesis to perform a wide variety of novel tasks, even outside the particular application of texture extraction.

1.2 Outline of System

The system described in this thesis utilizes an engineered structured light source to extract shape information from a visible surface. This structured light source originates from a single laser beam, which passes through a diffractive optical element (DOE). In general, DOEs are capable of producing various patterns ranging from lines to entire images; however, the DOE selected in this thesis produces a grid of dots (also referred to as a pyramidal laser projector). This grid of dots reflects off an object of interest, and is subsequently imaged by a camera. Thus, the shape estimation occurs from only a single image of some surface with the projected grid. Since the pyramidal projector samples the surface at a finite number of discrete points which are in turn imaged by a camera, the shape estimation procedure can be performed in rapid succession–enabling the system to scan moving objects without much motion distortion. Figure 1.1 depicts the basic hardware elements.

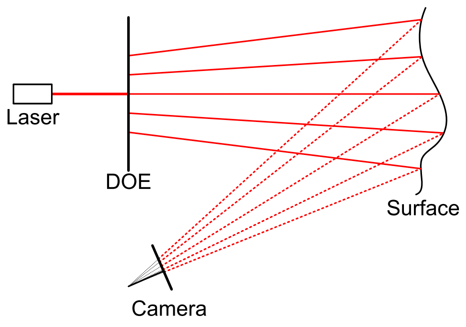


Figure 1.1: Diagram of shape estimation system.

The software to accomplish shape and texture estimation relies on several modules, as shown in Figure 1.2. The first module is the calibration module, which

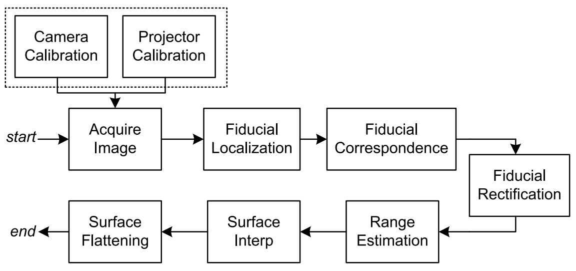


Figure 1.2: Diagram of shape and texture estimation software.

computes essential information for depth estimation. Fortunately, calibration can be pre-computed, and retained for many shape estimation sessions. The next two modules, fiducial location and labeling, contain the image-processing required to transform an input color image into a vector of fiducial image locations. The fiducials locations are rectified by the following block if appropriate calibration data is available. Next, the range from the camera to each reflected fiducial is estimated. Subsequently, the next module performs surface interpolation of the range data to generate a dense topographic map of the object's surface. Lastly, in the specific application of texture extraction, the flattening module produces a flattened version of the imaged texture using the original image and the dense estimated topographic map.

1.3 Outline of Thesis

This thesis first describes the problem of 3D shape estimation from 2D imagery. While there are numerous approaches to this problem, Chapter 2 focuses primarily on structured light due to its robustness, hardware simplicity, and scalability. After the general framework of structured light is presented, a detailed mathematical view into the mechanics of structured light and associated image processing requirements is given in Chapter 3. Chapter 4 uses the 3D shape estimation framework developed in the previous chapter to focus on the specific application of texture extraction using sparse 3D range measurements. This sparse measurement scheme is dealt with via interpolation methods, which are also outlined in Chapter 4. Chapter 5 provides the conclusion and future work relating to this thesis.

Chapter 2

3D Shape Estimation, Theory, and Techniques

This chapter introduces the problem of 3D shape estimation, and investigates the varied approaches taken to estimate the shape of an object. Since this thesis specifically focuses on structured light shape estimation, a review of structured light approaches is given, along with the physical considerations of the specific shape estimation system implemented.

2.1 Vision Based 3D Shape Estimation Schemes

As mentioned before, estimating the 3D shape of an object is one of the cornerstone problems in the field of computer vision. Humans have an innate understanding of the 3D world, yet observe objects with only 2D image sensors; likewise, cameras obtain only 2D projections of 3D objects. The study of 3D shape estimation is therefore important in order to develop algorithms and techniques that allow computers to mimic or even improve upon our shape estimation ability. The need for 3D shape estimation occurs in many fields such as medical imaging, robot control and planning, machine control, part inspection, and many others. There are numerous approaches one may use to extract the 3D shape from a 2D image, which can be separated into two categories, passive and active shape estimation[1].

Passive 3D shape estimation involves estimating the shape of an object by remotely sensing the object (typically with a digital image sensor), using only ambient excitation sources such as room lighting or outdoor illumination. The term "passive" in effect means that one must not transmit a signal to the object in attempts to recover information; in other words, a passive shape estimation system is a sensing only modality with no transmission. In computer vision, there are many approaches that fall under this category such as stereo-vision [3], shape-from-shading [4, 5], shape-from-texture [6, 7], shape-from-motion [8, 9], and many others. A vast amount of literature is available on these topics, as shape estimation from only a 2D projection is a compelling and difficult problem (that is often ill posed). These techniques often must assume substantial constraints or simplifications in order to become tractable, such as Lambertian reflectance in shape-from-shading, etc. To operate in a generalized capacity, active approaches that have both transmitting and sensing modalities provide additional information in the shape recovery problem.

Active vision[2] shape estimation schemes involve transmitting a signal of some sort to the object prior to sensing. It is intuitive that active shape estimation should simplify the 3D shape estimation problem since the observer now gets to "touch" the

[1]There are many other optical methods available to solve the 3D shape estimation problem, a good overview is provided by Chen et al. [1] and Besl [2].

[2]Though the term active can also be used to describe any system where the camera non-stationary, the term is used here to discriminate between sensing modalities that receive or both transmit and receive light

object directly via the transmitted signal. There are numerous methods available categorized as active shape estimation techniques including active stereo, light in flight, structured light, and many others. This thesis focuses on structured light approaches to the shape estimation problem because of their scalability, simplicity of hardware, and robust estimation results. The following section outlines the history of structured-light, and establishes the proposed system in a framework of the previous approaches.

2.2 Structured Light Approaches

Shape from Structured Light (SFSL) is a process in which range information (i.e. 3D shape) is estimated from an object by imaging a scene containing some projected pattern. Such systems contain an image sensor, and a light-projector of some sort (which can range from LCD projectors, to lasers projectors). The SFSL technique is analogous to stereo vision methods; with SFSL however, one of the passive cameras is replaced with a transmitting light source. As an *active* method for shape estimation, the transmitted signal determines in part what information is revealed by the observed scene. In particular, the arrangement (or structure) of the light pattern in SFSL determines how 3D shape information projects to the image domain. Thus, there exists two primary components of SFSL that engineers can change according to their application needs, the structure of the light (the coding)–that is, how the light is arranged spatially and temporally, and the shape estimation scheme–how the shape is estimated after observing the scene (including selecting the calibration scheme). There are many different structures of light used in the field of SFSL; we will examine three important categories, non-coded patterns, spatially encoded patterns, and spatio-temporally coded patterns in Section 2.2.1.

2.2.1 Structured Light Patterns

Various patterns have been used successively in the field of SFSL. These range from single scanning lines, to complex spatio-temporally encoded pattern sets, each with advantages and disadvantages. Salvi et al.'s review of pattern codification strategies [10] groups these patterns into three distinct groups: time-multiplexing, spatial neighborhood, and direct coding. We will first explore light-stripes, followed by spatially-encoded patterns, and lastly spatio-temporally encoded patterns. Light stripes and spatially-encoded patterns belong to the spatial neighborhood and direct coding categories, while spatio-temporal patterns belong primarily to time-multiplexing and secondarily to the other two groups.

Light Stripe Patterns: Light stripes are perhaps the oldest form of SFSL, originating in the 1970's with the pioneering work from Shirai et al. [11], who developed one of the first systems for estimating polyhedrons via a sliding slit projector, and Will et al. [12], who used a Fourier approach to infer information about the shape of objects. Shape estimation in these cases is performed by scanning for line segments, and hence detecting simple polyhedral scenes. Recent approaches such as [13], use light stripes in a probabilistic framework. All of these approaches involve projecting a stripe of light onto a target object, and imaging the resulting distorted pattern. Since the light stripe method utilizes a continuous line, dense range estimates are obtained wherever the stripe is projected upon. However, to fully scan an object, the light stripe must pass over the the scene (either by projector movement or object movement), thus restricting such systems to capturing static scenes, but with high depth resolution.

Spatially-Encoded Patterns: Related to light stripes are the projected grids/arrays patterns (direct encoding), and more generally spatially-encoded patterns. Where light stripe systems require scanning to obtain a full set of measurements, projected patterns require only a single image of the object overlaid with the projected pattern. Thus, projected grids/arrays sample the target surface at the grid points or intersections. If the target surfaces are fairly smooth, these sparse measurements can provide adequate reconstruction results in a fast, reliable manner. When the pattern is a grid of lines, such as in the case of [15] and [16], the spacing of the lines as well as the thickness of each line determines the accuracy of the surface sampling. When the pattern is instead a grid of dots such as in [17], the spacing of the dots determines the accuracy, as well as the difficulty of the correspondence problem. The advantage of such systems is the ability to capture non-stationary objects (by sacrificing some surface sampling density). More general spatially-encoded patterns extend the ability of direct coding by taking advantage of neighborhood relations among pixels such as in Salvi et al. [18] using a color-encoded grid, and Albitar et al. [19] using primitives based encoding. These encoding schemes can increase the sampling density of the system by eliminating the correspondence problem at the expense of complexity, as pixels must now be decoded in order to obtain the correct correspondences. Grid based patterns solve the correspondence problem by utilizing proper system configuration which ensure unique decodings at the expense of spatial resolution.

Spatio-Temporal Patterns: An alternative to the temporally-fixed patterns such as described before are spatio-temporally coded patterns. Since these patterns actually consist of several separate patterns, techniques such as multi-resolution, time-averaging, and space-coding may be taken advantage of. These methods typically involve projecting a set of patterns, and imaging the reflected scene for each pattern such as in [20, 21, 22]. The time-averaging aspect of spatio-temporal light patterns was investigated by Curless et al. [22], which provides a study of the error in single-shot triangulation schemes. Error sources include reflectance discontinuities, object corners, shape discontinuity, and sensor occlusion. By moving the light source, these errors are averaged out over time. Horn and Kiryati [21] derived the mean-squared error (MSE) optimal spatio-temporal structured light pattern, using the communications notion of embedding $l-$points in a $k-$dimensional space. The resulting optimal pattern is the 3D-Hilbert curve which may be interpreted as projecting k gray level planes. Since the spatio-temporal patterns generally rely on projecting several patterns sequentially, observing dynamic scenes is not feasible.

Given the available methods, it is clear that spatially coded patterns are the best suited patterns for a nominally real-time SFSL system for use with both static and dynamic scenes. Of these spatially coded patterns, direct-coded systems such as grid patterns allow for efficient 3D shape estimation without the overhead of pixel neighborhood decoding, thus allowing for scalability in size, resolution and speed. The selected pattern used in this thesis is the pyramidal grid pattern as discussed in Section 2.3.

2.3 Diffractive Optical Element Projection

A convenient method for generating a pyramidal grid pattern is via a diffractive optical element (DOE) which is a thin-film capable of diffracting light into engineered patterns [23]. The DOE uses principles from diffraction gratings to generate patterns from a single light source. Figure 2.1 depicts a simulated DOE projector with 4x4 output spot lasers. This model is parameterized by the equivalent "focal" length of the projector (represented here by the single red line segment), the number of lasers in the $x-$ and $y-$ dimensions, and the angle separating neighboring beams. In this figure, red lines denote laser path from the DOE. We designate two parallel planes that define the capture volume of the system. The interception of the pyramidal laser pattern with each plane is denoted by a blue maker, with a blue line segment connecting the nearest and furthest points of the capture volume.

When using a DOE projector, laser spots (fiducials) are projected into the scene and subsequently imaged by a camera. Since the resulting image will only contain some surface superimposed with a grid of laser fiducials, we rely on direct correspondences to identify which fiducial belongs to which laser in order to deter-

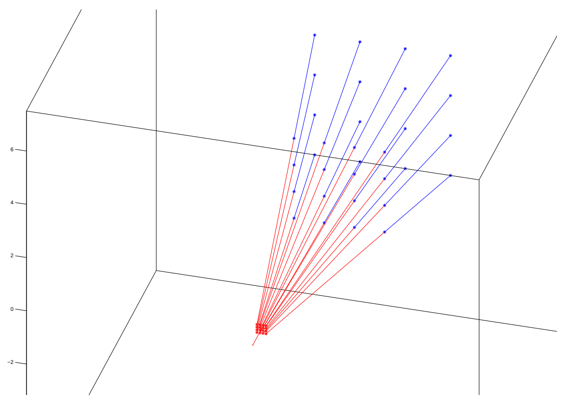


Figure 2.1: DOE laser projector with capture volume marked.

mine the range to that point in space. Previous work by Dipanda, et al. [17, 24] coined the term *Configurations of System* (COS), the relative position of the camera with respect to the laser projector. Valid COS provide unambiguous fiducial locations at every range within the specified capture volume, as well as sufficiently high spatial resolution. Thus, if the laser projector and camera are arranged within a valid COS, any surface imaged will yield a distorted pattern of fiducials from which a 3D shape can be estimated. The different valid COS cases for this SFSL system are considered in Section 3.1.1.

2.3.1 Surface Sampling

The structured light approach taken in this thesis gathers only sparse range estimates from target surfaces. While these few measurements allow the system to operate very quick and efficiently, it means that some surface features are not imaged by the SFSL system. Since the DOE projector can only project a finite number of lasers, the spatial density of these lasers will, to a large extent, determine the fidelity of 3D shape reconstruction. While this method will provide fewer samples than other SFSL methods, the trade off is increased speed and lower system complexity. The immediate questions are how to control the granularity of the resulting depth map and what does this entail physically for the DOE projector? To control the resolution of the surface sampling, one must change the angle between adjacent lasers from the DOE. In simulations, this is controlled by increasing the projector's focal length for a higher resolution, or decreasing it for a lower resolution, as illustrated by the 2D case in Figure 2.2. The obvious side effect is that the capture area decreases for a fixed number of lasers as the angle between lasers is decreased. A less obvious side effect is that for different surfaces, a higher-resolution sampling results in more laser registration errors, when, for example, point x'_2 moves to the left of point x_1 in the image. Thus, if we restrict our SFSL setup so that these errors are not permitted, we not only decrease the capture area, but the capture volume as well.

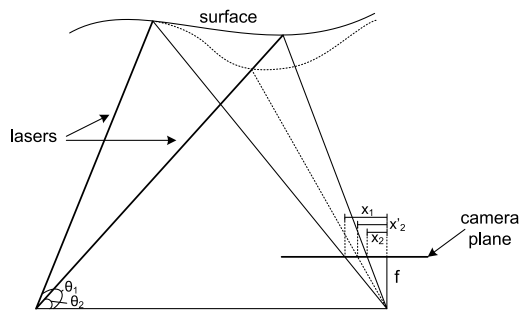


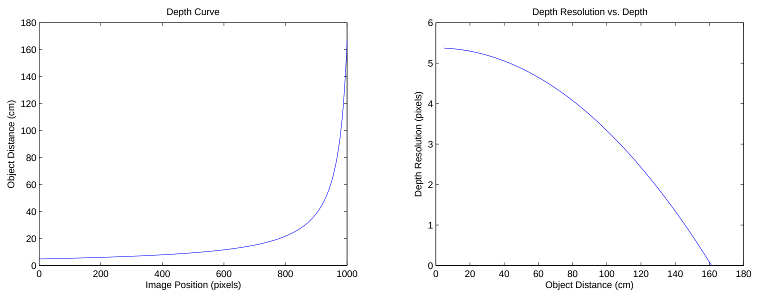
Figure 2.2: Diagram of surface sampling density's relationship to the image plane.

2.3.2 Depth Resolution Analysis

In the projective geometry of the pinhole camera, the resolution of an object projected onto the image plane is related to the distance the object lies from the camera. The general relationship between imaged points from an object at various depths can be visualized by a depth-curve; an example of which is shown in Figure 2.3, whose equation is given by,

$$d = \frac{A}{B - x}, \tag{2.1}$$

where A and $B \in \mathbb{R}$ are constants, and d and $x \in \mathbb{R}$ are variables. The mathematics governing this relationship are derived in Section 3.1. Figure 2.3(b) shows how the



(a) Depth Curve (b) Resolution vs Depth

Figure 2.3: Example depth curve with associated depth resolution plot.

depth resolution drops off significantly when the object is far away from the camera. Depth resolution is derived from the depth equation by computing the number of pixels per fixed distance amount. When designing a SFSL system, it is desirable to mitigate depth resolution drop off by careful selection of the camera/ projector geometry.

Chapter 3

3D Shape Estimation from Structured Light via DOE Projector

In order to design a SFSL system, it is necessary to understand the underlying mathematics. Chapter 1 outlined the algorithm requirements for SFSL and texture extraction; this chapter first formulates the mathematics behind SFSL in Section 3.1, and then presents the key algorithmic components of the system; namely, fiducial localization in Section 3.2 (finding laser fiducials in a single image), fiducial labeling in Section 3.3 (the correspondence problem), triangulation (depth estimation) and calibration in Section 3.5.

