DATA ACQUISITION AND PROCESSING INTERFACE DEVELOPMENT FOR 3D LASER RANGEFINDER

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ORÇUN ÇEVİKBAŞ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

AUGUST 2006

Approval of the Graduate School of Natural and Applied Sciences

Prof.Dr.Canan ÖZGEN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science

Prof. Dr. S. Kemal İDER
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science

Asst. Prof. Dr. E. İlhan KONUKSEVEN
Supervisor

Examining Committee Members

Prof. Dr. Samim ÜNLÜSOY (Chairman) (METU,ME) _____

Asst. Prof. Dr. E. İlhan KONUKSEVEN (METU,ME) _____

Prof. Dr. Reşit SOYLU (METU,ME) _____

Inst. Dr. Yiğit YAZICIOĞLU (METU,ME) _____

Asst. Prof. Dr. Afşar SARANLI (METU,EEE) _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Orçun ÇEVİKBAŞ

ABSTRACT

DATA ACQUISITION AND PROCESSING INTERFACE DEVELOPMENT
FOR 3D LASER RANGEFINDER

Çevikbaş, Orçun
M.Sc., Department of Mechanical Engineering
Supervisor : Asst. Prof. Dr. E. İlhan Konukseven

August 2006, 138 pages

In this study, it is aimed to improve the previously developed data acquisition program which was run under DOS and 2D surface reconstruction program under Windows. A new system is set up and both data acquisition and processing software are developed to collect and process data within just one application, running under Windows.

The main goal of the thesis is to acquire and process the range data taken from the laser rangefinder in order to construct the 3D image map of simple objects in different positions for indoor environments.

The data acquisition program collects data in helical way. To do this, appropriate parameters for the data acquisition interface are determined. In the data processing step, it is aimed to use basic triangulation algorithms and threshold conditions to calculate resolutions, detect noisy points and segment objects from the environment for line fitting.

The developed and implemented data acquisition and processing interfaces in the thesis are capable of creating 3D image map and obtaining the view of scanned environment in a short time with high accuracy.

Keywords: laser rangefinder, data acquisition, data processing

ÖZ

3 BOYUTLU LAZER MESAFE BULUCU İÇİN VERİ TOPLAMA VE İŞLEME ARAYÜZÜNÜN GELİŞTİRİLMESİ

Çevikbaş, Orçun
Yüksek Lisans, Makina Mühendisliği Bölümü
Tez Yöneticisi : Y. Doç. Dr. E. İlhan Konukseven

Ağustos 2006, 138 sayfa

Bu çalışmada, önceden geliştirilmiş, DOS ortamında çalıştırılan veri toplama programının ve Windows ortamında çalıştırılan 2 boyutta yüzey oluşturma programlarının geliştirtmesi hedeflenmiştir. Yeni bir sistem kurulmuş, veri toplama ve işleme yazılımları Windows ortamında çalıştırılabilen tek bir uygulama altında veri toplayacak ve işleyecek şekilde geliştirilmiştir.

Bu tezin asıl amacı, değişik pozisyonlardaki basit cisimlerin, iç ortamlarda 3 boyutlu görüntü haritasını oluşturmak için lazer mesafe bulucudan menzil verilerinin elde edilmesi ve işlenmesidir.

Veri toplama programı, verileri helisel şekilde toplamaktadır. Bunun için, veri toplama arayüzünde kullanılacak uygun parametreler belirlenmiştir. Veri işleme aşamasında, çözünürlüklerin hesaplanması, hatalı noktaların belirlenmesi ve cisimlere doğrular atanması öncesinde bölümleme yapılabilmesi için, temel üçgenleme algoritmaları ve eşik koşulları kullanılmıştır.

Tezde geliştirilen ve uygulanan veri toplama ve işleme arayüzleri, 3 boyutlu görüntü haritası yaratabilmekte ve taranan ortamın görüntüsünü kısa zamanda yüksek doğruluk ile elde edebilmektedir.

Anahtar Kelimeler: lazer mesafe bulucu, veri elde edilmesi, veri işlenmesi

To My Family...

ACKNOWLEDGEMENTS

I would like to thank my supervisor Asst. Prof. Dr. E. İlhan Konuseven for his guidance, advice, criticism, encouragements and insight throughout the research.

I would also like to thank Seçkin Kaplan, Murat Anıl Oral and Gökhan Oral for their suggestions and comments.

I owe many thanks to my friends who study in the METU Mechatronics Laboratory for their precious support and presence.

Lastly, I am deeply grateful to my family who supported and encouraged me all through my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

ϕ	Phase shift
d	Distance to target
λ	Modulation wavelength
c	Speed of light
f	Modulation frequency
RawE1	The raw count data from 1 to 6152 for azimuth encoder
RawE2	The raw count data from 1 to 4220000 for elevation encoder
P	Source data
Q	Destination data
D_{thd}	Threshold condition for segmentation
$D(r_i , r_{i+1})$	Euclidean distance between two consecutive scanned points
C_0	Constant parameter used for noise reduction.
σ_r	Residual variance
S_i	Segments
e	Error in least squares method
p_i	Each point in triangulation
$e_{a,b}$	Edge in triangulation surface normals vectors of n_i, n_j and n_k
q	Additional point in triangulation
O_{real}	Real world object
M^P	Polygonal mesh
B_P	Bounding *box*
C_P	Cell space
c_b	Boundary cell
c_o	Outer cell
f_b	Boundary face
f_i	Inner face
M^I	Initial mesh

α	Amount of the attracting force
$\mathcal{L}_t$	Tangential component of Laplacian vector
λ	Speed parameter

Acronyms

LIDAR	Light Detection and Ranging
NURBS	Non-Uniform Rational B-Splines
DME	Distance Measuring Device
LORAN	Long Range Navigation
EDM	Electrical Discharge Machining
GPS	Global Positioning System
VLBI	Very Long Base Interferometer
VLF	Very Low Frequency
HSIF	High Speed Interface
ICP	Iterative Closest Point
ABD	Adaptive Break Point
SWBF	Shrink-Wrapped Boundary Face

CHAPTER 1

INTRODUCTION

1.1. Introduction

A laser rangefinder is a device which uses laser beam to determine the distance to an object. The most common form of laser rangefinder operates on the time-of-flight principle by sending a laser pulse in a narrow beam towards the object and measuring the time taken by the pulse to be reflected off the target and returned to the sender. Laser rangefinders are also referred to as LIDAR (Light Detection And Ranging).

Different types of rangefinders have different applications, often determined by the type of material being measured. While radar is excellent at determining the range to metal objects, such as aircraft, it is much less effective with rocks and organic material, which may not reflect radio waves at all, and therefore provide no visible signal. Laser rangefinders, in contrast, are able to measure the distance to a much wider range of objects. Organic materials, rocks, and meteorological phenomena all have a large enough reflective footprint to show up clearly using a laser rangefinder. Some surfaces naturally absorb laser beam, however, various military have developed special laser-absorbent paints with which to coat vehicles, making them effectively invisible to a laser rangefinder.

A laser rangefinder may be used for a number of uses other than simply calculating the distance to an object. Application areas of the laser rangefinders can be divided

into three main parts; 3D modeling, military applications and industrial applications.

Laser rangefinders are used extensively in 3D object recognition, 3D object modeling, and a wide variety of computer vision related fields. This technology constitutes the heart of the so-called *time-of-flight* 3D scanners. Laser rangefinders offer high-precision scanning abilities, with either single-face or 360-degree scanning modes. A range map taken using a laser rangefinder is shown in Figure 1.1 [1].

Figure 1.1. A Range Map Taken Using a Laser Rangefinder [1]

A number of algorithms have been developed to merge the range data retrieved from multiple angles of a single object in order to produce complete 3D models with as little error as possible. One of the advantages that laser rangefinders offer over other methods of computer vision is that the computer does not need to correlate features from two images to determine depth information as in stereoscopic methods.

The laser rangefinders used in computer vision applications often have depth resolutions of tenths of millimeters or less. This can be achieved by using

triangulation or refraction measurement techniques as opposed to the time-of-flight techniques used in LIDAR [2].

Laser rangefinders are also used in military applications. The X20 long range professional laser rangefinder is designed for long distance acquisition, and tactical range measurement (Figure 1.2.). The simple to use, point and shoot operation, enables the X20 rangefinder operator to instantaneously measure distances from one meter to 2200 meters with an accuracy of +/-1 meter. [3]

Figure 1.2. X20 Long Range Laser Rangefinder for Military Applications [3]

In order to make laser rangefinders and laser guided weapons less useful against military targets, various military arms may have developed laser absorbing paint for their vehicles. Regardless, some objects don't reflect laser light very well and using a laser range-finder on them is difficult.

Laser rangefinders and measuring sensors are used in many industrial applications such as [1]:

- Hoist applications
- Road profiling
- Metal industry
- Lumber measurement
- Medical industry

A major aircraft engine maintenance and overhaul facility at San Francisco International Airport uses laser rangefinder to reduce their margin for error in critical hoisting applications. After the successful overhaul of a jet engine, maintenance engineers run the engine in a state-of-the-art quality assurance test cell. Overhead hoists lift the heavy engine from an adaptor and carry it to the test cell fixture. The laser sensor reports critical range information so the maintenance engineers can clear fixed obstacles and gently deliver the heavy engine into a secure fixture. This application is similar to that used to measure cranes and hoists in the steel industry or shipping industry.

Road profiling has become a crucial practice for transportation departments throughout the world. These departments are responsible for gathering statistical information on road surfaces including the longitudinal profile, mactrotextures, microtextures and roughness to determine coefficients of friction. Acuity [1] has worked with developers of road profiling equipment and produced laser displacement sensors. The result is an extremely accurate, cost-effective sensor for all surface types, vehicle speeds, and vibration, sunlight, and temperature conditions encountered in profilometry applications. In Figure 1.3, a vehicle retrofitted with a road profiling laser to test the smoothness of pavement is shown.

Figure 1.3. Pavement Smoothness Testing Vehicle [1]

The steel industry demands accurate measuring equipment to be used under difficult circumstances. The environment is harsh, with high dust content and hot temperatures. The targets, hot metal, can be especially difficult because they radiate glowing light at elevated temperatures. The sensors used in this applications are often protected behind insulating glass or within a protective enclosure. These enclosures can be either air or water cooled so that the sensor can be placed close to hot surface targets.

The forest products industry uses laser sensor products for the measuring of logs, lumber, veneer, paperboard, etc. Laser displacement sensors are commonly integrated into automated lumber finishing machines to measure board position, board dimensions and surface profiles (Figure 1.4). The long-distance triangulation sensors offer a valuable stand-off distance, keeping the measuring equipment away from the quickly moving lumber and potential hazards.

Figure 1.4. Laser Displacement Sensor Integrated to a Finishing Mill [1]

Laser displacement sensors have been designed into a medical application that brings new hope for treatment to patients with brain tumors in high-risk areas. The medical device facilitates the resolution of problems previously considered

inoperable. Medtronics' FAZER™ Contour Laser [1], is a first-of-its-type device that enables surgeons to scan surfaces of the patient's anatomy with a laser (Figure 1.5). This profile data helps surgeons associate anatomy features to the patent's CT or MR scan. The FAZER laser eliminates the need for anatomical markers to create this mapping. Mapping using the FAZER can be completed in seconds.

The device's location is tracked by an external sensor attached to the patient's head. The patient is asleep during the procedure and his/her eyes are covered for laser safety.

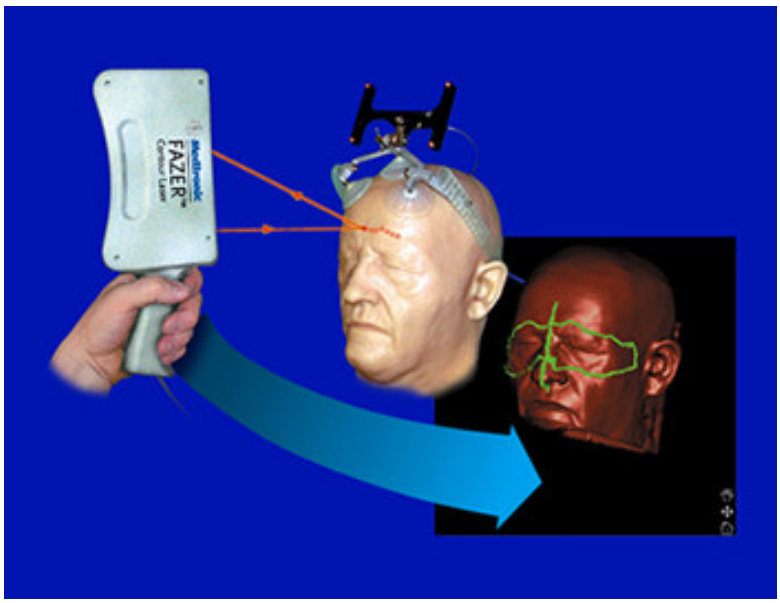

Figure 1.5. Laser Displacement Sensor Used in a Medical Application [1]

1.2. Literature Survey

Arkin et al. [4] worked on a mobile robot which can move through a building at relatively high speeds while generating a 3D model of the environment and transmitting the data in short bursts, so that it cannot easily be detected. Lifetime of the robot is limited since it may be destroyed by an enemy when detected. This model is used off-line to familiarize people with the building before entering it, for

example in a hostage situation. During the project, a SICK LMS 200 laser scanner is used which scans 180° in 0.5° steps from one side over the top of the other side. For each single measurement it returns a 13 bit integer value representing the distance to the next object, resulting in 361 such values for each complete 180° scan. The 3D model is displayed using OpenGL.

Huber et al. [5] presented techniques for building models of complex environments from range data gathered at multiple view-points. The challenges in this problem are the matching of unregistered views without prior knowledge of pose, the use of very large data sets and the manipulation of data sets of different resolutions and from different sensors. This approach is unique in that no prior knowledge of the relative viewpoints is needed in order to register the data. They showed results in building maps of interior environment from rangefinder data, building large terrain maps from ground-based and from aerial data, and from an operational of mapping from stereo data for hazardous environment characterization.

Sequeria et al. [6] described a technique for constructing a geometric model of an unknown environment based on data acquired by a laser rangefinder on board of a mobile robot. The geometric model would be most useful both for navigation and verification purposes. They represented all the steps needed for the description of the environment, including the range image acquisition and processing, 3D surface reconstruction and the problem of merging multiple images in order to obtain a complete model. Direct applications of the resulting 3D models are 3D scene analysis and interpretation, free space modeling for robot navigation and design verification measurements.

Park and Lee [7] proposed an algorithm to produce 3-D CAD model from a set of range data, based on Non-Uniform Rational B-Splines (NURBS) surface fitting technique. They aim to construct continuos geometric models, assuming that the topology of surface is unknown. In the approach, divide-and-conquer strategy is adopted, in which the whole range data is partitioned into surface patches. Each patch is sequentially processed to form the quadrilateral face model, which is used

to construct NURBS patch network. Experiments are carried out to evaluate the performance of the proposed algorithm and it is shown that continuos 3-D model is successfully generated automatically with tolerable computational complexity.

Klein and Sequeira [8] presented a new algorithm of planning the range image acquisition for the reconstruction of large, complex indoor scenes. Using a surface representation of seen and unseen parts of the environment, they proposed an objective function based on the analysis of occlusions. By optimizing this objective function, the parameters of the next view are computed efficiently for a large search space with eight degrees of freedom (3D position, viewing direction, field of view and resolution)

Goshtasby [9] concentrated on constructing 3D free-form models from multiview range images. He reviewed the literature related to three steps of the 3D model constructing; image integration, recovery of missing data, and model representation and editing. Then he presented new results in data recovery and model editing. Also examples demonstrating different steps of the model construction process are given using range images from a variety applications.

Liu et al. [10] described an algorithm for generating compact 3D models of indoor environments with mobile robots. Their algorithm employs the expectation maximization algorithm to fit a low complexity planar model to 3D data collected by rangefinders and a panoramic camera. The complexity of the model is determined during model fitting, by incrementally adding and removing surfaces. In a final post-processing step, measurements are converted into polygons and projected onto the surface model where possible. Empirical results obtained with a mobile robot illustrates that high-resolution models can be acquired in reasonable time.

1.3. Objective of the Study

In the previous study, Fidan [15] designed a laser rangefinder to scan an indoor environment and to construct the 3D image map of the scanned medium. He compiled two different softwares; data sampling software which was run under DOS and 2D surface reconstruction program under Windows.

.

Objective of this study is to set up a new system which can run under Windows and develop data acquisition and processing interfaces to collect and process data within just one application.

The study aims that the interfaces in the thesis are capable of creating 3-D image map and obtaining the view of scanned environment in a short time with high accuracy and time efficiency.

It is desired to acquire data in a helical way which is different from the other data acquisition systems. To do this, appropriate parameters for the data acquisition interface should be determined. In the data processing steps, it is aimed to use basic triangulation algorithms and threshold conditions to calculate resolutions, detect noisy points and segment objects from the environment for line fitting.

1.4. Outline of the Thesis

In this study, data acquisition and processing using a 3D laser rangefinder is introduced. The following chapters are organized to investigate each process separately.

In Chapter 2, laser measurement methods for distance measurement, experimental setup and main parts of the architecture and the data acquisition program compiled in Microsoft Visual C++ 6.0 are explained.

In Chapter 3, basic steps of the surface reconstruction process are explained and detailed information about surface fitting methods are given. Then, firstly preprocessing program, afterwards main processing program is handled with detailed explanation.

In Chapter 4, some case studies with objects positioned at different angles and places are given. Also a maximum range test is applied to a cardboard.

In Chapter 5, summary and conclusions of the thesis are given and recommendations for future work are presented.

In Appendix A, the laser sensor used in the thesis (AccuRange4000 Laser Sensor) is introduced. General information about the sensor, signal and power interface and basic commands of the sensor are explained.

In Appendix B, the interface board (AccuRange High Speed Interface) that takes samples from the sensor is introduced. General description about the card, I/O connectors and port interface are explained.

In Appendix C, the CTI-HSIF library, which is an interface layer between the sensor hardware and the application program is introduced. Communication scheme and usage of the library are explained.

CHAPTER 2

DATA ACQUISITION

2.1. Overview

This chapter consists of three main parts; laser measurement methods, experimental setup and data acquisition program. In the first part, three main laser measurement methods for distance measurement are dealt with and the principle of the laser sensor used in the thesis is focused on. In the second part, the experimental setup of the thesis is given and main parts of the system architecture are figured out. In the last part, the data acquisition program compiled in Microsoft Visual C++ 6.0 is explained. The parameters that have to be set by the user are handled one by one and ways of determining these parameters are investigated by some experimental tests.

2.2. Laser Measurement Methods

Optical distance measurement sensors are used in a wide variety of industrial, commercial, and research applications. Most sensors use a visible or infrared laser beams to project a spot of light onto a target, the surface to which the distance is to be measured. The distance from the spot back to the light-detecting portion of the sensor is then measured in one of several ways. Basic laser measurement methods for distance measurement are;

1. Triangulation systems
2. Pulse time-of-flight systems
3. Phase time-of-flight systems

All are used in practice for distance measurement depending on the particular applications. For distances of a few inches with high accuracy requirements, "triangulation" sensors measure the location of the spot within the field of view of the detecting element. Time of flight sensors derive range from the time it takes light to travel from the sensor to the target and return (Figure 2.1). For very long range distance measurements (up to many miles) "time-of-flight" laser rangefinders are used.

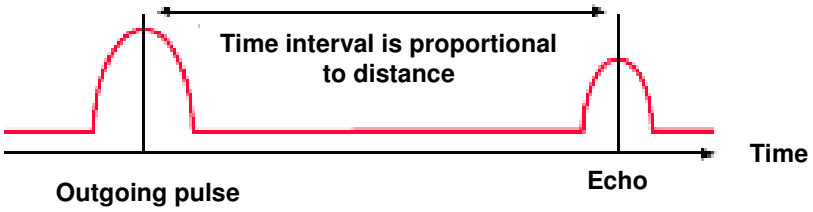

Figure 2.1. Time-of-Flight Measuring System

2.2.1. Triangulation Sensor Systems

Laser triangulation sensors are used in order to measure the distance to targets accurately through the use of laser triangulation sensors. They are so named because the sensor enclosure, the emitted laser and the reflected laser light form a triangle. (Figure 2.2)

The laser beam is projected from the instrument and is reflected from a target surface to a collection lens. This lens is typically located adjacent to the laser emitter. The lens focuses an image of the spot on a linear array camera. The camera views the measurement range from an angle that varies from 45 to 65 degrees at the center of the measurement range depending on the sensor model. The position of

the spot image on the pixels of the camera is then processed to determine the distance to the target. The camera integrates the light falling on it, so longer exposure times allow greater sensitivity to weak reflections. The beam is viewed from one side so that the apparent location of the spot changes with the distance to the target [1].

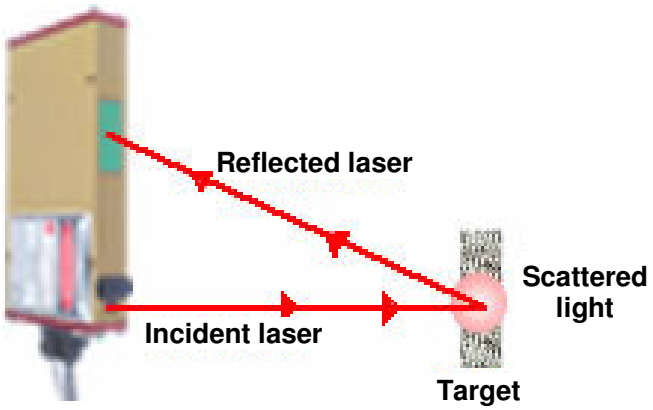

Figure 2.2. Laser Triangulation Sensor System [1]

Triangulation ranging utilize trigonometry to calculate the side of a triangle that correspond to the distance to a spot of interest. If the length of one side in the triangle is known, for instance the distance between two cameras, the distance to the target can be calculated from two measured angles. The basic principle is displayed in Figure 2.3 [12], where the distance B can be calculated from

$$B = A\frac{\sin\theta}{\sin(\theta+\phi)} \tag{2.1}$$

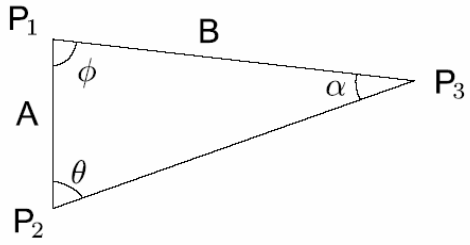

Figure 2.3. Triangulation System Range Calculation [12]

Triangulation devices are ideal for measuring distances of a few centimeters with high accuracy. These devices may be built on any scale, but the accuracy falls off rapidly with increasing range. The depth of field (minimum to maximum measurable distance) is typically limited, as triangulation sensors can not measure relative to their baseline, the distance between the emitter and the detector.

As the point of light falling on the object moves closer to or farther from the reference point, the spot position on the detector changes. Triangulation sensors are either diffuse or specular. The need for two types of sensors arises from differing reflectance characteristics of materials being examined. Smooth surfaces, such as mirrors are specular; others, such as anodized aluminum, are diffuse (Figure 2.4). Smooth or shiny surfaces typically require a specular sensor (the laser illumination hits the target such that the primary reflected light is reflected into the receive optics), while surfaces that scatter light are easier to measure with a diffuse sensor. So the selection of sensor type is dictated by the surface of the object being examined. A measure of the received signal strength (more = better), or the amount of time required to achieve a desired signal strength (less = better), is an indication that the right type of sensor has been chosen. Many surfaces, however, display both specular and diffuse characteristics and thus complicate the selection process. In those cases the user must test the surfaces using both specular and diffuse sensor types. [11]

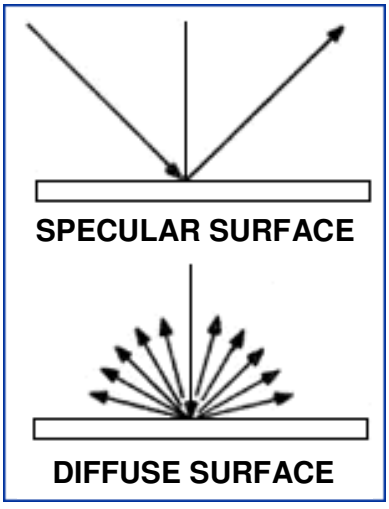

Figure 2.4. Specular and Diffuse Surfaces [11]

Triangulation ranging systems can be either passive or active. A passive system uses only the background light while an active system emits a beam to illuminate a certain spot. In the active system the beam emitter is one corner of the triangle and the camera corresponds to the second corner. For the passive system, the type and amount of background light has a large impact on the performance and in some circumstances artificial light has to be provided. In a passive two camera system, it can also be problematic to match points viewed in one camera to the same points viewed in the other camera. For both passive and active systems, it becomes increasingly difficult to determine longer distances with good accuracy. For a given angle measurement error, the error in the distance measurement increases with range.[12]

2.2.2. Pulse Time-of-Flight Sensor Systems

In pulse time-of-flight systems, the distance between the sensor and an object is calculated by measuring the time interval between an emitted laser pulse and reception of the reflected pulse. The amplitude of this signal is used to determine the reflectivity of the object surface.

This is the principle used in Radar, DME (Distance Measuring Device) for aircraft, LORAN (Long Range Navigation), Satellite Altimetry, Airborne Radar Altimetry , Lunar Laser Ranging etc. (Vaughan, 1998). Some of the newer EDM (Electrical Discharge Machining) instruments used by surveyor are also using pulse timing and accuracy of ±5μm are possible. Most of the military rangefinders also use pulse timing. The GPS (Global Positioning System) uses pulse timing for coarse distance measurement. VLBI (Very Long Base Interferometer) is also a pulse timing technique where signals from pulsars are timed from two or more radio telescopes and the difference in times of arrival are converted to intercontinental distances with a precision of a few centimeters.[13]

The overall pulse time-of-flight sensor performance is determined by a number of factors, such as the peak output power of the laser pulse, variations in the shape of the light pulse, the time resolution and dynamic range of the receiver electronics. Measuring short distances with high resolution is hence a challenging task. The resolution can be improved by sending out multiple pulses and averaging the delays. [12]

2.2.3. Phase Time-of-Flight Sensor Systems

In phase time-of-flight systems, the outgoing light intensity is sinusoidally modulated with a constant frequency and the phase difference between the local wave and the received wave is measured. The phase difference can be obtained by measuring the time differences between the zero crossings of the reference wave and the reflected wave. In both cases, the maximum distance that can be measured, while avoiding ambiguities, is determined by the selected modulation frequency. Two extensions of this technique, to improve the accuracy and avoid ambiguities when determining the distance, involve the use of more than one modulation frequency and/or adding a 90 degree phase shift to the local reference waveform [12].

There are many applications of this technique. A wide range of carrier frequencies are used ranging from visible through infrared to microwave and right down to VLF (Very Low Frequency). Precise positioning using GPS can be achieved by phase comparison of the carrier wave signals of the various satellites.[13]

Phase measurement is limited in accuracy by the frequency of modulation and the ability to resolve the phase difference between the signals. Some modulated beam rangefinders work on a range-to-frequency conversion principle, which offers several advantages over phase measurement. In these cases, laser light reflected from a target is collected by a lens and focused onto a photodiode inside the

instrument. The resulting signal is amplified up to a limited level and inverted, and used directly to modulate a laser diode. The light from the laser is collimated and emitted from the center of the front face of the sensor. This configuration forms an oscillator, with the laser switching itself on and off using its own signal. The time that the light takes to travel to the target and return plus the time needed to amplify the signal determines the period of oscillation, or the rate at which the laser is switched on and off. This signal is then divided and timed by an internal clock to obtain a range measurement. The measurement is somewhat nonlinear and dependent on signal strength and temperature, so a calibration process can be performed in the sensor to remove these effects. [1]

2.2.4. Laser Sensor Used in the Thesis

The laser sensor used in the thesis is “Acuity AR4000 laser rangefinder”. It employs a modified time-of-flight measurement principle that leads to very fast and accurate measuring speeds. Further information about Acuity AR4000 laser rangefinder is given in Appendix A.

The AR4000 differs from other long-distance rangefinders in that the laser emitter and return signal collection lens are concentric. The illustration (Figure 2.5) reveals the major functionality of the rangefinder. A collimated beam of laser light is emitted from a diode in the center of the collection lens. Light hits a target and is diffusely reflected, collected by the lens and focussed on an avalance photodiode[1].

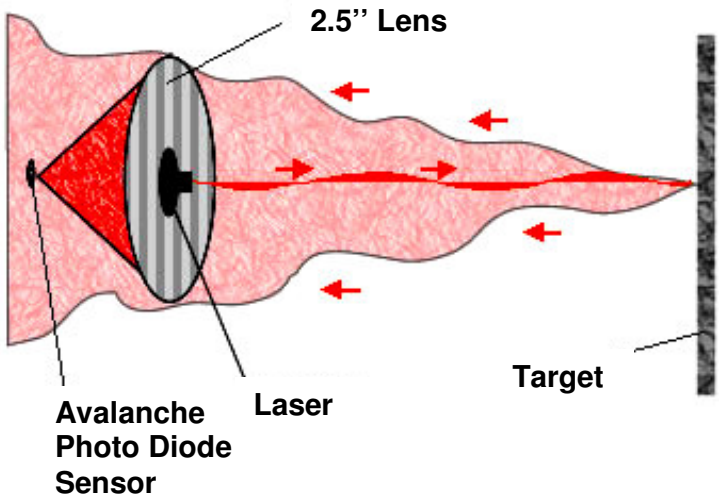

Figure 2.5. Working Principle of the AR4000 Rangefinder[1]

Because the emission and collection is concentric, the AR4000 rangefinder can be used to measure down narrow openings and tubes.

By this method, improved measurement accuracy and increased range can be achieved when cooperative targets (e.g. retroreflective tapes) are attached to the objects of interest to increase the power density of the reflected signal. The returned energy is compared to a simultaneously generated reference that has been split off from the original signal, and the relative phase shift between the two is measured to ascertain the round-trip distance the wave has traveled. The phase shift expressed as a function of distance to the reflecting target surface is [14]:

$$\phi = \frac{4\pi d}{\lambda} \tag{2.2}$$

where:

ϕ : phase shift

d : distance to target

λ : modulation wavelength

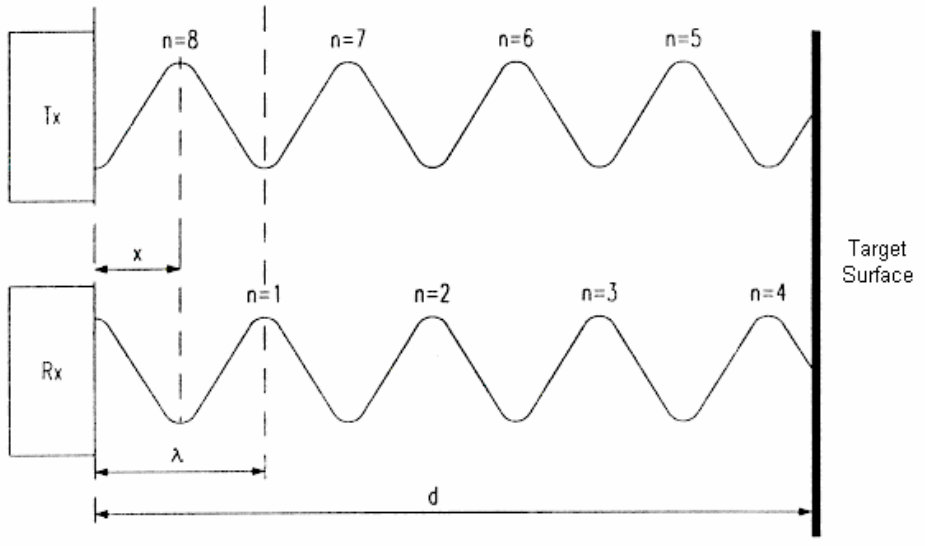

Figure 2.6. Relationship between Outgoing and Reflected Waveforms [14]

In Figure 2.6, x is the distance corresponding to the differential phase ϕ. The desired distance to target *d* as a function of the measured phase shift ϕ is:

$$d = \frac{\lambda\phi}{4\pi} = \frac{c\phi}{4\pi f} \tag{2.3}$$

where:

c: speed of light

f: modulation frequency

2.3. Experimental Setup

In the previous study, Fidan [15] designed a laser rangefinder to scan an indoor environment and to construct the 3D image map of the scanned medium. He compiled two different softwares; one for data sampling and one for data plotting.

The data sampling software is complied in Microsoft Visual C++ 5.0. The main task of the software is to read range and encoder data and to construct a 2D image of the cross section of the scanned environment resulted from a single rotation of the scanner head. The software reads raw range data, reflected signal amplitude, internal sensor temperature and encoder data from the DC Motor A. The output is the calibrated range data in mm, reflected signal amplitude and encoder position of DC Motor A. Therefore, the result gives the polar coordinates of a point on a plane. During the scanning process, the elevation angle is kept constant.

To enable visual inspection of the output, the data written to the output file is processed by a 2D plotter software which is compiled in Microsoft Visual Basic 4.0. The software reads the data (range, amplitude and encoder data) from the output file and converts the polar coordinates (range and encoder data) of each scanned point to the cartesian coordinates. Since the output file consists of recurring data after each revolution, the software takes only data from a single rotation of the scanner head. There is an algorithm in the 2D plotter software to filter out the erroneous and doubtful data. Since each set of sample contains an amplitude data, the range data with an amplitude less than predetermined value are plotted with a different color.