3.1 Geometry of SFSL: The Role of Epipolar Geometry

The use of a DOE projector requires a thorough examination of the geometry governing the projector-camera relationship. We begin by establishing the *central-projection* camera, the ideal pinhole camera[1], which maps points in $\mathbb{R}^3$ to the image domain $\mathbb{R}^2$. The pinhole camera model is illustrated in Figure 3.1. Simply put, the pinhole camera creates an image by connecting a 3D point to the camera center $\mathbf{C}$ with a line. The intersection of this line and the image plane (located f units away from the camera center C along the principal axis) determine the projected

[1]This section assumes the ideal pinhole camera model; nonlinearities and distortions will be dealt with later during the camera calibration stage in Section 3.5

location of the 3D scene point. In other words, this geometry produces a perspective projection from the 3D scene point, $\mathbf{X}$, to the 2D image point, $\mathbf{x}$. The mathematical

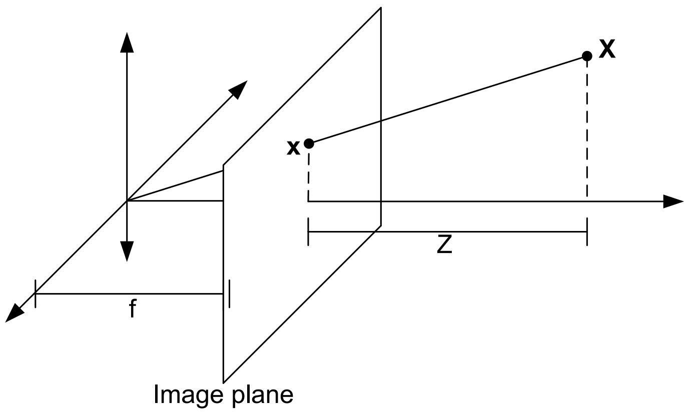


Figure 3.1: Ideal pinhole camera model.

expression for this geometry is best expressed using homogeneous coordinates, that is, a vector coordinate in $\mathbb{R}^d$ that have been augmented to a $d+1$ vector. Using this notation allows for the matrix-vector representation as follows,

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{3.1}$$

where image pixels are $x = \frac{U}{W}$ and $y = \frac{V}{W}$. More compactly expressed,

$$\tilde{\mathbf{x}} = \tilde{P}\tilde{\mathbf{X}} = [P|\mathbf{0}_3]\,\tilde{\mathbf{X}} \tag{3.2}$$

where $\tilde{P}$ is the $3x4$ projective matrix from Equation 3.1, $\tilde{\mathbf{X}}$ are homogeneous 3D points and $\tilde{\mathbf{x}}$ are 2D image points. Using Equation 3.1, we can now take points from the 3D scene and project them through the ideal pinhole camera to the image plane using the camera coordinate frame; however, it is often convenient to consider two different Euclidean coordinate frames, one called the world coordinate frame, the other being the camera coordinate frame. This allows for the trivial expression of 3D points such as ones defined on some plane such as a calibration board. We define the relationship between a 3D scene point in the world coordinate frame, and its corresponding image points via the rigid body equation,

$$\mathbf{X}_{cam} = R\mathbf{X} + t \tag{3.3}$$

where,

$$\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} \cos(\psi) & 0 & \sin(\psi) \\ 0 & 1 & 0 \\ -\sin(\psi) & 0 & \cos(\psi) \end{bmatrix} \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

represents the 3D rotation matrix about $x-$, $y-$, and $z-$ axis, and

$$\mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{3.5}$$

represents the translation vector. This relationship is visualized in Figure 3.2. Using

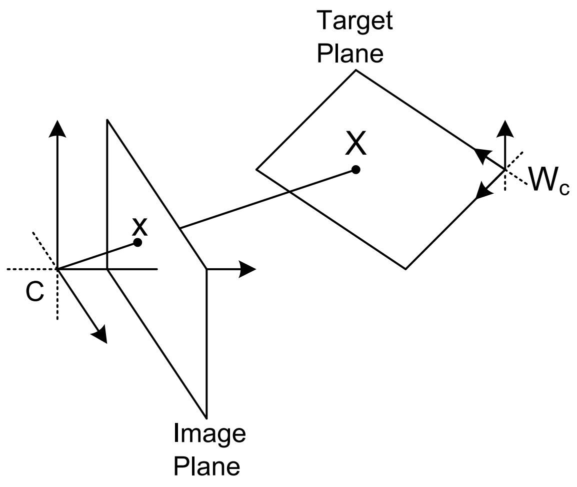


Figure 3.2: Extrinsic relationship between target plane and image plane.

this rigid body motion, we can write new equations for projecting 3D scene points in the world-coordinate system to the image plane (using the transformation between the world coordinate frame and the local camera coordinate frame),

$$\tilde{\mathbf{x}} = \tilde{P} D \tilde{\mathbf{X}} = \tilde{P} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_3^T & 1 \end{bmatrix} \tilde{\mathbf{X}} \tag{3.6}$$

equivalently,

$$\tilde{\mathbf{x}} = P \left[R | \mathbf{t} \right] \tilde{\mathbf{X}}. \tag{3.7}$$

Since SFSL is analogous to a stereo-vision system, it is intuitive that the geometry should be very much the same. The geometry that governs both SFSL and stereo systems is that of two-perspective view geometry, commonly referred to as Epipolar Geometry [25]. Epipolar geometry allows us to explore how 3D points project onto images, how points on the image plane can be transferred to other planes, and perhaps most importantly, how 2D image points from multiple views allow for estimation of their corresponding 3D points in space. The generic two-camera view is presented first, followed by SFSL. In Figure 3.3 below, the epipolar geometry is shown for a two-perspective view. In this case, there are two cameras represented by their image planes and camera centers. The 3D scene point $\mathbf{X}$ maps to the two image planes via the pinhole model described earlier. Without knowing $\mathbf{X}$ and observing only $\mathbf{x}$, the epipolar plane determined by the baseline between the two camera centers and the ray determined by $\mathbf{x}$ constrains the possible locations for $\mathbf{x}'$ in the second image. This constraint is termed the epipolar line. This is intuitive, since $\mathbf{X}$ was projected to $\mathbf{x}$ from anywhere along a specific ray (i.e. an infinite number of possible 3D scene positions), the corresponding projection $\mathbf{x}'$ will also have many possible locations. When one camera is replaced by a laser projector, the same epipolar geometry holds. In Figure 3.4, lasers from the DOE

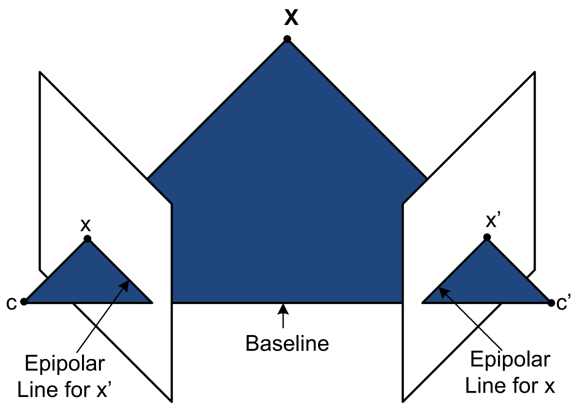


Figure 3.3: Epipolar geometry for two-perspective view.

projector travel outward in the 3D scene space. A surface that intersects these projected, such as points $\mathbf{X}_1$ and $\mathbf{X}_2$, will reflect the ray to the image plane, just as in Figure 3.3, and generate the corresponding $\mathbf{x}_1$ and $\mathbf{x}_2$.

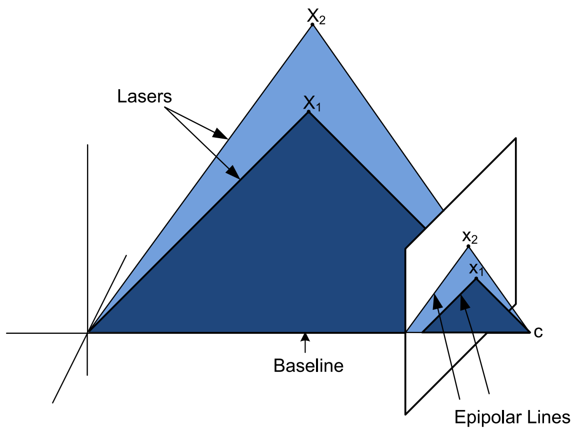


Figure 3.4: Two lasers and their corresponding epipolar lines.

If one can associate known range estimates with points along an epipolar line, then the epipolar line will serve as a measurement device for range estimation. To augment the DOE model presented in Section 2.3, we add a virtual pinhole camera as shown in Figure 3.5.

In this virtual camera, the blue line segments from the capture volume are pro-

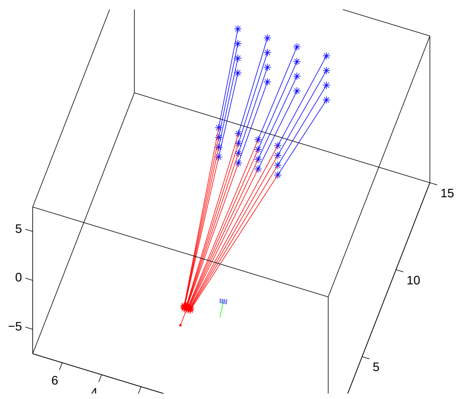


Figure 3.5: Laser projector with known distances marked, with virtual camera.

jected onto the virtual image, explicitly determining the epipolar line corresponding to each laser. The resulting virtual image is shown in Figure 3.6. Here, the leftmost point of each line corresponds to the *nearest* 3D point in the capture volume, while the rightmost point corresponds to the *furthest*. The orientation of these epipolar lines is determined by the orientation of the camera with respect to the projector.

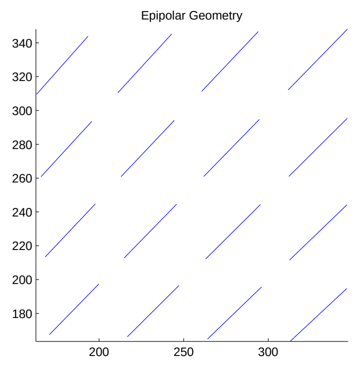


Figure 3.6: Epipolar lines associated with laser projector between two known distances.

The equations that represent the epipolar geometry are very similar to the projective equations discussed before. From the previous figures, we know that there is a mapping from $\mathbf{x} \mapsto \mathbf{l}'$, the epipolar line in the second image. First, we define P as the first view's projection matrix, and P' as the second view's projection matrix. The ray from the left camera's center through the image point x can be written as

$$X(\lambda) = P^{+}\mathbf{x} + \lambda\mathbf{C} \tag{3.8}$$

where C is the camera center defined by $P\mathbf{C} = \mathbf{0}$ and P^{+} is the pseudo-inverse[2]. The epipolar line in the second image, $\mathbf{l}'$, is defined by two points: the first is, conveniently, the left camera's center projected onto the right camera's image plane, and the second is the point along the previously defined ray at infinity projected to the image plane. Thus, $\mathbf{l}' = (P'C) \times (P'P^{+}\mathbf{x})$. In the SFSL system, each laser will have an associated epipolar line. The arrangement of these lines varies as the camera moves with respect to the laser projector. Section 3.1.1 discusses how the geometry changes with the position of the camera.

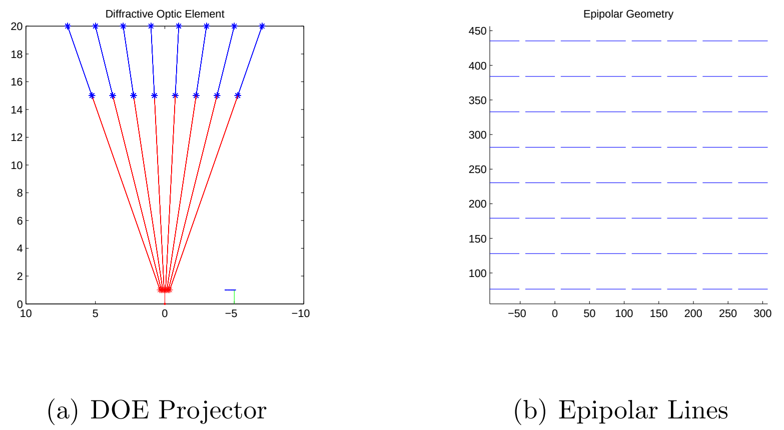
3.1.1 Camera Motion Effects

As the camera moves, the epipolar lines corresponding to the laser fiducial will transform. A valid COS is one where no epipolar line associated with a laser intersects another epipolar line within the capture volume. Below, three different motions are presented, which define all possible camera motions besides a z-axis translation, which would result in an overall zooming (in or out) of the epipolar line image. By understanding these motion effects, we can place the camera w.r.t. the projector to maximize the depth resolution while maintaining separation between

[2]The details of this back-projection equation are given in Section 3.5.2.3

epipolar lines (to ensure a straightforward correspondence).

Camera motion along the baseline: If the camera is displaced along a plane parallel with the projector plane, the epipolar lines will appear horizontal, as shown in Figure 3.7. In this configuration, the epipolar lines will intersect at their endpoints if the capture volume is large enough.

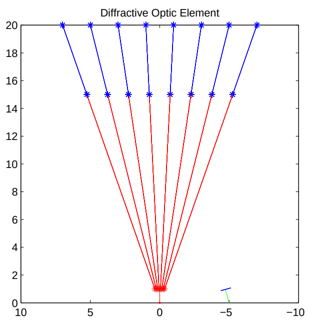


(a) DOE Projector (b) Epipolar Lines

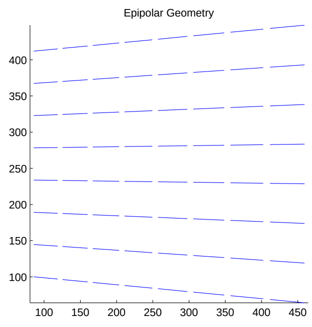
Figure 3.7: Epipolar geometry from y-translated camera.

Converging camera motion along the baseline: If the camera is displaced along the baseline between the projector and the camera (as before), and then allowed to converge (i.e. rotate so that the center ray from the projector and camera meet at a point) the epipolar lines will appear to converge, as shown in Figure 3.8. Note that the epipolar endpoints are still collinear, thus for a large enough capture volume, they will intersect.

Converging, camera motion parallel to projector plane: Most generally, if the camera



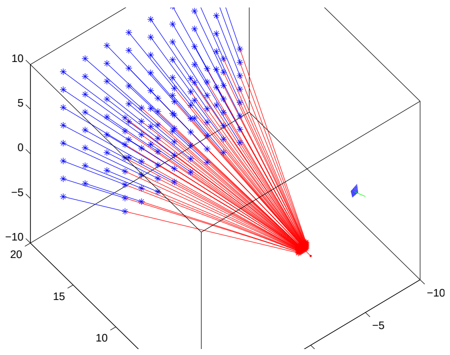
(a) DOE Projector



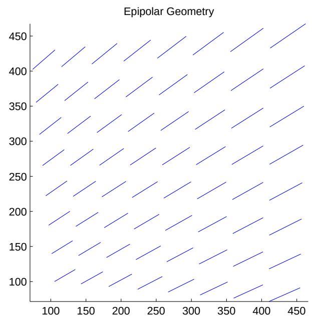
(b) Epipolar Lines

Figure 3.8: Epipolar geometry from converging y-translated camera.

is displaced by a two-dimensional translation from the projector center, and then allowed to converge, the epipolar lines will skew, as shown in Figure 3.9. Figure 3.9



(a) DOE Projector



(b) Epipolar Lines

Figure 3.9: Epipolar geometry from converging xy-translated camera.

illustrates a method by which the epipolar lines may be increased without risking epipolar intersections– by separating the projector and camera by the desired baseline, and translating one of them vertically, the epipolar lines travel in their own "lanes".

Given the geometric foundation for SFSL, we may now explore specific algorithms that are required to archive shape and texture estimation. The first such step involves locating the laser fiducials in the captured image.

3.2 Fiducial Localization

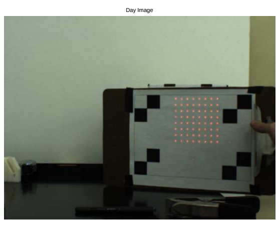
Fiducial localization is the process by which the image coordinates of each laser fiducial are estimated from an image. This process can be thought of as two separate tasks, the first is to extract the pixels corresponding to laser spots from the background, and the second is to find the precise centroid of each laser fiducial. As with the overall system design, we desire this process to be robust against noise, and be computationally efficient. There are several approaches one may take while performing image segmentation: binary thresholding [26], clustering methods, graph-cut methods [27] and many others. Sezgin et al. [28] provide a very thorough overview of such thresholding techniques. The following subsection will detail the approach taken for this thesis. After performing image segmentation, the task of centroid extraction must be carried out. This task is more sensitive to noise than the previous task, as it relates to subpixel level of accuracies.

3.2.1 Localization Methodology: Image Segmentation

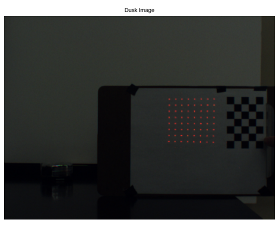
The hardware of this system provides some insight as to which method will work best. The lasers selected for this system operate at a nominal 650nm wavelength (visible red). The laser shape also plays a key role in the ability to easily segment the fiducials from the background. Through experimentation, round lasers proved the most reliable. Oval lasers tend to suffer from distortion when applied to surfaces at severe incident angles. The color camera selected relies on a Bayer image filter to separate incoming light into three bands of wavelengths. In the ideal setting, this Bayer pattern would block almost all of the red laser light in the green and blue channels; however, real Bayer patterns suffer from leakage. It is anticipated that the laser will still be very bright in the other two color channels, but brightest in the red color channel. The image segmentation scheme can take advantage of this by examining where the brightest green/blue pixels are versus the brightest red pixels. By selecting a threshold just past the green/blue responses, the remaining pixels should belong to the laser fiducials.

Data was captured under three illumination settings, bright ambient light (day), dim ambient light (dusk), and no ambient light (night) as shown in Figure 3.10. Since the laser fiducials are very bright, but also very small (in terms of the number of pixels), we expect long histogram tails, primarily along the red channel. Figure 3.12 shows the histogram results of the three images from Figure 3.10. These histograms have values that are the logarithm of the 256-bin histogram levels. The logarithm allows us to better visualize the differences at the brighter pixel end of the spectrum. As expected, the red channel in each illumination case has the longest histogram tail, though with few pixels in each bin (i.e. the histogram is roughly flat where the fiducials are located). Also, there is a strong green response to the red fiducial, as indicated by the long green channel response. Unexpectedly, the blue channel seems largely unaffected by the fiducials. This effect is exploited later during the texture extraction stage in Section 4.3.

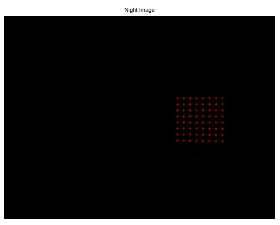
After closely examining these log-histograms, it is apparent that a simple



(a) Day Image

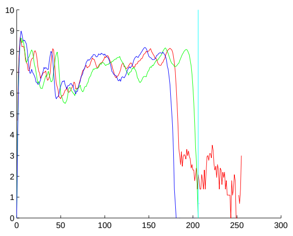


(b) Dusk Image

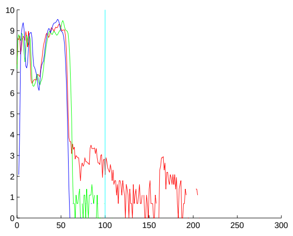


(c) Night Image

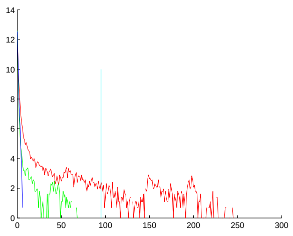
Figure 3.10: Illumination sample images for fiducial localization.



(a) Day Image



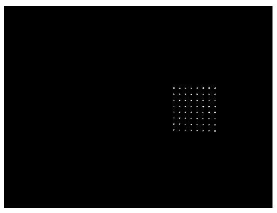
(b) Dusk Image



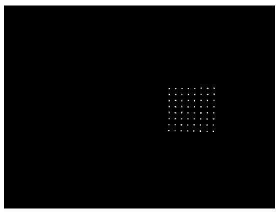
(c) Night Image

Figure 3.11: Log-Histograms for three illumination cases.

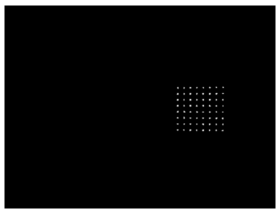
thresholding strategy will work to segment the fiducials from the background. When the expected number of fiducials is known, an iterative approach can be taken. This approach starts with a high threshold on the red channel, decreasing the threshold until the expected number of fiducials is found. In the lab setting, this has proven very reliable.



(a) Day Image



(b) Dusk Image



(c) Night Image

Figure 3.12: Binarized results for three illumination cases using thresholding.

3.2.2 Localization Methodology: Centroid Extraction

The next step in fiducial localization is to estimate the centroid location of each fiducial with high accuracy (see Section 3.4.5 for an tolerance/ error analysis).

Connected Component Analysis: Given a segmentation map, as shown in Figure 3.12, it is important to know how many discrete components there are in the image, and the characteristics of them. Connected Component Analysis (CCA)(colloquially known as Blob Coloring) is a pixel-level process by which neighboring pixels are assigned a unique label. Traditionally, CCA is implemented via a two-pass scan of a binary image produces; though, some modern CCA have been developed [29]. The segmented binary fiducial images are run through CCA which allows us to determine how many fiducials are present and statistics about each fiducial such as width/height/mass. Components outside the standard acceptable range (determined experimentally) are discarded, such as components with too many pixels (due to shadowing, etc.), or components whose nearest-neighbors are too far away (this occurs for spurious noise).

Centroiding: This is the process by which the center of mass of a connected component (a mass of pixels) is computed. Trivially, one computes the average pixel location for a group of pixels which is deemed the centroid. The drawback to the centroiding approach lies in the physics of the laser projector. Lasers exhibit so-called "speckle" noise, due to the random interference of many light wavefronts.

This random speckle pattern creates segmented objects that are inherently random in nature. That is, given identical laser/camera setups with a stationary target, the imaged laser fiducial will appear random about its true centroid[3]. While the spot intensity is approximately Gaussian, there is no guarantee that the binarized image will be centered at the true centroid because of this random speckle.

Matched Filter Approach: A more traditional signal processing approach is peak detection, which is readily accomplished via a matched filter. For this problem, our matched filter will be a 2D filter that corresponds to the nominal laser spot. The matched filter based fiducial localization may be computed very efficiently by the Fourier method, as discussed in [30]. We denote the template fiducial $f(x,y)$ whose Fourier transform, $\mathcal{F}\{.\}$, is given by:

$$F(\eta_x,\eta_y) = |F(\eta_x,\eta_y)| \exp(j\Phi(\eta_x,\eta_y)). \tag{3.9}$$

A 3D object with reflected laser fiducials is imaged and denoted as $g(x,y)$, with Fourier transform $G(\eta_x,\eta_y)$. The matched filter is then given by

$$H_{MF}(\eta_x,\eta_y) = F^*(\eta_x,\eta_y) = |F(\eta_x,\eta_y)| \exp(-j\Phi(\eta_x,\eta_y)). \tag{3.10}$$

To apply the filter, we must compute the correlation of the input image with the matched filter transfer function. Again, after using the Fourier correlation property, we derive:

$$C(x,y) = \mathcal{F}^{-1}\{G(\eta_x,\eta_y) H_{MF}(\eta_x,\eta_y)\}. \tag{3.11}$$

[3]Though, this deviation is well below the accuracy capability of the calibration system, Section 3.5

Estimates of the centroid positions are then given by the peaks found in the correlation.

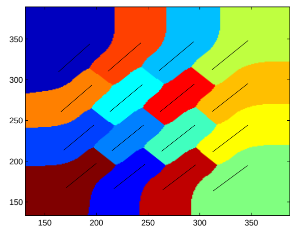
3.3 Fiducial Labeling (Correspondence Problem)

After every fiducial has been located, the task now is to associate each fiducial centroid to a laser beam. This is the so-called correspondence problem that occurs often in multiview geometry. Unlike stereo vision, SFSL enjoys a simplified correspondence problem. In fact, this is one of the main reasons SFSL is so widely used. In SFSL, the engineer explicitly controls the difficulty level of the correspondence problem in two ways. First, the epipolar geometry can be carefully measured to ensure that epipolar lines do not overlap. Second, structured light patterns can be designed to uniquely encode every pixel. Since the selected structured light pattern is a direct coding, we must rely on the validity of the COS in order to compute accurate fiducial labels. In this section, two methods are presented. The first method is useful for computing fiducial labels during calibration, when no a priori information is known about the configuration. Given a vector of fiducial centroids $\mathbf{C} = [\mathbf{f}_1 \; \mathbf{f}_2 \ldots \mathbf{f}_{nm}]^T$, we would like to sort $\mathbf{C}$ to match each $\mathbf{f}_i$ with its corresponding laser. First, we denote the grid of lasers by p_{ij} , $i = 1, \ldots, n$ and $j = 1, \ldots, m$. Arranging $\mathbf{C}$ into an n by m matrix, we get

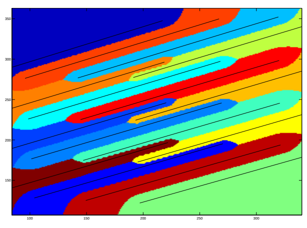
$$\mathbf{C} = \begin{bmatrix} \mathbf{f}_1 & \mathbf{f}_2 & \cdots & \mathbf{f}_n \\ \mathbf{f}_{n+1} & \mathbf{f}_{n+2} & \cdots & \mathbf{f}_{2n} \\ \vdots & & & \vdots \\ \mathbf{f}_{n+m-1} & \cdots & \cdots & \mathbf{f}_{nm} \end{bmatrix}. \tag{3.12}$$

Next, we sort each column vector according to the $x-$component of each f_i. The corresponding rows must then be sorted by the $y-$ component of each f_i, thus leaving the top-left most fiducial in the first matrix position, and the bottom-right most fiducial in the last matrix position. This method works only when the fiducial orientation is within $pi/2$ radian rotation of the nominal position (as the upper left most fiducial will not be the same).

Once initial labeling is performed on the calibration data, we may utilize a direct correspondence procedure for future data sessions. Other approaches to this problem such as [17] involve computing a homography H that maps the epipolar lines to a regular grid (where the correct label can simply be read from the grid coordinates of the transformed fiducial). This method does not allow epipolar lines such as in Figure 3.9. An alternative method is to instead precompute a map for each pixel that assigns a fiducial label. Figure 3.13 depicts such a mapping for two epipolar cases. To generate this mapping, the resulting epipolar image from calibration is fed into a nearest-neighbors algorithm. This algorithm scans through the entire image pixelwise, assigning a label based on the Euclidean distance from that pixel to pixels belonging to the epipolar lines. This map can be easily precomputed and stored. During runtime, a new fiducial centroid is labeled by looking up the corresponding cell in the map.



(a) Example 1



(b) Example 2

Figure 3.13: Voronoi cells from nearest neighbor on epipolar line segments.

3.4 Triangulation for Range Estimation

The process of triangulation provides range estimates for 3D shape estimation. The following subsections present the mathematical formulation of this procedure, along with simulation and experimental results.

3.4.1 Planar Triangulation

First, consider a 2D scene where the camera is 1D and the laser projector is 2D as shown in Figure 3.14. We know from the study of epipolar geometry that rays in space will project to epipolar lines on the image plane. A simple way to view this is, for a given laser beam, a change in range will correspond to a fiducial movement along the epipolar line. This means that if the epipolar lines are well estimated, we can use 1D equations to estimate depth for a given laser. This figure represents a transformation of the 3D geometric equations into simple 1D equations. The baseline between the laser and the camera is represented by b. It is assumed that the camera center and projector center are on this line; thus the camera plane is f away from this baseline for this laser. The laser is at some θ from the baseline. It is

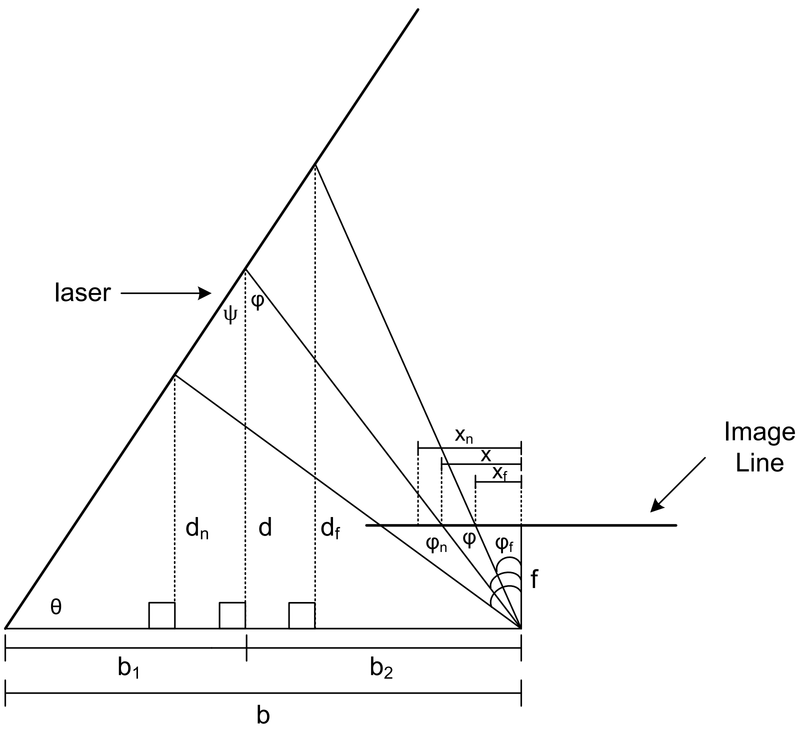


Figure 3.14: 1D Triangulation from camera to laser.

shown here that the laser is coplanar with the baseline and camera, though this will be expanded upon later. There are three range projections shown, d_n representing the nearest object that can be imaged, d_f representing the furthest object that can be imaged, and a generic d for any object in-between. These ranges project to the image line at locations x_n, x_f, and x [4], which invokes the corresponding angles φ_n,φ_f and φ (the angle between the chief ray and the principal point). This φ yields a new angle $\psi = \frac{\pi}{2} - \theta$. By the law of sines, we can then write

$$b_1 = \frac{d \sin(\psi)}{\sin(\theta)} = \frac{d \sin\left(\frac{\pi}{2} - \theta\right)}{\sin(\theta)} = \frac{d}{\tan\theta}; \tag{3.13}$$

[4]Note that the value of x here decreases as the object distance increases. Therefore, care must be taken when implementing using a different reference for x to ensure the correct polarity.

and, since $b = b_1 + b_2$,

$$b_2 = b - \frac{d}{\tan\theta}. \tag{3.14}$$

Forming the trigonometric relationship between the right triangle defined by the b_2 portion of the baseline and the projected d, we have

$$\tan(\varphi) = \frac{b_2}{d} \tag{3.15}$$

thus,

$$\tan(\varphi) = \frac{b - \frac{d}{\tan\theta}}{d}. \tag{3.16}$$

Finally, we can write the final equation for x in terms of d, f and θ,

$$x = f\tan(\varphi) = \frac{f\left(b - \frac{d}{\tan\theta}\right)}{d}. \tag{3.17}$$

Equivalently, the equation that yields the range for a given pixel is:

$$d = \frac{fb}{x + \frac{f}{\tan\theta}}. \tag{3.18}$$

In general, f, b and $\tan\theta$ are not known values, rather they must be estimated in some manner. To simplify the estimation problem, we can write Equation 3.18 in terms of two constants,

$$d = \frac{c_1}{x + c_2}, \tag{3.19}$$

which is an equation with only two unknowns. This thesis estimates these parameters in three ways, all of which are fast visual methods (i.e. they rely on camera measurements during calibration), the details of which are in Section 3.5.2.

3.4.2 Non-planar Triangulation

For the non-planar case, consider that the laser is allowed to rotate from along the opposite dimension (Figure 3.15). According to Hartley, "As the position of the 3D point X varies, the epipolar planes 'rotate' about the baseline. This family of planes is known as an epipolar pencil. All epipolar lines intersect at the epipole." [25]. Thus, this rotation generates a pencil along the baseline such that the projected range on the image plane is always mapped to the same x value (though the y values change correspondingly). We can thus estimate the range using only the $x-$coordinate in the ideal, noise free case. In actual experiments, there will

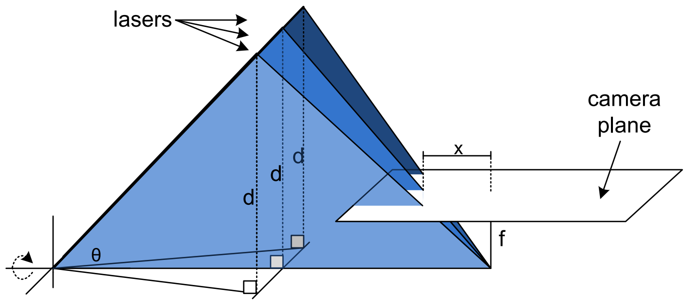


Figure 3.15: Non-planar triangulation case.

be estimation noise, thus relying solely on the $x-$coordinate will result in non-optimal solutions. Using the mean-square-error (MSE) as the optimality criterion, the optimization is given by

$$\hat{\mathbf{x}} = \arg\min E\left[\left(\hat{d} - d\right)^2\right] = \arg\min_{\mathbf{c}} E\left[\left(\frac{c_1}{\hat{x} + c_2} - \frac{c_1}{x + c_2}\right)^2\right] \; s.t. \quad \hat{\mathbf{x}} \in \mathbf{l}', \quad (3.20)$$

where $\mathbf{l}'$ is the estimated epipolar line. Since c_1 and c_2 are fixed, this optimal $\hat{\mathbf{x}}$ is the one such that the error is orthogonal to the MSE estimate (which must lie on the epipolar line because of the constraint). Thus, the error is perpendicular to the epipolar line. In geometric terms, we can simply compute the point along the epipolar line that is the intersection of the measured point and its perpendicular projection to the epipolar line. The optimal $\hat{x}$ is the projection of the measured fiducial location to the estimated epipolar line as shown in Figure 3.16.