Basic parts of the designed laser rangefinder are explained in three main categories. These are:

1. Main architecture of the rangefinder
2. Motors, encoders, slip ring and sensor
3. Miscellaneous parts.

2.3.1. Main Architecture of the Rangefinder

The 3D model of the laser rangefinder is given in Figure 2.7. The whole assembly fits in a box of dimensions 220mm x 260mm x 600mm, weighting 30kg, approximately. The height of the mirror axis (elevation axis) from the ground is 575mm.

To rotate the scanning mechanism about the azimuth axis, a DC motor (DC Motor A) is used. A continuous steady power transmission between this motor and the scanning mechanism is achieved by a timing belt transmission system.

Figure 2.7. 3D Model of the Laser Rangefinder

Elevation scanning is carried out by a second DC motor (DC Motor E) installed on the scanning mechanism. This second DC motor is attached to the scanning mechanism and it rotates about the elevation axis.

The rotation of DC Motor E brings out the problem of power and signal transmission to/from the motor terminals and encoder attached to this motor. Since the motor was expected to rotate, it would not be possible to transmit signals and

power via simple wiring. A slip ring is used to transmit signals between two relatively rotating frames.

®KESTAMID (registered trade name of Polikim A.Ş. for it's Cast Polyamide products) is selected as the material for the shaft. The mechanical resistance values of Kestamid products are higher than other polyamides produced by injection or extrusion.

2.3.2. Motors, Encoders, Slip Ring And Sensor

First DC motor (DC Motor A) is Maxon Motor's motor-encoder unit 145863 consisting of DC Motor 118797 and encoder 110514. Type 118797 is a 70-watt DC motor. Second DC motor (DC Motor E) is Minimotor's DC-Micromotor Type 2233 R 012 S-74-213 assembled with encoder type HEDL 5540 I14 and reduction gearhead type 22/5. The gearbox type 22/5 is a zero backlash spur gearhead with a reduction ratio of 2050:1.

The selected slip ring is AC4831 of Litton Poly-Scientific which allows continuous 360° rotation of power and data signals up to 250 rpm. It has six circuits that can carry a current up to 5 A. The outer and through-bore diameters are 100 mm and 38 mm respectively.

For laser sensor, Acuity Research's AccuRange 4000-LIR laser sensor is selected. It has an operating range of 0 to 15 m and it also has reflected signal amplitude, ambient light and temperature outputs. Further information about laser sensor is given in Appendix A.

AccuRange High Speed Interface (HSIF) is used with the sensor to increase the sample rate capability of AR4000-LIR, using a PC-compatible computer. HSIF increases the sample rates up to 50000 samples/second and improves resolution

compared to the AR4000-LIR alone. Further information about HSIF is given in Appendix B.

2.3.3. Miscellaneous Parts

In the miscellaneous parts section, mirror, mirror holder, bearings, structural body, alignment system and belt pushing mechanism are explained.

The mirror used in the thesis is a two-face photocopier mirror. One side is an ordinary mirror and the other side is silvered glass with no glass thickness on the reflective surface. The silvered side is used to prevent the laser beam from refraction in the change of media and shifting from its angle of incidence.

The mirror is mounted in the mirror holder, which is fixed between the DC Motor E shaft and another 3mm shaft housed in a series 623 bearing. The alignment of the mirror on the mirror holder is achieved by two M3 screw on the mirror holder. All these parts are located on the scanner head (Figure 2.8).

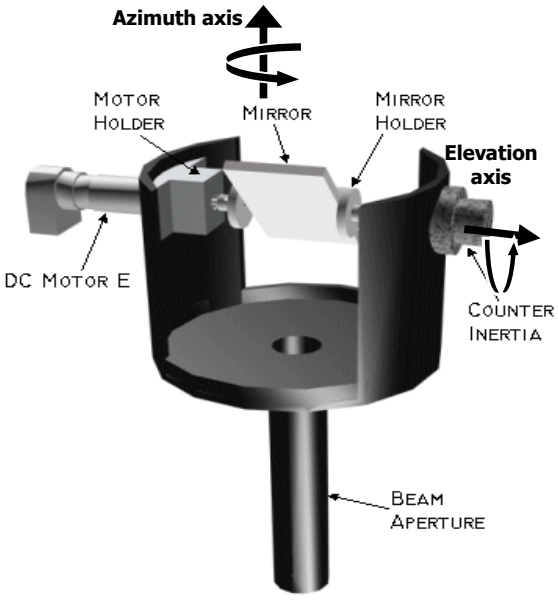

Figure 2.8. Scanner Head

The housing for the scanning mechanism and the slip ring consists of two pieces; housing and housing cover. Each piece contains a housing for the bearings so that the bearings are on two different parts. Figure 2.9 gives the cross sectional details of the housing assembly. The bearings are from two different series; 6007 and 6008. Driving belt gear which is not shown in the Figure 2.9 is placed under two bearings.

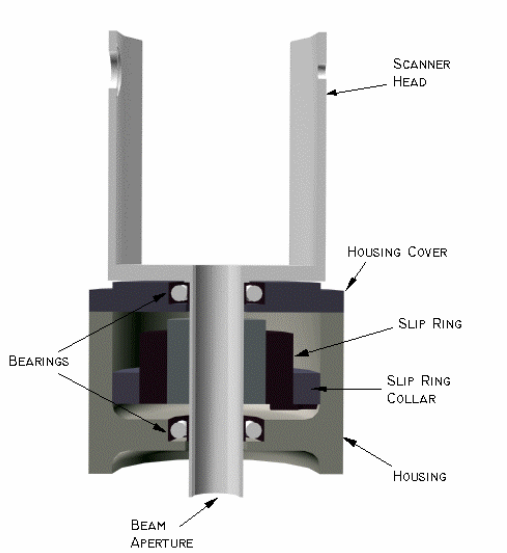

Figure 2.9. Cross Sectional View of the Housing Assembly

The modified structural body consists of four columns carrying the upper base on the lower base (Figure 2.10). The DC Motor A is housed in the polyamide motor holder by six M2.5 screws. The holder is mounted on a motor support. The scanning mechanism with its housing is assembled on Base 1. Base 2 is carried on three M12 screws. These screws are used to level the rangefinder to the horizontal plane.

Figure 2.10. Structural Body of the Laser Rangefinder

The laser sensor mounted on its holder is assembled on the aligner part 1 which can be translated in two directions. Aligner part 2 allows the translator screws to move in its slots. Aligner part 1 is fixed in position pressed between aligner part 2 and the tilter platform by screw set B when required translation is achieved. The rotation of the laser sensor is carried out by the tilter platform. The tilter platform is mounted on the base of the structural body by the screw set A with springs in between allowing the whole system to be tilted in two axes about approximately 5°.

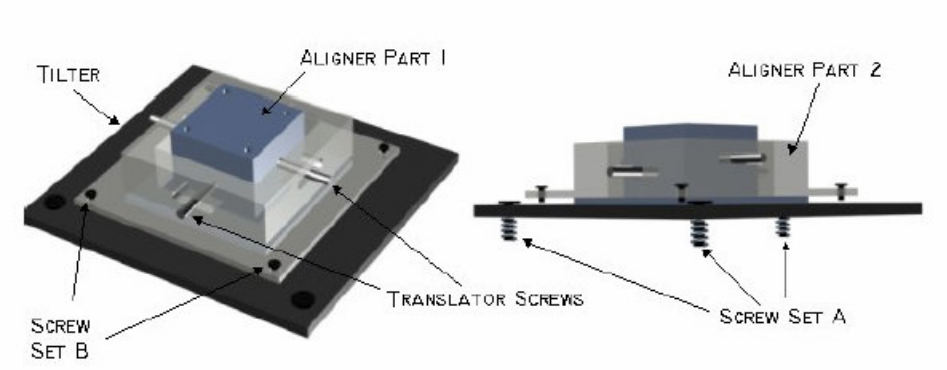

Figure 2.11. Alignment System

To keep the tension in the timing belt transmission system, a pushing mechanism is mounted on the upper base of the structural body. Figure 2.12 shows the belt pushing mechanism with the components labeled.

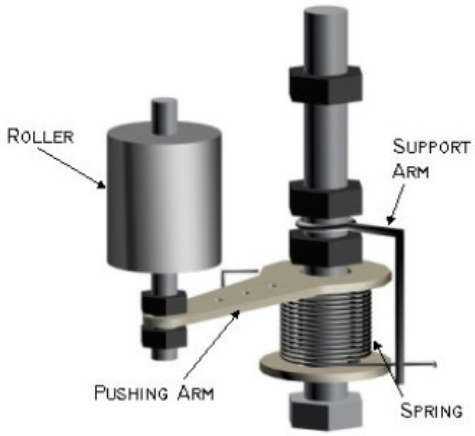

Figure 2.12. Belt Pushing Mechanism

2.4. Data Acquisition Program

The data acquisition program is compiled in Microsoft Visual C++ 6.0. The software is developed in order to read calibrated range data in mm, raw count (1-6152) (azimuth encoder), angle (0-360°) (elevation encoder), signal amplitude (0-255) and sensor internal temperature (°C).

Both of the motors used in the thesis have encoders which are connected to the encoder and general purpose input lines of the sensor's High Speed Interface (HSIF) board. Absolute positions of these encoders can be obtained either in angle or raw count format. Azimuth encoder and elevation encoder counts 6152 and 4220000 data respectively in each turn of the motor. The raw count data from 1 to 6152 for azimuth encoder (RawE1) and from 1 to 4220000 for elevation encoder (RawE2) corresponds to 0 to 360° angular data.

If encoder count/rev values are set wrong, obtained data for each turn of the scanner or mirror may slide in the amount of the error. In the previous study [15], azimuth encoder count/rev is set to a wrong value of 6141. In Figure 2.13 azimuth encoder counts 6141 pulses whereas in Figure 2.14 it counts 6152 pulses. A long and straight object is put in the environment to determine the amount of bending in the view.

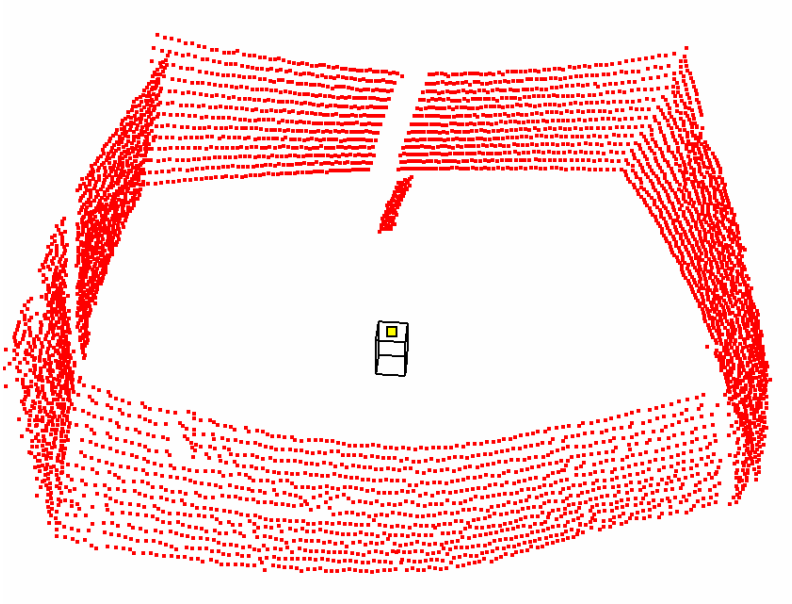

Figure 2.13. Sampled Data with an Azimuth Encoder count/rev of 6141

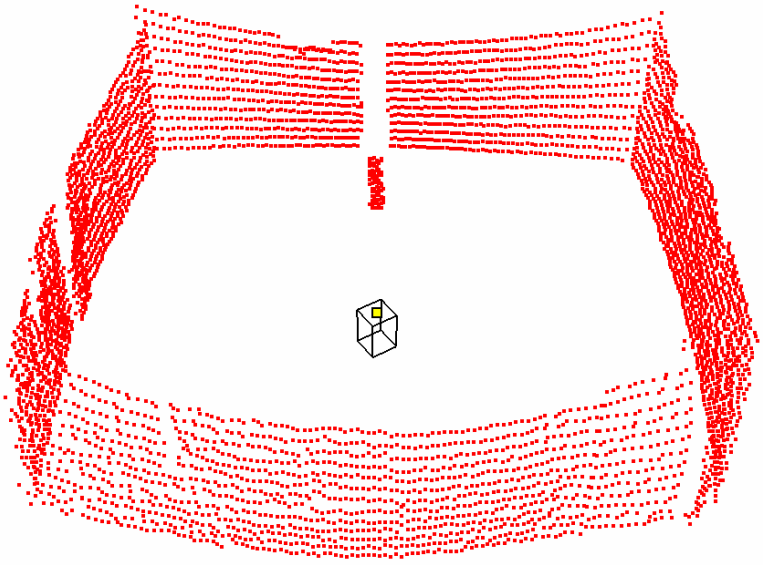

Figure 2.14. Sampled Data with an Azimuth Encoder count/rev of 6152

In order to guide the user, there is a “User Manual” button. The user can get information about the parameters that have to be set before the program is run. For some parameters, default values can be used. For the rest, there are tables and explanations.

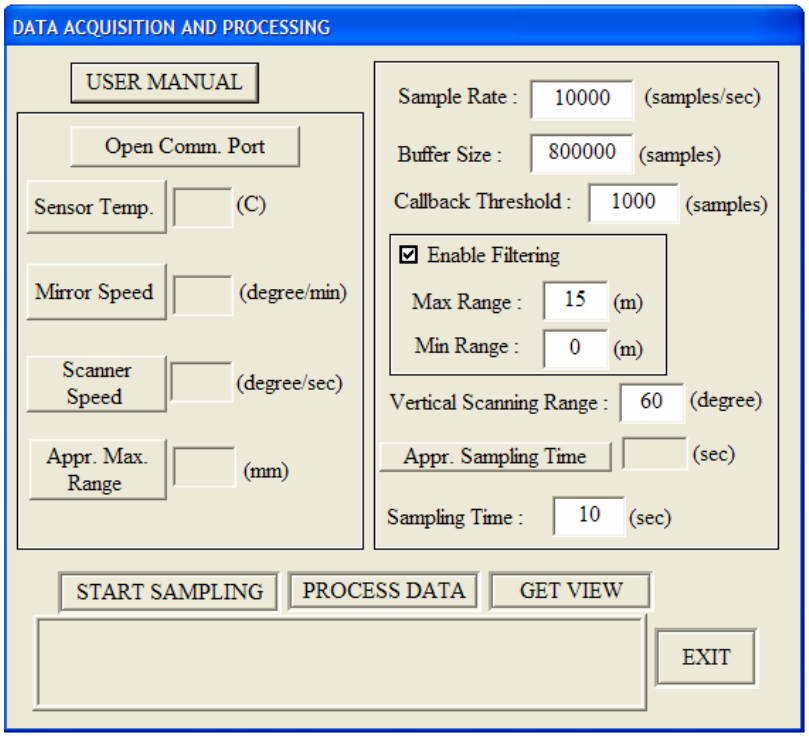

Figure 2.15. Screenshot of the Data Acquisition Program

2.4.1. Opening Serial Communication Port

Data sampling program starts with the opening of serial communication port. There are some parameters which are preset by the programmer. These parameters are High Speed Interface (HSIF) board’s I/O port address and interrupt number, calibration file, serial port and initial baud rate.

HSIF board’s I/O port address and the interrupt number have to match with the actual HSIF settings, as configured by the board’s hardware jumpers. Otherwise communications with the HSIF would fail.

HSIF's I/O address is set to 200Hex (HSIF's factory default) and interrupt number is set to 7. Further information about board parameters are given in Appendix B.

Each HSIF board has a different calibration data file which must match the particular laser sensor used, otherwise the calibrated sample data returned by the library will be inaccurate.

The serial port used to communicate with the sensor is "COM1". The selected serial port baud rate must match the actual baud rate of the sensor, otherwise communications with the sensor will fail. Serial port baud rate is set to the factory default of 9600 baud.

2.4.2. Getting Sensor Internal Temperature

When laser is first energized, it starts at whatever the room temperature is (15-25°C) and it slowly warms up. It takes 10-15 minutes for the laser to reach its target temperature of 33-34 °C.

The sensor temperature should be checked in small periods and it should not be started to get data before its temperature reaches to 33-34 °C.

2.4.3. Setting Mirror Speed

Mirror speed is used to determine vertical resolution of the scanned view according to the desired sample rate. Slower mirror speeds give better vertical resolution but increase total sampling time. On the other hand, faster mirror speeds decrease the total sampling time but gives worse vertical resolution compared to slower speeds. So by changing the mirror speed, an appropriate value should be adjusted by the user. For the desired optimum values of vertical resolution and sampling time, a mirror speed of 90-100 degree/min is selected in the study.

When "Mirror Speed" button is pressed, it starts getting Encoder 2 angle data and calculates mirror speed in degree/min, which helps the user to select the desired mirror speed.

2.4.4. Setting Scanner Speed

Scanner head (Figure 2.8) is responsible for horizontal data acquisition. Setting scanner speed plays a very important role in the determination of the sample rate because they are directly related with each other. There is an optimum sample rate for every scanner speed. For lower scanner speeds, sample rate should be kept low. However, for higher speeds, sample rate should be kept high in order to obtain certain scanning resolution.

There may be two different extreme cases; in the first case, the sample rate corresponding to the scanner speed can be below its optimum value. In such a case, the number of repetitions for the same azimuth encoder raw count (RawE1) readings will be very low(Table 2.1). But for very low sample rates, there might be discontinuities (jumps) between the consecutive RawE1 values (Table 2.2 and Table 2.3).

Table 2.1. Data Acquired at Sample Rate = 10000 samples/sec

Range (mm)	RawE1 (1-6152)	E2 angle (0-360°)	Amplitude (0-255)
853	1	0.0001	31
842	1	0.0003	34
834	2	0.0005	32
843	3	0.0006	32
847	4	0.0008	32
831	4	0.0011	33
837	5	0.0012	32
832	6	0.0014	30
845	7	0.0016	34
845	7	0.0017	34

Table 2.2. Data Acquired at Sample Rate = 1000 samples/sec

Range (mm)	RawE1 (1-6152)	E2 angle (0-360°)	Amplitude (0-255)
268	8	0.0017	98
270	16	0.0035	99
271	25	0.0052	99
271	33	0.0071	98
270	41	0.0087	99
269	50	0.0105	98
268	58	0.0122	99
271	67	0.0141	99
269	75	0.0157	98
268	83	0.0175	99

Table 2.3. Data Acquired at Sample Rate = 5000 samples/sec

Range (mm)	RawE1 (1-6152)	E2 angle (0-360°)	Amplitude (0-255)
278	2	0.0004	102
277	3	0.0007	103
277	5	0.0011	102
278	7	0.0014	102
274	8	0.0018	102
276	10	0.0022	102
276	12	0.0025	102
278	13	0.0029	102
278	15	0.0032	102
278	17	0.0036	102

In the second case, the sample rate corresponding to the scanner speed is above its optimum value. Here, repetitions for the same RawE1 values are high (2, 3 times or higher) so jumps cannot occur (Table 2.4). However, very high sample rates (Table 2.5) may bring the buffer overflowing problem depending on the selected buffer size. More information about this problem is given in the “Setting Buffer Size” section.

Table 2.4. Data Acquired at Sample Rate = 20000 samples/sec

Range (mm)	RawE1 (1-6152)	E2 angle (0-360°)	Amplitude (0-255)
279	1	0.00025	23
275	1	0.00034	24
262	1	0.00051	24
268	2	0.00059	24
255	2	0.00068	24
266	3	0.00076	23
268	3	0.00085	24
266	3	0.00093	23
268	4	0.00102	24
273	4	0.00111	23
268	4	0.00119	24
262	5	0.00127	24
266	5	0.00136	23
275	6	0.00145	24
275	6	0.00153	24
262	6	0.00162	24

Table 2.5. Data Acquired at Sample Rate = 30000 samples/sec

Range (mm)	RawE1 (1-6152)	E2 angle (0-360°)	Amplitude (0-255)
286	1	0.00034	106
286	1	0.00042	106
286	1	0.00051	106
286	1	0.00059	106
286	2	0.00059	106
286	2	0.00068	106
299	2	0.00076	106
286	3	0.00076	106
286	3	0.00085	106
286	3	0.00093	106
286	3	0.00093	106
286	4	0.00102	106
286	4	0.00111	106
286	4	0.00119	106
287	4	0.00121	106

After examining the Tables [2.1 to 2.5] given for various sample rates, it is obvious that data sampling rate should be selected in such a way that there shouldn't be any jumps within the consecutive RawE1 values. In Table 2.6, some scanner speeds and corresponding sample rates are shown. While acquiring data for Table 2.6, other parameters are preset in the program. These are buffer size (800000 samples), callback threshold (1000 samples) and sampling time (10 sec). This table should be used in the selection of the sample rate for changing scanner speeds. For example at a scanner speed of 450-470 degree/sec, the sample rate should be at least 10000 samples/sec.

Table 2.6. Relationship between Scanner Speed and Sample Rate

Scanner Speed (degree/sec)	Sample rate (samples/sec)	Number of jumps between RawE1	Number of turns of scanner head
350-370	6000	2642	9
	7000	30	9
	8000	0	10
	9000	0	9
400-420	7000	3413	11
	8000	215	11
	9000	0	11
	10000	0	11
450-470	8000	1055	12
	9000	23	13
	10000	0	13
	11000	0	12
500-520	9000	1203	14
	10000	14	14
	11000	0	13
	12000	0	14
550-570	10000	703	15
	11000	108	17
	12000	0	16
	13000	0	16
600-620	11000	508	17
	12000	3	17
	13000	0	17
	14000	0	17

When "Scanner Speed" button is pressed, it starts getting RawE1 data and calculates scanner speed in degree/sec.

2.4.5. Approximate Maximum Range

When "Approximate Max Range" button is pressed, it starts getting range data and finds out the highest range value. Then, amplitude of this data and the range values of the neighbour data are checked whether the highest range is a noisy data or not. If not, then it is given as the approximate maximum range. This value can give idea about the environment and can also be used in the filtering part of the program to set maximum range value.

2.4.6. Setting Sample Rate

After the scanner speed is set, appropriate sample rate for that speed is selected from Table 2.6. For lower scanner speeds, sample rate should be kept low. However, for higher speeds, sample rate should be kept high. Main criteria in the selection of the sample rate is that there shouldn't be any jumps within the consecutive RawE1 values.

The sensor's sampling interval may be set in discrete intervals of one microsecond. Due to the discrete sampling intervals, the actual sample rate may differ from the selected parameter value. For example, if sample rate is set to 101 samples/sec, it will result in a sample interval of 9900 microseconds, and the actual sample rate is obtained as approximately 101.01 samples/second.

The valid range for sample rate is approximately from 32 samples/sec to 50000 samples/sec if HSIF is used.

2.4.7. Setting Buffer Size

Buffer is used to hold data coming from the HSIF before that data is read by the application program. When the application program requests samples, they are retrieved from the buffer and returned to the application.

Small buffer size may result in a buffer overflowing problem. At higher sample rates, data cannot be written to disc as fast as the library can read samples from the sensor. Therefore, library's internal buffer may overflow and some samples may be lost. In order not to face with that problem, buffer size should not be changed unless it is necessary. If it has to be, then Table 2.7 can give idea to determine the correct buffer size for various sample rates. While acquiring data for Table 2.7, other parameters are preset in the program. These are scanner speed (460 degree/sec), callback threshold (1000 samples) and sampling time (15 sec). The buffer size should be at least the product of sampling time and sample rate parameters to be on the safe side. For example, for a sampling time of 15 sec and a sample rate of 12000 samples/sec, the buffer size should be minimum 180000. Also the buffer size cannot be smaller than library's default value (5000 samples).

Since data cannot be written to disc as fast as the library reads from the sensor, at the end of the sampling time, some data may still remain in the buffer. In Table 2.7 there are three subcolumns under data column. First column is the data written to disc at the end of the sampling time, second column is the data remaining in the buffer at the end of the sampling time and third column is the total data written to disc after the program finishes writing remained data in the buffer.

Discontinuities (jumps) between the consecutive RawE1 values may occur as a result of two reasons:

The first reason is to select low sample rates. For example, the first (sample rate of 8000) and second rows (sample rate of 9000) in Table 2.7 correspond to a buffer size of 300000, where jumps occurred in RawE1 data. Since the sample rate is not

high enough for the selected scanner speed, there becomes jumps in the RawE1 data.

Table 2.7. Effect of Buffer Size and Sample Rate on Data Sampling

Buffer Size	Sample Rate	DATA			Number of turns of scanner head	First jump in RawE1 values	Number of jumps between RawE1
		written to textfile	remaining in buffer	total			
300000	8000	115000	6853	121000	0	89	539
	9000	123000	14212	137000	0	4105	11
	10000	109000	43597	152000	18	-	0
	12000	114000	71316	185000	17	-	0
	15000	136000	97467	233000	18	-	0
	20000	101000	209184	310000	19	-	0
200000	8000	116000	5853	121000	0	692	11
	9000	111000	26213	137000	17	-	0
	10000	112000	40573	152000	18	-	0
	12000	110000	75338	185000	18	-	0
	15000	106000	127469	233000	17	-	0
	20000	89000	194000	283000	17	272957	10
100000	8000	115000	6853	121000	0	70	415
	9000	111000	25157	136000	0	13223	160
	10000	108000	44597	152000	17	-	0
	12000	107000	77250	184000	16	159008	123
	15000	102000	93000	195000	14	167973	27
	20000	96000	92000	188000	8	137978	50
50000	8000	113000	8853	121000	0	12	1878
	9000	123000	14213	137000	0	3930	27
	10000	118000	34572	152000	18	-	0
	12000	109000	42000	151000	11	108983	42
	15000	103000	43000	146000	7	83987	62
	20000	94000	45000	139000	4	67985	71
30000	8000	111000	10853	121000	0	2384	892
	9000	114000	21000	135000	16	128983	58
	10000	110000	24000	134000	12	91984	42
	12000	109000	21000	130000	6	62990	67
	15000	100000	26000	126000	4	48993	77
	20000	104000	16000	120000	2	40993	79

The second reason is to select low buffer size. For example, a buffer size of 30000 in Table 2.7, all the rows contain jumps because of the buffer overflowing problem. Data in the buffer cannot be written to disc fast enough and after a time buffer becomes fully loaded. Then coming data from the sensor is written on the remaining data and the continuity of the RawE1 data is corrupted and jumps in data occur. An example for buffer overflowing can be seen from Table 2.8 in between fifth and sixth rows.

Table 2.8. Buffer Overflowing

Range (mm)	RawE1 (1-6152)	E2 angle (0-360°)	Amplitude (0-255)
4622	1712	9.5463	54
4607	1712	9.5464	56
4615	1713	9.5465	55
4607	1713	9.5466	54
4629	1714	9.5468	54
1981	2677	9.7763	84
1981	2677	9.7763	84
1988	2678	9.7765	82
1980	2678	9.7766	82
1995	2679	9.7768	84

2.4.8. Setting Callback Threshold

Callback threshold is a part of callback function. When the number of samples in the library's internal buffer exceeds the callback threshold, the library will call the callback function and continuously take samples from the buffer and write these samples to disc until all the data in the library's buffer are over.

The data transferred from library to disc occurs in blocks of samples where each block has number of data equal to the callback threshold. Effect of callback threshold on the speed of writing data to disc is shown in Table 2.9.

While acquiring data for Table 2.9, other parameters are preset in the program. These are scanner speed (460 degree/sec), buffer size (800000 samples) and

sampling time (15 sec). Two different sample rates are selected for this test. For both of the sample rates, as callback threshold increases, data written to disc at the end of the sampling time decreases because writing to disc with higher threshold values takes more time than writing to disc with lower threshold values.

Table 2.9. Effect of Callback Threshold on the Speed of Writing Data to Disc

Sample Rate	Callback Threshold	DATA		
		written to textfile	remaining in buffer	total
10000	100	108600	43973	152500
	500	112500	40071	152500
	1000	108000	44572	152000
	3000	105000	48596	153000
	5000	105000	47572	150000
	10000	100000	52573	150000
	15000	90000	62572	150000
	20000	80000	72572	140000
20000	100	95400	215892	311200
	500	93500	217790	311000
	1000	97000	214286	311000
	3000	96000	215289	309000
	5000	90000	221291	310000
	10000	90000	221290	310000
	15000	75000	235003	300000
	20000	80000	231291	300000

Third part of the data column in the Table 2.9 illustrates the “total data” written to disc after the program finishes writing the remained data in the buffer. For all callback threshold values, total data obtained are nearly same for the same sample rates.

Same sample rates and sampling times are used in each row of the test, but total data for all callback threshold values are not same. For example, for a sample rate of 10000 and a callback threshold of 3000, 153000 data is written to disc. However, for the same sample rate and a callback threshold of 20000, 140000 data is written. For the latter case, nearly 13000 data are not written to disc because that test has a

callback threshold value of 20000 and blocks consisting of 20000 samples are taken from the library. 13000 data remains in the buffer because the callback threshold value could not exceeded. Therefore keeping callback threshold value so high is not a preferred case because data blocks smaller than the callback threshold cannot be read from the library.

For callback threshold, “1000” is used as the default value. If it is desired, it can be changed before running the data sampling software.

Callback threshold must be less than or equal to the library’s internal data buffer size. If the resultant buffer size is less than the current callback threshold, the threshold will be set to zero.

2.4.9. Setting Filtering Ranges

Data filtering functions enable the programmer to specify which samples acquired by the library will be returned to the application program. These functions should be used to discard samples that are not of interest. In the program, only maximum and minimum range readings that will be considered valid are set.

If maximum range is set, samples with more than the value specified are set to zero or discarded. If minimum range is set, samples with less than the value specified are set to zero or discarded. The approximate maximum range value obtained in the previous steps can give idea in the determination of maximum valid range parameter. A value slightly more than the approximate maximum range value can be set as maximum valid range.

2.4.10. Setting Sampling Time

After setting the mirror speed and entering the vertical scanning range to the program, it calculates the “approximate sampling time”. It should be noted that 1 degree of mirror turn gives 2 degrees of angle data in vertical. For instance, if mirror speed is set to 90 degree/min and environment to be scanned is 60 degrees in vertical, then the mirror should turn 30 degrees. Therefore approximate sampling time will be 20 sec. This calculated value can be used as the sampling time.

2.4.11. Starting Sampling

After all parameters are determined in the previous steps, sampling can be started in this step. These determined parameters should be set by the library commands in the correct order as follows:

- Set azimuth encoder and elevation encoder count/rev
- Set the data output format to metric units
- Open output data file
- Set sample rate
- Set buffer size
- Set callback threshold
- Set the callback function
- Set range offset
- Set maximum and minimum valid ranges
- Reset HSIF board
- Clear library’s internal buffer
- Set sampling time
- Start continuously reading samples from HSIF
- Stop continuously reading samples from HSIF
- Get number of the remained data in library after sampling
- Write the remained data in the library to file

CHAPTER 3

DATA PROCESSING

3.1. Overview

This chapter consists of two main parts; surface reconstruction methods and data processing programs. In the first part, basic steps of the surface reconstruction process are explained. Then, detailed information about surface fitting methods are given. In the second part, firstly preprocessing program is figured out. Then, main processing program is handled with detailed explanation. Calculating resolutions, detecting noisy points, segmentation, triangulation and line fitting processes are given.

3.2. Surface Reconstruction Methods

Surface reconstruction from 3D range data covers a variety of methods. Figure 3.1 shows these methods used in the process of converting an imaged scene into a surface model [16].

The whole process consists of two main steps; acquisition of physical data from the environment and processing these data with surface reconstruction methods.

There are many techniques used in constructing surface models. Each of these techniques can be applied in combination or in various orders. These techniques are as follows:

1. Registration and integration
2. Segmentation
3. Surface fitting
4. Mesh simplification

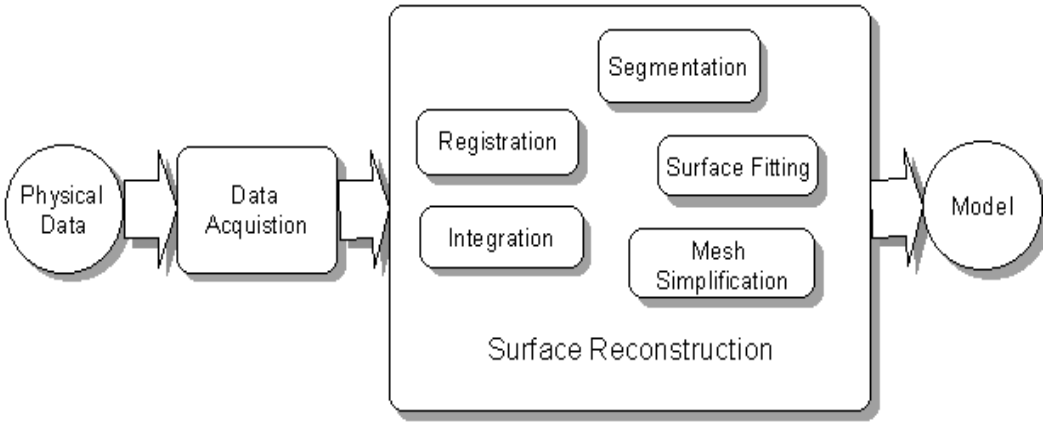

Figure 3.1. Process of Converting an Image Scene into a Surface Model [16]

3.2.1 Registration and Integration

A single range view can only contain points in the scene that are visible to the data acquisition device so it is necessary to merge multiple range views into a single point set to obtain a complete scene model. This involves *registration* and *integration*.

Registration is deriving and applying a transformation to bring a pair of range views into the same coordinate system. Registration creates redundant data which have to be integrated to a less dense point set. Integration is the process of either removing redundant data, that is points visible in both views, or averaging out common points.