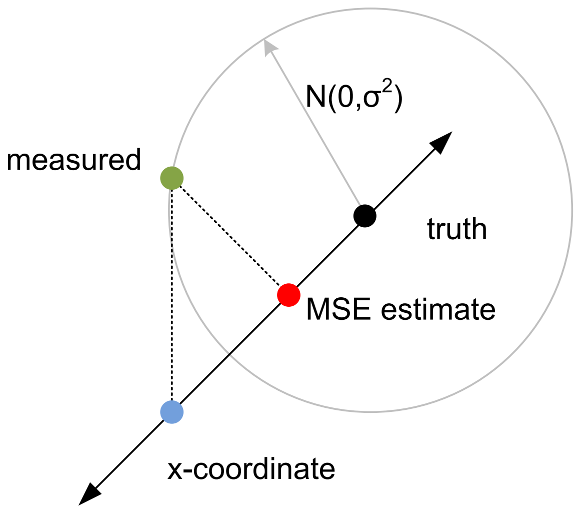


Figure 3.16: Measured fiducial projected onto estimated epipolar line.

3.4.3 Shape Estimation Simulations

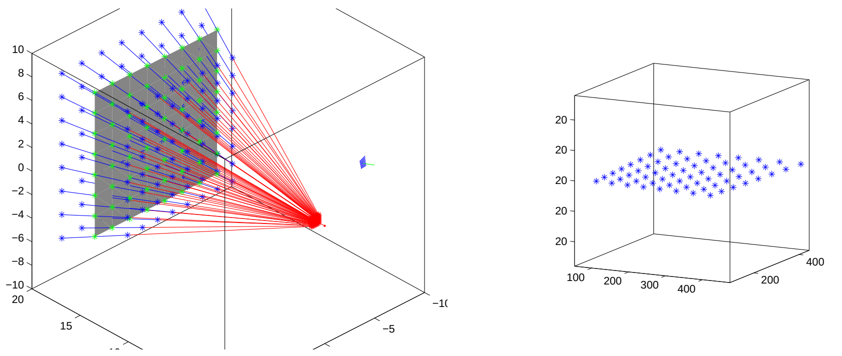
To verify the triangulation equations, and the calibration procedure, simulations were performed with synthetic surfaces. Each simulation involved constructing a virtual surface in 3D space, and computing the intersection points of the surface with the laser fiducials to give ground truth range values. These points were then imaged by the virtual camera and processed. Since the fiducial localization algorithm is trivial in the simulation case, various levels of noise must be added to test the reconstruction capabilities. The residual error from between the ground truth range and estimated range, i.e.

$$\epsilon_{k,j} = d\left(k,j\right) - \hat{d}\left(k,j\right), \quad x \in \{1,\ldots,N_x\}, \quad y \in \{1,\ldots,N_Y\} \tag{3.21}$$

with MSE percentage given by,

$$\epsilon_{MSE} = \frac{1}{N_x N_y}\frac{100}{d_f - d_n}\sum_{k}^{N_x}\sum_{j}^{N_y}\left|\epsilon_{k,j}\left(k,j\right)\right|^2. \tag{3.22}$$

The first simulation was a a planar surface perpendicular to the laser projector center line as shown in Figure 3.17(a) with the noise-free triangulated mesh shown in Figure 3.17(b). Table A.1 shows the residual error for each fiducial after triangulation in the noise-free setting, while Table A.2[5] shows how additive Gaussian noise (zero mean, one pixel variance) affects the results.

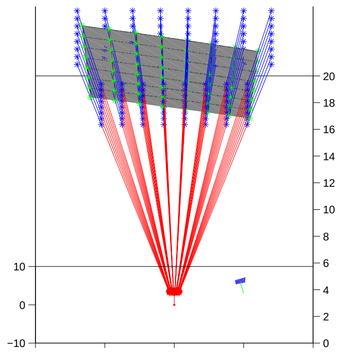


(a) DOE Projector (b) Resulting Model

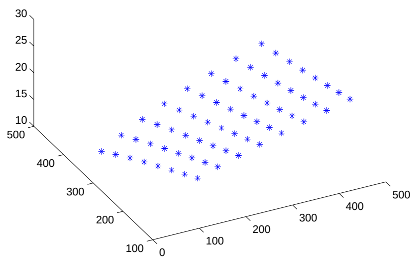
Figure 3.17: Triangulation simulation for orthogonal planar surface.

The second simulation rotated the surface in both the $x-$ and $y-$dimensions, as shown in Figure 3.18(a). The estimated range points are shown in Figure 3.18(b). Table A.3 shows the error performance.

[5]Any 3D model shown in this Thesis is oriented so as to best show planarity, curvature, or other important feature.



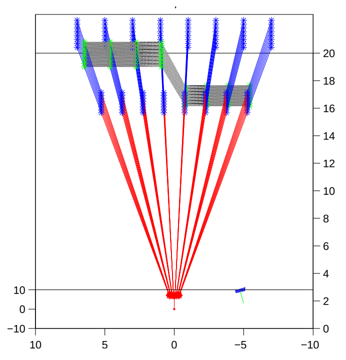
(a) DOE Projector



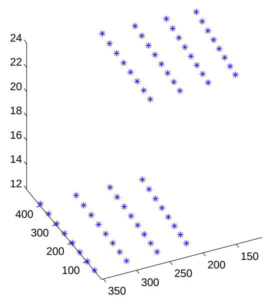
(b) Resulting Model

Figure 3.18: Triangulation Simulation for angled surface.

Finally, a step-discontinuity was used as the surface as shown in Figure 3.19(a), with the triangulated mesh surface shown in Figure 3.18(b). Table A.4 shows the error performance.



(a) DOE Projector



(b) Resulting Model

Figure 3.19: Triangulation Simulation for step surface.

These simulations confirm that the triangulation and calibration equations are accurate (as indicated by the $10e{-}12\%$ error in the noise free case). They also show how the sensitivity to noise perturbations occurs as the fiducial/ surface intersection point moves towards d_f. Since the error at some points was over 2% for a noise variance of only one pixel, understanding how noise and misestimation is crutial to the design of the system. Secion 3.4.5 analyzes this problem in detail.

3.4.4 3D Shape Estimation Experiments

We have seen that the 3D shape estimation algorithm works very well on simulated data, now we turn our attention to real surfaces imaged with an actual camera and DOE laser projector, as seen in Figure 3.20.

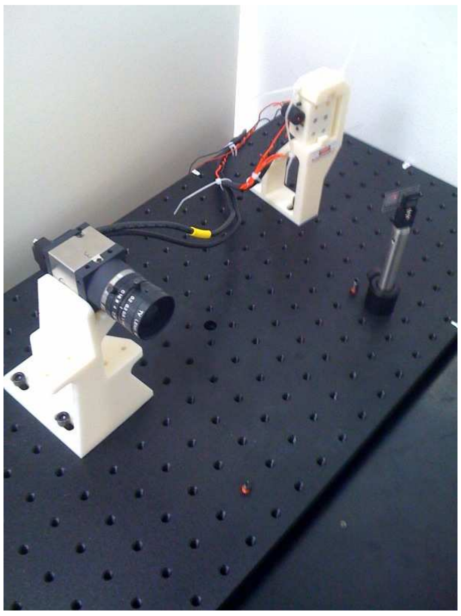


Figure 3.20: SFSL hardware setup.

Planar Surface: The first experiment performed was a flat surface situated orthogonally to the camera's principal axis. Figure 3.21 shows the estimated ranges at each fiducial.

Split Planar Surface: The next test featured two parallel planar surfaces that split the projected pattern into two levels, similar to the simulation in Figure 3.19. Figure 3.22 shows the estimated ranges at each fiducial.

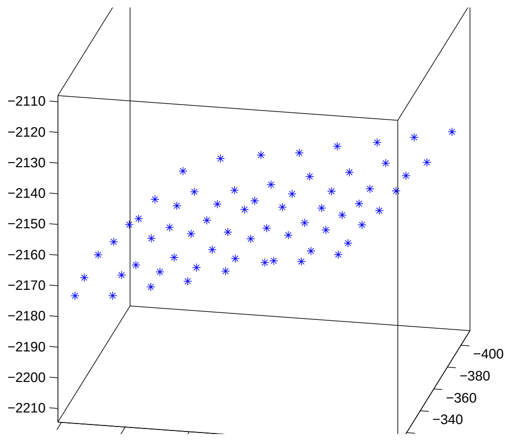


Figure 3.21: Planar surface point estimates.

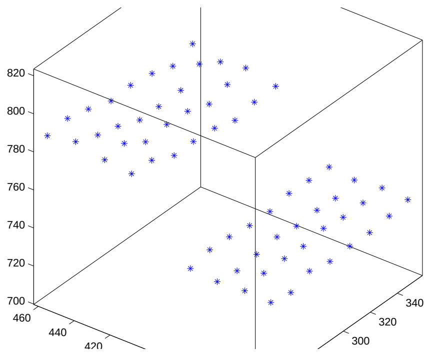


Figure 3.22: Split planar surface point estimates.

Cylindrical Surface: The following experiment involved a cylindrical surface. Figure 3.23 shows the resulting point cloud.

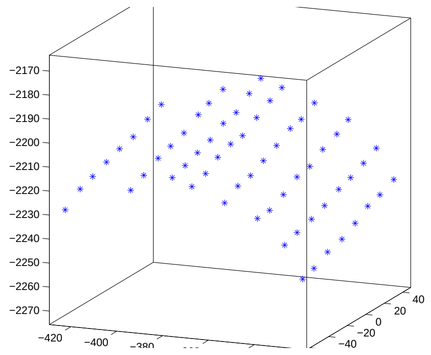


Figure 3.23: Cylindrical surface point estimates.

Due to the lack of available calibrated surfaces (where the 3D model is known a priori), measuring the performance of the triangulations on arbitrary surfaces is a difficult task. To measure the performance of the 3D reconstruction of simple geometric objects, one can try to fit a geometric model to the reconstructed data, and compare that model to what is known to be true. For instance, a plane located at 955mm from the camera is imaged, and the resulting 3D surface is estimated. A least-squares plane can be fit to the data; the fit plane is then compared to the ground truth (a plane located 955mm away from the camera). The residual error between the 3D points and the fit surface also gives an estimate of the triangulation error. The error performance of triangulation in these experiments is given in Table 3.1. Here sparse refers to fiducial-only measurements, while dense refers to the smoothed triangulated surfaces via methods in Section 4.2. The error percentage is the amount of triangulation error over a given capture volume.

Table 3.1: Triangulation Results for Various Surfaces

	Norm of Residual Error	**Error Percentage**
Sparse Orthog. Plane	6.6474	3.2547
Dense Orthog. Plane	9.6727	4.7358
Sparse Slanted Plane	5.3943	2.6411
Dense Slanted Plane	6.5360	3.2001
Sparse Cyl.	7.9151	3.8753
Dense Cyl.	0.7291	0.3570

3.4.5 Epipolar Error/ Tolerance Analysis

There are many variables in the 3D shape estimation system, all of which may contribute some sort of error in range estimation. As this system relies on epipolar line estimation, any perturbation in these estimates will directly influence the final depth estimate. The system also relies on estimation of the each fiducial centroid. The algorithm to detect the fiducial centroids is very accurate, but may incorrectly estimate the centroid because of the laser speckle noise as described in Section 3.2.2; also, the fiducials themselves may deviate from the epipolar line because of erroneous estimation of the epipolar lines, imperfect camera calibration (localized), and so on. A detailed analysis of the error sources that arise in SFSL systems is provided by Yang et al. in [31].

To analyze these sources of error, Monte Carlo simulations were performed. For each simulation, a specific error source was allowed to deviate from the noise free position via additive Gaussian noise. Each simulation began by generating 500 random surfaces, drawn from a uniformly random range for each fiducial within the d_n and d_f limits. It is important for the random surface to be uniformly generated since error is more significant towards the d_f plane, as the range resolution decreases with distance. For each random surface, the noise source under study was amplified by incrementing the noise variance. For a comparison, each simulation was run using the simple 1-dimensional triangulation equations (x-projection) as well as the MSE optimal projected-triangulations.

The first simulation studied fiducial localization error. Figure 3.24 shows the results of the fiducial localization noise test. Here, the epipolar lines are assumed to be error free, thus the only error source is the incorrect fiducial centroid estimate. The performance metric used was the MSE between the measured range and the ground truth, as in Equation 3.22.

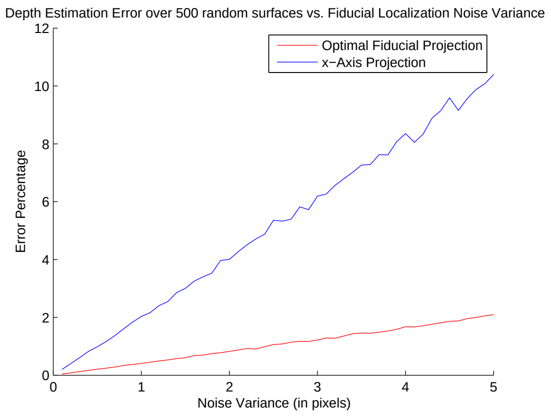
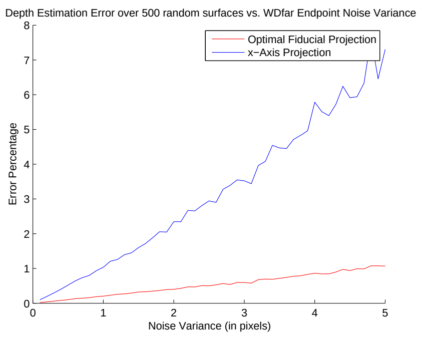
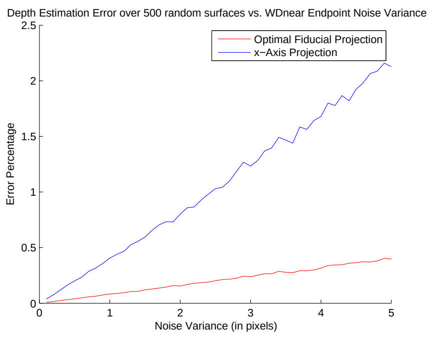


Figure 3.24: Simulation of range estimation error from fiducial localization noise.

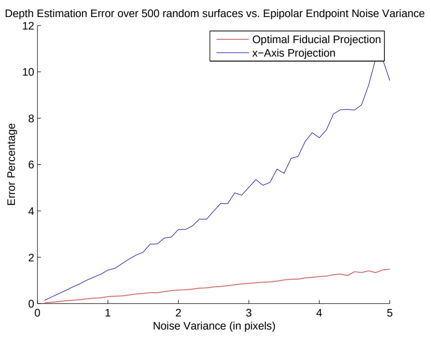
The second set of simulations performed analyzed how estimation error of the epipolar lines leads to range estimation noise. Three scenarios were examined as shown in Figure 3.4.5. The first scenario involved adding Gaussian noise (zero mean, with a variable variance) to the rightmost point of each epipolar line (the d_f measurement). The second involved adding noise to the leftmost point (the d_n measurement); and the last simulation added independent noise to both endpoints. Again, 500 random surfaces were generated, each surface being tested under increas-



(a) Far WD Epipolar Endpoint Noise



(b) Near WD Epipolar Endpoint Noise



(c) Both Epipolar Endpoint with Noise

Figure 3.25: Epipolar estimation error analysis.

ing noise variance. As evident by Figure 3.25(b), mis-estimation of the leftmost epipolar endpoint contributes to the range estimation error negligibly (since the depth resolution is quite high close to the camera), whereas error in the rightmost endpoint estimation is significant when the uncertainty is four pixels or greater (since the depth resolution is significantly lower).

Lastly, both the epipolar endpoints, and the fiducial centroid were subjected to additive Gaussian noise as shown in Figure 3.26. As expected the error increased significantly from previous simulations. To achieve a range estimation error of under 1%, every parameter must have estimation error of approximatly 1 pixel.

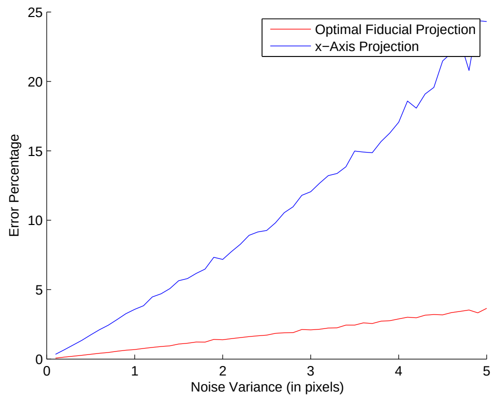


Figure 3.26: Range error versus epipolar endpoint and fiducial localization noise.

3.5 Calibration Methodology

The proposed system for SFSL relies on pixel level accuracies for estimating range. There are two calibration stages that take place in this system. The first stage is the intrinsic camera calibration, which attempts to remove distortion from the captured images. Depending on the required level of reconstruction accuracy, this stage may be omitted or simplified. The Projector-to-Camera calibration stage however, is essential to the operation of the system; though, every attempt has been made to simplify this process to the minimum requirements.

3.5.1 Camera Calibration

Camera calibration is the process by which a physical camera and lens system is transformed into an ideal pinhole camera (up to a certain degree of accuracy) by a distortion model. Much work has been done on this topic, including the work by Zhang, et al. [32], Heikkila, et al. [33] and Clarke, et al. [34]. The Caltech Matlab camera calibration software [35] was used in this system's implementation. The camera distortion model used corrects the distortion-free Equation 3.1 by transforming the distorted image coordinates (x, y) into undistorted image coordinates (x_p, y_p) via:

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} f_1 & s & x_0 \\ 0 & f_2 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} \tag{3.23}$$

which is more compactly expressed as,

$$\mathbf{m}_p = K\tilde{\mathbf{m}}. \tag{3.24}$$

We define $(\tilde{x}, \tilde{y})$ by,

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} = \begin{bmatrix} (1 + k_1 r^2 + k_2 r^4 + k_5 r^6)\, x + (2k_3 xy + k_4 (r^2 + 2x^2)) \\ (1 + k_1 r^2 + k_2 r^4 + k_5 r^6)\, y + (k_3 (r^2 + 2y^2) + 2k_4 xy) \end{bmatrix} \tag{3.25}$$

where $r^2 = x^2 + y^2$ and $s = \alpha_c f_1$ (the skew between pixel-axis). These parameters reflect the intrinsic camera parameters as well as radial and tangential lens distortion. Instead of a fixed f, we now have f_1 and f_2 that allow for non-square pixels, as is the case with many CCD and CMOS sensors. The principal point offset (x_0, y_0) allows for a camera center that is not located at the principal point, p (See Figure 3.1). To compute these constants, the Caltech camera calibration toolbox for Matlab was used [35]. This software works by imaging a known calibration plane at several positions. This plane yields image points corresponding to known 3D world points via Harris corner detection on the checkerboard pattern. From this correspondence, the homography between the image plane and each calibration plane can be estimated via a variety of methods such as random-sample consensus (RANSAC), maximum-likelihood estimation (MLE), and others. The intrinsic parameters can then be estimated directly from the estimated homographies. Extrinsic parameters such as rotation and translation are given by the factorization of the estimated homographies. This technique is used to aid the projector-camera calibration in Section 3.5.2.

3.5.2 Projector-to-Camera Calibration

The goal of the projector-to-camera calibration is to estimate the epipolar geometry between the two devices. As shown before, the epipolar lines induced by the projector's lasers are crucial to the triangulation of range data.

3.5.2.1 Two-Plane Epipolar Line and Depth Curve Estimation

The minimum calibration required is the two-plane epipolar line and depth curve estimation approach. The most direct method for projector-to-camera calibration is to explicitly measure the image position of each laser fiducial (along some unknown epipolar line) at known ranges. The operator must capture an image of the fiducial grid on a planar surface orthogonal to the principal camera axis some known distances. Recalling the triangulation equations from before, we have an equation with two unknowns for a given image coordinate x and range d. Thus, if we measure the fiducials' image coordinates at two known distances, d_n and d_f, we can determine the value of the constants as follows,

$$C_1 = \frac{d_f\left(x_f - x_n\right)d_n}{d_n - d_f} \tag{3.26}$$

$$C_2 = \frac{d_f\left(x_f - x_n\right)}{d_n - d_f} - x_n. \tag{3.27}$$

So, by simply viewing a plane parallel with the image plane at two distances, we can quickly estimate the required parameters for triangulation. Using the measured image coordinates from each fiducial at d_f and d_n also allows us to construct the epipolar line corresponding to each laser by trivially "connecting the dots". Recalling Section 3.4.2, the epipolar lines are used to minimize the MSE of new fiducials that are imaged.

Unfortunately, the two-plane method is susceptible to many forms of error. As with any method, there exists localization error when measuring the laser fiducials' image coordinates. In addition, it is highly unlikely that the plane is truly parallel with the image plane, thus introducing errors in d_f and d_n.

3.5.2.2 Least-Squares Epipolar Line and Depth Curve Estimation

To improve upon the two-plane method, we may over-determine the system of equations for C_1 and C_2 by taking into consideration many parallel planes such as in [24]. The procedure is very similar to the two-plane method, except instead of measuring fiducial image locations from planes at d_f and d_n, many more planes spanning the range between are imaged. A linear least-squares epipolar line estimate is then performed by

$$\arg\min_{\mathbf{c}} \sum_{i=1}^{N} \left(y_i - (c_1 x_i + c_2)\right)^2 = \arg\min_{\mathbf{c}} \left\|\mathbf{y} - [\mathbf{x}|\mathbf{1}]\,\mathbf{c}\right\|^2 \tag{3.28}$$

which is readily solved by the pseudo-inverse

$$\hat{\mathbf{c}} = \left(X^T X\right)^{-1} X^T y = X^+ y. \tag{3.29}$$

While the two-plane method is the minimum, capturing the fiducial locations at more ranges yields a better epipolar line estimate as shown by Figure 3.27. For this experiment, the number of calibration planes ranged from two to fifteen. For each set of calibration planes, 1500 random surfaces were generated as ground truths. Noisy fiducials were used for triangulation on the epipolar lines generated from just the two end planes, and on the LS generated epipolar lines. As expected, the LS generated epipolar lines are more accurate than the two-plane approach. Since each experiment (i.e. the number of calibration planes)

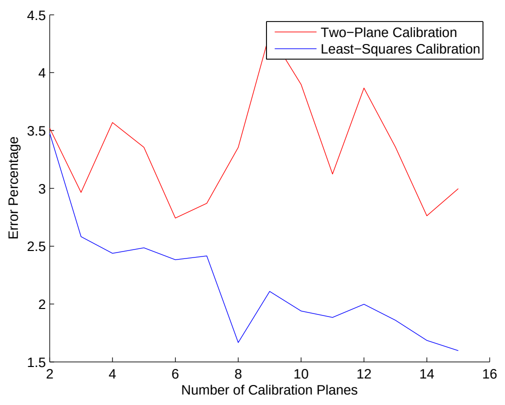


Figure 3.27: Comparison of calibration method error percentage performance.

For depth-curve estimation, the linear LS regression approach will not work due to the nonlinear nature of the depth-curve equation 3.19. Thus, we must examine non-linear LS regression. Since the equation for depth is only locally convex (in the region we are interested), proper initial starting points must be selected in order for the algorithm to converge to reasonable estimates. The function we wish to optimize is,

$$\arg\min_{\mathbf{C}} \sum_{i=1}^{N} \left(d_i - \frac{C_1}{x_i + C_2} \right)^2 \tag{3.30}$$

which may be solved by gradient-methods such as *nlinfit* in MATLAB. In order convergence, we must select proper initial values for C_1 and C_2. The method implemented simply uses the two-plane estimates for C_1 and C_2, which are then further refined through the nonlinear LS regression. To visualize the improvement of this method over the epipolar only LS esitmate, Figure 3.28 shows the results of a simulation similar to that of Figure 3.27

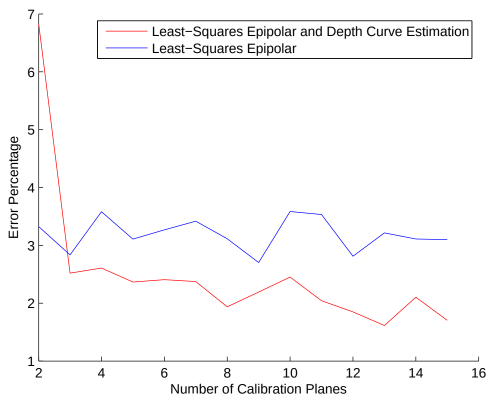


Figure 3.28: Comparison of least-squares depth-curve estimation vs. previous approach.

3.5.2.3 Visual-Feedback Depth Curve Correction

While the least-squares epipolar and depth-curve estimation works quite well when the only source of noise is measurement noise, the method requires that the calibration planes be approximately orthogonal to the principal axis. This is inconvenient since it requires calibrated hardware to hold the plane at a right angle to the camera at a known distance. A more sophisticated approach is to use a visual-feedback mechanism. The procedure is carried out the same as before, except the target calibration plane carries a calibration fiducial pattern as shown in Figure 3.29.

This pattern is imaged by the camera, which enables the estimation of the extrinsic relationship between the camera and the target board. This extrinsic relationship is then used to more accurately compute the range at each imaged fiducial location via back-projection. Back-projection is a method to map image points ($\mathbb{R}^2$) to scene points ($\mathbb{R}^3$) (given that there are some additional constraints).

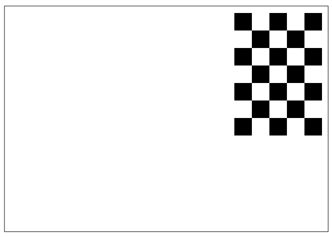


Figure 3.29: Calibration target board.

From before, we define the projection from a world-coordinate point to an image coordinate via

$$\tilde{\mathbf{x}} = [P|\mathbf{p}_4]\,\tilde{\mathbf{X}} \tag{3.31}$$

as seen in Figure 3.30.

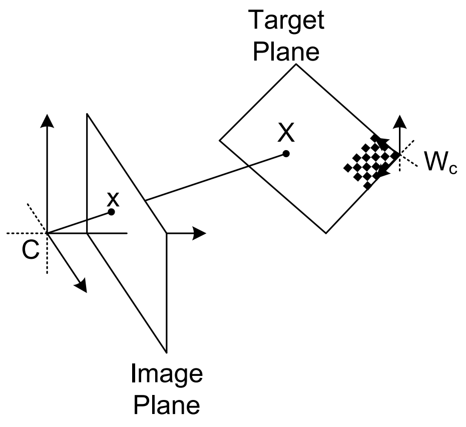


Figure 3.30: Calibration board's extrinsic relationship with camera.

We denote a calibration plane $\pi = [\pi_1\ \pi_2\ \pi_3\ \pi_4]^T$ by its normal vector $[0\ 0\ 1]^T$, or equivalently, by this the vector's point at infinity, $\tilde{\mathbf{D}} = [0\ 0\ 1\ 0]^T$. The ray joining the camera center $\mathbf{C}$ with the imaged fiducial point $\mathbf{x}$ is defined by

$$L(\lambda) = \tilde{\mathbf{C}} + \lambda \tilde{\mathbf{D}}_x, \tag{3.32}$$

where $\tilde{\mathbf{C}}$ is the homogeneous camera center, and $\tilde{\mathbf{D}}_x$ is the fiducial at infinity. At the heart of the problem is solving for where this ray intersects the calibration plane. This intersection occurs at $\mathbf{X}$, the fiducial point on the 3D surface. We can then write,

$$\tilde{\mathbf{X}} = \tilde{\mathbf{C}} + \lambda \tilde{\mathbf{D}}_x, \tag{3.33}$$

Since any point on the calibration plane must be orthogonal to the normal vector, $\tilde{\mathbf{X}}^T \tilde{\mathbf{D}} = \tilde{\mathbf{D}}^T \tilde{\mathbf{X}} = 0$, thus

$$\tilde{\mathbf{D}}^T \tilde{\mathbf{X}} = \tilde{\mathbf{D}}^T \left(\tilde{\mathbf{C}} + \lambda \, \tilde{\mathbf{D}}_x \right) = 0. \tag{3.34}$$

Solving for λ,

$$\lambda = \frac{-\tilde{\mathbf{D}}^T \tilde{\mathbf{C}}}{\tilde{\mathbf{D}}^T \tilde{\mathbf{D}}_x}. \tag{3.35}$$

The camera center $\mathbf{C}$ is defined as the 1-dimensional null-space of $[P|\mathbf{p}_4]$, which is defined (homogeneously) as

$$\tilde{\mathbf{C}} = \begin{bmatrix} -P^{-1}\mathbf{p}_4 \\ 1 \end{bmatrix}. \tag{3.36}$$

The point $\tilde{\mathbf{D}}_x$ represents the back-projected point on the plane at infinity, i.e.

$$\tilde{\mathbf{D}}_x = \begin{bmatrix} -P^{-1}\mathbf{x} \\ 0 \end{bmatrix}. \tag{3.37}$$

Thus, with λ determined, the back-projected estimate for $\tilde{\mathbf{X}}$ is obtained. This point represents the 3D location of the laser fiducial in the world-coordinate frame, thus inversing the rotation and translation transformation to the camera coordinate frame is necessary to obtain the positive range from the camera to the fiducial. Using this method, we arrive at more-accurate estimates for each d_i, x_i pair. These accurate pairs are then fed into the LS depth-curve fitting algorithm as described in the previous subsection.

A side benefit to performing the visual-feedback calibration is that an accurate model of the laser projector may be computed. Since we obtained $\tilde{\mathbf{X}}$ at several points along the laser, we can perform a 3D linear regression fit to these points, thus estimating the path of each laser from the projector. The regression is much like before, except using a 3D line equation, $Z = aX + bY + C$, and performing a regression for each dimension. The resulting model appears in Figure 3.31; blue markers indicate the back-projected 3D coordinate estimates a fiducial, and the red lines represent the linear regression fit to the points.

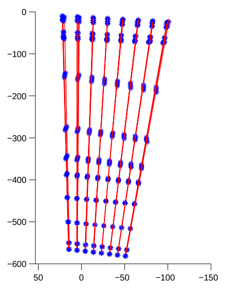
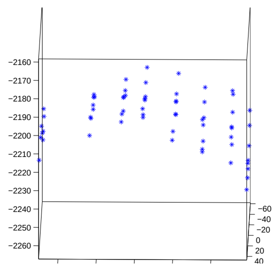
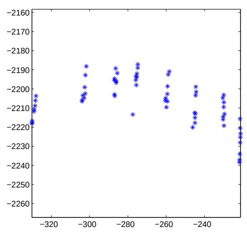


Figure 3.31: Reconstructed model of laser paths via visual-feedback calibration.

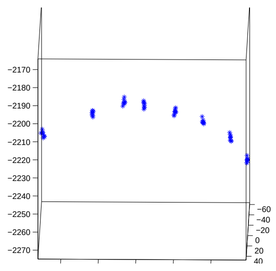
To compare the effects of the various calibration methods, a cylindrical object was imaged, and its depths computed using the two-plane approach, the LS epipolar line and depth-curve approach, and the visual-feedback approach. Figure 3.32. As expected, the visual-feedback approach combined with the LS fitting performs much better than the other methods.



(a) Two-Plane Calibration



(b) LS Calibration



(c) Visual Feedback Calibration

Figure 3.32: Comparison of the presented calibration methods.

Chapter 4

Texture Estimation from 3D Shape

4.1 Texture Estimation Introduction

The goal of texture estimation in this thesis is to find a *isometric* mapping from our sparse 3D range data to a 2D plane which represents the "unrolled" or "shape-distortion"-free image texture[1]. Glasbey [36] separates warping into two categories, parametric (such as bilinear, affine, polynomial, etc.) and non-parametric (such as elastic deformations, thin-plate splines, Bayesian approaches, etc.). Parametric transforms are widely used to correct image warping due to their simplicity, but suffer greatly when there are localized distortions in the data [37]. What we seek is an isometric mapping, where the distance between any two points along the estimated 3D surface (termed *geodesic distances*) and the distance between the corresponding points on the flattened 2D surface are preserved [38].

This chapter first examines the sparse range data interpolation problem, followed by the 3D model flattening (and image warping) problem.

[1]Here the term texture is meant not as a 3D construct but as a 2D image that when applied (mapped) to the estimated 3D shape and imaged, the resulting view is the same as the actual object. This is a common problem in cartography, e.g. flattening a view of the Earth onto a 2D plane.

4.2 Sparse Range Data Interpolation

The output of the SFSL algorithms is a collection of noisy, irregularly-spaced range estimates that represent samples along the target surface. Depending on the design of the DOE, this sampling may be very dense, or very sparse. As shown before in Section 2.3.1, sparse sampling allows for a larger capture volume, at the expense of spatial resolution. The problem of 3D shape data interpolation can be viewed as a special case of a $k-$dimensional data interpolation, where general algorithms apply, or more problem-specific approaches can be taken. This section examines both, general data interpolation applied to range data; then range-data specific interpolation schemes.

4.2.1 Range Driven Interpolation

Given only the noisy range data, the goal is to interpolate between the sparse samples and reconstruct a surface. Amidror [39] explores four common families of scattered data interpolation that occur in imaging systems, namely: Triangularization, Inverse Distance Weighted Functions, Radial Basis Functions, and Natural Neighbors. Of these, we will explore triangularization, and radial basis functions. Mathematically, we are given a set of points G_i, where $i \in (1, 2, \ldots, M)$ belonging to the 3D surface, and we seek a set of points $P_{i,i+1,k}$ where $k \in (1, 2, \ldots, N)$ such that each $P_{i,i+1,k}$ lies between the bounding G_i and G_{i+1}. We desire that these $P_{i,i+1,k}$ also belong to the set of points along the true surface.