A summary of work in the registration area is given by Besl and McKay [17], who also present their own iterative closest point (ICP) algorithm. Most registration techniques consist of two major components:

1. Solving the closest point problem
2. Determining the transformation to register the data.

In the registration process, the task is to find a transformation T, to apply to the *source* data P, to align it with the *destination* data Q, such that the relation $T\mathbf{p} = \mathbf{q}$ holds for any pair of points ($\mathbf{p}$, $\mathbf{q}$), where $\mathbf{p} \in P$ and $\mathbf{q} \in Q$. T is a rigid transformation which consist of 3D rotation and translation.

For real data, the relation $T\mathbf{p} = \mathbf{q}$ cannot hold. Then the problem becomes minimizing the expression $|T\mathbf{p} - \mathbf{q}|^2$. There is a need to establish a pairwise point correspondence between P and Q. This is done by taking a point on P and finding the closest point on Q. Besl and McKay [17] allow the registration to be done on various types of data representations including points, simple surfaces and implicit surfaces and present methods for obtaining the closest point on all of these.

3.2.2. Segmentation

The process of segmentation aims to group points in a range image into regions of homogeneous properties. Since these properties are often surface based, segmentation and surface fitting (described below) are closely interrelated. Surface fitting aims to extract geometric primitives and segmentation can group points based on being a part of a single surface.

Segmentation can be used as a part of the surface reconstruction process, since the segmented regions are often described in terms of surfaces. Also, if there are unwanted objects in a scene, segmentation provides a means to remove them.

Besl and Jain [18] give a summary of earlier segmentation efforts. Segmentation of a range data may be done on the basis that, many surfaces in a scene can be approximated by planar or quadratic surfaces and that range data points exhibit a

local spatial or surface coherence. The segmentation process needs to balance the aims of producing a minimum number of regions against having a maximum degree of coherence in each region.

Besl and Jain [18] describe their own iterative region growing segmentation method. Initially, each point in a range image is categorized according to its mean and Gaussian curvature. The curvatures are computed on a 3×3 neighborhood about the points. These curvatures are chosen because they are invariant to rotation and translation of the data and that points of the same curvature are likely to belong to the same surface. A seed point in each region is chosen and a planar or quadratic surface is fitted. Neighboring points are incrementally added until the goodness of fit of the surface is below a certain threshold. Then, either the process stops or a surface of higher order is used.

Premebida and Nunes [19] analyzed the segmentation process with a different kind of algorithm. Schematic representation of a hypothetical scan data is shown in Figure 3.2 where r defines measured range and Δα defines the sensor angular resolution.

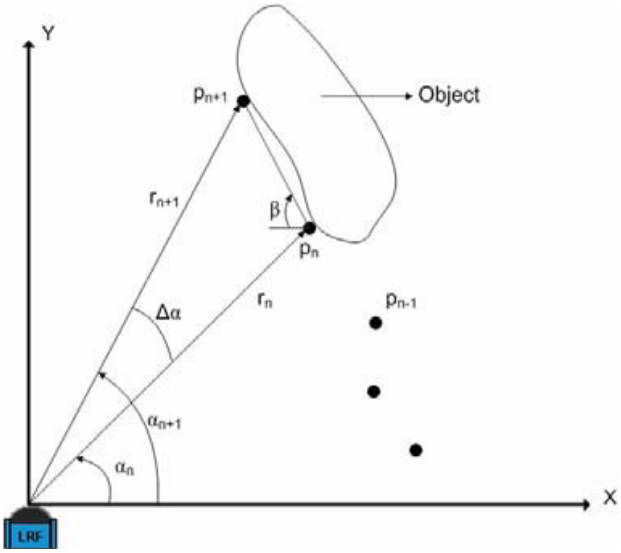

Figure 3.2. Schematic Representation of a Hypothetical Scan Data [19]

If $D(r_i , r_{i+1}) > D_{thd}$ then, segments are separated, else segments are not separated.

D_{thd} is the threshold condition and $D(r_i , r_{i+1})$ is the Euclidean distance between two consecutive scanned points.

$$D(r_i, r_{i+1}) = \sqrt{r_i^2 + r_{i+1}^2 - 2.r_i .r_{i+1}.\cos\Delta\alpha} \tag{3.1}$$

A well known approach, uses the following threshold condition [20]:

$$D_{thd} = C_0 + C_1.\min\{r_i, r_{i+1}\} \tag{3.2}$$

where $C_1 = \sqrt{2.(1-\cos\Delta\alpha)} = D(r_i, r_{i+1})/r_i$ and C_0 is a constant parameter used for noise reduction.

Based in the previous approach, Santos and friends [21] have included a new parameter (β) in the Eq. (3.2), aiming to reduce the dependence of the segmentation with respect to the distance between the rangefinder and the objects, resulting the threshold condition:

$$D_{thd} = C_0 + \frac{C_1.\min\{r_i, r_{i+1}\}}{\cot(\beta).[\cos(\Delta\alpha/2) - \sin(\Delta\alpha/2)]} \tag{3.3}$$

where C_0 and C_1 are the same as in Eq. (3.2)

A simple method for segmentation is presented by Lee [22], where the threshold condition is given by:

$$D_{thd} = \left| \frac{r_i - r_{i+1}}{r_i + r_{i+1}} \right| \tag{3.4}$$

Another method, called Adaptive Breakpoint Detector (ABD) proposed by Borges and Aldon [23] which specifies a new equation for the threshold condition as

$$D_{thd} = r_i \frac{\sin \Delta\alpha}{\sin(\lambda - \Delta\alpha)} + \sigma_r \tag{3.5}$$

where λ is an auxiliary parameter and σ_r is a residual variance to encompass the stochastic behavior of the sequence scanned points $\{P_n\}$ and the related noise associated to r_n.

Premebida and Nunes [19] present the results using real data scanned by a laser rangefinder sensor. In Figure 3.3, a typical segmentation using Dietmayer's method [20] is shown. Top view is segmentation result and bottom view is a snapshot of an indoor environment. In Figure 3.4, a typical segmentation using Lee's method [22] in an outdoor environment is shown. The numbers inside the figures represent the segments (S_i) identified by the algorithms. In Figure 3.4, segment 1 is related to a tree; segments 3, 5, 7 and 8 are related to walls and the others are related to a person's legs.

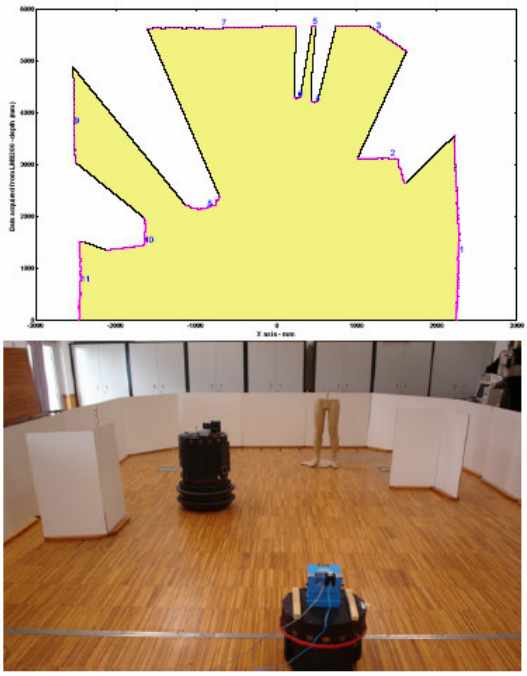

Figure 3.3. Typical Segmentation Using Dietmayer Method [20]

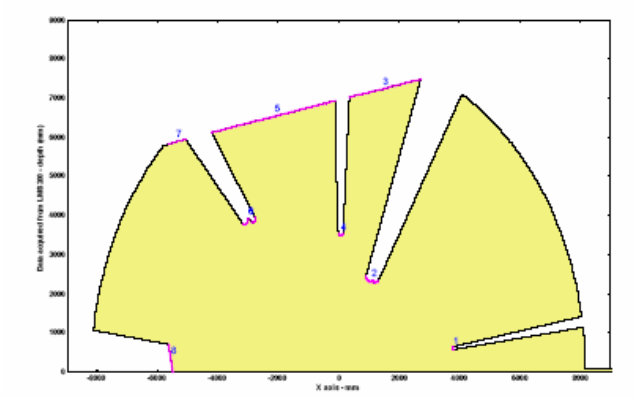

Figure 3.4. Typical Segmentation Using Lee Method [22]

3.2.3. Surface Fitting

There are several ways of fitting surfaces to 3D range data. Basic methods are as follows:

1. Functional fitting
2. Triangulation Methods
3. Deformable Models

3.2.3.1. Functional Fitting

Functional fitting is one of the basic of all surface fitting methods. In this method, surface representation is done by using surface functions. These functions may be either explicit functions (f(x,y) = z) or implicit functions (f(x,y,z) = c) [16].

In this method, obtained point data is not used directly; it is just reduced to a functional description and this function is matched to existing physical models. In order to do this operation, segmentation is necessary for each object and this brings the problem of time loss during mathematical operations. Also, this method cannot fit surfaces for objects with sharp edges correctly.

Surface fitting in explicit form often uses spline techniques, where the data is sparse. The explicit form is usually not as useful as the implicit form in fitting surfaces to range data.

Simple shapes such as planes, cylinders and spheres fall into the class of implicit functions, as well as quadratic and conic surfaces. Some of the basic quadratic surfaces are shown in Figure 3.5.

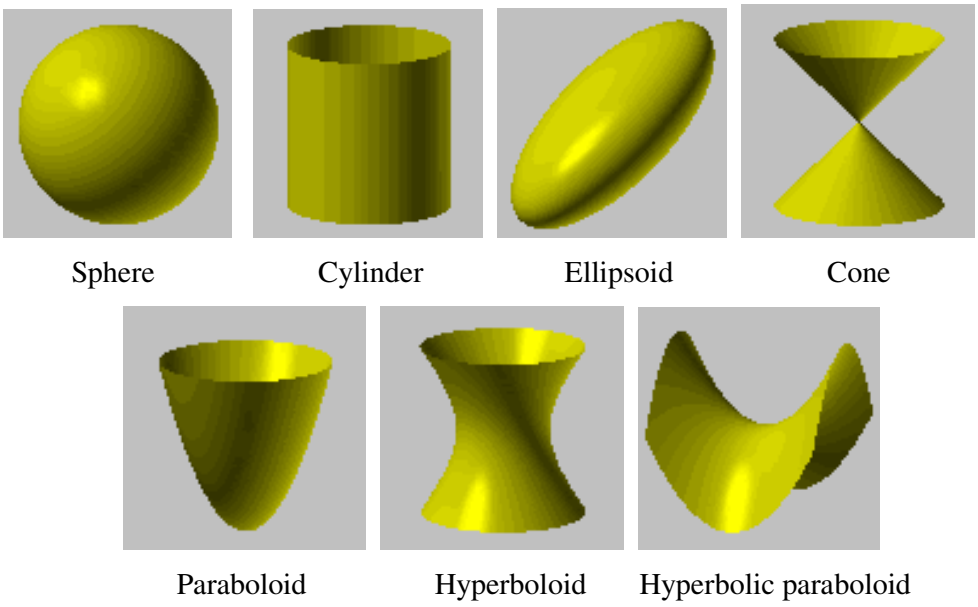

Figure 3.5. Basic Quadratic Surfaces

The equation of a quadratic surface (second degree with three variables) is;

$$f(x, y, z) = s_{11}x^2+s_{22}y^2+s_{33}z^2+2s_{12}xy+2s_{13}xz+2s_{23}yz+2s_{41}x+2s_{42}y+2s_{43}z+s_{44}+ e = 0 \quad (3.6)$$

In the above function [24], "e" is the error and used in determination of the coefficients. Given a set of N range data points, $\{P_i = (x_i, y_i, z_i)\}$, where $i \in 1,2,3,..,N$, the coefficients of the above equation can be obtained by using least

squares method. In this method, sum of the squared errors is minimized. To find the minimum value, partial derivatives according to the constants are taken and set to zero. The sum of the squared error function is:

$$\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(s_{11}x_i^2 + s_{22}y_i^2 + s_{33}z_i^2 + s_{12}x_i y_i + s_{13}x_i z_i + s_{23}y_i z_i + s_{41}x_i + s_{42}y_i + s_{43}z_i + s_{44} \right)^2 \tag{3.7}$$

To avoid a trivial solution ($s_{11}= s_{22}= s_{33}= s_{12}= s_{13}= s_{23}= s_{41}= s_{42}= s_{43}= s_{44}=0$), the above equation can be normalized. There are many different normalizations and the comparisons of these normalization in the literature [25]. Here, instead of normalization, to avoid a trivial solution and to decrease the number of undetermined coefficients, the above equation can be changed. It is assumed that the whole equation is divided into one of the coefficients, lets say s_{41}. Although the other coefficients also change by this division, for sake of simplicity, names of these coefficients are not changed and continue to the method with the same names.

$$\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(s_{11}x_i^2 + s_{22}y_i^2 + s_{33}z_i^2 + s_{12}x_i y_i + s_{13}x_i z_i + s_{23}y_i z_i + x_i + s_{42}y_i + s_{43}z_i + s_{44} \right)^2 \tag{3.8}$$

Partial derivative according to s_{11}:

$$\frac{\partial \sum e_i^2}{\partial s_{11}} = 2\sum x_i^2 \left(s_{11}x_i^2 + s_{22}y_i^2 + s_{33}z_i^2 + s_{12}x_i y_i + s_{13}x_i z_i + s_{23}y_i z_i + x_i + s_{42}y_i + s_{43}z_i + s_{44} \right) = 0 \tag{3.9}$$

$$s_{11}\sum x_i^4 + s_{22}\sum x_i^2 y_i^2 + s_{33}\sum x_i^2 z_i^2 + s_{12}\sum x_i^3 y_i + s_{13}\sum x_i^3 z_i + s_{23}\sum x_i^2 y_i z_i + s_{42}\sum x_i^2 y_i + s_{43}\sum x_i^2 z_i + s_{44}\sum x_i^2 = \sum x_i^3 \tag{3.10}$$

After partial derivatives of other parameters are taken as s_{11}, 9 equations with 9 unknown coefficients are obtained. These coefficients are solved by a matrix operation:

$$\begin{bmatrix} \sum x_i^4 & \sum x_i^2 y_i^2 & \sum x_i^2 z_i^2 & \sum x_i^3 y_i & \sum x_i^3 z_i & \sum x_i^2 y_i z_i & \sum x_i^2 y_i & \sum x_i^2 z_i & \sum x_i^2 \\ \sum x_i^2 y_i^2 & \sum y_i^4 & \sum y_i^2 z_i^2 & \sum x_i y_i^3 & \sum x_i y_i^2 z_i & \sum y_i^3 z_i & \sum y_i^3 & \sum y_i^2 z_i & \sum y_i^2 \\ \sum x_i^2 z_i^2 & \sum y_i^2 z_i^2 & \sum z_i^4 & \sum x_i y_i z_i^2 & \sum x_i z_i^3 & \sum y_i z_i^3 & \sum y_i z_i^2 & \sum z_i^3 & \sum z_i^2 \\ \sum x_i^3 y_i & \sum x_i y_i^3 & \sum x_i y_i z_i^2 & \sum x_i^2 y_i^2 & \sum x_i^2 y_i z_i & \sum x_i y_i^2 z_i & \sum x_i y_i^2 & \sum x_i y_i z_i & \sum x_i y_i \\ \sum x_i^3 z_i & \sum x_i y_i^2 z_i & \sum x_i z_i^3 & \sum x_i^2 y_i z_i & \sum x_i^2 z_i^2 & \sum x_i y_i z_i^2 & \sum x_i y_i z_i & \sum x_i z_i^2 & \sum x_i z_i \\ \sum x_i^2 y_i z_i & \sum y_i^3 z_i & \sum y_i z_i^3 & \sum x_i y_i^2 z_i & \sum x_i y_i z_i^2 & \sum y_i^2 z_i^2 & \sum y_i^2 z_i & \sum y_i z_i^2 & \sum y_i z_i \\ \sum x_i^2 y_i & \sum y_i^3 & \sum y_i z_i^2 & \sum x_i y_i^2 & \sum x_i y_i z_i & \sum y_i^2 z_i & \sum y_i^2 & \sum y_i z_i & \sum y_i \\ \sum x_i^2 z_i & \sum y_i^2 z_i & \sum z_i^3 & \sum x_i y_i z_i & \sum x_i z_i^2 & \sum y_i z_i^2 & \sum y_i z_i & \sum z_i^2 & \sum z_i \\ \sum x_i^2 & \sum y_i^2 & \sum z_i^2 & \sum x_i y_i & \sum x_i z_i & \sum y_i z_i & \sum y_i & \sum z_i & N \end{bmatrix} \begin{bmatrix} s_{11} \\ s_{22} \\ s_{33} \\ s_{12} \\ s_{13} \\ s_{23} \\ s_{42} \\ s_{43} \\ s_{44} \end{bmatrix} = \begin{bmatrix} \sum x_i^3 \\ \sum x_i y_i^2 \\ \sum x_i z_i^2 \\ \sum x_i^2 y_i \\ \sum x_i^2 z_i \\ \sum x_i y_i z_i \\ \sum x_i y_i \\ \sum x_i z_i \\ \sum x_i \end{bmatrix}$$

(3.11)

The above operation can be summarized as follows:

$$[A]\{S\}=\{C\} \tag{3.12}$$

In order to find {S} ,inverse of [A] must be taken and multiplied by {C}:

$$\{S\}=[A]^{-1}\{C\} \tag{3.13}$$

After getting 9 coefficients, the quadratic function is obtained and matched to existing models. The equations of the basic quadratic surfaces are shown in Table3.1.

Table 3.1 Quadratic Models and Their Functions

MODEL	FUNCTION
Sphere	$x^2 + y^2 + z^2 = r^2$
Cylinder	$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$
Ellipsoid	$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$
Cone	$\frac{x^2}{a^2} + \frac{y^2}{b^2} = \frac{z^2}{c^2}$
Paraboloid	$\frac{x^2}{a^2} + \frac{y^2}{b^2} = \frac{z}{c}$
Hyperboloid of one sheet	$\frac{x^2}{a^2} + \frac{y^2}{b^2} - \frac{z^2}{c^2} = 1$
Hyperbolic paraboloid	$\frac{x^2}{a^2} - \frac{y^2}{b^2} = \frac{z}{c}$

3.2.3.2. Triangulation Methods

In triangulation methods, triangular meshes are obtained directly from the data. These are purely interpolation methods and do not approximate the data. These methods are suitable where the noise in the data is small.

Triangulation methods can be divided into two main parts; direct triangulation and Delaunay triangulation.

3.2.3.2.1. Direct Triangulation

Surface reconstruction with direct triangulation method is presented by C. Oblonsek and N. Guid [26]. In this method, a new fast three stage procedure for

object reconstruction from 3D scattered points is presented. The procedure is divided into the following steps:

1. Triangulation of the scattered point set
2. Feature extraction
3. Improving the triangular mesh

The triangulation algorithm is based on an adapted algorithm for 2D Delaunay triangulation presented in [27] and later modified in [28]. In order to obtain the correct triangulation of the scattered point set $S = \{p_i\}_{i=1}^{n}$, the conditions given below must be satisfied:

- The maximal distance between each point $p_i \in S$ and the closest point $p_j, j \neq i$ has to be smaller than half the radius of the maximal curvature at point p_i.
- The point set must be isotropic, i.e., a couple of *k* closest points to p_i must be on both sides of each normal plane through each point $p_i \in S$.
- No point on the opposite side of the object with regard to point $p_i \in S$ must be in the *k*-neighborhood of point p_i .

The upper three conditions for the point set must be satisfied, otherwise triangulation procedure may fail to produce a proper triangulation of the object surface.

Figure 3.6 shows two point sets obtained by scanning a shoe; a good point set in Figure 3.6.b, a bad point set in Figure 3.6.a and a triangular mesh (Figure 3.6.c) obtained from the good point set [26]. The point set in Figure 3.6.a does not fulfill the conditions so the triangulation algorithm fails. If there are too many points on a cross section, then all *k* closest points are also on that cross section. In Figure 3.6.b

there are not too many points on a cross section so, that case is better. After all, the triangular mesh (Figure3.6.c) is produced using the good point set.

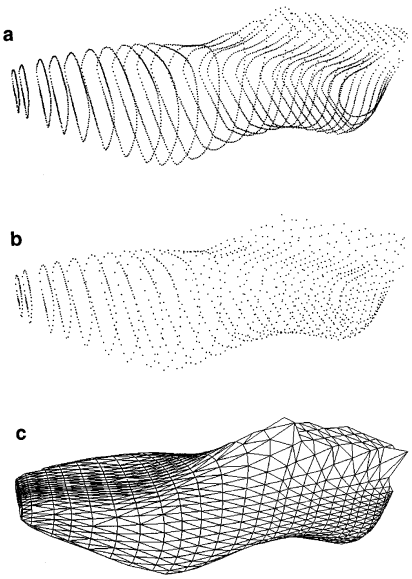

Figure 3.6. Bad and Good Point Sets and the Triangular Mesh [26]

The triangulation algorithm consists of the following three steps:

1. Calculation of the *k*-neighborhood for each point $p_i \in S$
2. Initialization of triangulation
3. Adding triangles into the triangulation

To calculate the *k*-neighborhood for each point; firstly, a bounding box of the scattered point set is calculated. Then, the size of a cell and the resolution of a space partitioning box is computed. All cells inside or those ones intersecting the sphere with radius $d_k + s/2$ (s = size of a cell) and with the center at point p_i are searched. At the start, the radius is set to be infinite and is then updated during the search. The search begins with the cell containing point p_i. In each step, all adjacent cells that are inside or intersect the bounding sphere are put in an order, so that they can be inspected later. When all cells inside or those ones intersecting the sphere are visited, all indices of *k* closest points are found.

In the initialization of triangulation step, point p_{first} with the maximum z-coordinate is chosen because, at this point, the surface normal vector pointing outside an object is oriented in the z-direction. This selection helps the algorithm later to orient properly normal vectors of approximate tangent planes. Then, the closest point to point p_{first}, denoted with p_{second} is found. These two points are used to create the first edge in triangulation.

In the last step, all points $p_i \in S$ are added one after the other into triangulation. Let $e_{curr;nxt} = (p_{curr} ; p_{nxt})$, where $nxt = next_{curr}$, be the current edge in the triangulation border. The third point p_q , that together with points p_{curr} and p_{nxt} forms a new triangle, must be found.

The triangle is added into triangulation in one of the following three cases (Figure 3.7.):

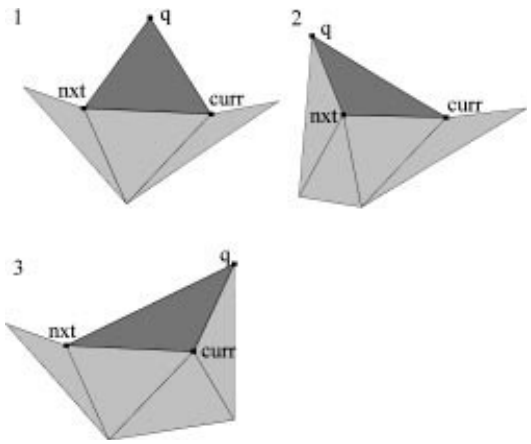

Figure 3.7. Adding Triangles into Triangulation [26]

This procedure is performed iteratively for edges on the border of triangulation, until all points are in triangulation and the border is empty (for closed surfaces). Figure 3.8.a displays 1500 scattered points on a surface of a sphere, while, in Figure 3.8.b, the triangulation obtained by the described triangulation algorithm is seen [26].

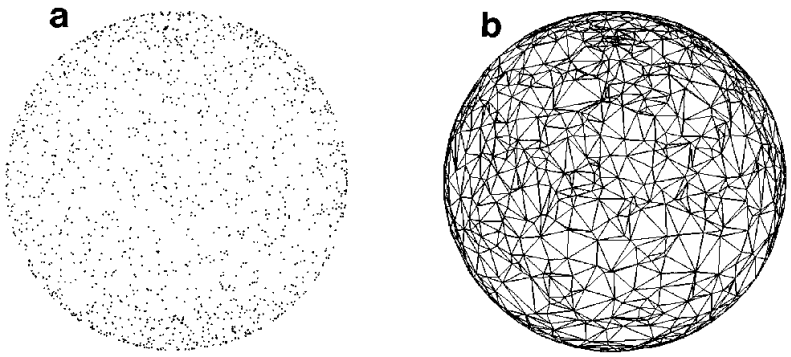

Figure 3.8. Scattered Points and Triangulation of a Sphere [26]

In Figure 3.9, 2200 scattered points on a surface of a glass and the obtained triangulation is displayed [26].

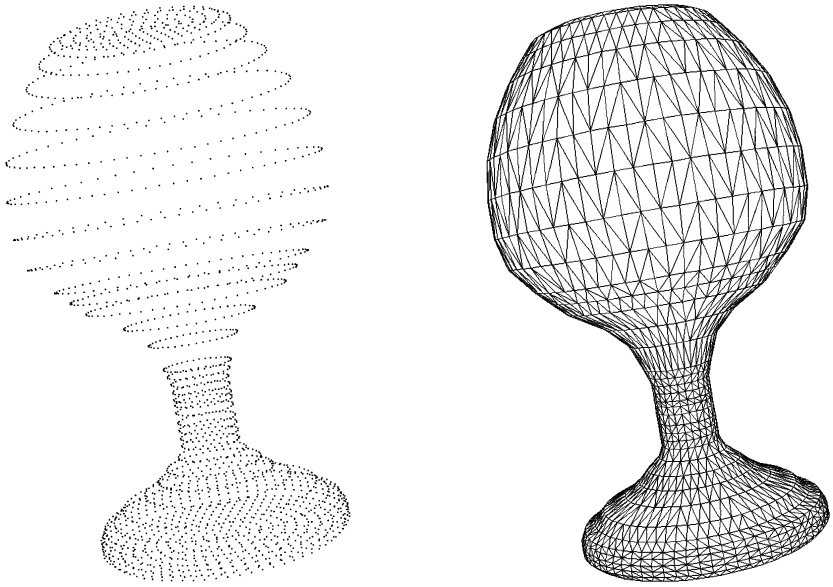

Figure 3.9. Scattered Points and Triangulation of a Glass [26]

Second step in direct triangulation is feature extraction. In this step, both view-independent (sharp edges and corners) and view-dependent features (silhouette polylines) of the reconstructed objects are detected. Only view-independent features are used in the phase of triangular mesh fairing.

The edge swapping procedure described in [29] is applied as a preprocessing step. It corrects some unwanted deviations of triangulation (Figure 3.10) and simply checks both possible diagonals of concave quads (built from two triangles). Then it chooses the one that minimizes the maximal angle between the normals of both triangles. This procedure improves significantly the triangulation around sharp edges.

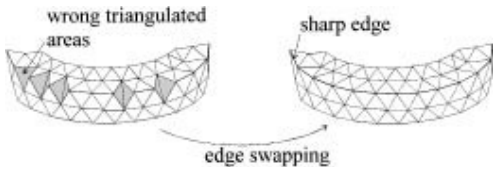

Figure 3.10. Edge Swapping Procedure [26]

After applying the edge swapping procedure as a preprocessing step, sharp edges can be extracted. For sharp edge detection, the following condition is used:

Let $e_{a,b} = (p_a, p_b)$ be an edge in triangulation with two adjacent triangles $\Delta(p_a, p_b, p_c)$ and $\Delta(p_b, p_a, p_d)$ with normals n_1 and n_2. Edge $e_{a,b}$ is then marked as sharp if the angle between normals n_1 and n_2 is greater than 0.5 radians.

A corner is then simply defined as a point on the surface where three or more sharp edges meet. In the last step of the procedure, sharp edges and corners are used as constraints for mesh decimation and mesh refinement procedures.

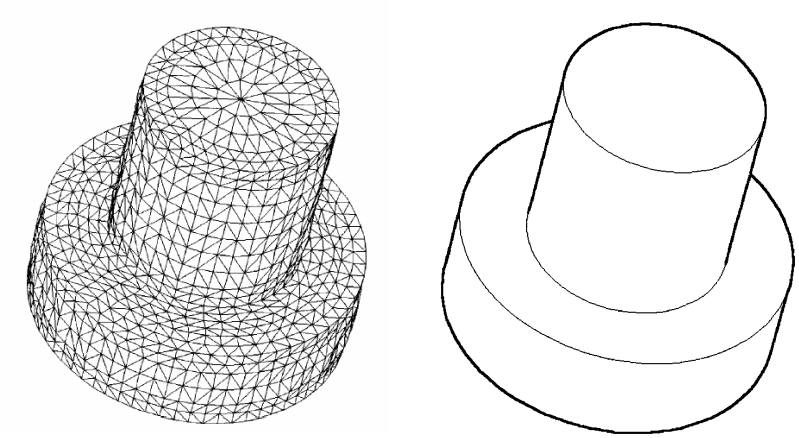

Figure 3.11. Triangulation and Wire-Frame Model of a Screw [26]

Figure 3.11 displays the triangulation and wire-frame model of an object similar to a screw [26]. In the wire-frame model, bold and thin lines denote visible silhouette polylines and sharp edges, respectively.

Last step of the reconstruction procedure is improving the triangular mesh. In this step, a number of triangles in triangulation are reduced in flat regions or increased in curved regions. Since the condition that the scattered point set must be isotropic is respected, the density of the set in flat regions is greater than necessary for quality presentation (but not for the triangulation procedure). Moreover, the density of the scattered point set in flat regions is increased when the object is thin (owing to the third condition for a scattered point set). Therefore, many unnecessary triangles can be eliminated there. However, in curved regions, the density of the object triangular mesh is often not sufficient for quality rendering.

Two procedures, mesh decimation and mesh refinement, can be used combined or separately, depending on the specific application.

In the mesh decimation procedure, the number of triangles in flat regions are reduced, so that the surface approximation quality is not decreased significantly. For this purpose, the decimation algorithm described by Schroeder [30] is modified. Each vertex in triangulation is checked by the method. First, it is classified as a simple vertex, a vertex lying on a sharp edge, or a corner. The vertex classified as a simple one or a vertex lying on a sharp edge is a candidate for removal. A simple vertex is removed if the distance between the vertex and the average plane is below some threshold, which is set by the user (Figure 3.12.a). Similarly, a vertex lying on a sharp edge is eliminated if the distance between the vertex and the line connecting two incident points on a sharp edge is below a value (Figure 3.12.b). When a vertex is removed, all triangles which include it are also eliminated from triangulation as well. The created gap is filled with 2D constrained triangulation. For this purpose, all vertices on the loop that encircles the gap are projected on the average plane. The projected loop is then filled with triangles by dividing recursively the loop into

two smaller ones. The recursion is stopped when the number of vertices in one loop is equal to three.

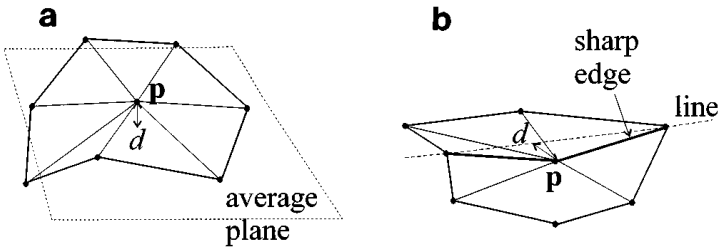

Figure 3.12. Test for Removal of a Candidate Vertex [26]

In Figure 3.13, the number of triangles in flat regions on the glass mesh is reduced from 4400 to 1800 [26].

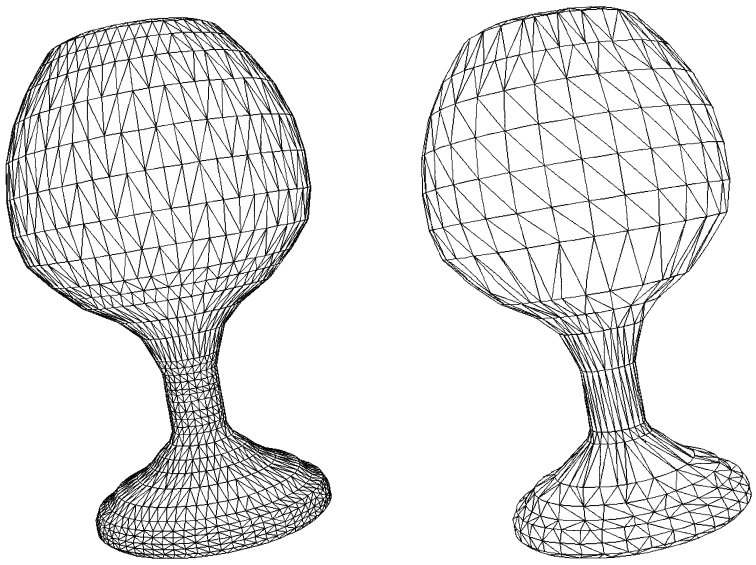

Figure 3.13. Decimated Triangular Mesh of a Glass [26]

In the mesh refinement procedure, the number of triangles in more curved areas are increased. For each triangle $\Delta(p_i, p_j, p_k)$ in triangulation, a test is made if the triangle should be divided into four or even six subtriangles. In each triangle corner, the normal is calculated by averaging all incident triangle normals (except those that are separated with a sharp edge). Let these normals be n_i ; n_j ; and n_k . The triangle is divided if the condition is satisfied,

$$\left|(n_j - n_i)\times(n_k - n_i)\right| > d_{threshold} \tag{3.14}$$

where $d_{threshold}$ is a user-specified parameter that controls the density of triangulation. To split the triangle into four subtriangles $\Delta(p_i, r_i, r_k)$, $\Delta(r_i, p_j, r_j)$, $\Delta(r_j, p_k, r_k)$ and $\Delta(r_i, r_j, r_k)$, new vertices r_i, r_j, and r_k must be calculated (Figure 3.14.a). If it is desired to split the triangle into six subtriangles $\Delta(p_i, r_i, q)$, $\Delta(r_i, p_j, q)$, $\Delta(p_j, r_j, q)$, $\Delta(r_j, p_k, q)$, $\Delta(p_k, r_k, q)$ and $\Delta(r_k, p_i, q)$, an additional point q must be inserted (Figure 3.14.b). The decision when to split a triangle into four or six subtriangles can be made upon the outcome of the test (e.g., how much the absolute value of the vector product in Eq.(3.14) exceeds the threshold). The described division is used only when all three adjacent triangles are split as well and when no triangle edge is in the sharp edge list. Otherwise, some subtriangles and additional points are omitted (Figure 3.14.c and Figure 3.15.d) [26].

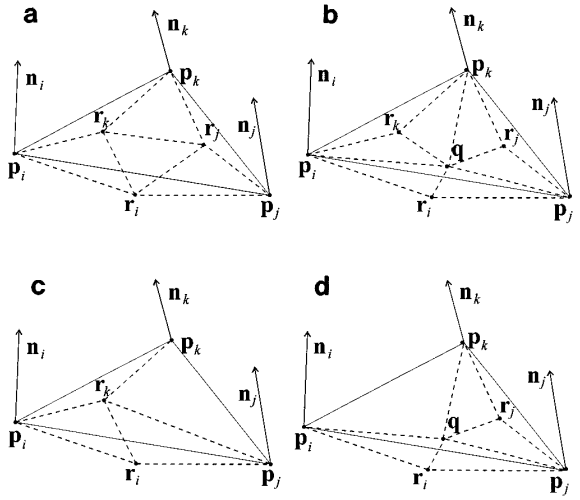

Figure 3.14. Mesh Refinement Procedure [26]

Figure 3.15.a displays a magnified detail of the original triangular mesh of the reconstructed glass. Figure 3.15.b presents the same part of the mesh refined by the four-subtriangle division procedure [26]. It can be spotted that the silhouette polylines are smoothed as well.

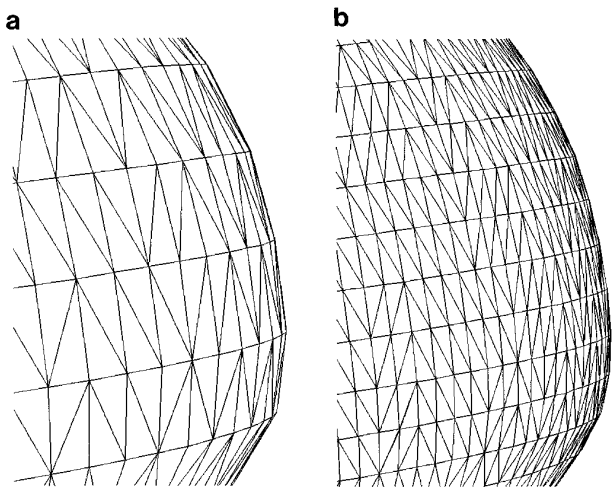

Figure 3.15. Original and Refined Mesh [26]

3.2.3.2.2. Delaunay Triangulation

The Delaunay triangulation is the dual structure of the Voronoi diagram. The Voronoi cell of a point separates it from its neighbour points. The edges of the Voronoi cells are the bisectors of the connection line segments of a point to its neighbour points. [31].

If Voronoi cells are neighbours to each other, then by connecting the points of these cells, Delaunay triangulations are obtained [32].In Figure 3.16 and Figure 3.17, Voronoi diagrams and their Delaunay triangulations are shown.

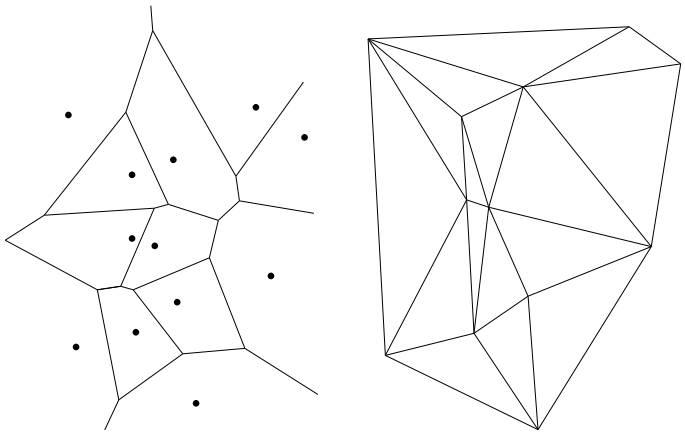

Figure 3.16. Voronoi Diagram and its Delaunay Triangulation [31]

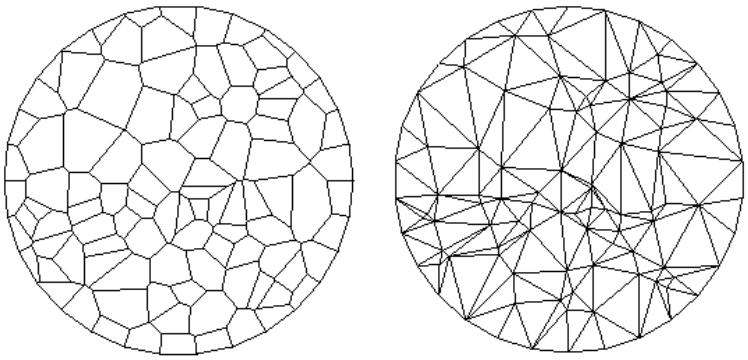

Figure 3.17. Voronoi Diagram and Delaunay Triangulation of a Point Set [32]

The definition of Voronoi diagrams and Delaunay triangulation in 3D space is straight forward [33]. Instead of triangles, there are tetrahedrals, besides Voronoi edges there are Voronoi faces and the empty circle property translates in empty sphere property.

Some basic properties of the Delaunay triangulation are as follows:

- The number of triangles in the Delaunay triangulation is at most (2N-5), where N is the number of vertices in the triangulation.
- The circumcircle (circumsphere) of every triangle (tetrahedron) does not contain any other points of the triangulation.
- Each line segment that connects a point to its nearest neighbour point is an edge of a triangle.
- The Delaunay triangulation maximizes the minimum angle over all triangulations. (This property is related with the maximum-minimum criteria which is explained in the following pages)

There are two criteria which constitute the basis of the Delaunay triangulation. First one is the circumcircle criteria, second is the maximum-minimum criteria[31].

The circumcircle (circumsphere) of each Delaunay triangle (tetrahedron) cannot contain any points of the triangulation. If four or more points are on the same circumcircle (circumsphere), this will be an exceptional case.

While determining the third point of a triangle, it is checked whether any other points lie in the circumcircle of these three points. If so, then the point lying in the circumcircle is a candidate for the third point of the triangle. This operation (Figure 3.18) is repeated until the circumcircle of the triangle has no points lie in it [31].

To check the circumcircle criteria, r (radius of the circle) is compared with d (distance between a point to the center). If d < r, then point E lies in the circle and it is the third corner of the triangle.

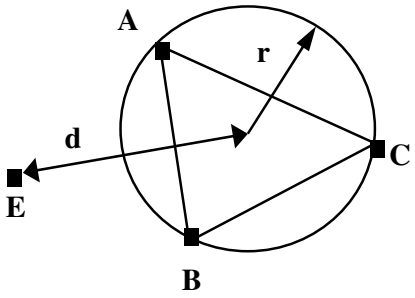

Figure 3.18. Circumcircle Criteria Check [31]

The second criteria which constitute the basis of the Delaunay triangulation is maximum-minimum criteria. In the maximum-minimum criteria [34], the diagonal of each convex quadrangle should be selected carefully. It is desired for the triangles created by the diagonal to be nearly equilateral. This is done by edge flipping process. In Figure 3.19, edge flipping is basically shown.

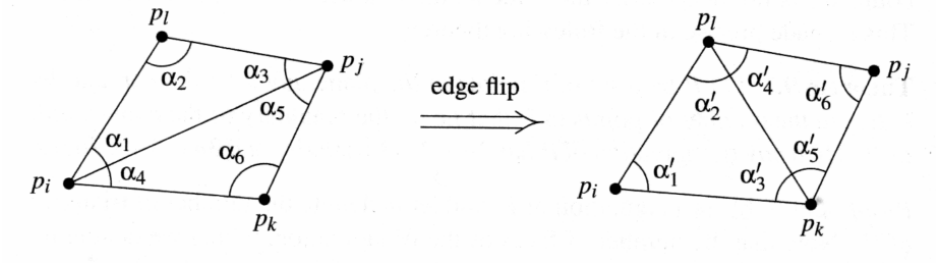

Figure 3.19. Edge Flipping Process

There are two possible diagonals for each convex quadrangle. The one which will create better triangles should be selected. Edge p_ip_j is illegal and p_{kl} is legal if the below condition is satisfied.

$$min\ \alpha_i < min\ \alpha_i^{'}\ (1 \le i \le 6) \tag{3.15}$$

Shortly, in order to obtain an optimal triangulation, angles of the two triangles are compared and the triangle that maximizes the minimum angle is selected.

3.2.3.3. Deformable Models

Another mesh generation method from 3D point cloud is creating an initial mesh and deforming it to match the range data. Shrink-wrapped boundary face (SWBF) algorithm [35] produces the final surface by iteratively shrinking the initial mesh generated from the definition of the boundary faces.

Originally, the shrink-wrapping-based mesh generation technique was proposed by Kobbelt [36]. He introduced a deformable surface scheme for converting a given unstructured triangle mesh into one having subdivision connectivity based on a simulation of the shrink wrapping process.

Recently, Jeong [37] extended the shrink-wrapping concept to produce a mesh model from unorganized points. For a given 3D point cloud, a bounding box of M_b,

which consists of 12 triangles (two triangles per each face of a bounding box) is computed (Figure 3.20.a). Then the box is linearly subdivided up to a user-defined level; i.e., each triangle is divided into four subtriangles and new vertices are placed at the middle of each edge. To simulate a shrink-wrapping process, *projection (shrinking)* and *smoothing* operations are applied repeatedly to metamorphose the initial mesh into one similar to the surface of the original object. Since the initial mesh is always a box shape, the reconstructed mesh is restricted. This method cannot be applied to objects containing holes such as a ring. Furthermore, for each vertex of the initial mesh, this method has to find the nearest point from all of the input points during the shrinking process, which is very time consuming.

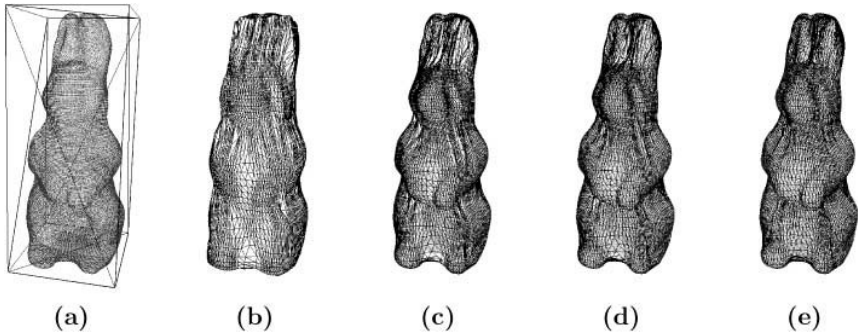

Figure 3.20. The Initial Bounding Box and Shrink-Wrapping Process [37]

The major difference in shrink-wrapped boundary face (SWBF) algorithm [35] from the method of Jeong [37], is the shape of the initial mesh. Instead of using the six faces of a bounding box, cell boundary faces are used as the initial mesh. This idea enables to overcome the restriction of the previous methods [35], [36] and considerably improves the computational time efficiency of generating a surface mesh.

Let O_{real} be a real world object and $P=\{p_1, p_2, ...,p_n\}$ be a point cloud sampled from O_{real} and also no connectivity information between points $p_1, p_2, ...,p_n$ is given. SWBF generating M^P (polygonal mesh) from an unorganized point cloud P includes the following four major stages:

1. Partitioning the 3D space into a set of cubes
2. Defining a boundary face and generating an initial mesh
3. Shrinking
4. Surface smoothing

3.2.3.3.1. Partitioning of the 3D Space into a Set of Cubes

Firstly, a bounding box B_P of P is estimated. Then, from the bounding box of P, a cell and cell space are defined. The cell space (C_P) is a dissection of B_P by three orthogonal sets of equally spaced parallel planes. Each cube, a component of the cell space, is called a cell and is denoted by c or $c(i, j, k)$.

The size of each cell, or the resolution of the cell space, should be carefully selected according to the density of P . If the resolution is too low (compared with the point sampling density), the resulting mesh becomes coarse and cannot represent the detailed topology of the object. If it is too high, there may be some unwanted holes in the resulting surface mesh. But its precise value is not very crucial compared with the previous method of [37].

3.2.3.3.2. Generating the Initial Mesh

The cells in C_P can be divided into two groups. The *boundary cell* (c_b), which contains one or more physical 3D points of P and the *outer cell* (c_o) which does not contain any 3D points of P.

The neighbors of cell c are assumed to be cells which are adjacent to one of the six faces of c and are denoted as $n(c)$. The term *O(1)-adjacency* for this kind of neighbour definition can be used.

For boundary cell, six faces exist in the directions of the O(1)-adjacent neighbour cells. A face of c_b is defined to be a *boundary face(f_b)* if the adjacent cell is an outer one. Otherwise, the face is defined to be an *inner face (f_i)*.

A boundary face can be interpreted as a simple approximation of the actual object surface, which is contained in the corresponding boundary cell. Consequently, the *initial mesh* denoted as M^I is defined to be the set of boundary faces as

$$M^I = \{\forall f_b(c_b) | c_b \in C_P \tag{3.16}$$

The initial mesh M^I in this method is a crude approximation of the surface of P for further processing, but it usually preserves the topology of the original object unless the hole in object surface is too small compared with the resolution of the cell space. The crude surface is then iteratively metamorphosed into the original surface by shrinking and smoothing operations.

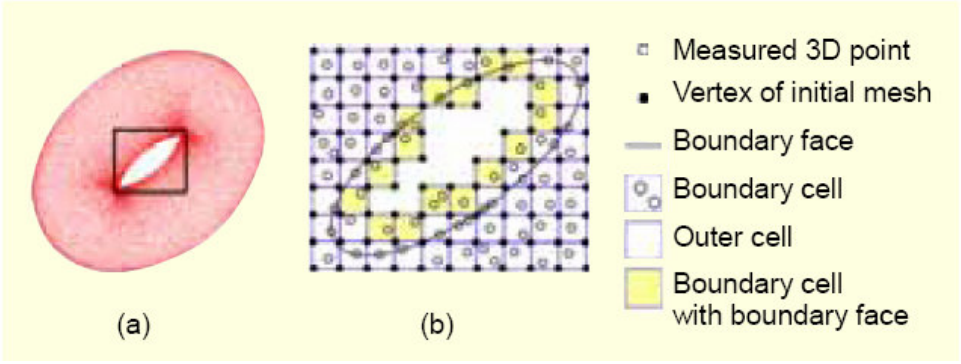

Figure 3.21. Conceptual Illustrations of the Definitions in [35]

3.2.3.3.3. Shrinking

Shrinking is applying an attracting force to each vertex. The attraction force of a vertex is a vector between the initial mesh M^I and the given point cloud P. In the method of Jeong [37], for each vertex q_i of the bounding box, the nearest points is

searched in the whole 3D point cloud. However, in SWBF algorithm this search step differs. For a vertex q_i of M^I, there is a boundary cell c_b containing it. The nearest 3D point in P minimizing the Euclidean distance from q_i to that point is searched. Unlike in the method of [37] requiring a global search for optimal results, shrinking in SWBF can be done in a local manner to obtain a similar result. Since the nearest point p_{near} should always be inside of c_b or inside of the *O(3)-adjacent* neighbors of c_b, it is sufficient to search for only the 3D points inside a total of 27 cells for the nearest one from a mesh vertex. This greatly saves the processing time for the shrinking process.

After finding p_{near}, the attracting force pushes q_i towards p_{near} as

$$q_i \leftarrow q_i + \alpha\{p_{near} - q_i\} \tag{3.17}$$

The weight α (0.0 to 1.0) determines the amount of the attracting force. Since there is a possibility of sharing the same point by more than two mesh vertices, the full attraction force (α =1) may meet these vertices at the same position causing a non-manifold region in the surface. To avoid this problem, a value less than 1.0; experimentally chosen 0.5 should be assigned for α [35].

3.2.3.3.4. Surface Smoothing

Smoothing is relaxing the shrink-wrapped surface for achieving a uniform vertex sampling, and the method used in [37] is also used in SWBF algorithm. An approximation of Laplacian $\mathscr{L}$ as in [38] is employed; this is the average vector of 1-neighbor edge vectors of a given vertex, and its application usually results in shrinkage. Thus, the tangential component $\mathscr{L}_t$ (Figure 3.22.a) of $\mathscr{L}$, which is perpendicular to the vertex normal is taken.

The final tangential Laplacian $\mathcal{L}_t$ of a given vertex v_i and an iterative smoothing equation are as follows:

$$\mathcal{L}_t(v_i) = \mathcal{L}(v_i) - (\mathcal{L}(v_i).\mathrm{n})\,\mathrm{n} \tag{3.18}$$

$$(M^I)' = M^I + \lambda\,\mathcal{L}_t \tag{3.19}$$

In Eq.(3.19), λ is the speed parameter that controls the convergence speed of the smoothing operation. If a small value for λ is applied (λ=0.5 in Figure 3.22.b), a detailed result can be obtained but there are wrinkles since several vertices may share the same position induced by the strong projection force. Also, a uniformly sampled mesh cannot be obtained . If a large value for λ is used (λ=0.8 in Figure 3.22.c), an opposite result is obtained because of the strong relaxation force; no wrinkles exist and a uniformly sampled mesh is obtained, but also the shrink-wrapping procedure might fail to capture detailed regions. Hence, finding the appropriate speed parameter is important. In the experiments, λ=0.8 is found to be a reasonable starting choice[37].

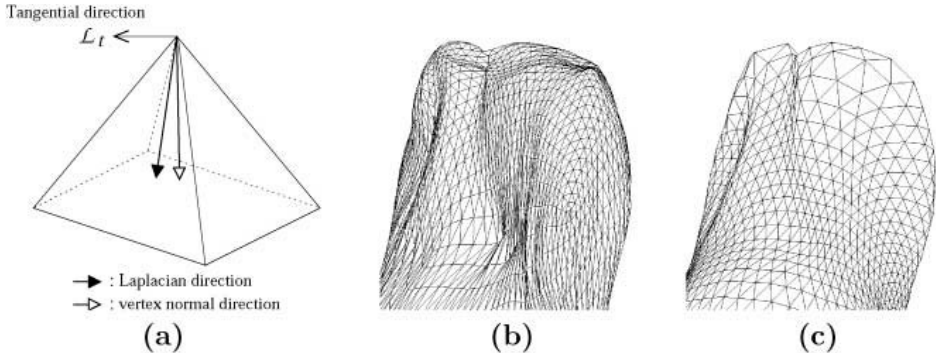

Figure 3.22. Tangential Laplacian and the Comparison of Shrink-Wrapping with Different Speed Parameters [37]

Figure 3.23 illustrates the difference between SWBF [35] and the method proposed by Jeong [37] for a ring data containing a hole in the surface. The initial mesh in SWBF (Figure 3.23.e) envelops all of the original 3D points in Figure 3.23.a. As shown in the resulting mesh (Figure 3.23.g), four iterations of the shrink-wrapping process would be sufficient in SWBF for general cases. But as Figure 3.23.d shows, it is still not enough in [37] because the shape of the initial mesh (Figure 3.23.b), is far from the real object.

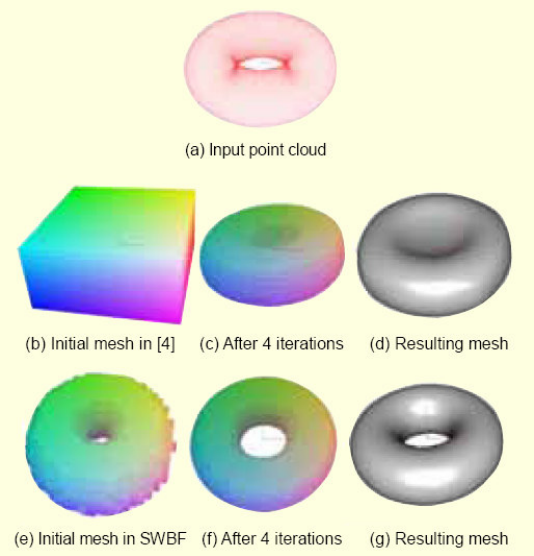

Figure 3.23. Difference between SWBF Method[35] and Jeong's Method [37]

Figure 3.24 shows the reconstruction results for the General data [35]. SWBF works well for generating a mesh even with only three occurrences of the shrinking and smoothing process. Table 3.2 summarizes the reconstruction results. The overall processing time was less than 60 seconds on a Pentium 2.0 MHz PC. Since SWBF searches only O(3)-adjacent neighbors in finding the nearest points, it is much faster than previous works [37], [39].

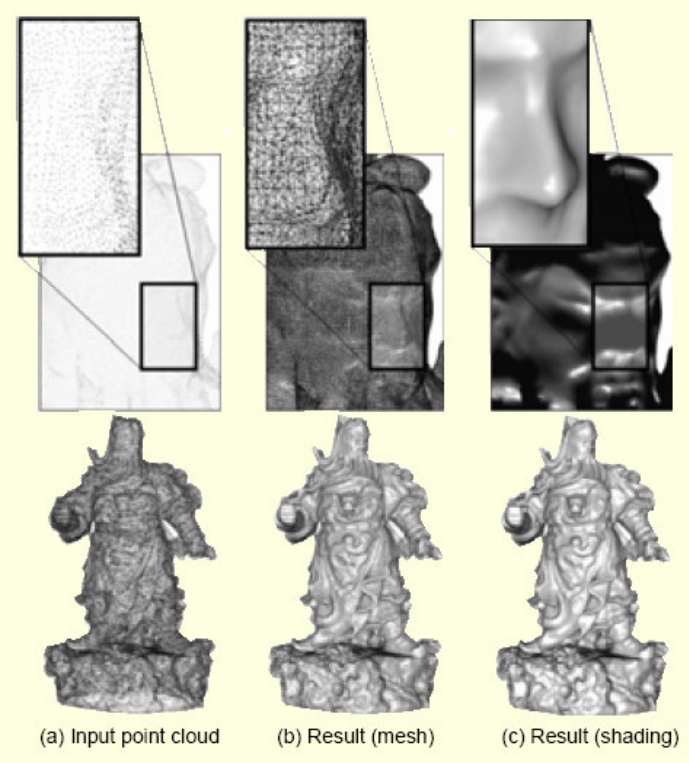

Figure 3.24. Reconstruction Result for the General Data [35]

Table 3.2. Reconstruction Summary for the General data [35]

Input point cloud	Number of 3D points	1,658,574
Cell space	Maximum resolution	Y:200
	Number of boundary cells	96,164
	Average number of points in a boundary cell	17.2473
Reconstructed mesh	Number of vertices	198,267
	Number of triangular faces	398,784

3.3. Data Processing Programs

In the data processing step, there are two programs which are both compiled in Microsoft Visual Basic 6.0. First one is a preprocessing program which makes obtained sensor data ready for the main processing program.

Basic steps of the preprocessing program are as follows:

1. Checking whether there are discontinuities (jumps) between the consecutive RawE1 values.
2. Determining repetitions for the same RawE1 values and taking the averages of range, angle2 and amplitude values for that RawE1 value.
3. Converting polar coordinates into cartesian coordinates.

Basic steps of the main processing program are as follows:

1. Calculating Resolutions
2. Detecting Noisy Points
3. Segmentation
4. Triangulation
5. Line Fitting to Objects

3.3.1 Preprocessing Program

The preprocessing program firstly reads calibrated range data in mm, raw count (1-6152) (encoder 1), angle (0-360°) (encoder 2) and signal amplitude (0-255) values from the texfile which is created by the data acquisition step.

After reading data, it is checked whether there are jumps between the consecutive RawE1 values. If the Eq. (3.20) is satisfied, then there are not any jumps between the consecutive RawE1 values.

$$RawE1(k+1) - RawE1(k) \leq 1 \qquad (3.20)$$

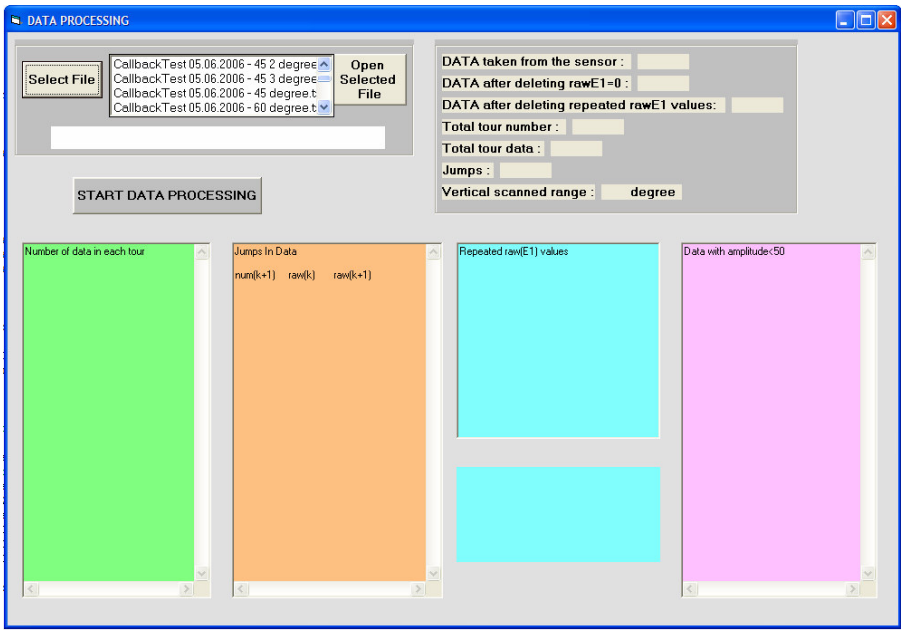

Figure 3.25. Screenshot of the Data Preprocessing Program

Then, repetitions for the same RawE1 values are determined. Table 3.3 shows a sample data which has repetitions for two or three times.

Table 3.3. Data Taken from the sensor

RawE1	Range (mm)	Q_2 (angle)	amplitude
1	279.853	0.000255924	23
1	275.338	0.000341232	24
1	262.107	0.000511848	24
2	268.722	0.000597156	24
2	255.491	0.000682464	24
3	266.603	0.000767772	23
3	268.722	0.000853081	24
3	266.603	0.000938389	23

Table 3.4 shows the average range, angle2 and amplitude data for the repeated RawE1 values.

Table 3.4. Data after Taking Averages

average RawE1	average range	average Q_2	average amplitude
1	272.432	0.000369668	23.666
2	262.106	0.000639810	24
3	267.309	0.000853080	23.333

After passing the previous steps without any jumps in the RawE1 values and taking averages of the repeated values for each RawE1, now all data is converted from polar coordinates into cartesian coordinates. The below equations are used for the conversion:

$$Q_1 = rawE1.\frac{360}{6152} \tag{3.21}$$

$$ccx = range.\cos Q_2.\cos Q_1 \tag{3.22}$$

$$ccy = range.\sin Q_2 \tag{3.23}$$

$$ccz = range.\cos Q_2.\sin Q_1 \tag{3.24}$$

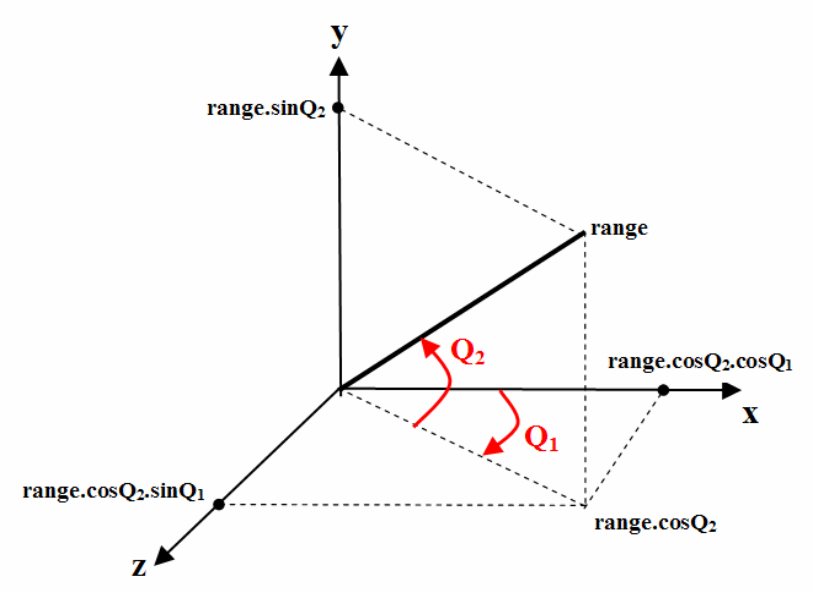

Figure 3.26. Converting Polar Coordinates into Cartesian Coordinates

As the final step of the preprocessing program, each line with order number, cartesian coordinates and amplitude are written to a textfile for the main processing program to use.

3.3.2. Main Processing Program

Main processing program calculates resolutions in horizontal, vertical and diagonal directions, detects noisy point, segments objects, creates meshes and fits lines to objects.

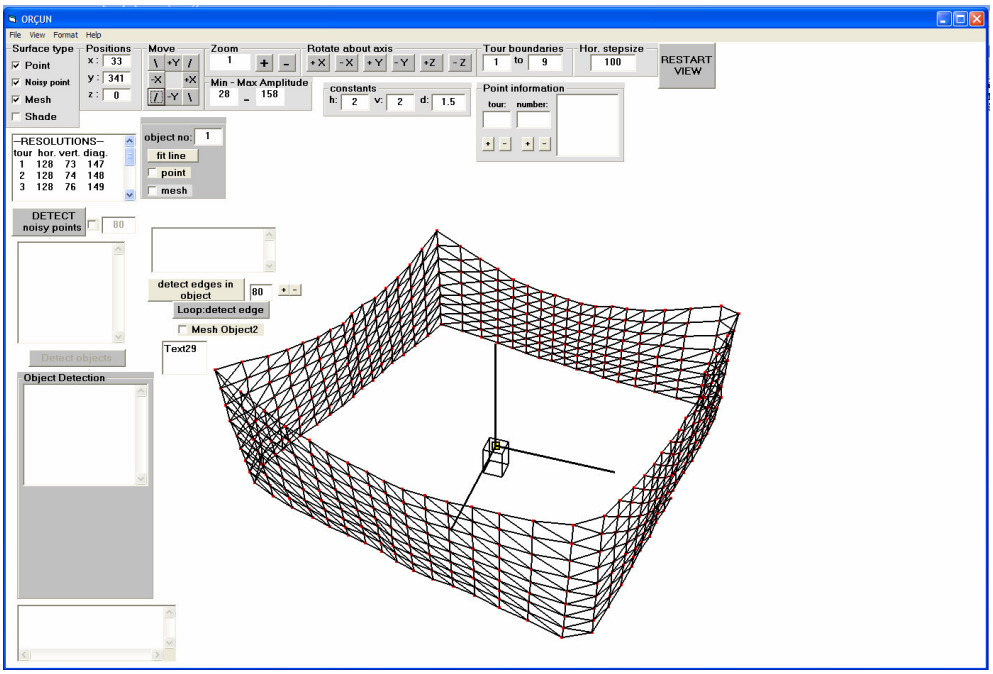

Figure 3.27. Screenshot of the Data Processing Program

3.3.2.1. Calculating Resolutions

Calculation of the resolutions in horizontal, vertical and diagonal directions is very important for the triangulation step because these values are directly used in the determination of the threshold values in triangulation step.

First step is to calculate horizontal resolution. For each tour, an average range value is calculated by dividing the sum of the range values in that tour to the total points in one tour(6152).

In the Eq. (3.24), *i* designates tour number and *j* designates the order of the encoder pulse in that tour number.

$$Average\ range\ (\ i\ ) = \frac{\sum_{j=1}^{6152} range(i, j)}{6152} \tag{3.25}$$

Azimuth encoder gives 6152 pulses in each turn so the angle between two pulses is:

$$\alpha = \frac{360}{6152} \tag{3.26}$$

Finally, horizontal resolution is calculated by cosine theorem:

$$Horizontal\ resolution(\ i\ ) \sqrt{\left(2.\left(averagerange(i)\right)^2 - 2.\left(averagerange(i)\right)^2.\cos(\alpha)\right)} \tag{3.27}$$

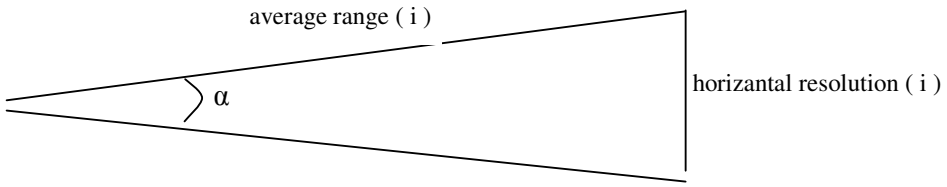

Figure 3.28. Horizontal Resolution Calculation

After horizontal resolution, vertical and diagonal resolutions are calculated. Vertical resolution is calculated by dividing the sum of the distances between two consecutive vertical points (point(i,j) & point(i+1,j)) to the total points in one tour(6152).

$$Vertical\ resolution\ (\ i\ ) = \frac{\sum_{j=1}^{6152}\left(point(i,j) - point(i+1,j)\right)}{6152} \tag{3.28}$$

$$Diagonal\ resolution\ (\ i\ ) = \sqrt{\left(Horizontalresolution^2(i) + Verticalresolution^2(i)\right)} \tag{3.29}$$

3.3.2.2. Detecting Noisy Points

Noisy points generally occur between objects which have different range values. Laser light may focus on two or more different points while trying to measure the edges of objects. In these cases, an average range value for that measurement is calculated. This situation is shown in Figure 3.29. An object is placed in front of a wall and the medium is scanned with the laser sensor. Violet points in the figure show noisy data.