4.2.1.1 Triangularization and Tetrahedralization

Two of the most studied areas of range data interpolation are triangularization (of 2D data) and tetrahedralization (of 3D data). One can use triangularization or tetrahedralization to generate a mesh between points in the dataset, which then provides a guide for data interpolation (i.e. data generates a cell, and new points in that cell are interpolated from the cell-generating data). Triangularization is an inherently local operation, that tries to join a set of 2D data points with triangles [39]. There are obviously many possible ways to accomplish this task, but in general, we seek a "good" surface that consists of approximately equilateral triangles. The very popular Delaunay triangularization generates a triangle net in which every circle that circumscribes a triangle in the net contains *only* the three nodes that define the triangle, as shown in Figure 4.1. By enforcing this property, the resulting tri-

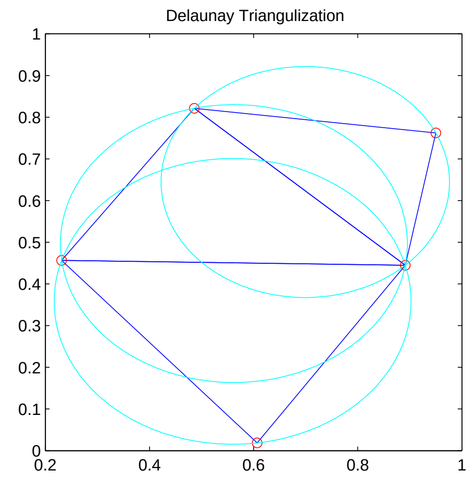


Figure 4.1: Example Delaunay triangularization with circumscribed circles.

angle meshes are composed of triangles with the "largest minimal angles" [39]. To extend the 2D triangle net to a 3D net, we must use tetrahedrons. Choi [40] explores the conversion of Delaunay triangles into tetrahedrons. Essentially, we extend the circumscribed circle into a sphere, for which the enclosed points all belong to a single tetrahedron.

For a given interpolation point P, (x, y), we wish to determine its corresponding $z-$coordinate. We can do so using a variety of methods including linear interpolation, cubic interpolation, etc.

Linear-Based Interpolations: Given the Delaunay triangle or tetrahedral nodes G_i, we wish to determine the estimated $z-$ value for some $P(x, y)$. We do so by assuming a planar surface connecting the nodes. Thus we must simply solve the general equation for any point P belonging to plane by three known points,

$$\begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = a \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + b \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} + c\mathbf{1}. \tag{4.1}$$

A linear-based tetrahedralization will result in a continuous piecewise-linear surface that covers the convex-hull of the scattered input range points. This is the first-order spline [36]. For the application of texture-extraction, this may pose some problems, e.g. the triangularization is not unique, nor is it always optimal (for non-convex shapes [39]). By exploiting more information (i.e. using neighboring nodes), we hope to achieve a better surface interpolation.

Cubic-Based Triangle Interpolations: Unlike the planar linear-based triangularizations, the cubic-based methods use cubic-faced patches that create a continuous and differentiable surface, since the partial derivatives of two neighboring triangles "that share the same triangle edge, in the direction normal to the edge, should be the same" [41]. The very popular approach to solving this problem is the

Clough−Tocher method (1965), which requires 12 known or estimated values: the value and gradient for each of the three vertices, and the normal derivatives at each edge midpoint. However, this system is underdetermined from the cubic relationship, which only provides 10 unknowns,

$$p(x, y) = a_1x^3 + a_2x^2y + a_3xy^2 + a_4y^3 + a_5x^2 + a_6xy + a_7y^2 + a_8x + a_9y + a_{10}, \quad (4.2)$$

To solve this dilemma, the Clough−Tocher method splits every initial triangle into three sub-triangles in order to generate enough relations to solve the system of equations. To achieve a continuous differentiable surface, $m = 3$ order Bernstein-Bézier curves are applied to the Barycentric coordinates, i.e.

$$p(u, v, w) = \sum_{i+j+k=3} \frac{m!}{i!j!k!} b_{i,j,k} u^i v^j w^k, \quad s.t. \quad 0 \leq i, j, k \leq 3 \quad (4.3)$$

which is an equation with 10 unknown constants. As seen in [39], these Bézier control points must be estimated by imposing constraints on cross-boundary derivatives. The control net equations are provided in [41].

4.2.1.2 Thin-Plane-Splines, (TPS)

TPS interpolation has fast become a very popular method for scattered data interpolation [42]. A subclass of RBF, TPS interpolation is analogous to bending a thin sheet of metal to conform to the scattered to set of input data. Put mathematically, we seek $F(u, v)$ of the form $(F_u(x, y), F_v(x, y))$ to map a set of points (x, y)

to (u, v):

$$u = \sum_{i=1}^{m} c_i f\left(\sqrt{(x - x_i)^2 + (y - y_i)^2}\right) + a_{10}x + a_{01}y + a_{00} \tag{4.4}$$

$$v = \sum_{j=1}^{n} d_j f\left(\sqrt{(x - x_j)^2 + (y - y_j)^2}\right) + b_{10}x + b_{01}y + b_{00}, \tag{4.5}$$

where the functions to be minimized are,

$$F_u(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\left(\frac{\partial^2 u}{\partial x^2}\right)^2 + 2\left(\left(\frac{\partial^2 u}{\partial x \partial y}\right)\right)^2 + \left(\frac{\partial^2 u}{\partial y^2}\right)^2\right) dxdy \tag{4.6}$$

$$F_v(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\left(\frac{\partial^2 v}{\partial x^2}\right)^2 + 2\left(\left(\frac{\partial^2 v}{\partial x \partial y}\right)\right)^2 + \left(\frac{\partial^2 v}{\partial y^2}\right)^2\right) dxdy. \tag{4.7}$$

It has been shown ([42], [36]) that selecting $f(t) = t^2 \log t^2$ with $t = \sqrt{(x - x_i)^2 + (y - y_i)^2}$ minimizes the bending energy equations. If we wish to perform smoothing in addition to the TPS interpolation, a regularized minimization can be constructed as follows:

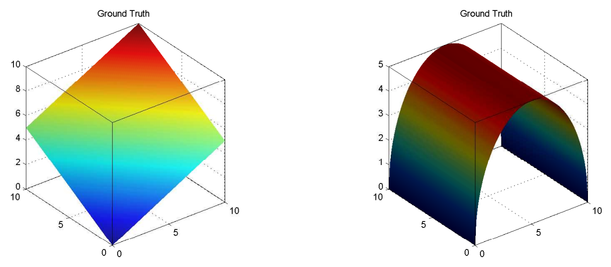
$$G_u(x, y) = \sum_{i=1}^{m} (u_i - u(x_i, y_i))^2 + \lambda F(u) \tag{4.8}$$

$$G_v(x, y) = \sum_{i=1}^{n} (v_i - v(x_i, y_i))^2 + \lambda F(v). \tag{4.9}$$

4.2.2 Interpolation Simulations

Appendix B presents the results of the interpolation simulations. Two surfaces are shown in the results section (though many others were tested during this thesis), a slanted planar surface, and a cylindrical surface. Both surfaces were first created in a noise-free, densely sampled manner, as shown in Figure 4.2.

To simulate the SFSL 3D range data, these ground-truth surfaces were then sparsely sampled by taking one-tenth the number of original samples. Three interpo-



(a) Slanted Planar Surface (b) Cylindrical Surface

Figure 4.2: Original noise-free, densely sampled surfaces.

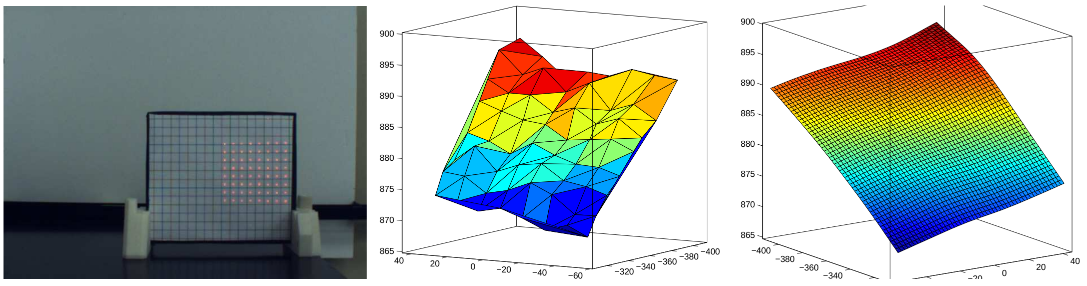
lation methods are shown for each surface, linear-triangulation, cubic-triangulation, and TPS. For each simulation, the interpolation results were compared to the ground-truths to determine the performance. These results are displayed for the noisy cases in Appendix B. Table 4.1 summarizes the interpolation results. The TPS interpolation performed best in each case.

Table 4.1: Interpolation Mean-Squared Error Percentages

	Noise-Free	**Noisy**	**Residual**
Cylinder, Linear	0.696	2.459	1.764
Cylinder, Cubic	0.538	2.708	2.171
Cylinder, TPS	0.496	1.426	0.930
Plane, Linear	0.000	1.406	1.406
Plane, Cubic	0.000	1.801	1.801
Plane, TPS	0.000	1.082	1.082

4.2.3 3D Shape Estimation and Interpolation Experiments

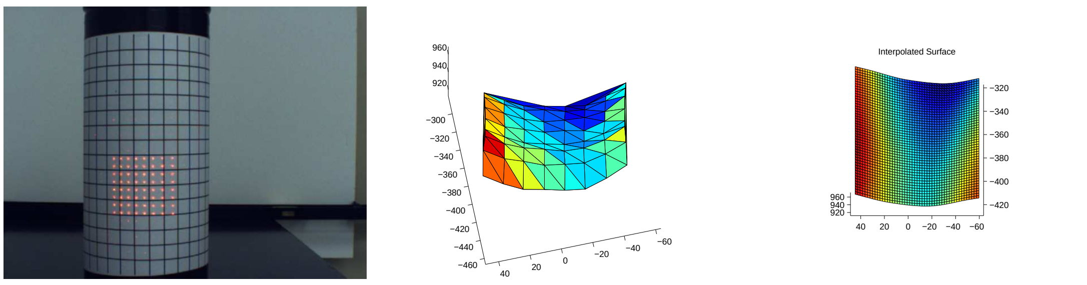
To test the entire SFSL system, various objects were imaged and reconstructed. Similar objects to those in the previous simulations were selected for easy visual comparison between simulation and actual experimental results. The first object imaged was a planar surface, shown in Figure 4.3. The first image is the the scene view from the camera, the second is the linear-interpolated 3D shape, and the third is the TPS interpolated 3D shape.



(a) Original Image (b) Linear Interpolation (c) TPS Interpolation

Figure 4.3: Interpolation experiment for planar surface.

Next, a cylindrical object was imaged, as shown in Figure 4.4.



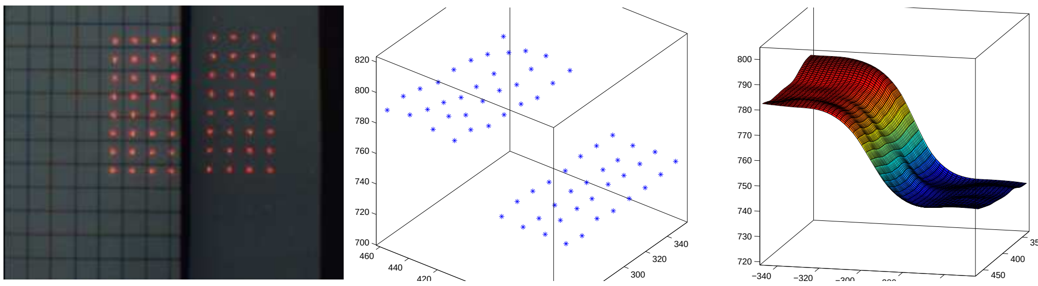
(a) Original Image (b) Linear Interpolation (c) TPS Interpolation

Figure 4.4: Interpolation experiment for cylindrical surface.

To test the how TPS dealt with sharp surface edges, a step surface was imaged, The last object is a corrugated foam surface, Figure 4.6.

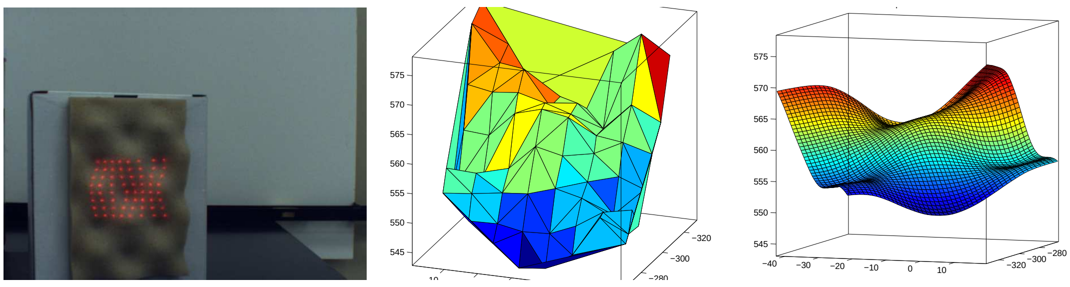
4.3 Image warping from 3D Shape

Up to this point, this thesis has demonstrated a 3D shape estimation system that can quickly and accurately determine the shape of an object. An example



(a) Original Image (b) Point Cloud (c) TPS Interpolation

Figure 4.5: Interpolation experiment for step surface.



(a) Original Image (b) Linear Interpolation (c) TPS Interpolation

Figure 4.6: Interpolation experiment for corrugated surface.

application is that of texture extraction via image warping from 3D shape. After data interpolation, image warping performs the final task of generating a shape-distortion free 2D image from the original (distorted) 2D image. This process is known as image-warping, more specifically, image-flattening. There are many (actually infinite) number of ways to take a 3D shape and project it to a 2D planar surface. A common method used by cartographers for centuries is orthographic projection [43]. Unfortunately, this orthographic projection (along with any other projection) does not capture the inherent distortion arising from the 3D shape. Therefore, we consider warping models that can in some sense preserve features that lie on the 3D surface when projected to two dimensions. Such a mapping is an *isometric* mapping from the estimated 3D surface to the 2D image plane in which local distances are preserved. Unfortunately, Gauss proved that such an isometric mapping between two curves with different intrinsic curvature is *not* possible [44]. In other words, for a true isometric mapping to exist, the 3D surface must have zero Gaussian curvature [38]. This section explores two methods for embedding a 3D graph such as the ones resulting from the SFSL technique on a 2D plane. This embedding serves as the shape-distortion correction for the original captured image. The first method explored is a graph flattening technique using parameterization, while the second is a dimensionality reduction technique applied to the 3D to 2D embedding problem.

4.3.1 Free-boundary Mesh Parameterization

Given an arbitrary graph, a common question is how to best draw or display it. For a 2D embedding, a visually-appealing method to arrange and display graphs is the Tutte embedding [45] which places each graph node in the center of gravity of its neighboring nodes. This is can equivalently be thought of as a physical spring-mass models (SPM) such as [46, 47] where each node is a mass, and each edge is some string. By applying a force to the graph, the graph can be flattened to a 2D plane. Tutte's embedding begins by examining the boundary of the mesh. If this boundary is embedded as a convex shape in the 2D plane, and each interior vertex of the graph is convex combination of its neighbors (Barycentric coordinates) with edges being straight lines, then the 2D planar embedding contains only convex faces [46]. The resulting planar embedding contains graph edges which do not intersect. To generalize, we may define a mesh parameterization as a mapping $P : M \mapsto U$ where M is the original 3D triangulated graph and U is the planar triangulation with original points $\mathbf{x}_i \mapsto \mathbf{u}_i$ (the 2D representations). According to Floater [48], by parameterizing the surface via $\mathbf{u}$, which are Barycentric or other convex-combinations of neighboring nodes, and by specifying a shape-preserving scheme we can generate a planar triangulation that is isomorphic to the original surface triangulation (the proof of which is given in [48], for certain convex surfaces). Mathematically, choose $\mathbf{u}_{n+1}, \ldots, \mathbf{u}_N$ to be the vertices of a $K-$sided convex polygon in M. We can compute the new coordinates by solving

$$\mathbf{u}_i = \sum_{j=1}^{N} \lambda_{ij} \mathbf{u}_j, \quad i = 1, \ldots, n, \quad \lambda_{ij} \geq 0, \quad \sum_{j=1}^{N} \lambda_{ij} = 1. \tag{4.10}$$

The nodes $\mathbf{u}_i, \ldots, \mathbf{u}_n$ represent the new nodes for U with straight line edges and triangular faces. To generate a shape-preserving parameterization, we must select appropriate $\lambda_{i,j}$. One such method explored (see [48] for others) uses the area ratio as a guide to compute λ's for a specific Barycentric point p,

$$\lambda_{i,j_1} = \frac{area\,(p, p_1, p_2)}{area\,(p_1, p_2, p_3)}, \quad \lambda_{i,j_2} = \frac{area\,(p_1, p, p_3)}{area\,(p_1, p_2, p_3)}, \quad \lambda_{i,j_3} = \frac{area\,(p_1, p_2, p)}{area\,(p_1, p_2, p_3)}. \tag{4.11}$$

While classic Tutte-embedding requires a fixed boundary, methods such as Desbrun et al. [49], Karni et al. [50] and Wang [51] allow for free-boundary graph parameterizations. This is useful for the problem of texture extraction since most of the 3D shape estimations will not be homeomorphic to a disk, rather they will have some arbitrary boundary. The free-boundary graph parameterizations are achieved by minimizing some distortion measure (see [49] for details) for each vertex placement. For example, some distortion measures for the 3D to 2D embedding include area, perimeter, and Euler characteristics. A wonderful implementation of such a free-boundary parameterization was used for this thesis by Peyré [52].