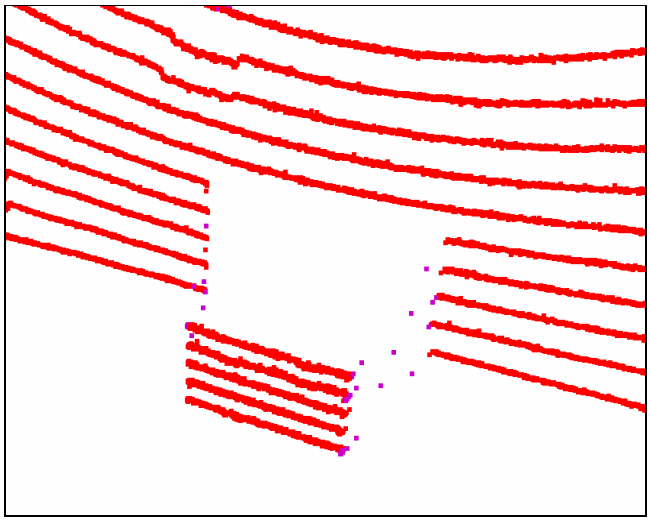

Figure 3.29. Noisy Points between an Object and Wall

The main factor in the existence of noisy points is the range difference so, in order to detect noisy points, range values of obtained data are compared. This case is shown in Figure 3.30.

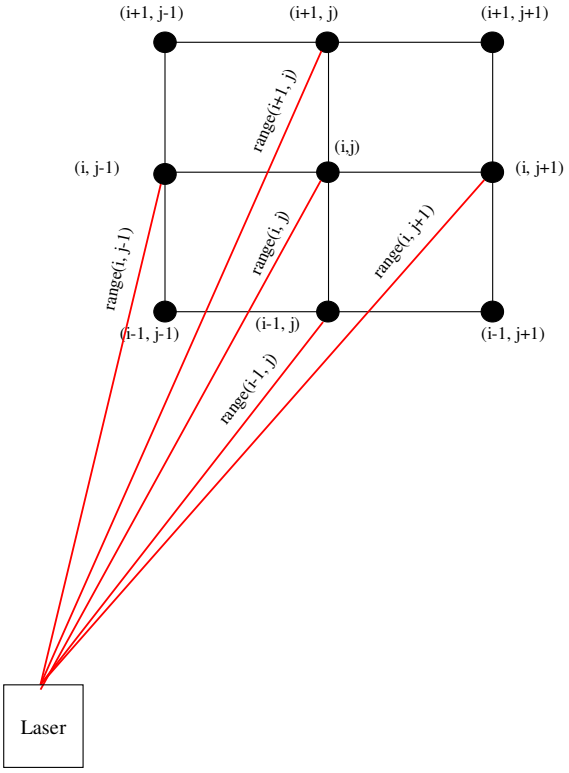

Figure 3.30 Range Values of Sampled Data

If one of the below conditions is satisfied, then point(i , j) is said to be a noisy point. Condition 1 checks the range difference between points in horizontal direction and condition 2 checks the range difference between points in vertical direction.

Condition 1:

range (i,j)-range (i,j+1) > threshold(i) & range (i,j)-range (i,j-1) > threshold(i) (3.30)

Condition 2:

range (i,j)-range (i+1,j) > threshold(i) & range (i,j)-range (i-1,j) > threshold(i) (3.31)

In the above conditions, determination of the threshold is very important. For each tour, a different threshold value is determined because range values increase as the tour numbers increase. Each tour's threshold value is equalized to vertical resolution in that tour which was calculated before. Main reason for this equalization is that; during triangulation, it is desired for the triangles to be nearly equilateral. After setting mirror speed and obtaining data in the data acquisition section, vertical resolution in the image cannot be changed so in order to obtain nearly equilateral triangles, range differences between a point and its neighbour points should be close to vertical resolution. As explained in condition 1 and condition 2, if the range difference between a point and its neighbour points in either horizontal or vertical direction is greater than vertical resolution, then that point cannot be included in the triangulation algorithm and is said to be a noisy point.

3.3.2.3. Segmentation

As explained in the previous section, most important criteria of the segmentation is the determination of the threshold value. If the distance between two consecutive scanned points is greater than the threshold, then segments are separated.

There are many different threshold conditions in the literature; Dietmayer et al.[20] used the condition in Eq. (3.2), Santos et al. [21] used the condition in Eq. (3.3), Lee [22] used the condition in (3.4) and Borges and Aldon [23] used the condition in Eq. (3.5). None of the above conditions are suitable for the segmentation process in the thesis so a constant value is selected as the threshold condition.

3.3.2.4. Triangulation

After the data acquisition step, obtained data is not in the form of point cloud like other surface fitting methods. Number of data in each tour and number of tours are known so arrays are created for each point. Arrays are in the form of point(i , j) where *i* designates the tour number and *j* designates the order of the encoder pulse in that tour number. This situation brings a very important advantage to the triangulation algorithm; there is no need to search for the neighbour points because arrays give that information. Triangulation process is briefly shown in Figure 3.31.

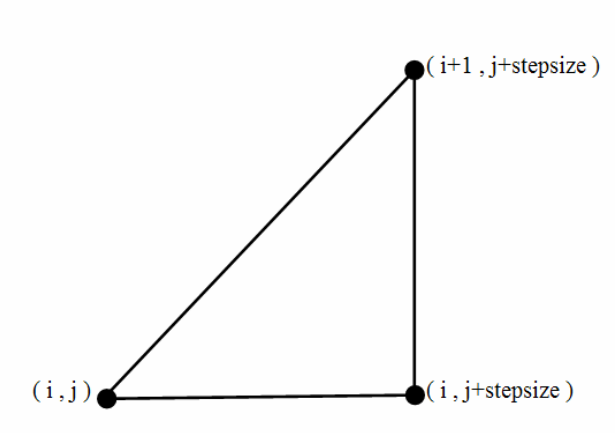

Figure 3.31. Triangulation Process Used in the Thesis

Point (i,j) is connected with its neighbour points in horizontal (point(i , j+stepsize)) and diagonal (point (i+1 , j+stepsize)) and point (i , j+stepsize) is connected with its neighbour point in vertical (point (i+1 , j+stepsize)). Stepsize is the interval of points in horizontal when output image is shown. For example if a stepsize of 100 is selected by the user, then in each tour only 61 points is shown as the output image because there are totally 6152 points in each tour.

The meshing process of neighbour points is done according to a condition; distance between two consecutive points in horizontal (distanceh), vertical (distancev) and diagonal (distanced) are compared with threshold values. If distances are smaller than threshold values, then meshes are created. Thresholds determined in three directions are as follows:

$$Horizontal\ threshold = horresolution(i).\frac{range(i)}{averagerange(i)}.horconst \quad (3.32)$$

$$Vertical\ threshold = verresolution(i).\frac{range(i)}{averagerange(i)}.verconst \quad (3.33)$$

$$Diagonal\ threshold = diaresolution(i).\frac{range(i)}{averagerange(i)}.diaconst \quad (3.34)$$

In Eq.(3.32), horresolution (i) is the horizontal resolution for the i^{th} tour, range(i,j) is the range value of the point which triangulation is started, averagerange(i) is the average range value of all points in the i^{th} tour and verconst is a constant value for the equation.

Fraction part in the threshold equations (range(i) / averagerange(i)) is useful for closer objects to have a smaller threshold value and distant objects to have a higher threshold value because distances between points in three directions gets higher for the distant objects and smaller for the closer objects.

As the constants in the threshold equations are increased, creating meshes between far away points becomes easier. Some different cases for selection of the constants are shown in Figures [3.32-3.36]

In Figure 3.32, all constants are equal to 1 so there is no effect of constants in the threshold equations. In other figures, just one constant is changed and other two are kept constant in order to see the effect of the changed constant. Horizontal constant is increased into 1.5 (Figure 3.33), vertical constant is increased into 1.2 (Figure 3.34) and diagonal constant is increased into 1.5 (Figure 3.35). As a final case all three constants are increased at the same time (Figure 3.36),

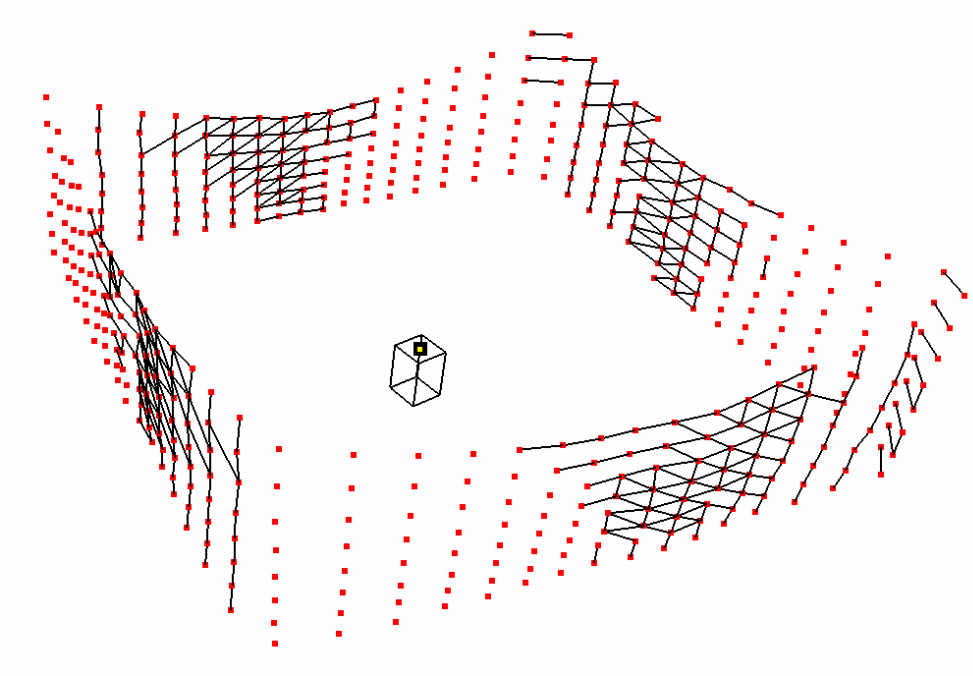

Figure 3.32. Output Image

(Horizontal Const: 1, Vertical Const: 1, Diagonal Const: 1)

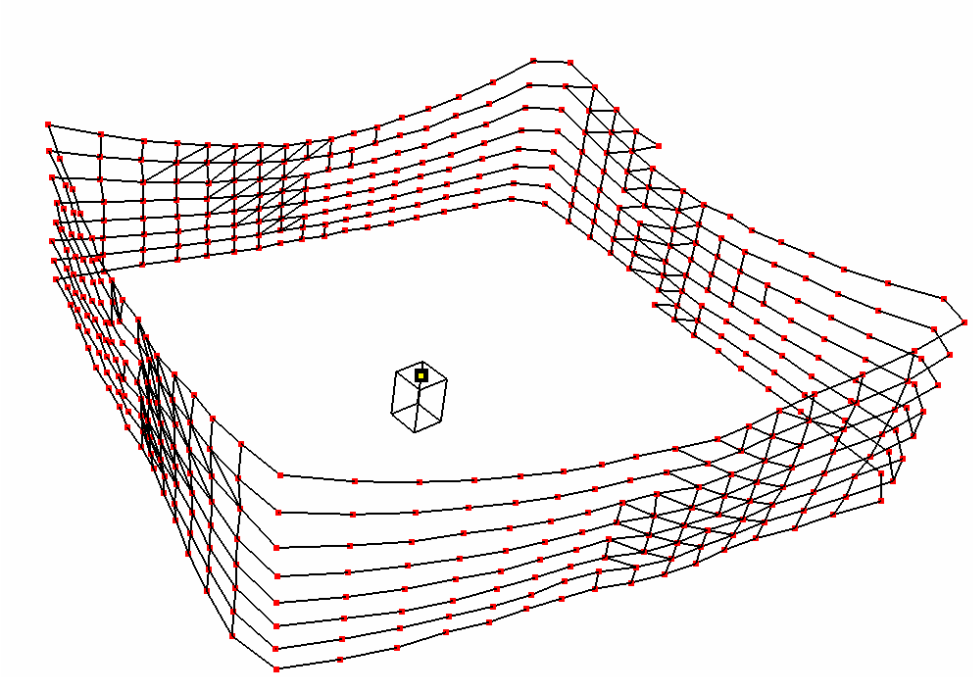

Figure 3.33. Output Image

(Horizontal Const: 1.5, Vertical Const: 1, Diagonal Const: 1)

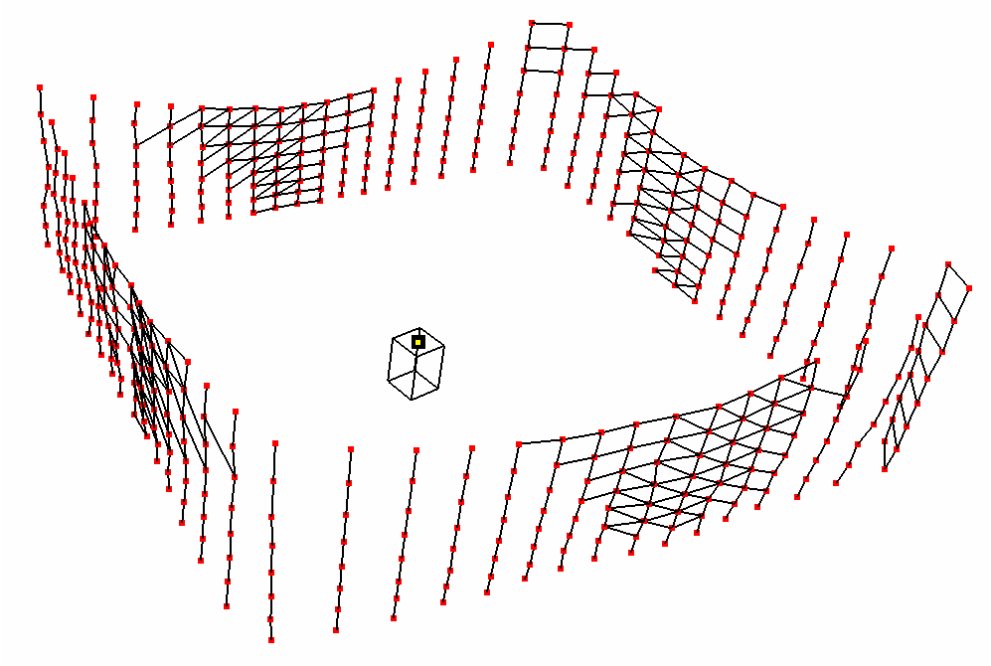

Figure 3.34. Output Image

(Horizontal Const: 1, Vertical Const: 1.2, Diagonal Const: 1)

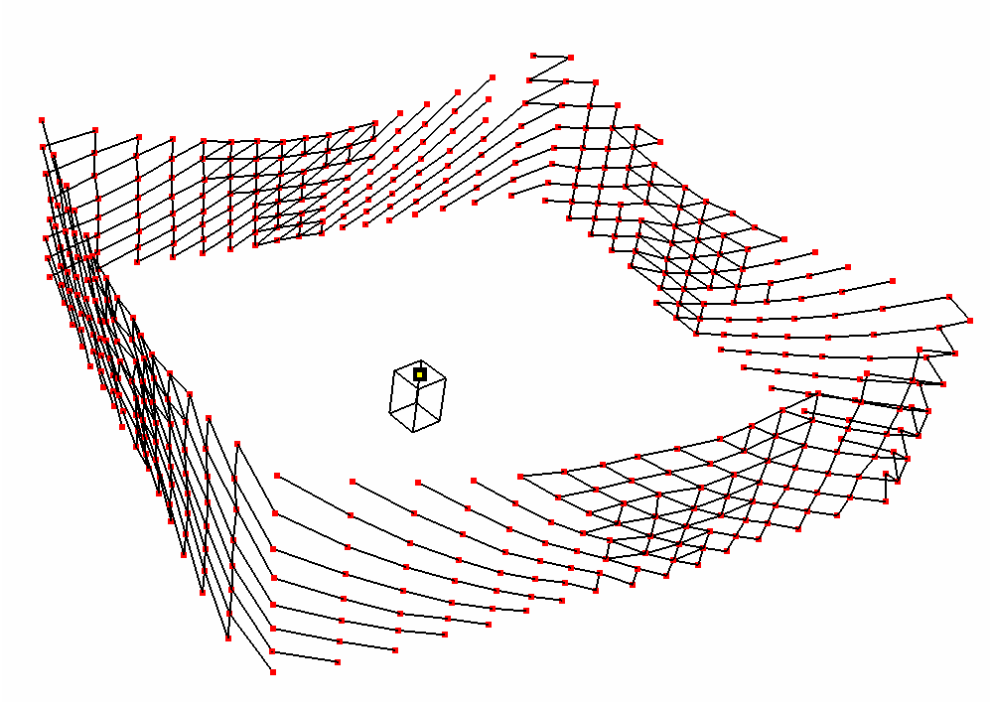

Figure 3.35. Output Image

(Horizontal const: 1, Vertical Const: 1, Diagonal Const: 1.5)

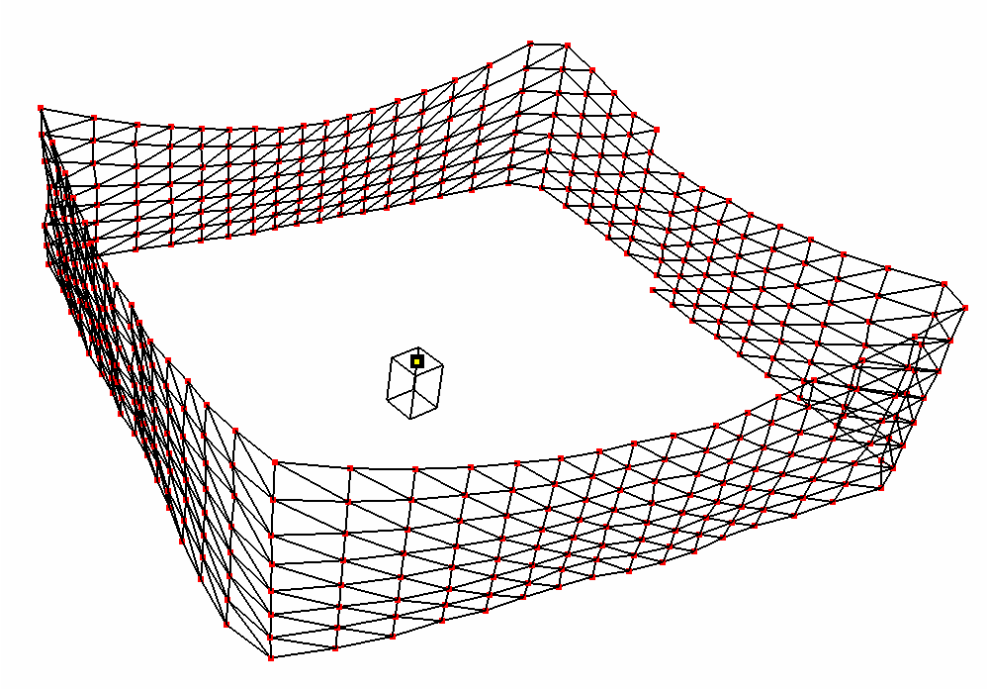

Figure 3.36. Output Image
(Horizontal Const: 1.5, Vertical Const: 1.2, Diagonal Const: 1.5)

Advantages of the triangulation method used in the thesis against other surface fitting methods:

1. Unlike functional fitting, obtained data is used directly; no approximation to the data is applied in order to match with an existing physical model. Also, surfaces for objects with sharp edges can be fit.
2. There is no need to satisfy any condition as in the direct triangulation method and to search for nearest points for each point.
3. Obtained triangles are nearly equilateral and do not contain any other points in the circumcircle of triangles so there is no need to check for the two criteria which constitute the basis of Delaunay triangulation.
4. Unlike deformable models, triangulation algorithm is straight forward, it does not contain so many steps like partitioning the 3D space into a set of cubes, generating initial mesh and shrinking.

3.3.2.5. Line Fitting

After segmentation and triangulation steps, lines for the detected objects can be fit and the projections of the obtained points to these lines can be taken. Then, by using the projected points, new meshes are created for objects. Projection process is shown in Figure 3.37.

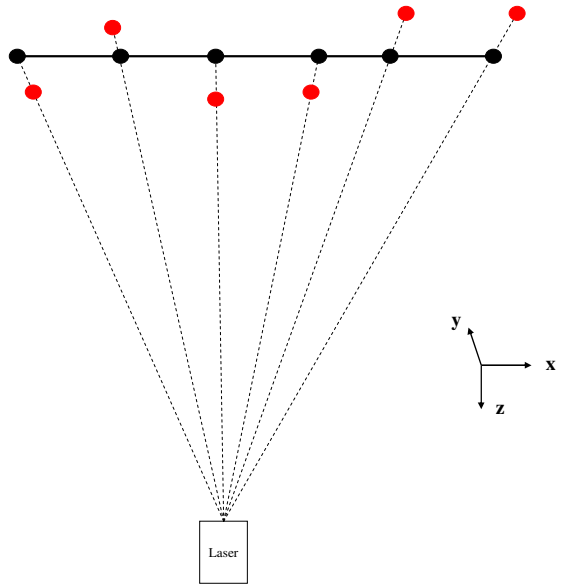

Figure 3.37. Projection of the Scanned Points to the Fit Line

The basic steps of the projection process are explained below:

Equation of a line is:

$$z = a_0 + a_1 x \tag{3.35}$$

From least squares technique, the coefficients of the line equation are:

$$a_1 = \frac{n.\sum(x_i.z_i) - \sum x_i.\sum z_i}{n.\sum x_i^2 - \left(\sum x_i\right)^2} \tag{3.36}$$

$$a_0 = \frac{\sum z_i}{n} - a_1 . \frac{\sum x_i}{n} \tag{3.37}$$

Equation of a plane is:

$$x + by + cz = d \tag{3.38}$$

If this plane passes through y-axis (sensor) and a point (x_1,y_1,z_1), then new plane equation is:

$$x - \frac{x_1}{z_1} . z = 0 \tag{3.39}$$

The intersection point (x_2,y_2,z_2) of the plane and the line is:

$$x_2 = \frac{a_0}{\frac{z_1}{x_1} - a_1} \tag{3.40}$$

$$z_2 = \frac{z_1}{x_1} . \frac{a_0}{\left(\frac{z_1}{x_1} - a_1 \right)} \tag{3.41}$$

Point (x_2,y_2,z_2) is the projection of the point (x_1,y_1,z_1) on the line $z = a_0 + a_1 x$.

This projection process is applied to each point that belongs to an object. Then, by using the projected points, new meshes are created. In Figure 3.38, a flat object is placed in front of a wall. After detecting and clearing noisy points, object is ready for line fitting. In Figure 3.39, the surface of the object is created with normal triangulation process but in Figure 3.40 the surface is created by line fitting. The surface created with line fitting is smoother than the surface created with normal triangulation.

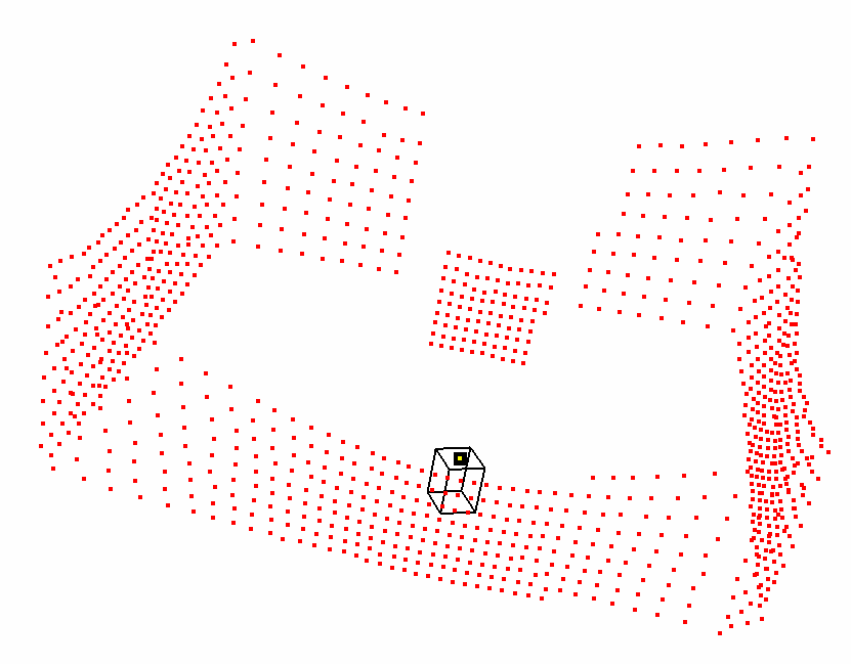

Figure 3.38. A Flat Object Placed in front of a Wall

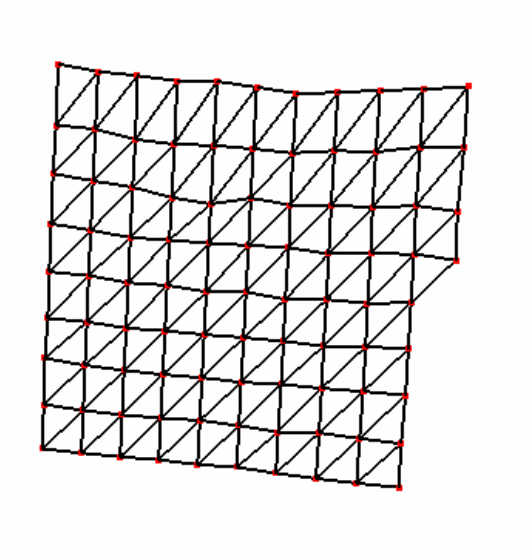

Figure 3.39. A Flat Surface before Line Fitting

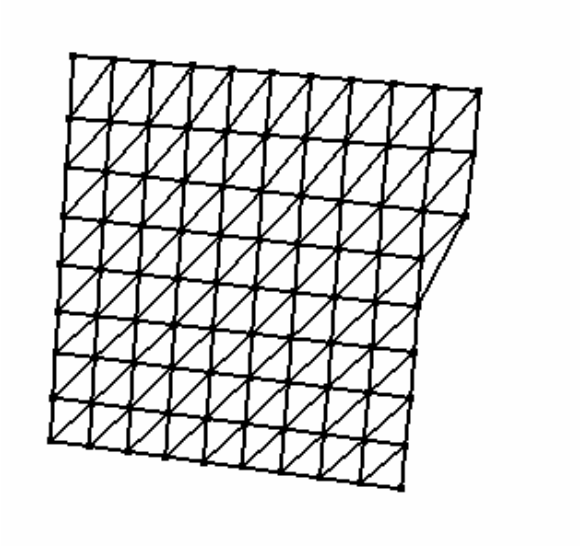

Figure 3.40. A Flat Surface after Line Fitting

In Figure 3.41, an object which has a sharp edge is placed in front of a wall. Violet points between the object and the wall are noisy points. In Figure 3.42, the surface of the object is created with normal triangulation process whereas in Figure 3.43 the surface is created by line fitting. In the case of line fitting, triangles on the surface are smoother than the triangles in Figure 3.41. Also it is easier to detect the sharp edge in Figure 3.43

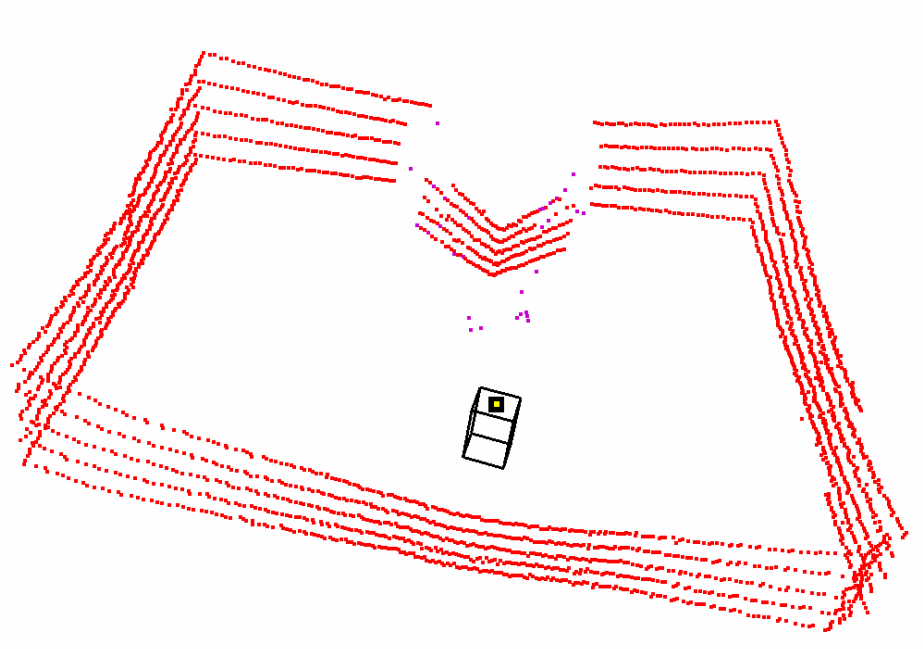

Figure 3.41. An Object with a Sharp Edge

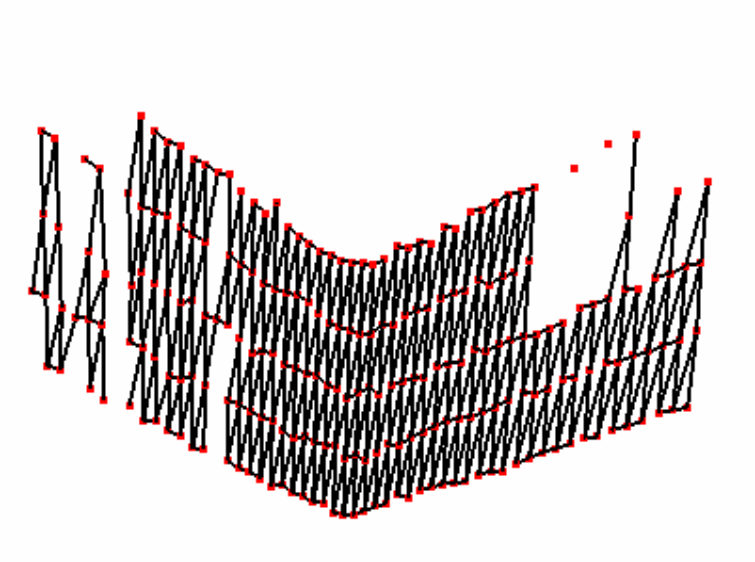

Figure 3.42. An Object Surface before Line Fitting

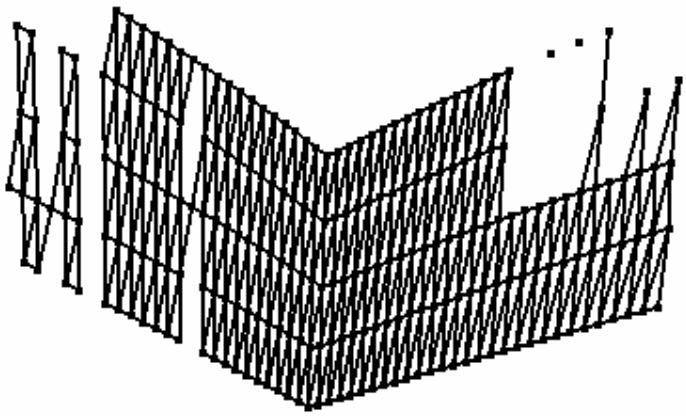

Figure 3.43. An Object Surface after Line Fitting

CHAPTER 4

CASE STUDIES

In this chapter some case studies are done and the output images of these studies are figured out. Parameters that are set during data acquisition for these case studies are; mirror speed of 100 degree/min, scanner speed of 450 degree/sec, sample rate of 15000 samples/sec, buffer size of 800000 samples, callback threshold of 1000 samples and sampling time of 10 sec. In Figure 4.1, a flat object which is placed perpendicularly to the laser rangefinder is scanned. In Figures 4.2 and 4.3, a cardboard is tilted 45° to the back and in Figures 4.4 and 4.5, it is tilted 45° to the front. In Figures 4.6 and 4.7, the same cardboard is tilted 45° to the right. In Figures 4.8 and 4.9 a cylindrical object is scanned. Figure 4.10 is the top view of the first tour of cylindrical object scanning.