4.3.2 Isomap Manifold Method

Whereas the SPM lacked a geometric foundation, the Isomap method is based on the inherent geometry of the surface (even if this surface is embedded in a higher dimension). The concept of image flattening via manifolds is rooted in the notion that high-dimensional data (such as 3D range points) really reside on a manifold which can be thought of as a surface that behaves Euclidean locally, but exhibits non-Euclidean behavior globally. For instance, we live on the surface of the Earth which appears flat locally, in which Euclidean geometry pertains; while globally the Earth is (roughly) spherical, which behaves non-Euclidean. For example, measuring the distance between two objects at the local scale is a simple matter of computing the Euclidean distance, while measuring the distance between objects separated by a large distance along the Earth's surface requires a non-Euclidean measurement known as the *geodesic* distance. Thus, the goal of using a manifold approach to image flattening is to exploit the manifold relationship to create a 3D to 2D mapping that serves our texture extraction needs. The key behind such methods lies in graph based methods for computing the geodesic distances between graph nodes.

To approximate geodesic distances, we must first construct a mesh with nodes and connecting edges. Fortunately, this mesh has already been generated via the interpolation methods in Section B. For a given interpolation scheme, a triangulated mesh can always be computed from the known sparse ranges, and interpolated ranges. The triangle edges connecting each node serves as a first nearest-neighbor distance. One of the most widely used methods for computing the minimal geodesic distance between two point is Dijkstra's method [53]. More recently, Kimmel and Sethian's Fast-Marching approach on triangulated surfaces [54, 55].

The Dijkstra algorithm works on tree-graphs (that is, graphs where only one path exists between two nodes). For a given node v_s in graph G with nodes $\mathbf{v}$ and edges $\mathbf{e}$, we wish to determine the minimal geodesic distance to some other node v_d. The process involves three steps. The first step is to construct a list of all other nodes which connect to v_s through at least one edge. We begin by setting $v_0 = v_s$, and computing T for every other node,

$$T(v_i) = \min\left(T(v_i), T(v_0) + d_{0,i}\right) \tag{4.12}$$

where $T(v_i)$ is initialized to be zero for the source node, and infinity for every other node, and $d_{0,i}$ is the distance along the triangle edge connecting v_0 and v_i. The list of vertices is updated according to this rule (using a min-heap data structure); each time the $v_i = \arg\min_{v_i} T(v_i)$ is chosen as the new v_0 if $T(v_i)$ is a value less than infinity. This process is repeated until the node v_d is reached. The collection of $v_i's$ designates the geodesic path from the source v_s to destination v_d.

Once the geodesic distances have been approximated, we wish to find the mapping that preserves distances along the curved surface (geodesic distances) d_{ij} to the distances along the planar surface (Euclidean distances) $\hat{d}_{ij}$. We can do so by minimizing the following (i.e. apply MDS),

$$L = \frac{1}{c} \sum_{i<j}^{i=N} \frac{\left(d_{ij} - \hat{d}_{ij}\right)^2}{\hat{d}_{ij}}, \tag{4.13}$$

where $c = \sum \hat{d}_{ij}$. L may be solved via gradient methods as described before. Another perspective of the same problem is to treat the Isomap as a Kernel Principle Component Analysis (PCA) method. PCA is a classic data dimensionality technique that projects high dimensional data to a lower dimension in such a way so that the data variance is preserved. PCA fails when data is not linearly-separable in its native space. Kernel PCA addresses this problem by first projecting the data (typically nonlinearly) into a higher-dimensional space in which the data is linearly-separable by some hyperplane. Thus, we can view the Isomap problem in the same light. The given data belongs to $\mathbb{R}^3$, but might not have a clear projection into $\mathbb{R}^2$; thus we must first project the data into a higher dimension. To do so, we construct a kernel as follows:

$$K = \frac{1}{2}CDC \tag{4.14}$$

where $C = I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$ is a centering matrix, and D is the element wise square of geodesic distances, i.e. $D(i,j) = d_{i,j}^2$. Following the Kernel-PCA procedure, we avoid directly computing the covariance matrix, and instead use the known kernel and its eigen-decomposition,

$$N\lambda\mathbf{a} = K\mathbf{a}. \tag{4.15}$$

The two eigenvectors associated with the two largest eigenvalues represent the mapping of the data onto the 2D plane.

4.3.3 Warping Simulations

This section presents the warping simulations ran in order to test the validity of the selected warping methods. First, simple 3D shapes were modeled, and orthographically texture mapped with a known texture. The texture extraction problem is the dual of the texture mapping problem[2], thus we can judge the performance of texture extraction equivalently by texture mapping. This process begins by extracting a non-shape distorted texture from the 3D surface. By flattening this non-shape distorted texture, we are actually applying shape-distortion. The first example is

[2]Texture mapping and texture extraction are dual problems trivially because determining a mapping from some surface S_1 to surface S_2 is the same as determining the mapping from S_2 to S_1.

a test to verify that the geodesic distances closely approximate euclidean distances when the surface is well-behaved. The surface is a plane, as shown in Figure 4.7. The parameterized and Isomap flattened graphs are shown in Figure 4.8. To vi-

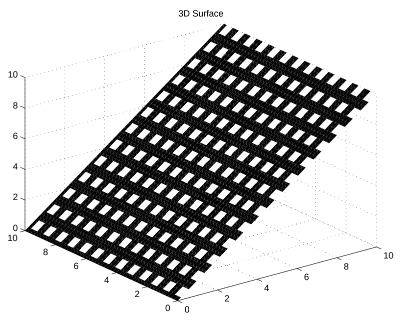
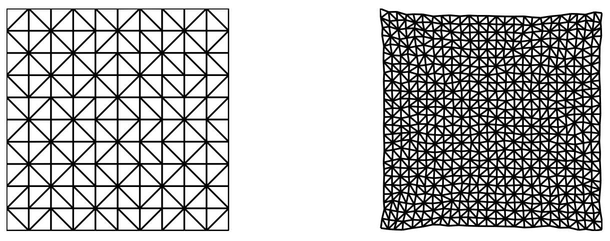


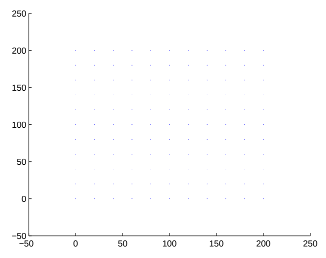
Figure 4.7: Slanted planar surface with regular texture.



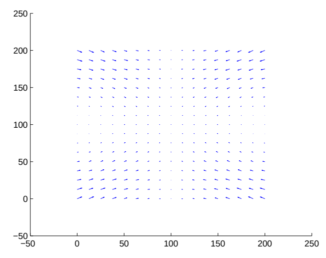
(a) Parameterization (b) Isomap

Figure 4.8: Distortion correcting graphs for planar surface.

sualize how the distortion correcting graphs move image pixels, a quiver plot was generated using the vector difference between the original (regular) grid, and the correcting grid. The results for the planar surface are shown in Figure 4.9. Given the new mappings, we can now perform texture extraction. Using the graphs as control points, a TPS morph is used to generate the estimated texture as shown in Figure 4.10. The next surface tested was a cylindrical surface, as shown in Figure 4.11. The distortion correcting graphs are shown in Figure 4.12. The quiver

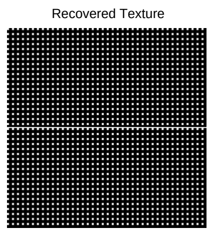


(a) Parameterization

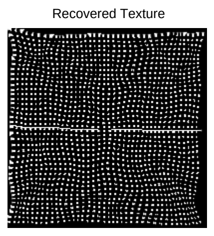


(b) Isomap

Figure 4.9: Quiver plots for planar surface.



(a) Parameterization



(b) Isomap

Figure 4.10: Estimated texture for planar surface.

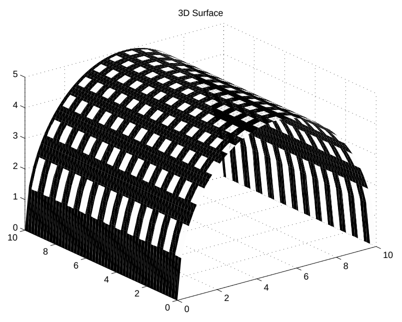
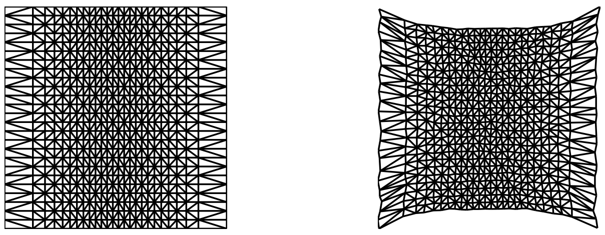


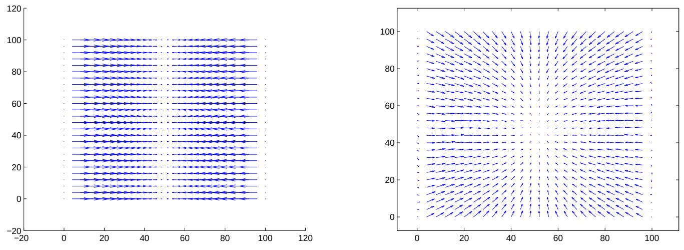
Figure 4.11: Cylindrical surface with regular texture.



(a) Parameterization (b) Isomap

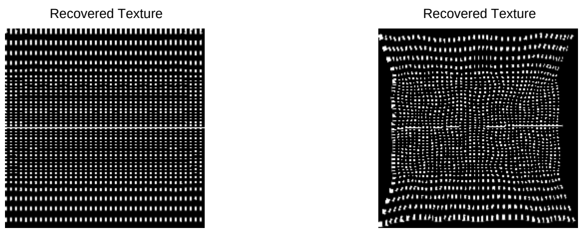
Figure 4.12: Distortion correcting graphs for cylindrical surface.

plots are shown in Figure 4.13. The estimated textures are shown in Figure 4.14.



(a) Parameterization (b) Isomap

Figure 4.13: Quiver plots for cylindrical surface.

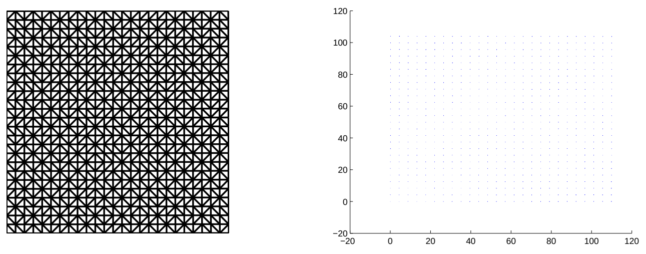


(a) Parameterization (b) Isomap

Figure 4.14: Estimated texture for cylindrical surface.

4.3.4 Warping Experiment

The first set of warping experiments involved imaging a known grid-pattern on various surfaces, including planes, and cylinders. The first experiment tested how the image warping worked in a zero-warp scenario. A plane was positioned roughly orthogonal to the camera's principal axis, as shown in Figure 4.3. From the estimated surface, the distortion-correcting graph (using the free-boundary method) was generated, along with the quiver plot as shown in Figure 4.15. The distortion-



(a) Parameterized distortion-correcting graph

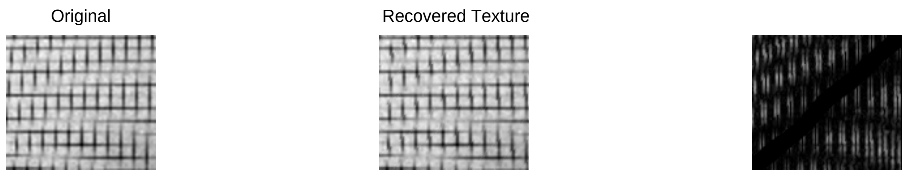
(b) Quiver Plot

Figure 4.15: Distortion correcting graph and resulting quiver plot for orthogonal plane experiment.

correction graph is nearly identical to a normal grid, as shown in the quiver plot, thus indicating the warping function is not trying to correct a shape distortion. After performing TPS morphing using the new graph, the estimated texture was extracted, as shown in Figure 4.16. As anticipated, the resulting estimated texture is nearly identical to the input texture. Next, the plane was then tilted, and the experiment was repeated yielding a straightened version of the grid texture as shown in Figure 4.17.

(a) Original Imaged Texture (b) Estimated Texture

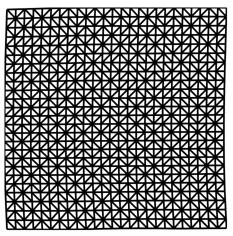
Figure 4.16: The imaged texture (blue channel) and the output of the TPS morphing for the orthogonal plane.



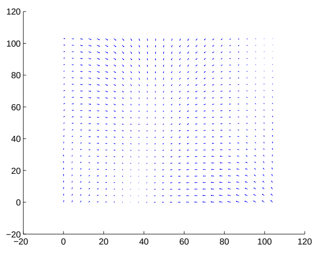
(a) Original Imaged Texture (b) Estimated Texture (c) Abs. Difference

Figure 4.17: The imaged texture (blue channel) and the output of the TPS morphing for the slanted plane.

The next target was a cylinder with the same known grid texture applied to its surface. The original image and model were shown in Figure 4.4. From the estimated surface, the distortion-correcting graphs were generated. Comparing this distortion correcting graph to a synthetic graph such as in Figure 4.12(a) we can immediately see that the amount of shape distortion is significantly less. This is due to the cylinder's position being sufficiently far away from the camera. If the cylinder were placed closer, the shape distortion would be aggravated. After performing TPS morphing using the new graph, the final texture was extracted, shown in Figure 4.19. The image warping algorithm attempts to straighten the horizontal lines occurring from the curvature of the cylinder, in addition to displacing the vertical lines on the edges of the image to account for the perspective distortion.

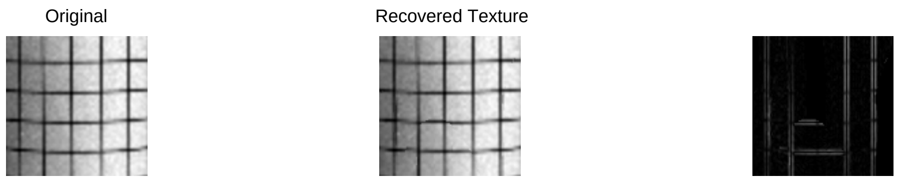


(a) Parameterized distortion-correcting graph



(b) Quiver Plot

Figure 4.18: Distortion correcting graph and resulting quiver plot for cylinder experiment.



(a) Original Imaged Texture (b) Estimated Texture (c) Abs. Difference

Figure 4.19: The imaged texture (blue channel) and the output of the TPS morphing.

Chapter 5

Conclusion

This thesis presented a system comprised of only a DOE laser projector and camera capable of 3D shape estimation. By using a static structured-light pattern that sparsely samples surfaces, such a system is capable of very efficient implementations. As such, the presented SFSL is appropriate for measuring the 3D shape of a stationary or moving object via a hand-held device. While the proof-of-concept system is limited in spatial and depth resolution, future versions of the system could achieve much higher resolution by using different optics (DOE and camera). Along with the efficient shape estimation algorithm, the supporting algorithms such as projector-to-camera calibration improve upon previous designs by eliminating the need for calibrated mechanisms or complicated procedures. In the current version, calibration of both the intrinsic camera parameters and the extrinsic projector-to-camera parameters is accomplished via one set of calibration images, thus reducing the time to setup the system while improving its calibration accuracy over other such SFSL systems. Compared to other SFSL systems, the pyramidal laser pattern combined with epipolar geometry estimation allows for easy-setup and a simplified correspondence problem.

This thesis explored the SFSL problem, including the geometry of two-view perspective, triangulation, identifying fiducials, the correspondence problem, and surface interpolation /smoothing. In addition, this thesis examined application of texture extraction from SFSL. SFSL simulations verified that the mathematics for triangulation were correct, followed by experiments with various surfaces. The experimental results using the prototype system were satisfactory, showing that such a system is capable of 3D shape estimation even with limited hardware (such as the DOE in the prototype). Future systems will use a higher-quality laser and DOE capable of projecting denser patterns on surfaces at a close range, thus enabling for even higher-quality 3D shape estimations. The texture extraction simulations showed that recovering the texture from a 2D image and a 3D model is possible via the parameterization and manifold methods contained in this thesis. While the simulations were very promising, the experimental results suffered more from 3D model noise and insufficient camera resolution, which created disturbances in the distortion-correcting grids, thus making this particular application rather limited. When combined with the future SFSL system, accurate texture extraction will be highly achievable. One possible extension to this work is to use random laser patterns paired with compressive sensing for surface sampling. Such a system may combine the efficiency of the current system with higher-accuracy of surface reconstruction.