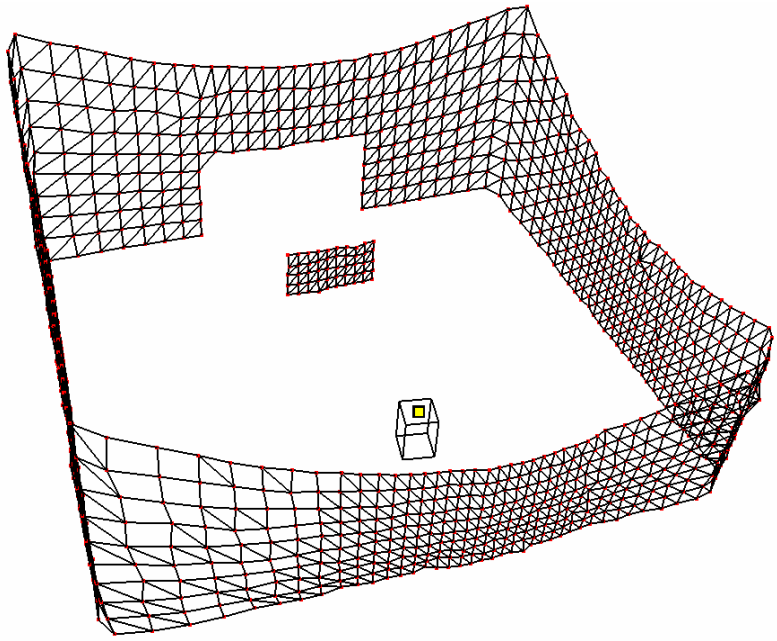

Figure 4.1. Output Image (A Flat Object)

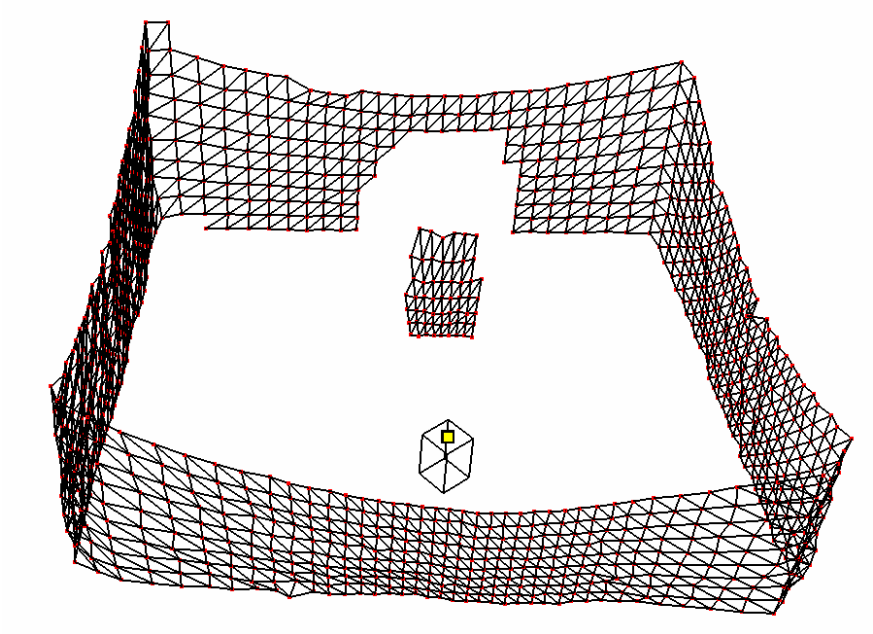

Figure 4.2. Output Image (A Flat Object Tilted back 45°)

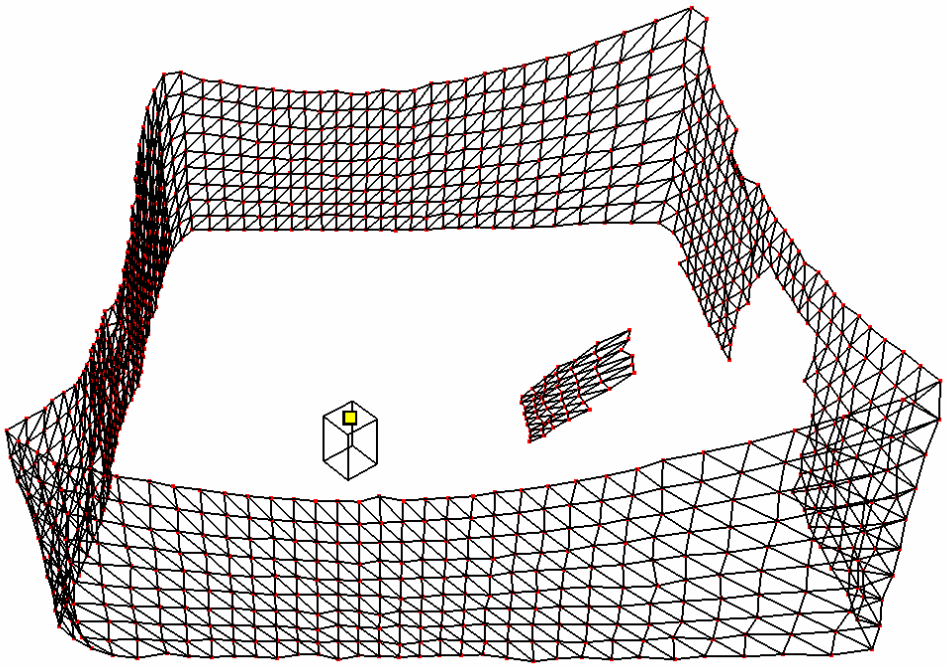

Figure 4.3 .Output Image (A Flat Object Tilted back 45°)

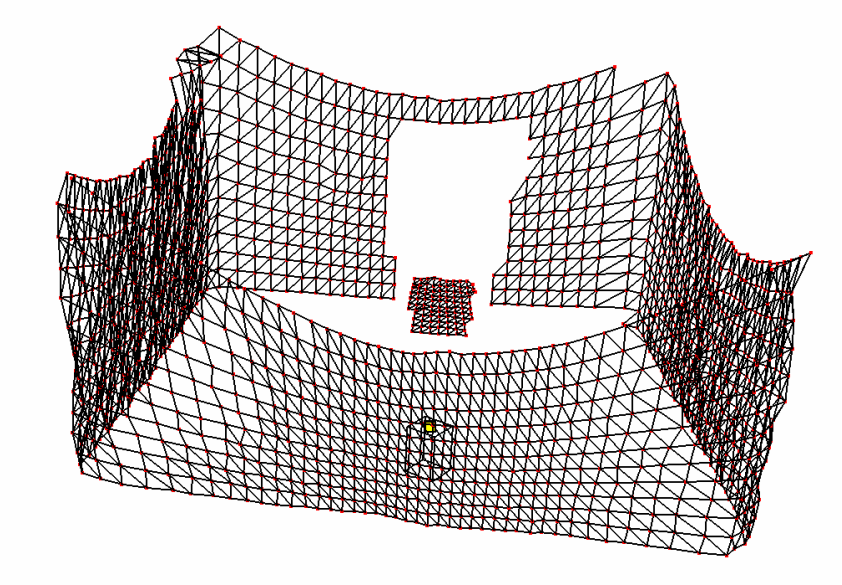

Figure 4.4. Output Image (A Flat Object Tilted front 45°)

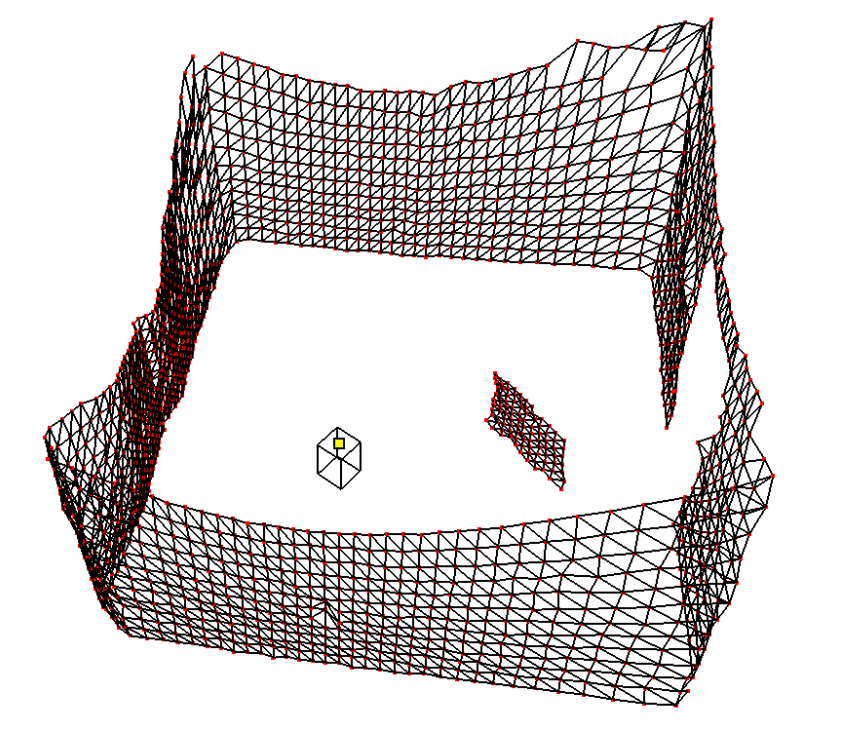

Figure 4.5. Output Image (A Flat Object Tilted front 45°)

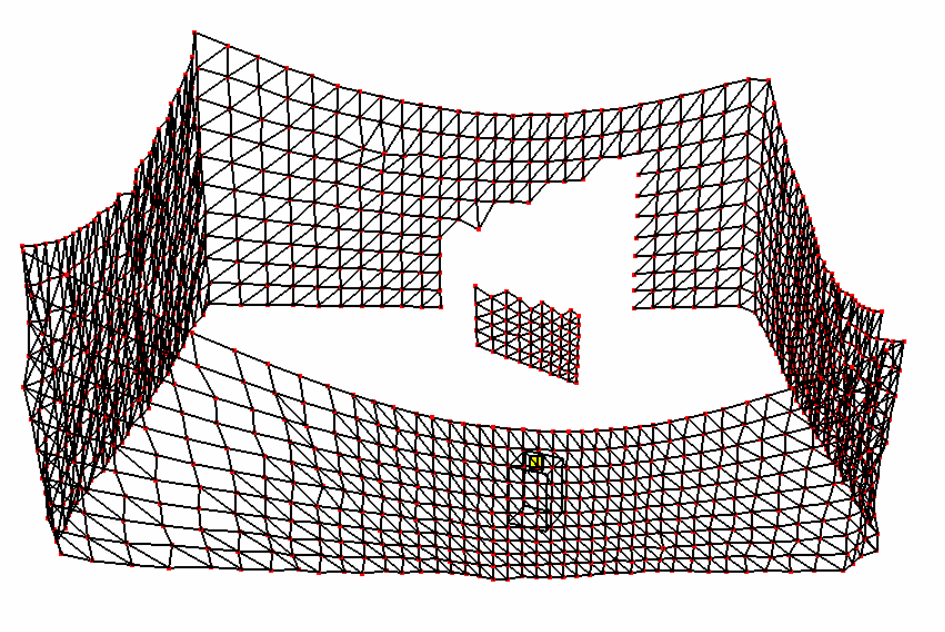

Figure 4.6. Output Image (A Flat Object Tilted right 45°)

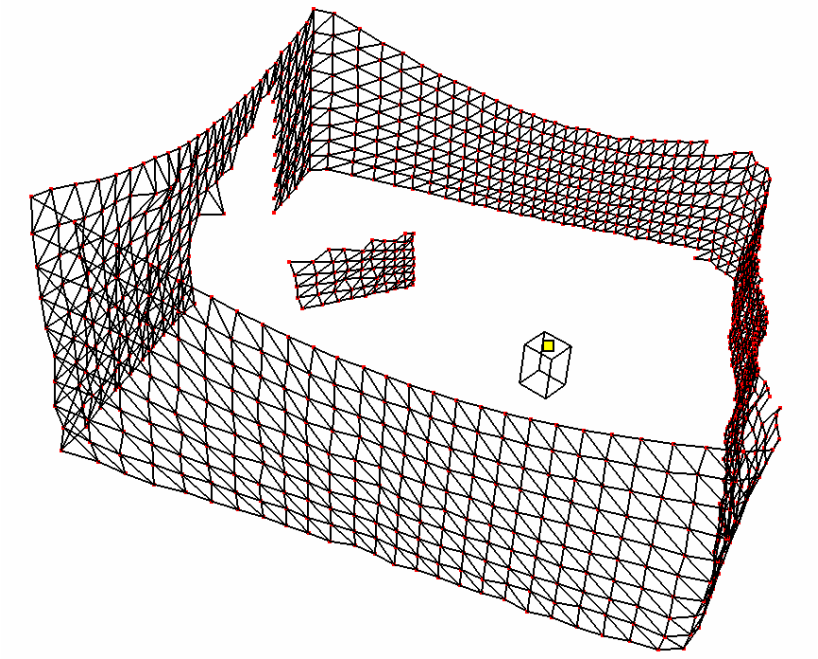

Figure 4.7. Output Image (A Flat Object Tilted right 45°)

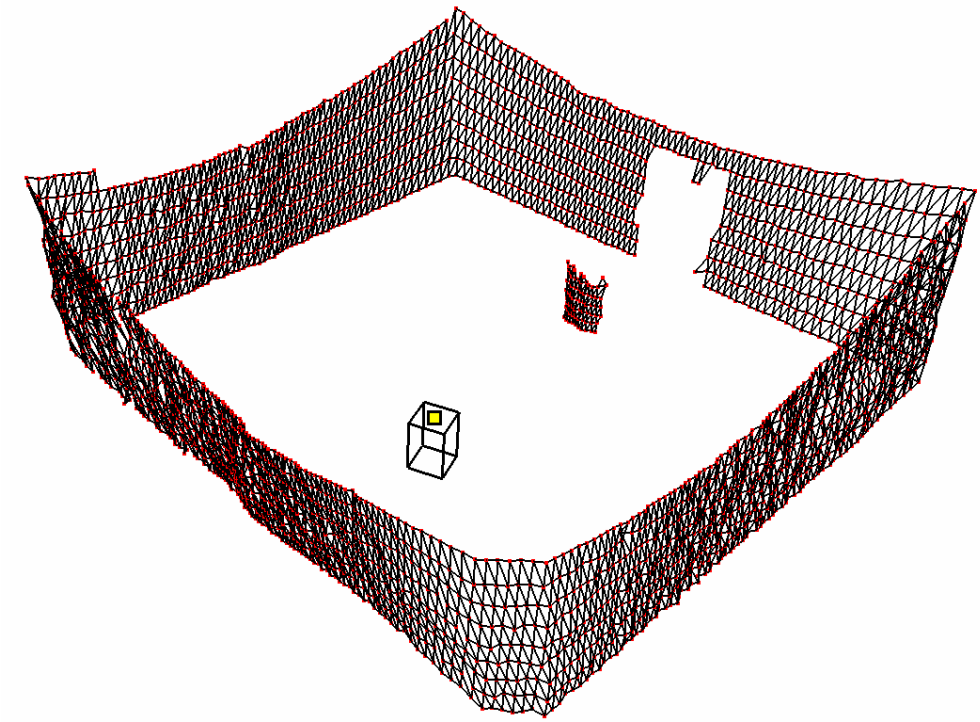

Figure 4.8. Output Image (A Cylindrical Object)

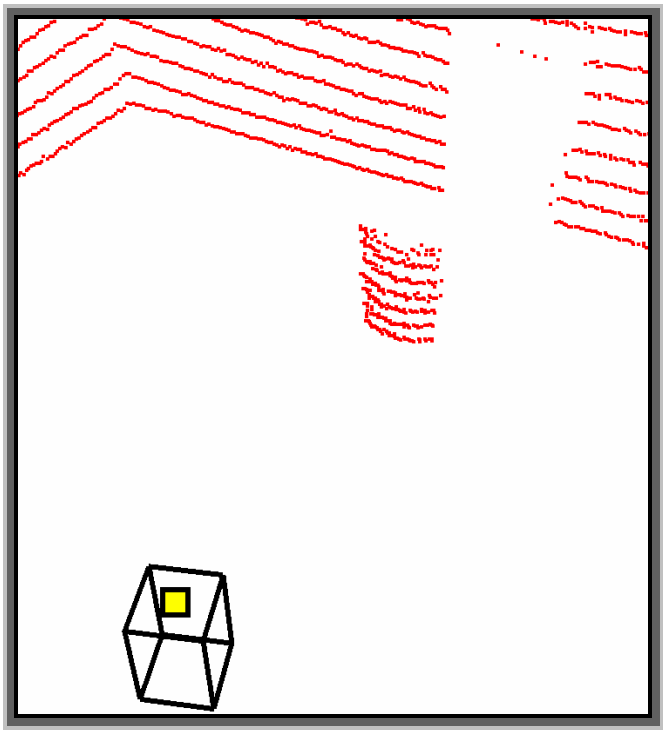

Figure 4.9. Output Image (A Cylindrical Object)

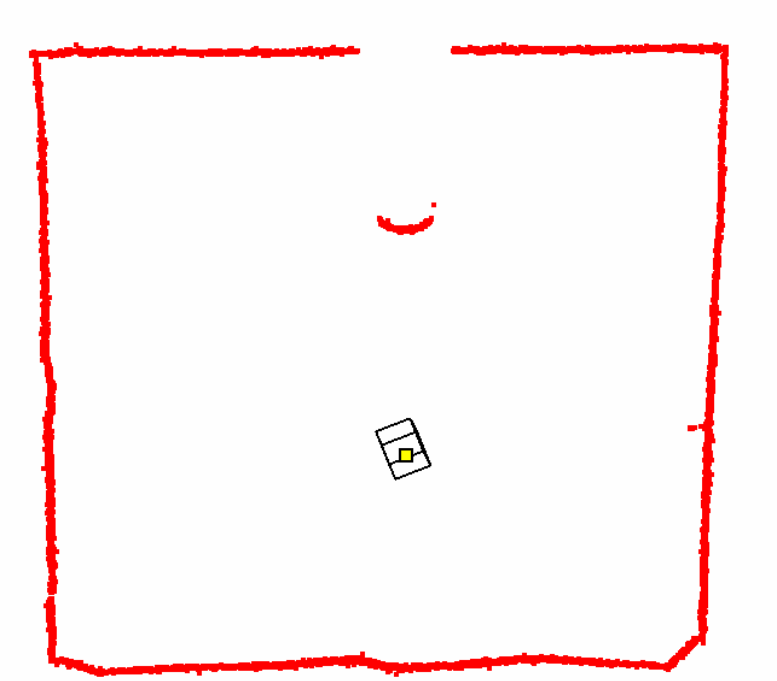

Figure 4.10. Top View of the First Tour of Cylindrical Object Scanning

In Figures 4.11 to 4.17, two cardboards are combined in order to make an edge towards the sensor. In Figures 4.11 and 4.12 angle between the surfaces of the cardboards is 45°, in Figures 4.13 and 4.14 angle is 60° and in Figures 4.15 and 4.16 angle is 90°. In Figure 4.17 two different objects are placed in the same environment.

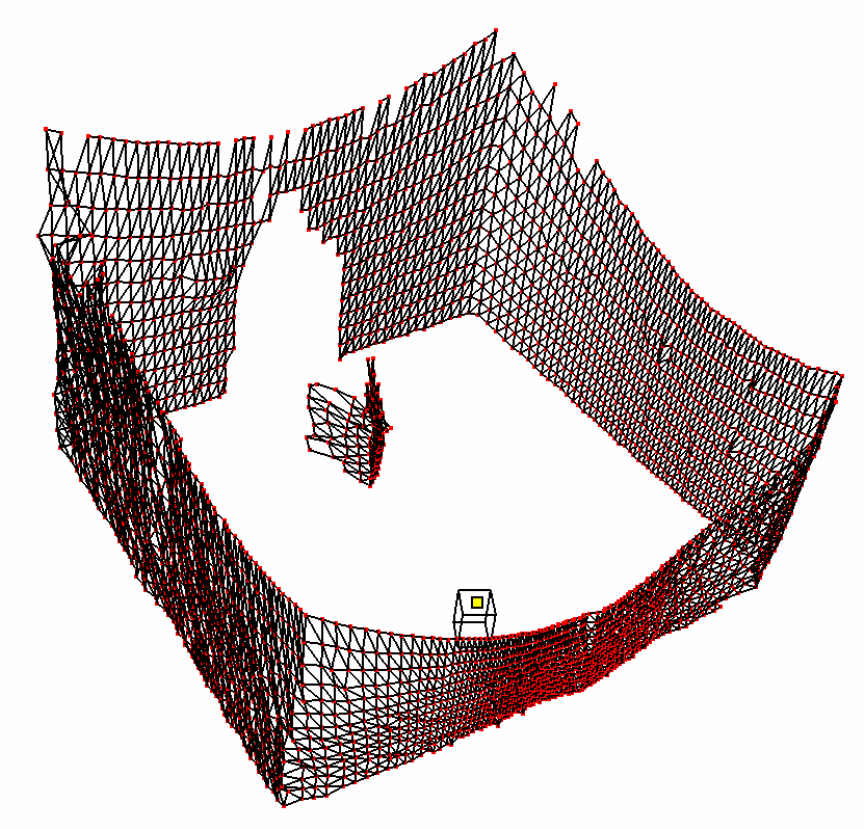

Figure 4.11. Output Image (Angle between Surfaces is 45°)

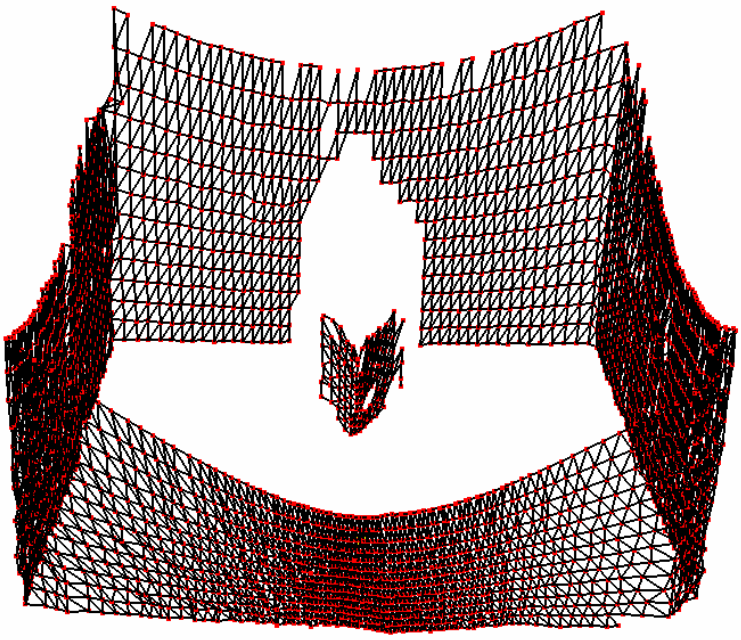

Figure 4.12. Output Image (Angle between Surfaces is 45°)

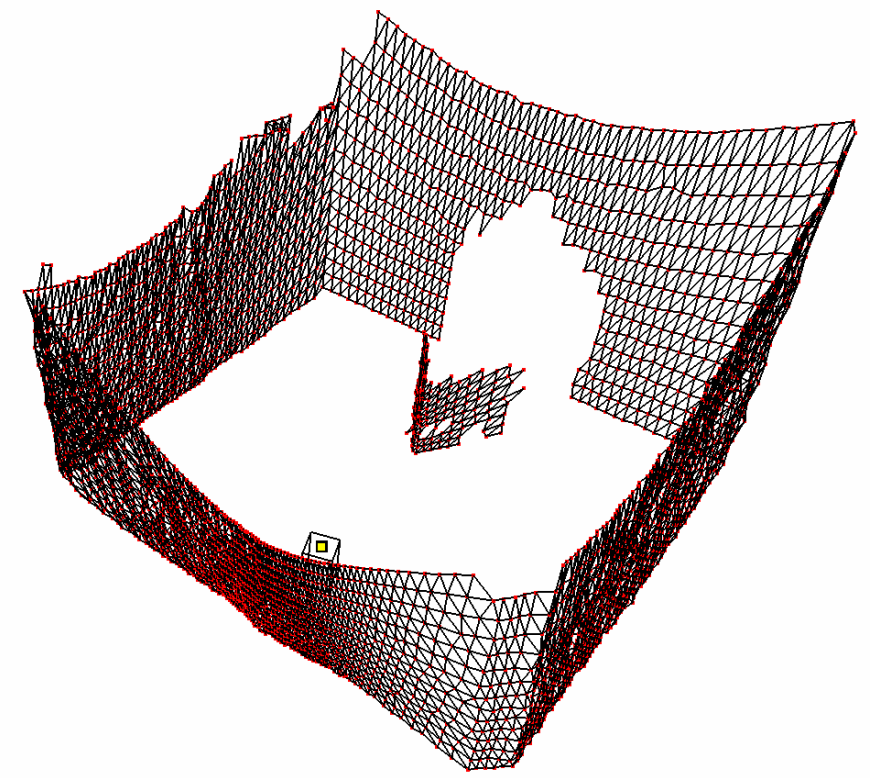

Figure 4.13. Output Image (Angle between Surfaces is 60°)

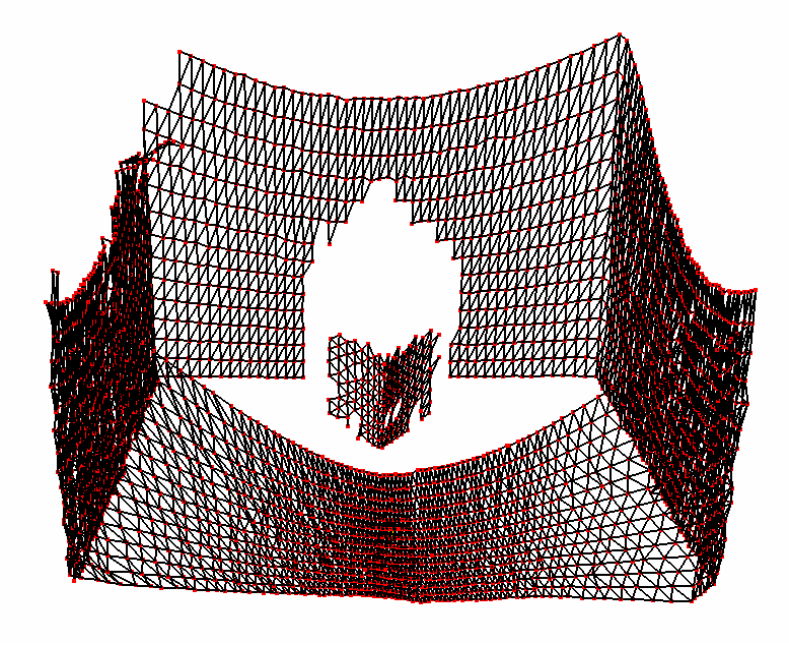

Figure 4.14. Output Image (Angle between Surfaces is 60°)

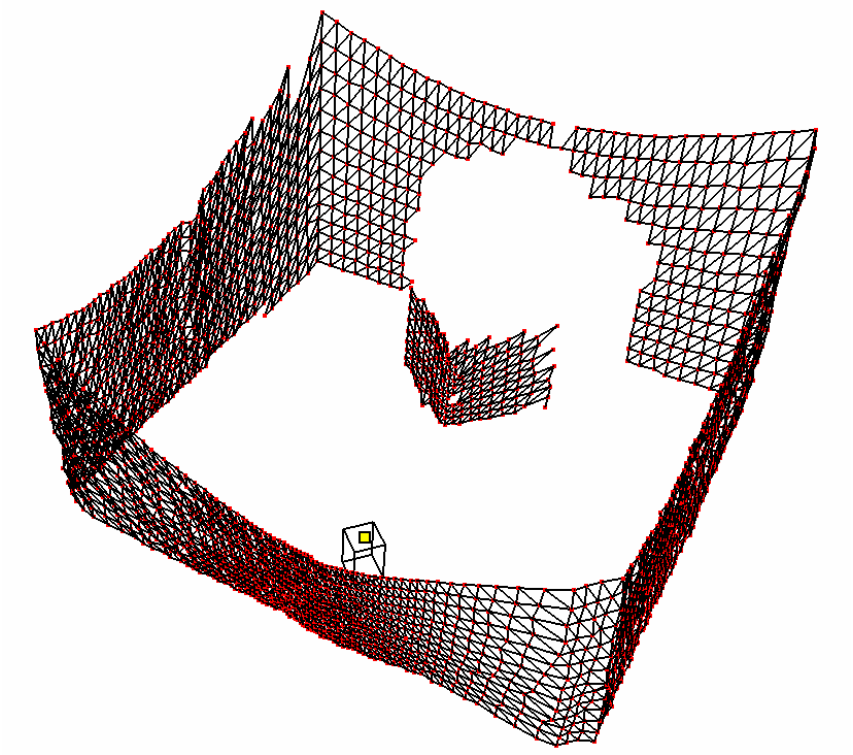

Figure 4.15. Output Image (Angle between Surfaces is 90°)

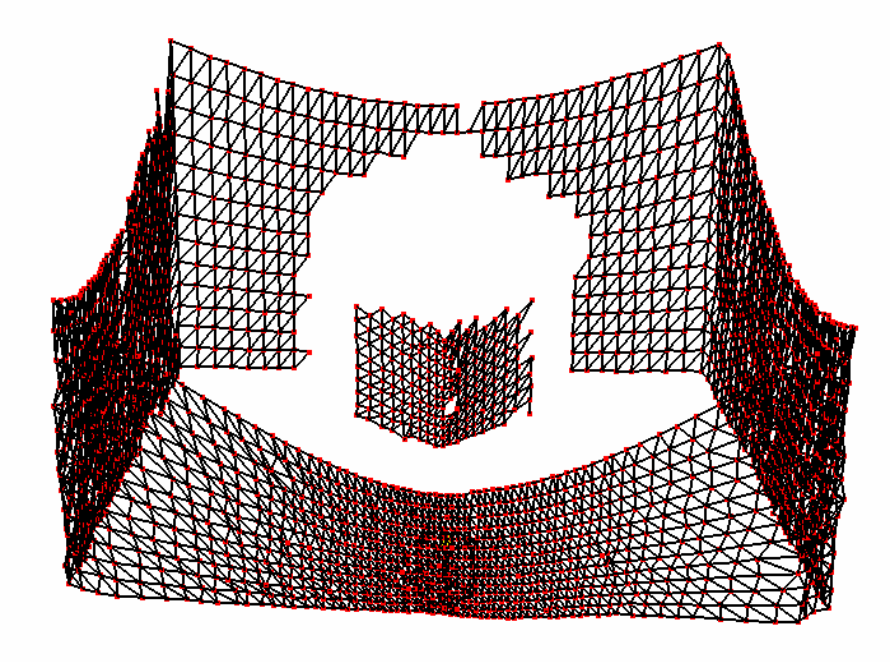

Figure 4.16. Output Image (Angle between Surfaces is 90°)

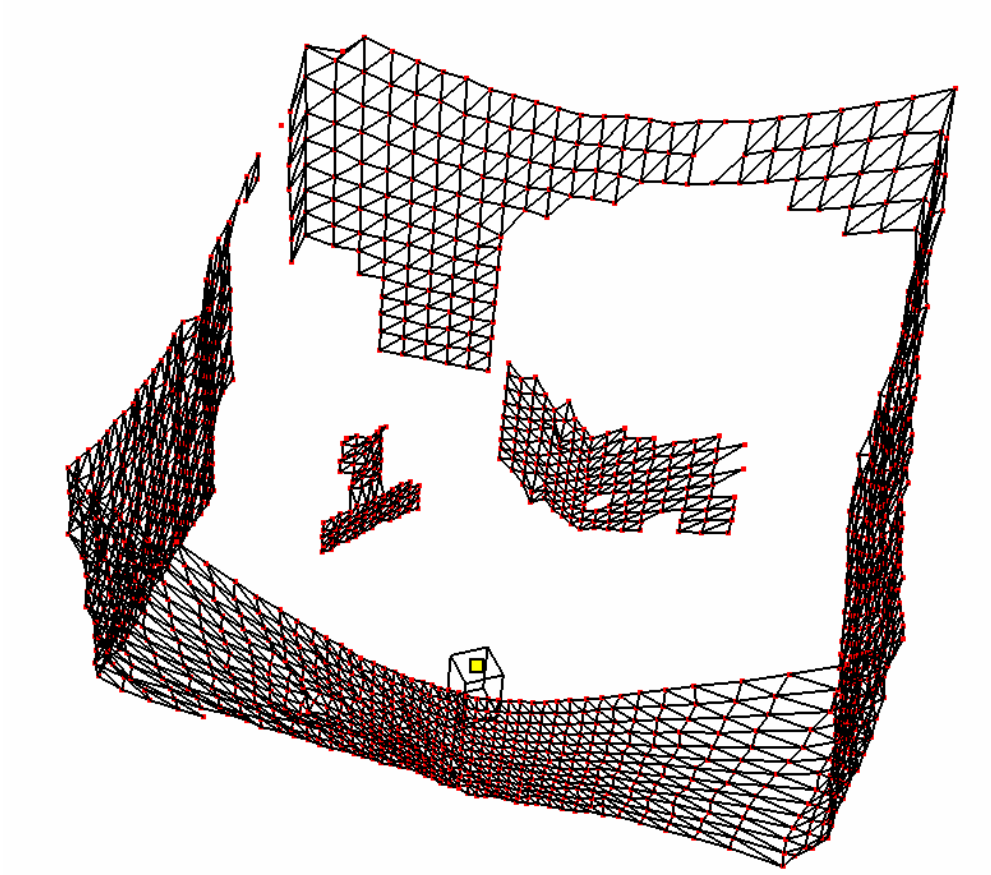

Figure 4.17. Output Image (Two Objects)

It can be seen from the Figures 4.11 to 4.17 that the triangulation algorithm used in the thesis can create meshes of object's sharp edges from an angle of 45° up to 90° successfully. Also two different objects located in the same environment (Figure 4.17) can be sampled and 3D view of these objects can be constructed.

Figures 4.18 and 4.19 are the front and back view of the scanned corridor from two different views. Output images created in these corridor are shown in Figures 4.20 and 4.21. To protect the user from the laser beams, a cardboard seen in Figure 18 is located in the environment. This cardboard can be seen as a flat surface in Figures 4.20 and 4.21. In the environment, there are three doors which are coated with shiny paint. One is at the back of the laser as seen in Figure 4.19 and others are at the right and left sides of the protective cardboard. Images of these doors cannot be obtained but instead, noisy data as in the form of point cloud are obtained.

Figure 4.18. Front View of the Corridor

Figure 4.19. Back View of the Corridor

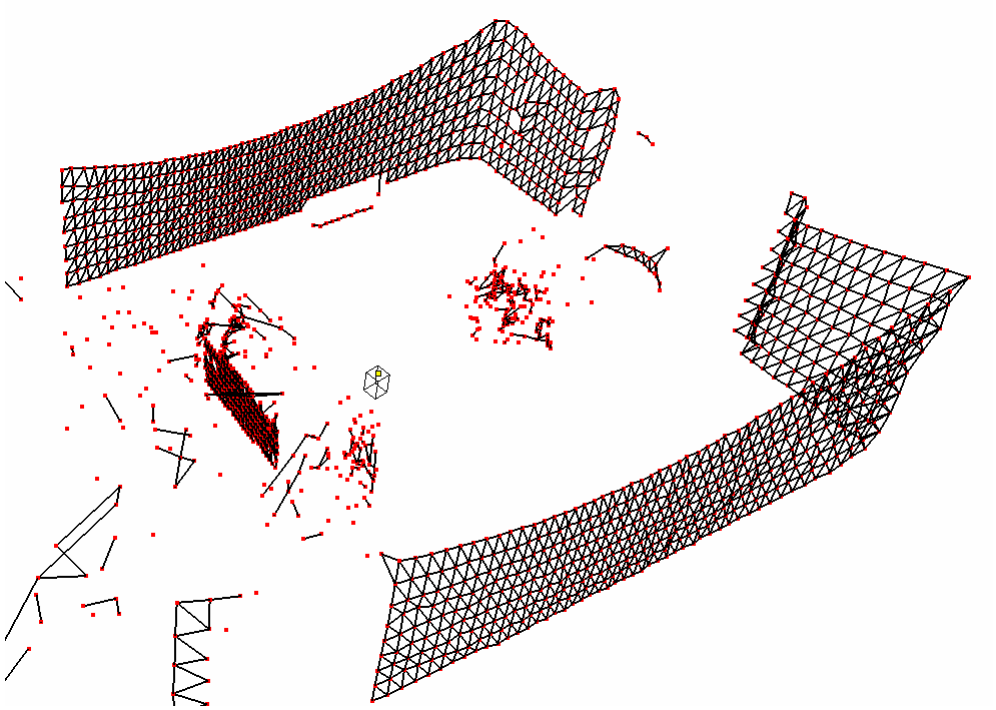

Figure 4.20. Output Image (Corridor)

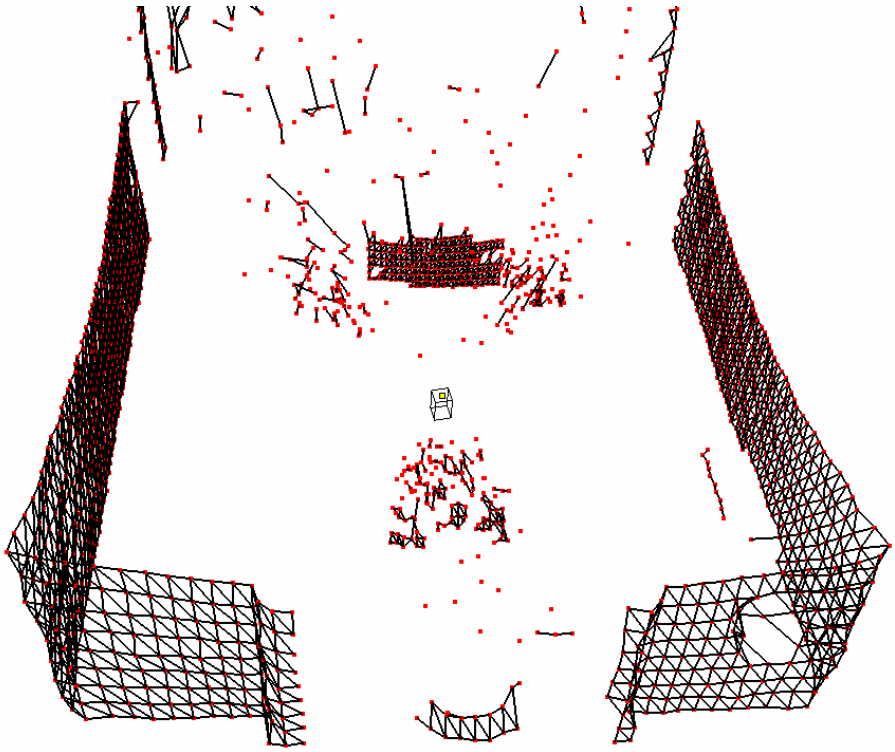

Figure 4.21. Output Image (Corridor)

Figures 4.22 and 4.23 are the front and back sides of the scanned corridor from two different views. In this case, a big cardboard is placed at different distances. In Figures 4.24 and 4.25 distance is 5 m, in Figures 4.26 and 4.27 distance is 7.5 m and in Figure 4.28 distance is 10 m.

Figure 4.22. Front View of the Corridor

Figure 4.23. Back View of the Corridor

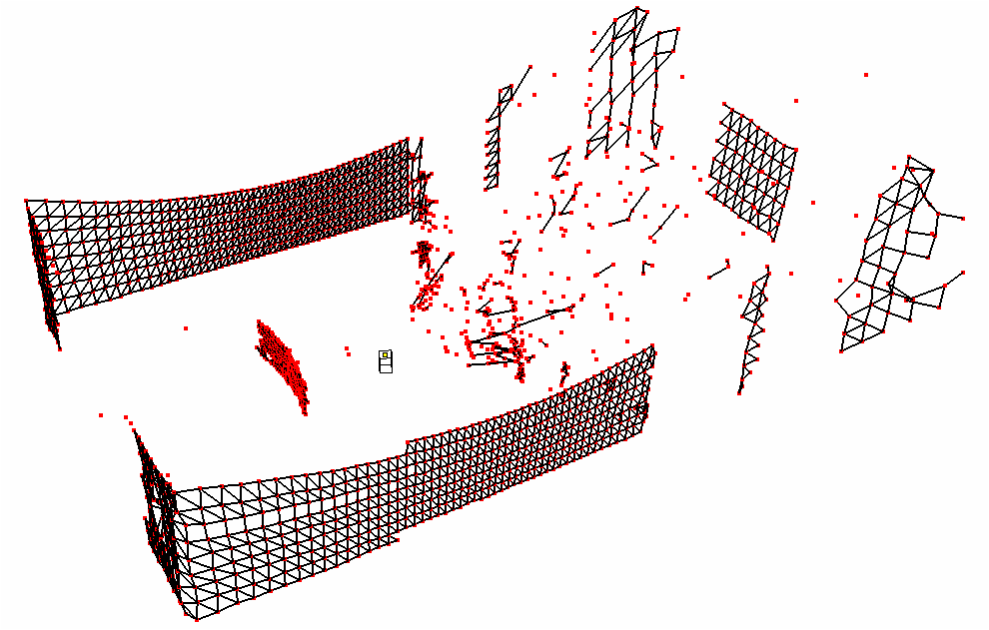

Figure 4.24. Output Image (Big Cardboard is at 5 m)

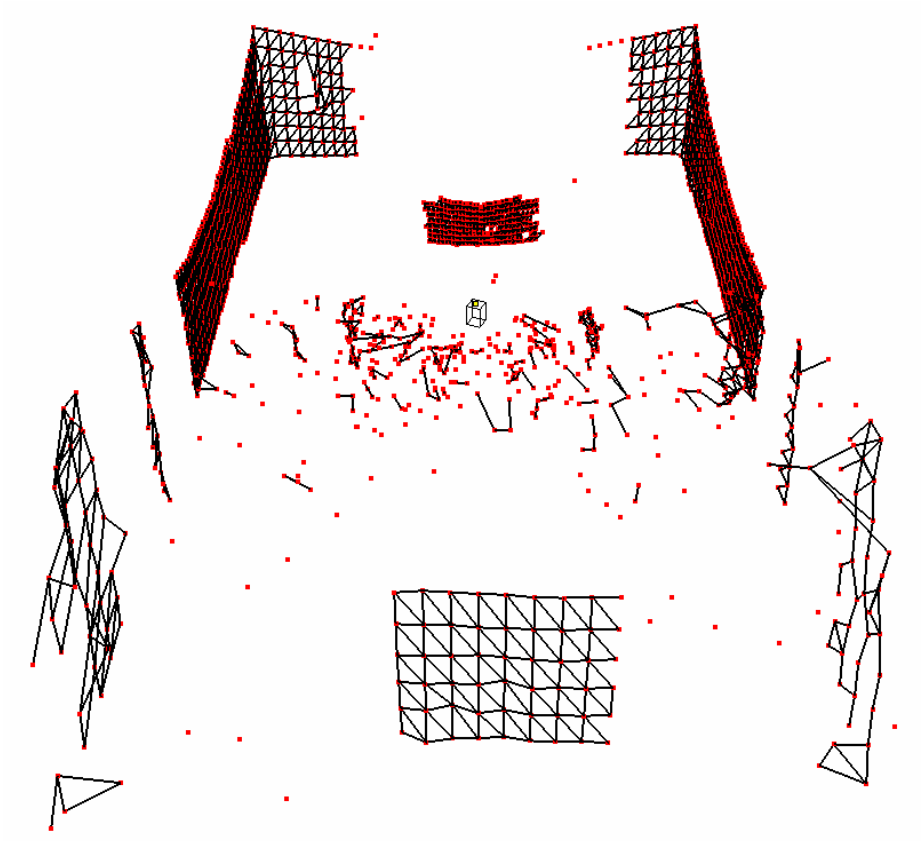

Figure 4.25. Output Image (Big Cardboard is at 5 m)

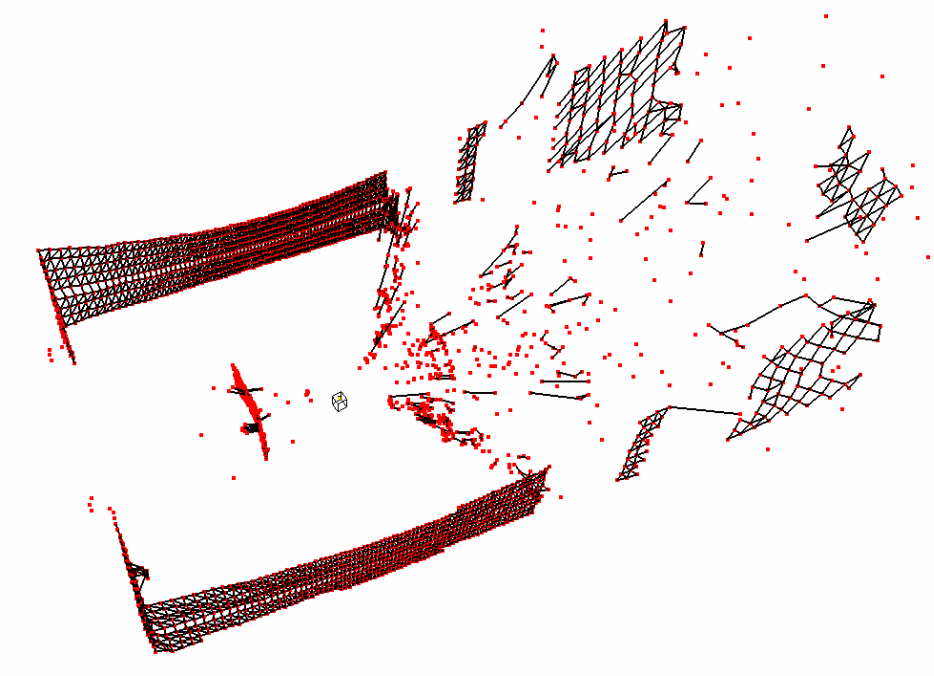

Figure 4.26. Output Image (Big Cardboard is at 7.5 m)

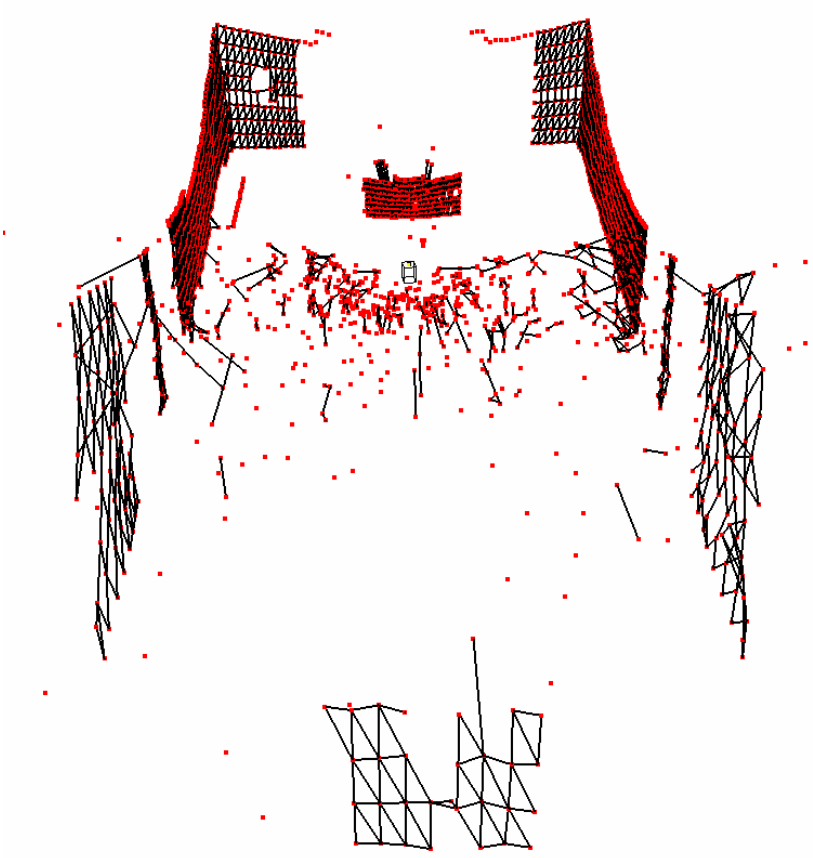

Figure 4.27. Output Image (Big Cardboard is at 7.5 m)

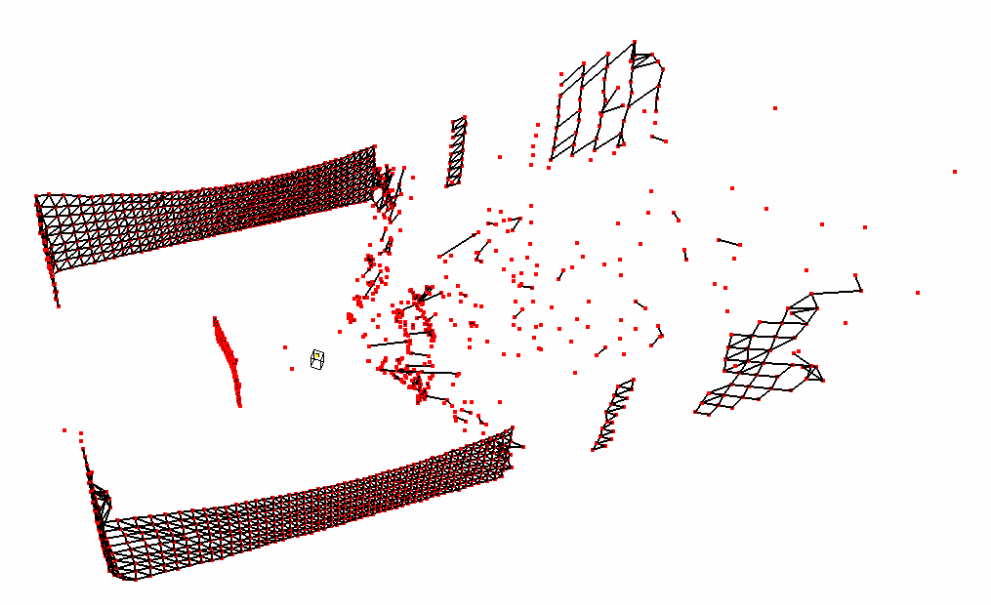

Figure 4.28. Output Image (Big Cardboard is at 10 m)

When the big cardboard is placed 10 m away from the sensor and the scanned points are shown in a stepsize of 50 points in the processing program, the cardboard can not be recognized in the view. If all points are shown one by one (stepsize of 1 point) in the processing program, then cardboard can be recognized.

Three scanning cases are done with 8 m, 8.5 m and 9 m in order to find the closest range that the object can be recognized. In Figure 4.29, distance form cardboard to the sensor is 9 m, in Figures 4.30 and 4.31 distance is 8 m and in Figures 4.32 and 4.33 distance is 8.5 m.

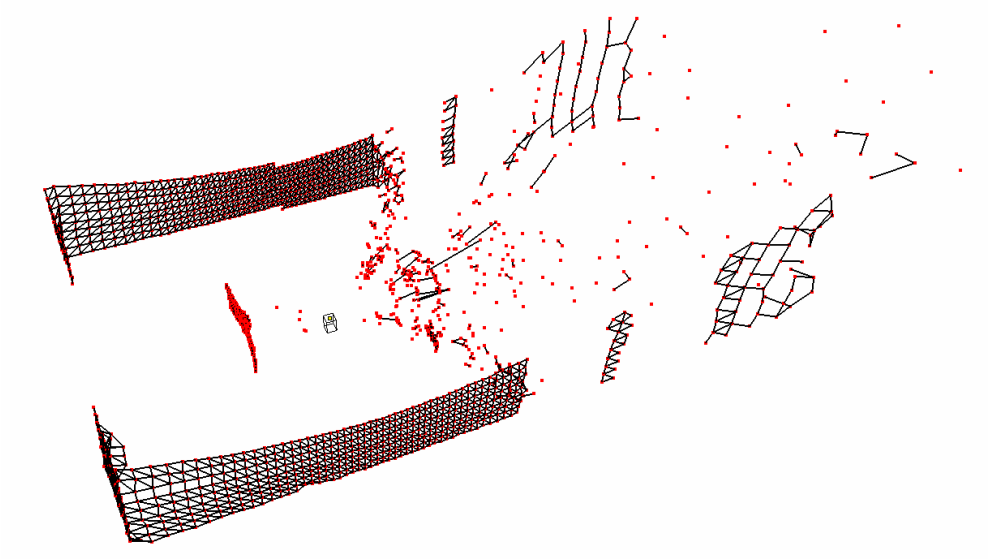

Figure 4.29. Output Image (Big Cardboard is at 9 m)

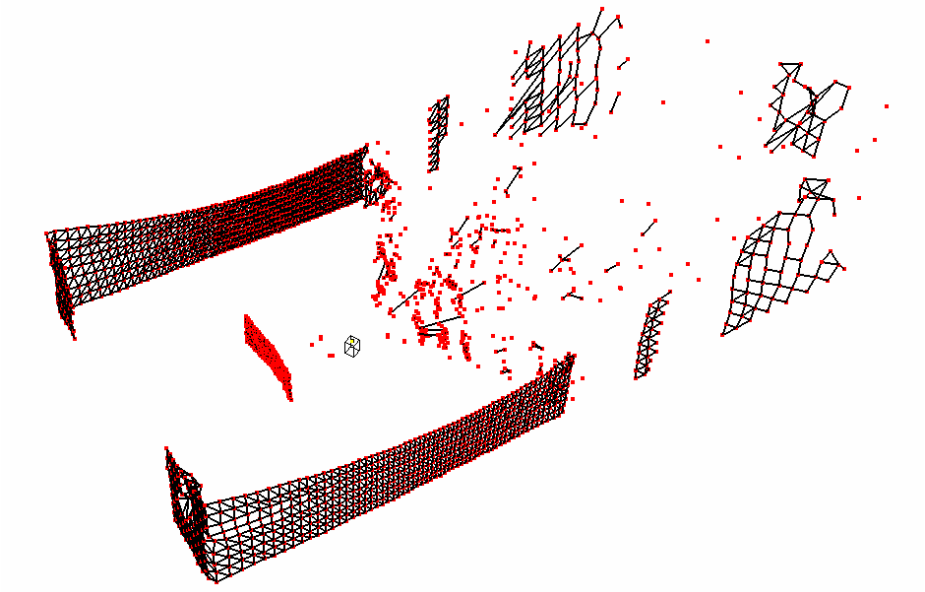

Figure 4.30. Output Image (Big Cardboard is at 8 m)

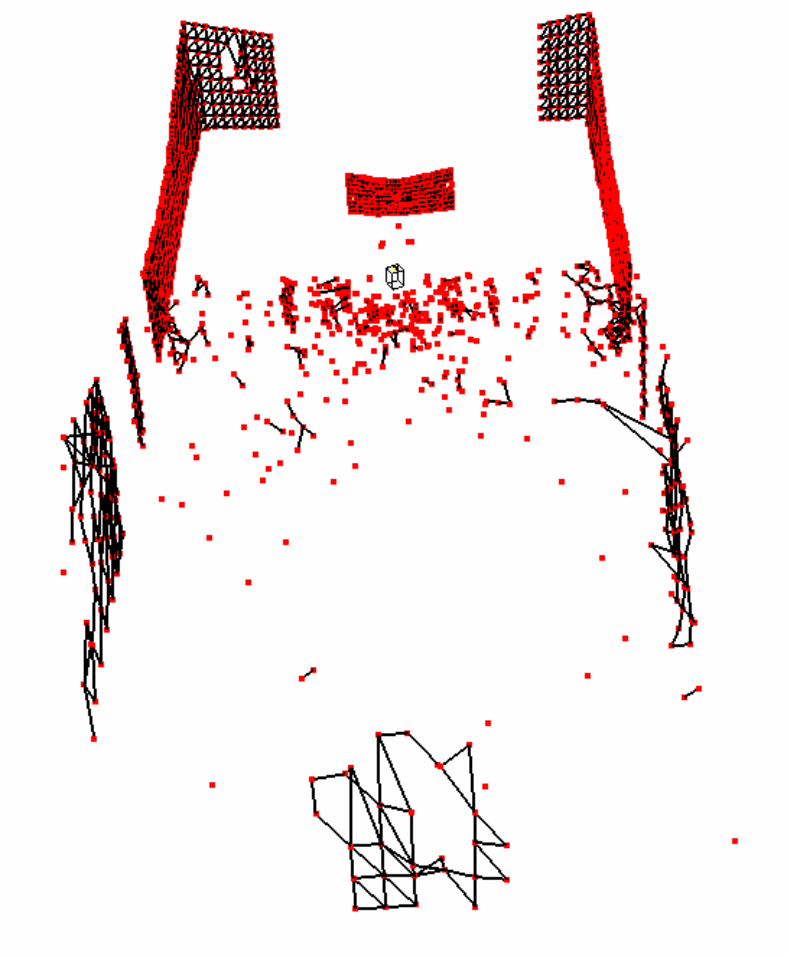

Figure 4.31. Output Image(Big Cardboard is at 8 m)

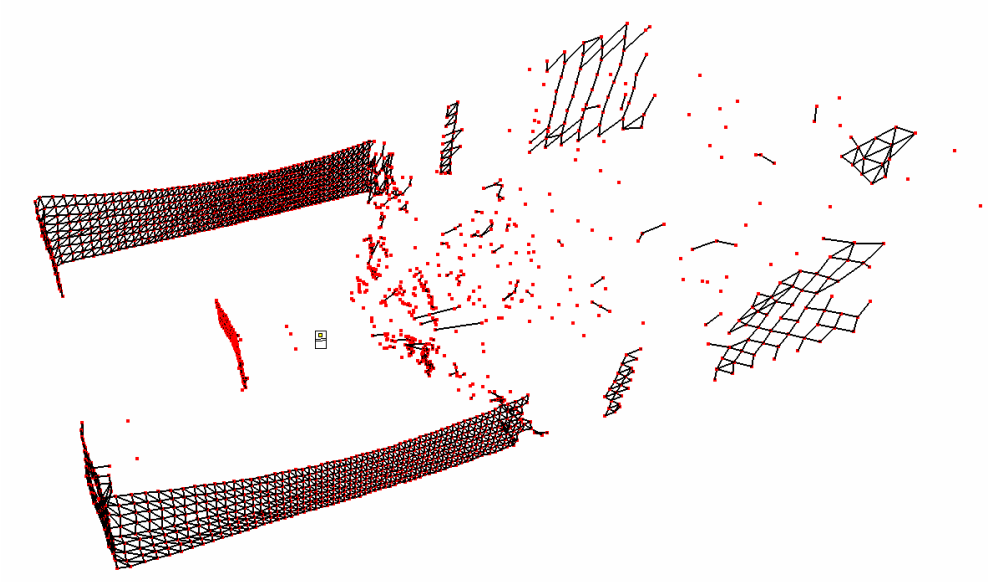

Figure 4.32. Output Image (Big Cardboard is at 8.5m)

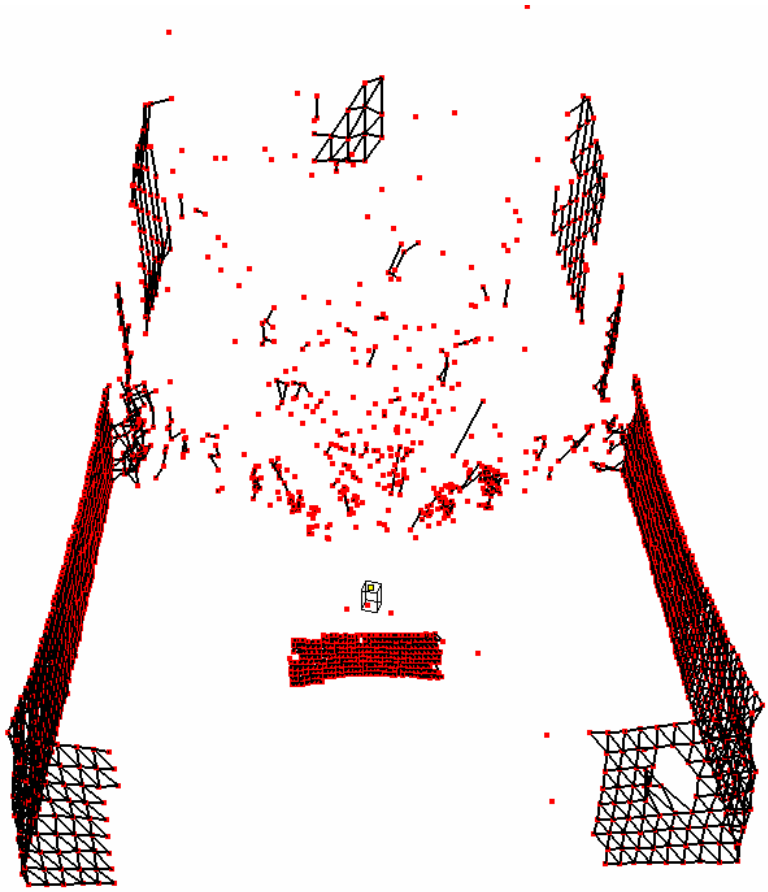

Figure 4.33. Output Image (Big Cardboard is at 8.5 m)

After scanning the big cardboard in different distances, it is seen that for distances greater than 8.5 m and a stepsize of 50 points in the processing program, an image for the cardboard cannot be recognized.

CHAPTER 5

CONCLUSION

5.1. Conclusions

In the previous study [15], data acquisition program was run under DOS and 2D surface reconstruction program under Windows. However, in this study, a new system is set up and both data acquisition and processing software are developed to collect and process data within just one application, running under Windows.

The developed and implemented data acquisition and processing interfaces in the thesis are capable of creating 3D image map and obtaining the view of scanned environment in a short time with high accuracy and time efficiency.

The data acquisition algorithm used in the study can acquire data in helical way. If optimum sample rate for each scanner speed is specified correctly and enough buffer size for the sensor data is determined, then data acquisition program runs efficiently. Afterwards, by using the basic triangulation algorithm for consecutive points and threshold conditions for noisy data detection, segmentation and line fitting, 3D views of basic objects in simple environments are obtained. As explained in the previous section, if more powerful data processing algorithms and threshold conditions are developed, then more complex objects can be sampled in complex environments.

In the thesis, a simple segmentation is applied to the acquired data in order to detect objects basically and pass to the line fitting steps. For objects of either flat surface or edged surfaces, successful results are obtained at the end of the line fitting.

Results of the case studies show that tilted objects to the front, back and right with 45° and cylindrical objects can be sampled and output images for these cases can be obtained successfully. Also for a big cardboard in different distances, it is seen that for distances greater than 8.5 m and a stepsize of 50 points in the processing program, an image for the cardboard cannot be recognized.

5.2. Recommendations For Future Work

During the thesis, it is assumed that scanner and mirror rotate at constant speed but in real, there can be deviations in their speeds because there are not servo controls on the DC Motor A (for scanner speed) and DC Motor E (for mirror speed). Servo controls to the both of the motors can be included and control algorithms for these systems can be developed.

Since a single range view can only contain points in the scene that are visible to the data acquisition device, environment can be scanned from at least two different points and multiple range views can be merged into a single point set to obtain a complete 3D model. By this way, invisible parts in the environment for one view can be seen in the final 3D model.

Data can be acquired in different ways. In the thesis, helical data acquisition is used. There may be two other ways for this process. Firstly, data can be acquired in different planes. In this method, after each turn of the scanner, mirror height is increased in the amount of desired vertical resolution so all environment can be scanned. In the second method, data can be acquired by sweeping the environment in vertical direction with a preset angle value. For instance, all environment can be scanned in six steps where each step scans a medium of 60°.

The whole data acquisition process can be speeded up by using a more efficient hardware system and developing the interface. For example, using a higher capacity computer may result in shorter sampling and processing times so in the same period, more accurate results can be obtained via higher vertical resolutions.

Data processing interface can be developed in order to work under complex environments. Distant and close objects can be viewed in the same resolutions by determining the stepsize in an adaptive way. Also, triangulation threshold conditions can be improved and noisy points can be eliminated more effectively; projections of the noisy points can be taken and new meshes can be created with those projected points.

Segmentation process can be handled with developed threshold conditions to detect objects in complex environments. Also, after segmentation, object recognition process can be applied to objects in order to classify them in basic types like sphere, cube, prism, etc.

REFERENCES

[1] Acuity, http://www.acuityresearch.com, Last access date: 15.08.2006

[2] Wikipedia, http://en.wikipedia.org/wiki/Laser_range-finder, Last access date: 15.08.2006

[3] X20 Laser Rangefinder, http://www.x20.org/nightvision/x20rangefinder.htm, Last access date: 15.08.2006

[4] Arkin R., Rossignac J., Kaess M., Compact Map Building, College of Computing, Georgia Institute of Technology, Atlanta, 2001

[5] Huber D., Carmichael O., Hebert M., 3-D Map Reconstruction from Range Data, IEEE International Conference on Robotics & Automation, San Francisco, CA, Volume 1, Issue , pp. 891 - 897, 2000

[6] Sequeira V., Gonçalves J.G.M., Ribeiro M.I., 3D Environment Modelling Using Laser Range Sensing, Robotics and Autonomous Systems, Volume16, Issue 1, pp. 81-91, 1995

[7] Park I.K. and Lee S.U., Geometric Modeling from Scattered 3-D Range Data, Image Processing, Volume 2, Issue , pp. 712 – 715, 1997

[8] Klein K. and Sequeira V., The View-Cube: An Efficient Method of View Planning for 3D Modelling from Range Data, Applications of Computer Vision, Fifth IEEE Workshopon, pp. 186-191, 2000

[9] Goshtasby A. A., Three-Dimensional Model Construction from Multiview Range Images: Survey With New Results, Pattern Recognition, Volume 31(11), pp. 1705-1714, 1998

[10] Liu Y., Emery R., Chakrabarti D., Burgard W., Thrun S., Using EM to Learn 3D Models of Indoor Environments with Mobile Robots, Proceedings of the Eighteenth International Conference on Machine Learning, pp. 329 – 336, 2001

[11] The Basics of Triangulation Sensors, http://archives.sensorsmag.com/articles/ 0598/ tri0598/, Last access date: 15.08.2006

[12] Daniel Nordin, Optical frequency Modulated Continuos Wave Range and Velocity Measurements, Ph. D. Thesis, Department of Computer Science and Electrical Engineering, Lulea University of Technology, Lulea, Sweden, 2004

[13] Rangefinding Devices, http://show.docjava.com:8086/scanners/rangefin.htm, Last access date: 15.08.2006

[14] Woodbury, N., Brubacher, M., Woodbury, J.R., Noninvasive Tank Gauging with Frequency-Modulated Laser Ranging, Sensors, pp. 27-31, September 1993.

[15] Fidan, U., Design and Construction of a 3D Laser Rangefinder for Indoor Applications, M.Sc. Thesis, Mechanical Engineering Department, Middle East Technical University, Ankara, Turkey, 2001.