Appendix A

Triangulation Performance Simulation Results

Table A.1: Triangulation Error Percentage on Orthogonal Planar Surface, zero noise

0.0e+000	0.0e+000	0.0e+000	7.1e-014	1.4e-013	7.1e-014	-7.1e-014	-1.4e-013
0.0e+000	-7.1e-014	7.1e-014	7.1e-014	7.1e-014	0.0e+000	-7.1e-014	1.4e-013
1.4e-013	7.1e-014	-7.1e-014	-7.1e-014	-2.1e-013	-1.4e-013	1.4e-013	7.1e-014
0.0e+000	-7.1e-014	7.1e-014	0.0e+000	1.4e-013	7.1e-014	7.1e-014	0.0e+000
-7.1e-014	0.0e+000	1.4e-013	0.0e+000	-1.4e-013	-4.3e-013	3.6e-013	1.4e-013
7.1e-014	-7.1e-014	3.6e-013	0.0e+000	-7.1e-014	-7.1e-014	-7.1e-014	-2.8e-013
1.4e-013	-7.1e-014	-7.1e-014	2.1e-013	7.1e-014	-2.1e-013	7.1e-014	-1.4e-013
2.1e-013	7.1e-014	0.0e+000	-1.4e-013	-2.8e-013	-3.6e-013	7.1e-014	-1.4e-013

Table A.2: Triangulation Error Percentage on Orthogonal Planar Surface, with additive Gaussian noise

-2.1	1.2	-2.2	-2.7	0.0	1.3	-3.7	2.1
0.4	-1.4	0.2	-0.4	-1.8	2.0	0.6	2.4
0.9	-1.1	-2.7	5.1	-2.8	1.9	1.6	-2.5
-1.1	-1.8	1.3	3.4	4.7	-6.1	1.2	0.3
-1.9	0.4	0.5	-0.1	3.5	0.2	-1.2	0.7
3.0	-1.1	-1.1	0.7	-0.5	-0.3	3.9	1.9
-1.1	-0.8	0.7	4.1	0.1	-3.5	-0.3	-2.4
-0.5	1.0	-3.7	-0.1	-1.3	0.6	2.5	-0.8

Table A.3: Triangulation Error Percentage on Angled Surface, with additive Gaussian noise

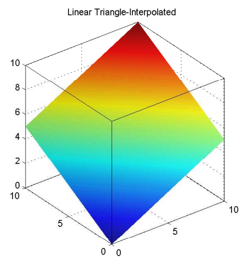
-1.4	0.5	-0.2	-0.6	-4.2	1.3	-1.6	-2.4
0.5	-0.9	3.5	-0.4	-1.8	-0.5	-3.3	-1.3
2.3	-0.0	1.0	-3.4	-1.3	-0.9	2.7	0.1
1.3	-2.5	-1.4	-4.3	-0.5	-1.9	-3.8	1.9
-0.1	0.4	-0.7	-1.3	-2.1	1.7	1.6	0.2
-2.7	2.3	-1.3	1.2	-0.5	2.4	5.1	3.8
1.9	-0.7	-2.2	-0.0	-1.5	-2.0	2.1	1.4
-0.2	0.7	2.5	0.6	0.6	3.3	-1.4	-1.8

Table A.4: Triangulation Error Percentage on Step Surface, with additive Gaussian noise

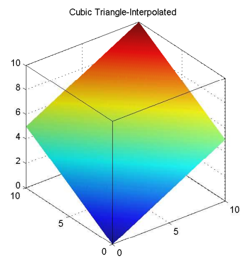
0.6	-3.8	-0.2	-1.0	-1.7	0.6	2.5	3.1
0.2	-2.4	1.4	-0.7	2.4	0.5	-0.3	-2.6
-0.6	2.0	0.7	3.9	1.3	1.3	-4.4	1.3
2.0	-2.7	0.4	0.4	-0.8	-1.7	1.8	3.0
0.5	-1.9	-0.3	2.2	-0.6	-1.4	3.8	1.7
-0.7	0.8	0.0	2.9	-6.8	3.9	-1.2	-6.6
1.3	-0.8	0.5	-2.6	0.1	2.4	4.8	2.5
-1.3	1.5	0.4	0.6	0.5	2.0	-0.7	0.2

Appendix B

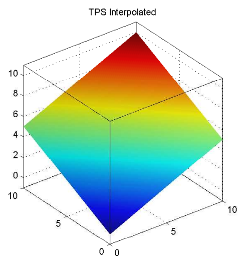
Interpolation Simulation Results



(a) Linear Interpolation

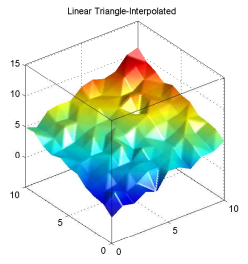


(b) Cubic Interpolation

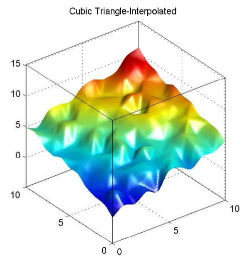


(c) TPS Interpolation

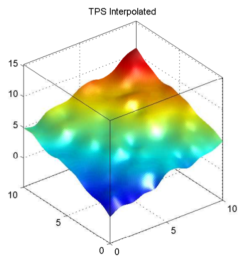
Figure B.1: Interpolation results for noise-free slanted surface.



(a) Linear Interpolation

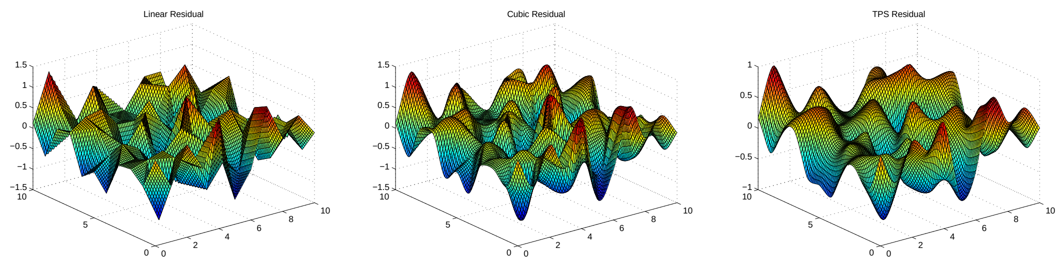


(b) Cubic Interpolation



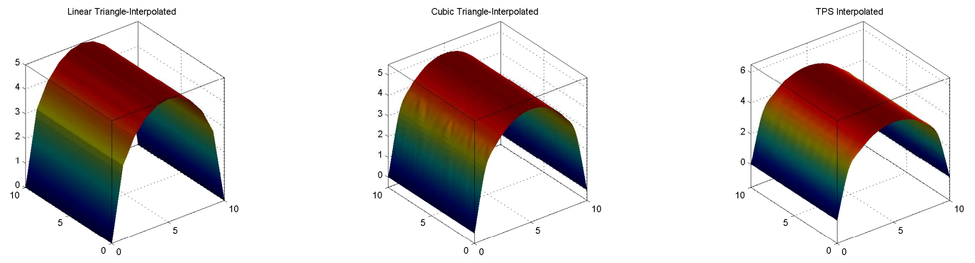
(c) TPS Interpolation

Figure B.2: Interpolation results for noisy slanted surface.



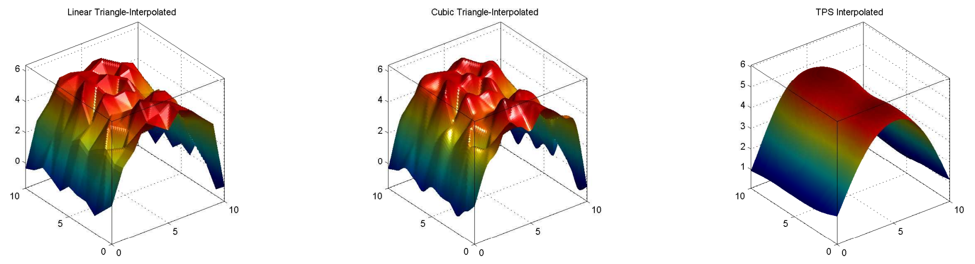
(a) Linear Interpolation (b) Cubic Interpolation (c) TPS Interpolation

Figure B.3: Residual difference of interpolation and ground truth.



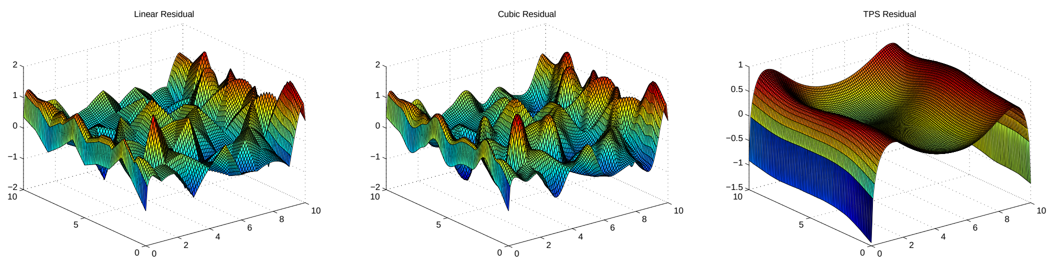
(a) Linear Interpolation (b) Cubic Interpolation (c) TPS Interpolation

Figure B.4: Interpolation results for noise-free cylinder surface.



(a) Linear Interpolation (b) Cubic Interpolation (c) TPS Interpolation

Figure B.5: Interpolation results for noisy cylinder surface.



(a) Linear Interpolation (b) Cubic Interpolation (c) TPS Interpolation

Figure B.6: Residual difference of interpolation and ground truth.

Bibliography

[1] F. Chen, G. Brown, and M. Song, "Overview of three-dimensional shape measurement using optical methods," *Optical Engineering*, vol. 39, p. 10, 2000.

[2] P. Besl, "Active, optical range imaging sensors," *Machine vision and applications*, vol. 1, no. 2, pp. 127–152, 1988.

[3] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, 1993.

[4] B. Horn and M. Brooks, "The variational approach to shape from shading," *Computer Vision, Graphics, and Image Processing*, vol. 33, no. 2, pp. 174–208, 1986.

[5] R. Zhang, P. Tsai, J. Cryer, and M. Shah, "Shape from shading: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 21, no. 8, pp. 690–706, 1999.

[6] J. Malik and R. Rosenholtz, "Computing local surface orientation and shape from texture for curved surfaces," *Int. J. Comput. Vision*, vol. 23, no. 2, pp. 149–168, June 1997.

[7] J. Garding, "Direct estimation of shape from texture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1202–1208, 1993.

[8] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," *International Journal of Computer Vision*, vol. 9, no. 2, pp. 137–154, 1992.

[9] J. Li and R. Chellappa, "Structure from planar motion," *IEEE Transactions on Image Processing*, vol. 15, no. 11, p. 3466, 2006.

[10] J. Salvi, J. Pages, and J. Batlle, "Pattern codification strategies in structured light systems," *Pattern Recognition*, vol. 37, no. 4, pp. 827–849, 2004.

[11] Y. Shirai, "Recognition of polyhedrons with a range finder," *Pattern Recognition*, vol. 4, no. 3, pp. 243 – 244, 1972. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-48MXMTT-29/2/d0fb691ab4fd33594128b16dab164239

[12] P. Will and K. Pennington, "Grid coding: a novel technique for image processing," *Proceedings of the IEEE*, vol. 60, no. 6, pp. 669–680, 1972.

[13] J. Guehring, "Reliable 3d surface acquisition, registration and validation using statistical error models," *3D Digital Imaging and Modeling, International Conference on*, vol. 0, p. 224, 2001.

[14] K. Nakano, Y. Watanabe, and S. Kanno, "Extraction and recognition of 3-dimensional information by projecting a pair of slit-ray beams," in *ICPR88*, 1988, pp. II: 736–743.

[15] N. Shrikhande and G. Stockman, "Surface orientation from a projected grid," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 6, pp. 650–655, 1989.

[16] L. Guisser, R. Payrissat, and S. Castan, "PGSD: an accurate 3D vision system using a projected grid for surface descriptions," *Image and vision computing*, vol. 18, no. 6-7, pp. 463–491, 2000.

[17] A. Dipanda and S. Woo, "Towards a real-time 3d shape reconstruction using a structured light system," *Pattern Recognition*, vol. 38, no. 10, pp. 1632–1650, 2005.

[18] J. Salvi, J. Batlle, and E. Mouaddib, "A robust-coded pattern projection for dynamic 3d scene measurement," *Pattern Recognition Letters*, vol. 19, no. 11, pp. 1055–1065, 1998.

[19] C. Albitar, P. Graebling, and C. Doignon, "Robust structured light coding for 3d reconstruction," in *IEEE Int. Conf. on Computer Vision*, 2007.

[20] J. Posdamer and M. Altschuler, "Surface measurement by space-encoded projected beam systems," *Computer Graphics and Image Processing*, vol. 18, no. 1, pp. 1–17, 1982.

[21] E. Horn and N. Kiryati, "Toward optimal structured light patterns," in *NRC '97: Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling.* Washington, DC, USA: IEEE Computer Society, 1997, p. 28.

[22] B. Curless and M. Levoy, "Better optical triangulation through spacetime analysis," in *Proceedings of IEEE International Conference on Computer Vision*, vol. 95, 1995, pp. 987–994.

[23] H. Herzig, *Micro-optics: Elements, systems and applications.* CRC, 1997.

[24] A. Dipanda and S. Woo, "Structured light system configuration determination for efficient 3D surface reconstruction," in *Image Processing, 2004. ICIP'04. 2004 International Conference on*, vol. 5, 2004, pp. 3005–3008.

[25] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521623049, 2000.

[26] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition).* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[27] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[28] M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 146–168, 2004.

[29] L. di Stefano and A. Bulgarelli, "A simple and efficient connected components labeling algorithm," *Image Analysis and Processing, International Conference on*, vol. 0, p. 322, 1999.

[30] A. Awwal, K. Rice, and T. Taha, "Fast implementation of matched-filter-based automatic alignment image processing," *Optics and Laser Technology*, vol. 41, no. 2, pp. 193–197, 2009.

[31] Z. Yang and Y. Wang, "Error analysis of 3d shape construction from structured lighting," *PR*, vol. 29, no. 2, pp. 189–206, February 1996.

[32] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *International Conference on Computer Vision*, vol. 1, 1999, pp. 666–673.

[33] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* IEEE, 1997, pp. 1106–1112.

[34] T. Clarke and J. Fryer, "The development of camera calibration methods and models," *Photogrammetric Record*, vol. 16, no. 91, pp. 51–66, 1998.

[35] J. Bouguet, "Matlab camera calibration toolbox," in *TR*, 2000.

[36] C. Glasbey and K. Mardia, "A review of image-warping methods," *Journal of Applied Statistics*, vol. 25, no. 2, pp. 155–172, 1998.

[37] A. Goshtasby, "Piecewise linear mapping functions for image registration," *Pattern Recognition*, vol. 19, no. 6, pp. 459–466, 1986.

[38] E. Schwartz, A. Shaw, and E. Wolfson, "A numerical solution to the generalized mapmaker's problem: flattening nonconvex polyhedral surfaces," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 11, no. 9, pp. 1005–1008, Sep 1989.

[39] I. Amidror, "Scattered data interpolation methods for electronic imaging systems: a survey," *Journal of Electronic Imaging*, vol. 11, no. 2, pp. 157–176, 2002.

[40] S. Choi, "The delaunay tetrahedralization from delaunay triangulated surfaces," in *SCG '02: Proceedings of the eighteenth annual symposium on Computational geometry.* New York, NY, USA: ACM, 2002, pp. 145–150.

[41] A. Goshtasby, "Piecewise cubic mapping functions for image registration," *Pattern Recognition*, vol. 20, no. 5, pp. 525–533, 1987.

[42] F. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 11, no. 6, pp. 567–585, 1989.

[43] J. Snyder, *Flattening the earth: two thousand years of map projections.* University of Chicago Press, 1997.

[44] G. Zigelman, R. Kimmel, and N. Kiryati, "Texture mapping using surface flattening via multidimensional scaling," *IEEE Transactions on Visualization and Computer Graphics*, pp. 198–207, 2002.

[45] W. Tutte, "The Use of Numerical Computations in the Enumerative Theory of Planar Maps," *Jeffery-Williams Lectures, 1968-1972*, p. 73, 1972.

[46] M. Brown and W. Seales, "Document restoration using 3D shape," in *ICCV01*, 2001.

[47] J. Shen, B. Mastuszewski, L. Shark, C. Moore, and U. Manchester, "Deformable image registration using spring mass system," in *Proceedings of Medical Image Understanding and Analysis.* Citeseer, 2006.

[48] M. Floater, "Parametrization and smooth approximation of surface triangulations," *Computer Aided Geometric Design*, vol. 14, no. 3, pp. 231–250, 1997.

[49] M. Desbrun, M. Meyer, and P. Alliez, "Intrinsic parameterizations of surface meshes," in *Computer Graphics Forum*, vol. 21, no. 3. Citeseer, 2002, pp. 209–218.

[50] Z. Karni, C. Gotsman, and S. Gortler, "Free-boundary linear parameterization of 3D meshes in the presence of constraints," in *Proceedings of the International Conference on Shape Modeling and Applications.* Citeseer, 2005, pp. 268–277.

[51] C. Wang, "Computing length-preserved free boundary for quasi-developable mesh segmentation," *IEEE transactions on visualization and computer graphics*, vol. 14, no. 1, pp. 25–36, 2008.

[52] G. Peyré, "Graph toolbox for matlab," 2007. [Online]. Available: http://www.ceremade.dauphine.fr/ peyre/matlab/graph/content.html

[53] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[54] J. Sethian *et al.*, *Level set methods and fast marching methods.* Cambridge university press Cambridge, 1999.

[55] R. Kimmel and J. Sethian, "Computing geodesic paths on manifolds," *Proc. Natl. Acad. Sci. USA*, vol. 95, no. 15, pp. 8431–8435, 1998.