[16] Myers A., Introductory Literature Review Surface Reconstruction from Three Dimensional Range Data, April 1999

[17] Besl P. and McKay N., A Method for Registration of 3-D Shapes, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 14, no 2, pp 239 – 256, 1992

[18] Besl P. and Jain R., Segmentation Through Variable-Order Surface Fitting, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 10, no 2, pp 167 - 192, 1988

[19] Premebida C. and Nunes U., Segmentation and Geometric Primitives Extraction from 2D Laser Range Data for Mobile Robot Applications, Robótica 2005 – Actas do Encontro Científico Coimbra, pp17 – 25, April 2005

[20] Dietmayer, K.C.J., J. Sparbert and D. Streller, Model based Object Classification and Object Tracking in Traffic scenes from Range Images, Proceedings of IV IEEE Intelligent Vehicles Symposium, Tokyo, 2001

[21] Santos S., Faria J.E., Soares F., Araujo R. and Nunes U., Tracking of Multi-Obstacles with Laser Range Data for Autonomous Vehicles, 3rd National Festival of Robotics Scientific Meeting (ROBOTICA), pp. 59-65, Lisbon, Portugal, 2003

[22] Lee K.J., Reactive navigation for an outdoor autonomous vehicle, Master Thesis, University of Sydney, Department of Mechanical and Mechatronic Engineering, 2001

[23] Borges G.A. and. Aldon M.J., Line Extraction in 2D Range Images for Mobile Robotics, Journal of Intelligent & Robotic Systems, v. 40, n. 3, pp. 267-297, 2004

[24] Bolle R. M. and Vemuri B.C., On Three-Dimensional Surface Reconstruction Methods, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 13, no 1, pp 1 - 13, January 1991

[25] Zhang Z., Parameter Estimation Techniques: A Tutorial with Application to Conic Fitting, Image and Vision Computing, vol 15, pp 59-76, 1997

[26] Oblonsek C. And Guid N., A Fast Surface-Based Procedure for Object Reconstruction from 3-D Scattered Points, Computer Vision and Image Understanding, vol 69, no 2, pp. 185 – 195, February 1998

[27] Fang T. P. and Piegl L. A., Algorithm for Delaunay triangulation and Convex-Hull Computation Using Sparse Matrix, Computer-Aided Design 24, pp. 425–436, 1992

[28] Fang T. P. and Piegl L. A., Delaunay Triangulation Using a Uniform Grid, IEEE Computer Graphics Applications 13(3) , pp. 36–47, 1993

[29] Choi B.K., Shin H.Y., Yoon Y.I. and Lee J.W., Triangulation of Scattered Data In 3D Space, Computer-Aided Design 20, pp. 239–248, 1988

[30]Schroeder W.J., Zarge J.A. and Lorensen W.E., Decimation of Triangle Meshes, Computer Graphics 26(2), pp. 65–70, 1992.

[31] Yanalak M. , Yüzey Modellemede Üçgenleme Yöntemleri, Harita Dergisi, Volume 126, July 2001

[32] Delaunay Triangulation In Two and Three Dimensions, http://www.gris.uni-tuebingen.de/gris/proj/dt/dteng.html, Last access date: 15.08.2006

[33] Boissonnat J.D., Geiger B., Three Dimensional Reconstruction Of Complex Shapes Based On The Delaunay Triangılation, Institut National de Recherche en Informatique et en Automatique, April 1992

[34] Lawson C.L., Generation of Triangular Grid With Application to Contour Plotting: California Institute of Technology, Technical Memorandum, No 229, 1972

[35] Koo B.K., Choi Y.K., Chu C.W., Kim J.C., and Choi B.T., Shrink-Wrapped Boundary Face Algorithm for Mesh Reconstruction from Unorganized Points, ETRI Journal, Volume 27, Number 2, pp. 235-238, April 2005

[36] Kobbelt L., Vorsatz J., Labsik U. and Seidel H. A Shrink Wrapping Approach to Remeshing Polygonal Surfaces, Proc. Eurographics '99, vol. 18, no. 3, pp. 119-129, September 1999

[37] Jeong W. and Kim C., Direct Reconstruction of Displaced Subdivision Surface from Unorganized Points, Graphical Models, vol. 64, issue 2, pp. 78-93, March 2002

[38] Taubin G., A Signal Processing Approach to Fair Surface Design, SIGGRAPH '95, pp. 351-358, August 1995

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

[40] Crandun Technologies Inv., http://www.crandun.com/ ,Last access date: 15.08.2006

APPENDIX A

ACCURANGE 4000 LASER SENSOR

A.1. General Description

The AccuRange 4000 is a laser diode based distance measurement sensor for ranges up to 50 feet, with 0.1 inch accuracy. Type AccuRange 4000-LIR uses near infrared light (780 nm wavelength) and it is a Class IIIb laser product, available in power levels of 8 mW (standard), or up to 20 mW with the High Power Laser option. It also has lower measurement noise and greater sensitivity and maximum range.

The AccuRange 4000 operates by emitting a modulated, collimated beam of laser light and converting the distance to the target surface to an RS-232 or RS- 422/485 output. The range may be read via the serial cable as digital data, or from the optional analog current loop output. A second cable supplies power to the AccuRange 4000 and brings out other signals which include reflected signal strength, sensor temperature, and background light level. A photo of the AccuRange 4000 is shown in Figure A.1.

Figure A.1. AccuRange 4000 Laser Sensor

Key features of the AccuRange 4000 laser sensor are as follows:

- Zero to 50 feet operating range for most surfaces.
- 0.1 inch (2.5mm) accuracy, 0.02 inch (0.5mm) short-term repeatability.
- Optional RS-485/422, 4-20 mA current loop and pulse width outputs. RS-232 serial output standard.
- Reflected signal amplitude output for grayscale images.
- Fast response time: 50 KHz maximum sample rate.
- Lightweight, compact, low power design.
- Tightly collimated output beam for small spot size.
- Three output beam configurations are available: Visible, infrared, or eye safe infrared for reflective tape targets.
- Ideally suited to level and position measurement, machine vision, autonomous vehicle navigation, and 3D imaging applications.

The laser beam is emitted from the center of the front panel, and the central 2.5 inch diameter of the front panel is a collector for return light. The bottom of the sensor has 4 blind holes which are threaded for 6-32 bolts for mounting the sensor. The back of the sensor has a switch for configuration and reset, LED, and two 6 foot cables. The first is for RS-232 communication, the second contains power and analog signals. The weight of the AccuRange 4000 is 22 ounces. Mechanical dimensions of the AccuRange 4000 is shown in FigureA.2.

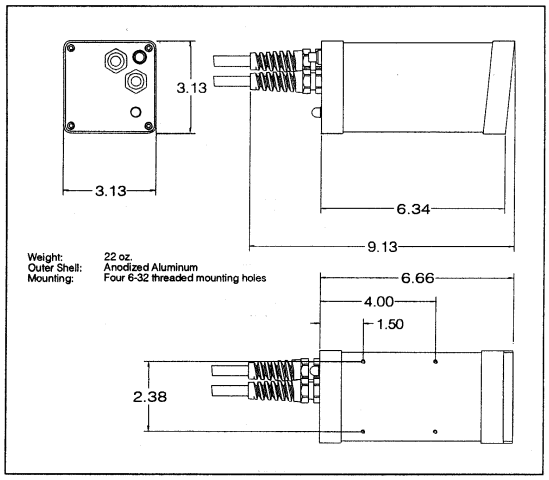

Figure A.2. Mechanical Dimensions of the AccuRange 4000

A.2. Operation Guidelines

- Use protective eyewear whenever there is a risk of being exposed to the output beam of the 4000-LIR.
- Do not point the sensor at any person, particularly a person's eyes or face.
- Do not attempt to disassemble the sensor.
- Do not operate the sensor in areas where the sensor case is exposed to direct sunlight for more than a minute or where the air temperature is more than 45°C (113°F).
- Do not point the sensor at the sun or other intense heat sources.
- Avoid excessive vibration and shocks.
- Do not scratch the front face of the sensor, particularly in the central area.

A.3. Signal and Power Interface

The AccuRange 4000 has 2 cables. The cable with the 9 pin connector is a standard RS-232 serial connection to the sensor which can be connected to an IBM-PC compatible serial port (Table A.1). The other cable is an 8 pin power/signal cable (Table A.2). If the 4000 is ordered with a power supply, the power/signal cable will pass through the power supply.

Pin #	RS-232 Function
1	CTS
2	TxData
3	RxData
4	DTR
5	GND
6	DSR (Connected to CTS)
7	NC
8	NC
9	NC

Table A.1. RS-232 Serial Port Wiring

Wire	Function	Direction
Red	No Connection	
Black	Ground	
Orange	No Connection	In
Brown	No Connection	
Yellow	Temperature, 0/5 Volt	Out
Blue	Pulse Width Range or Optional Current Loop Range	Out
Green	Ambient light signal, 0-5 V	Out
Purple	Amplitude signal, 0-5 V	Out
Shield	No Connection	

Table A.2. AccuRange 4000 LIR Power Supply Signal Cable Wiring

The optional AC to DC power supplies for the AccuRange 4000-LIR supply operating power and temperature stabilization heater power to the sensors. They are housed in NEMA-4 polycarbonate enclosures and are permanently attached to the AccuRange 4000 power/signal cable, with 6 feet of cable between the sensor and power supply. An additional 4 feet of cable extends beyond the power supply for reading the sensor's optional current loop and other outputs (Figure A.3).

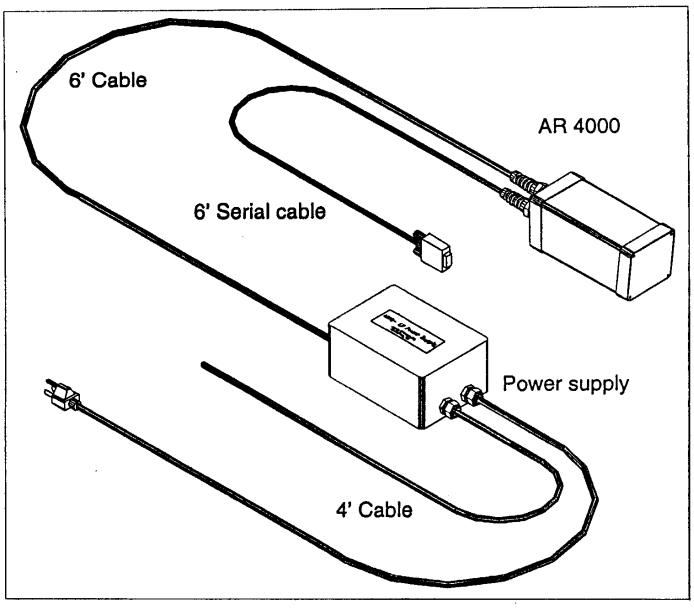

Figure A.3. AccuRange 4000 LIR with Power Supply

When power is applied, the green LED on the back panel should come on. After approximately 6 seconds, the LED will flash briefly and the laser will come on. The beam will be invisible or just barely visible. Use an IR viewing card in the path of the beam for viewing. The sensor should give reasonable range information after the 6 second delay has passed and the laser turns on, although it will take 5-10 minutes for the range readings to stabilize fully.

A.4. Performance and Measurement Accuracy

The AccuRange 4000-LIR will detect diffuse reflections from objects of any color with the greatest sensitivity falling at about 8 feet, although short distances right up to the front face of the sensor can be measured. If the sensor is configured with the close focus optics options, the greatest sensitivity will be 3 to 4 feet from the sensor unless adjustments have been made for a specific application. The sensor has no trouble picking up walls, floors, carpets, and even surfaces such as CRT screens

from almost any angle. Shiny surfaces such as glossy plastic or paint can be more difficult to detect, depending on the angle at which the beam hits them.

The sensor is calibrated with the temperature control active and set to 95° F. Lower laser settings and operation at other temperatures may reduce the accuracy of the measurements taken.

One of the configuration options for the AccuRange 4000 is the maximum range expected. This allows the sensor to obtain readings with the best possible resolution and accuracy. Internally, the time required to take a single sample depends on the distance being measured and the resolution used to take the measurement. If the ranges are known to be short, better resolution and accuracy at high sample rates may be obtained by reducing the maximum range. For most applications the default of 650 inches should be adequate. If you are measuring ranges greater than this, or ranges much shorter in situations where maximum resolution and high sample rates are required, your maximum should be specified using the Set Maximum Range command.

Most significant of the factors that affect the indicated range output is the amplitude of the return signal, or the reflectivity of the target. Indicated range can vary as much as 3 inches between very weak signals and very strong ones. The sensor has a signal strength output, which is an analog signal that ranges from 0 to 4 volts and is approximately logarithmic with received light intensity. The calibrated output compensates for varying reflectivity. The amplitude output can also be used to create grayscale images of objects over which the beam is scanned, and to determine whether a signal is valid or too weak to be reliable. Temperature and the ambient light level also affect the measurement slightly. Analog temperature and ambient light outputs allow these effects to be compensated for in software, but typically they are not significant unless the sensor is used in an environment where they vary widely.

A.5. Command Set

All configuration of the sensor may be done via commands sent over the serial port or by using the push-button switch and acknowledgment LED on the back panel. The serial port commands are ASCII commands that may be entered under computer control or from the keyboard of a terminal connected to the port. Configuration information may be stored in nonvolatile EEPROM with the Write command, and is then retained through power cycling. Commands are shown below:

- Set Sample Interval
- Set Maximum Range
- Set Zero Point
- Laser Power On / Off
- Enable / Disable Serial Data Output
- Set Baud Rate
- Set Serial Output to ASCII / Binary
- Set Analog Zero Current
- Set Span
- Set Analog Output Mode
- Read Configuration Data from EEPROM
- Write Configuration Data to EEPROM
- Reset Configuration to Factory Defaults
- Set Temperature Hold Level
- Take Single Sample (Serial Entry Only)
- Set Minimum / Maximum Valid Amplitude
- Show Version Number

APPENDIX B

ACCURANGE HIGH SPEED INTERFACE

B.1. General Description

The AccuRange High Speed Interface (HSIF) is an interface board that takes samples from the AccuRange 4000 optical rangefinder. Two models are available. One model plugs into an IBM-PC or compatible ISA or EISA bus and the other model is PC/104 compatible. In the thesis, the model compatible to ISA bus is used. Samples come over the bus in an 8 byte format that includes a 19 bit range value and 1 byte values for signal strength, ambient light, and sensor internal temperature as well as status and general purpose input bits.

The interface board operates by measuring the range-dependent pulse width output of the AccuRange 4000. To use the 4000 with the High Speed Interface, the Current Loop option must not be installed in the sensor. Each pulse on the pulse width output is timed on the Interface board by a timer with a clock rate of 80 MHz. The sample rate of the interface is therefore controlled by the sample rate for which the 4000 is configured. Since the pulse width output can be set to repeat at up to 50 kHz, that is the maximum sample rate of the interface.

Other features of the interface include memory buffer empty, half full, and overflow status indicators, external sample start/stop control, and three general purpose input bits that allow synchronous recording of events while sampling.

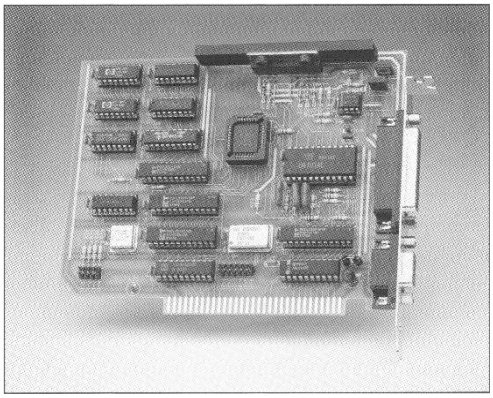

Figure B.1. AccuRange High Speed Interface

Key features of the AccuRange HSIF are as follows:

- Sample rates to 50,000 samples per second.
- Samples distance, amplitude, ambient light and temperature outputs.
- Supplies 8-byte samples over the standard PC ISA bus.
- 2K byte or optional 16K byte buffer for sample data onboard.
- Buffer empty , half-full and overflow indicators, and half-full interrupt.
- Start/stop sampling control input.
- 3 general purpose input data lines for event recording.
- Improved resolution and higher sample rate capability than the AccuRange 4000 alone.
- Supplies power to the AccuRange sensor.
- Optional motor power control and shaft encoder reading capability .

B.2. Motor Power

The AccuRange High Speed Interface can be ordered with two motor power control and encoder reading channels. Each motor may be set to one of 64 software controlled power levels via commands to the board. If the motors have encoders which are connected to the encoder inputs, two 8-bit values from the encoders are

decoded and inserted into the data stream, giving the position of each motor, modulo 256. If the encoders provide index pulses, these can be applied to two of the general purpose input lines and used to determine the absolute positions of the motors. The description of the 25 pin I/O connector for encoder connection details are shown in Table B.2.

If motors are to be driven by the power amplifier on the board, the motors and motor power must be connected to P2. Motor 1 should be connected between pins 14 and 16, and motor 2 between pins 1 and 2. A separate power supply is required to drive the motors. Connect the motor power supply to pin 3 and the power supply ground to pin 15.

B.3. I/O Connectors

There are 2 connectors on the High Speed Interface. The 9 pin connector supplies power and receives signals from the AccuRange 4000. The 25 pin connector is used for powering the motors and reading the motor encoders, general purpose inputs, and sample control input.

B.3.1. 9 Pin Power and Signal Connector P2

The line descriptions for P1 are same as the descriptions of the power and signal lines in the AccuRange 4000 Power and Signal Cable Wire Description section. Pins 1-4 supply sensor power and sensor heater power and ground lines. The remaining lines are inputs for the signals from the AccuRange 4000. Pins 5, 7, and 8 are the inputs for the analog signals, with 2K impedance. Pin 6 is the input for the pulse width range signal (Table B.1).

Pin	4000 Wire	Function	Direction
1	Red	Power, +5V (5-6V)	Out
2	Black	Ground	
3	Orange	Heater Power, +5V (4.5-7V)	Out
4	Brown	Heater Power Return	
5	Yellow	Temperature, 0-5 V	In
6	Blue	Pulse Width Range Signal	In
7	Green	Ambient light signal, 0-5 V	In
8	Purple	Amplitude signal, 0-5 V	In

Table B.1. Power and Signal Connector Wiring

B.3.2. 25 Pin I/O Connector P2

P2 includes general purpose input lines, a sample start/stop control line, quadrature encoder input lines, and power for encoders or other applications (Table B.2).

	Top Row			*Bottom Row*	
Pin	**Function**	**Direction**	**Pin**	**Function**	**Direction**
1	Motor 2 Control	Out	14	Motor 1 Control	Out
2	Motor 2 Return	Out	15	Motor Power Ground	
3	Motor Power Supply	In	16	Motor 1 Return	Out
4	No Connection		17	No Connection	
5	+5V Power, 100 mA.	Out	18	+5V Power, 100 mA	Out
6	Ground		19	Motor 2 Encoder Ch A	In
7	Ground		20	Motor 2 Encoder Ch B	In
8	Ground		21	Motor 1 Encoder Ch A	In
9	Ground		22	Motor 1 Encoder Ch B	In
10	Ground		23	No Connection	
11	No Connection		24	General Purpose Input 1/ Encoder 1 Index Pulse	In
12	Start/Stop Sample Ctrl	In	25	General Purpose Input 3	In
13	General Purpose Input 2/ Encoder 2 Index Pulse	In			

Table B.2. I/O Connector

B.4. I/O Port Interface

The High Speed Interface is accessed as a set of I/O ports on the ISA or PC/104 bus. The board occupies a group of 8 contiguous port address locations starting at the address determined by the port address jumper J1 (Table B.3).

	Jumpers				Address
ISA/EISA Model:	J1-4	J1-3	J1-2	J1-1	
PC/104 Model:	JMP-4	JMP-3	JMP-2	JMP-1	
	off	off	off	off	000Hex
	off	off	off	on	040Hex
	off	off	on	off	080Hex
	off	off	on	on	0C0Hex
	off	on	off	off	100Hex
	off	on	off	on	140Hex
	off	on	on	off	180Hex
	off	on	on	on	1C0Hex
	on	off	off	off	200Hex Default
	on	off	off	on	240Hex
	on	off	on	off	280Hex
	on	off	on	on	2C0Hex
	on	on	off	off	300Hex
	on	on	off	on	340Hex
	on	on	on	off	380Hex
	on	on	on	on	3C0Hex

Table B.3. Port Address Base Jumper Map

Any one of 16 locations between 0 Hex and 3C0 Hex may be selected for the port base. A location that does not interfere with other peripherals in the system can be chosen.

B.5. Interrupt Driven Operation

The interface board may be jumpered to generate an interrupt when the on-board data buffer is half full. One jumper pair in Jumper J2 may be closed to activate the

corresponding interrupt as shown in Table B.4. The interrupt can only be cleared by reading samples until the buffer is less than half full or by issuing a Reset command to the board.

Jumper Pair		Interrupt Enabled
ISA/EISA	PC/104	
J2-1	JMP2-6	IRQ 3
J2-2	JMP2-5	IRQ 4
J2-3	JMP2-4	IRQ 5
J2-4	JMP2-3	IRQ 6
J2-5	JMP2-2	IRQ 7
J2-6	JMP2-1	IRQ 9

Table B.4. Interrupt Enable Jumper

B.6. Interface Resolution and Sample Rates

The resolution for the high speed interface is 33% better than the resolution of the sensor data transmitted over the serial port or current loop, and the maximum possible sample rate is much higher, since the serial and current loop output rates are limited by the 4000's on-board processor.

At relatively low sample rates, the 4000 sets a higher internal rate and averages multiple samples for best resolution. Since the high speed interface sample rate is the same as this internal rate, setting a combination of low sample rate and short maximum range will result in a higher than expected high speed interface sample rate. The slowest sample period with a max range setting of 650 inches is about 2250 microseconds, or 440 Hz, and rises to 5.4 kHz (185 microseconds) with a max range setting of 1 inch. To obtain a lower sample rate, set the max range to a larger value, and then set the sample rate desired, or for best accuracy sample at the higher rate and average multiple samples in software.

APPENDIX C

CTI-HSIF SOFTWARE LIBRARY

C.1. Product Overview

CTI-HSIF software library is a set of kernel drivers, Dynamic Link Libraries (DLLs), object files and header files that provide a high-performance, high-level interface to the functionality of the Acuity Research Inc. AR4000 series laser distance sensors, using the AccuRange High-Speed Interface card.

The CTI-HSIF library allows the user to select a number of formats for the acquired data. English or metric output, calibrated or uncalibrated data, or both, may be selected. Additional information (signal amplitude, ambient light, temperature) may optionally be returned.

The HSIF card's onboard memory may be augmented by a user configurable buffer within the library, which will be filled as data arrives over the HSIF. When a predefined number of samples is available, the library sets a queryable status flag, or calls a user-defined callback function, enabling the application program to easily determine when a desired number of samples is ready for processing.

C.2. Library Conceptual Overview

The CTI-HSIF library is an interface layer between the Acuity Research AR4000 sensor hardware and the application program(s). The library communicates with the sensor via the computer's serial port, and the HSIF card, while the application

program communicates with the library via function calls. All commands sent to the sensor by the application program pass through the library, and all data sent by the sensor is processed by the library and made available for use by the application program.

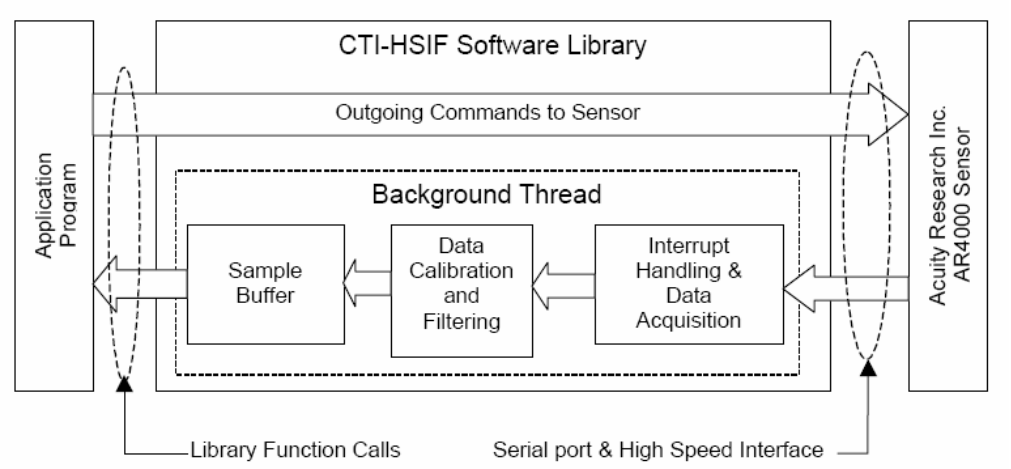

Figure C.1. Communication Scheme of the CTI-HSIF Software Library

One of the major functions of the library is to manage the flow of data to and from both the serial interface and the HSIF card, and ensure that both outbound data (commands sent to the sensor), and inbound data (range samples from the HSIF) are properly processed. To accomplish these tasks, the library consists of several distinct functional blocks. These are shown pictorially in Figure C.1.

C.2.1. Commands to Sensor

When an application program calls a library function that sends a command to the sensor or HSIF, the outbound data is passed directly to the computer's serial port for transmission to the sensor or is sent directly to the HSIF card. The library will not return control to the application program until the command has been sent. Inbound data however, goes through a more complex series of steps as explained below.

C.2.2. Background Processing Thread

A key design concept of the CTI-HSIF library is a separation of responsibilities – the library is responsible for reading data from the sensor, and translating that data to range samples, while the application program is responsible for processing those samples, once read. To accomplish its task, the library uses a background processing thread. The background thread is started by the library when the setCommOpen function is called, and runs continuously until the setCommClosed function is called. Without any interaction by the application programmer, the thread reads data from the computer's serial port and the HSIF card, as it is received from the sensor and passes it to the other parts of the library for subsequent processing.

Since, during use, the AR4000 sensor is typically outputting a continuous stream of data samples via the HSIF, this ensures that those samples are captured and stored for use by the application program. Because the background thread runs independently of whatever else the application program may be doing, this design permits the application programmer to concentrate on processing samples already retrieved from the library, while the background thread assumes the responsibility of acquiring any new incoming data simultaneously.

C.2.3. Interrupt Handling and Data Acquisition

This functional block implements the low-level driver that handles hardware interrupts from the HSIF card, and reads samples from the card's hardware buffer into the library's internal memory.

The HSIF has the capability of generating a hardware interrupt whenever the card's onboard buffer is half-full. Through a Windows kernel mode driver, the CTI-HSIF library recognizes when this interrupt occurs, and using a high-priority background thread reads the data from the buffer. Using the HSIF's interrupt generation

capability is the preferred mode of execution, since only when the interrupt occurs, indicating that data is actually available, does the library's background thread run.

Alternatively, if the HSIF's interrupt capability cannot be used (for example, if no free hardware interrupts exist on the computer), then the CTI-HSIF library can poll the HSIF to determine when its buffer is half-full. This method is not preferred, as the continual polling may impose excessive load on the computer, potentially slowing down other portions of the library, and generally reducing the responsiveness of the system.

C.2.4. Data Calibration and Filtering

This functional block converts the raw encoder readings and range samples into angular measurements and calibrated samples. The raw samples from the HSIF are calibrated using the data from the calibration file provided with the sensor hardware. Any filtering and transformation rules set by the application programmer are then applied to the data samples.

C.2.5. Sample Buffer

All samples that pass the data filtering stage are stored internally in the library's internal data buffer. When an application program requests samples, they are retrieved from the buffer and returned to the application. If sufficient samples are not currently available in the buffer, the library will wait until additional samples are inserted in the buffer by the background processing thread.

C.3. Using the Library

C.3.1. Library Functions

All interaction with the AR4000 sensor or the HSIF card is done through one of the functions provided by the CTI-HSIF Library. These provide for configuring the library itself, and configuring and acquiring data from the sensor. Each function is explained in detail in the CTI-HSIF Software Library Reference Manual [40]. The functions fall into five main groups:

1. Library Configuration Functions
2. Sensor and High Speed Interface Configuration Functions
3. Data Acquisition Functions
4. Data Format Functions
5. Data Filtering and Transformation Functions
6. Error Handling and Miscellaneous Functions

The library configuration functions are used for configuring various aspects of the CTI-HSIF library itself, such as which serial port the library will use to communicate with the sensor, which I/O address is used for the HSIF card, the library's internal buffer size, the callback threshold, etc..

The sensor and HSIF configuration functions interact with the AR4000 sensor hardware and HSIF card to configure the sensor. Since the AR4000 sensor and HSIF hardware do not provide means of determining their current settings, the CTI Library tracks the current state of each setting, for easy determination by an application programmer.

When using the AR4000 sensor and HSIF card, most sensor configuration commands are sent to the sensor via the serial port. However, some operations (such as setting the motor power and direction or clearing the HSIF's buffer) require interacting directly with the HSIF board.

The data acquisition functions are used to turn the laser on or off, set the sensor sampling rate, acquire data from the sensor, turn on or off continuous sampling, etc..

The data format functions set the sample units to English units (inches) or metric units (millimeters), and the angle measurements to polar or Cartesian coordinates.

The data filtering functions allow filtering the raw samples from the sensor based on any combination of range, amplitude, ambient light, or temperature reading.

The error handling and miscellaneous functions perform various miscellaneous operations, such as querying the library's version number, the sensor's firmware version, and retrieving error messages.

C.3.2. Data Reading

When acquiring data from a hardware device, such as the AR4000 sensors, the application program must have some means of determining when data is available to be read from the device. Broadly speaking, there are three ways in which this can be done:

1. The device can be polled (queried) to see if it has valid data available. If so, the application program reads the data, otherwise the application program can perform some other function (such as process already received data) and loop back at a later time to poll the device again.

This method has the advantage of being simple to program. However, its disadvantages are:

- The application must poll the device frequently in order to ensure that no data is lost while the 'other work' is being done.

- It is typically inefficient, since the majority of the time it is likely that no data is yet available, and the frequent polling simply wastes CPU time that could be used by other processes.

2. The application program can perform a "blocking read" in which it starts a read operation that does not return control to the application program until the device has the desired amount of data ready.

This method is also simple to program, and is typically much more efficient than the polling approach. However this method suffers from the disadvantage that the application cannot do other work while waiting for data to be available from the hardware device.

3. The application program can install a 'callback function' that is called by the hardware device, or the device's driver software when data is available.

This method can be more complex to program than the other two approaches, however it has the significant advantage that the device *tells* the application when it has data available, rather than the application program having to *ask* the device if data is ready.

In general, the last two methods are preferred, since polling the library's sample counter is typically inefficient.

C.3.3 Data Filtering

Once data samples have been read from the library, the data acquisition program typically processes those samples according to the needs of the particular application.

The CTI-HSIF library permits filtering of the raw samples from the sensor, based on any combination of range, amplitude, ambient light, temperature reading, or angular measurement.

C.3.4. High-Speed Interface Interrupt Usage Considerations

When using the CTI-HSIF library with the HSIF's interrupt generation capability, samples will be read from the HSIF card only when the half-full interrupt is triggered. Each time the interrupt is triggered, the CTI library reads a block of samples equal to one-half the card's hardware buffer size. Thus, the CTI library acquires samples from the hardware in half-buffer 'chunks'.

The above behavior affects the results of any functions that query the amount of data available from the library. For example, assuming a 16KB hardware buffer (1024 samples per half-buffer), the getNumSamples() function will return zero samples until the first interrupt is received from the HSIF, after which it will return 1024 samples until the second interrupt is received, after which it will return 2048 samples, and so on.

Similarly, although any arbitrary callback threshold may be set, the callback will not be called until *both* the callback threshold has been exceeded, *and* enough half-buffers of data have been read to provide sufficient samples to the CTI library. Again assuming a 16KB hardware buffer, if the callback threshold is 500 samples, the callback will not be called until 1024 samples (one-half buffer) have been read.

C.3.5. Encoder Support

The HSIF provides two 8-bit encoder counters that record the encoder position simultaneously with the range data measurement.

The 8-bit count captured by the HSIF will wrap around to zero after reaching 255, so the CTI-HSIF library manages this wrap-around in order to return the full encoder counts per revolution to the application. Furthermore, the CTI-HSIF library converts the raw encoder count and returns to the application a standard angular measurement, using the encoder index pulse as the zero-angle point. From the index pulse point, the angular measurements will increase in the direction that the mirror or scanner is rotating.

C.3.6. Performance Considerations

To achieve the best performance from the CTI-HSIF library and HSIF hardware, the user or application programmer should consider the following:

- Use the interrupt generation capability of the HSIF card especially at high sample rates (above 5000-10,000 samples/second).
- Use a sample rate only as high as required to achieve the desired application results. A higher than necessary rate imposes additional overhead on the library, increasing the chance that the application program may not be able to process the data fast enough. Calculate the sample rate required carefully.
- Whenever possible, use the library's filtering functions to discard unwanted or 'out of range' samples. This can result in a performance improvement by avoiding relatively time-consuming operations (such as calibration of the raw range) that would otherwise be done on the raw data.
- Use the setBufferSize function to allocate as large a buffer as can be accommodated within the available memory of the computer used. A large library buffer gives the application programmer more leeway in processing samples from the library without being concerned with the library's buffer overflowing. (Note however that allocating too large a buffer may use up all the physical memory on the computer, and result in paging to disk as the virtual memory manager takes effect. This will cause a severe loss in performance.)

- In general, at high sample rates, application performance will be better from C or C++, rather than Visual Basic or Visual Basic for Applications (Excel). Although the internal performance of the CTI library is identical regardless of the language used, the overhead of the Visual Basic and VBA runtime systems can be avoided by using C or C++.
- When using the library from Visual Basic or Visual Basic for Applications (VBA), it is strongly recommended that interrupts be used, instead of polling the card (i.e. specifying interrupt=0). The overhead resulting from the Visual Basic runtime system may cause poor performance if polling is used.