3D Modelling of Buildings and Environments using Laser Scanning and Surface Reconstruction

Ramin Bakhshi

Submitted to the
Institute of Graduate Studies and Research
in partial fulfilment of the requirements for the Degree of

Master of Science
in
Electrical and Electronic Engineering

Eastern Mediterranean University
September 2011
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Elvan Yılmaz
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Electrical and Electronic Engineering.

Prof. Dr. Aykut Hocanın
Chair, Department of Electrical and Electronic Engineering

I certify that I 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 in Electrical and Electronic Engineering.

Prof. Dr. N.Suha Bayındır
Supervisor

Examining Committee

1. Prof. Dr. N.Suha Bayındır _____
2. Assoc. Prof. Dr. Hasan Demirel _____
3. Asst. Prof. Dr. Rasime Uyguroğlu _____

ABSTRACT

3D models of environments and buildings are widely used in Geographical Information Systems (GIS), building information models, constructional management, environmental planning, city guides, path finding and Robotic applications, where the accuracy of data collection and resolution of the 3D model have been the main concern. 3D models of buildings and other objects can be constructed by following three main steps, namely, data acquisition, alignment and surface reconstruction.

This project aims at introducing the process of forming a 3-D model from 3-D scan data and describing the data acquisition, alignment and surface reconstruction sequences in detail. Hardware and software design and implementation has been made for each stage of the 3D modelling process and a full grasp of the system is achieved. However, due to the complexity of the system, and also due to time limitation, it was not possible to achieve sufficient performance from the designed system. Instead, a commercial 3D laser scanner was used for the sake of completing the requirements of the 3D Modelling process with a reasonable performance.

Current data acquisition systems have been reviewed and compared to discuss their advantages and drawbacks, as a result of which, 3D laser scanner has been chosen to be the most accurate and fast data acquisition system appropriate for scanning indoor and outdoor environments. Due to some limitations of the available 3D laser scanners, a commercial 1D-laser scanner has been converted into a 3D laser scanner,

by designing and constructing a pan-tilt mechanism for the 3 axis control of the 1D laser scanner. The new 3D laser scanner is a simple and light in weight, which is easily adopted to a remote operating and monitoring Vehicle (ROMV) which has been designed in this project to add an indoor and outdoor mobility feature to the device. Although the new 3D Laser scanner operates properly, the accuracy and resolution of the scan results are not as expected yet. In order to complete the 3D modelling process with a reasonable accuracy and resolution, data acquisition is achieved by using the 3D laser scanner provided form the Stevens Institute of technology (CAD eye Scanner). This system is not only able to scan indoor and outdoor environments with acceptable resolution, but also it is able to collect RGB data corresponding to each scanned point in the scanned environment.

For the Alignment or Registration of the acquired coordinate data obtained by the 3D Laser scanner, this research has used the semi- automatic method produced by an academic software tool called Mesh-lab. Point cloud model of the Techno park building was obtained by using professional software called Pointools. Surface reconstruction methods are investigated to obtain models with seamless and smooth surfaces from the point cloud model. It is realized that the existing methods fail to produce realistic surfaces under noisy data and a new method based on implicit surface reconstruction using isotropic basis functions has been developed to represent the sharp features more close to their real appearance. Some initial simple results of this method are presented in the thesis. A further work is needed to apply this method to reconstruct the surfaces of a complete 3D building model.

Keywords: 3D Laser scanner, surface reconstruction, registration, radial basis functions, generalization.

ÖZ

Çevrenin ve binaların 3B modelleri, Coğrafi Bilgi Sistemleri (CBS), bina bilişim modelleri, yapı yönetimi, çevre planlaması, şehir rehberleri, yol bulma ve Robotik uygulamalarında yaygın olarak kullanılmaktadır. Bu uygulamalarda, veri toplamadaki doğruluk ve 3B modellerin çözünürlüğü en önemli ilgi odağıdır. Binaların ve diğer nesnelerin 3B modelleri, aşağıdaki üç ana aşamada oluşturulur: veri toplama, yerleştirme ve yüzeylerin düzeltilmesi.

Bu proje, taranmış 3B verilerinden, 3B modelleri oluşturma sürecini anlatmayı ve veri toplama, yerleştirme ve yüzey düzeltme süreçlerini detaylı olarak tanımlamayı hedefler. 3B modelleme işleminin her aşaması ile ilgili çeşitli donanım ve yazılım tasarımları ve uygulamaları yapılmış ve sistem tümüyle kavranmıştır. Ancak, sistemin çok karmaşık bir yapıda olması ve zaman sınırlaması nedeniyle, tasarlanan sistemden yeterli performans elde edilememiştir. Bunun yanı sıra, 3B modellemenin tüm aşamalarını, makul bir performans ile gerçekleştirebilmek için, ticari bir 3B Lazer tarayıcı kullanılmıştır.

Mevcut veri toplama sistemleri taranmış, avantaj lar ile dezavantajları karşılaştırılmış ve sonuç olarak, 3B Lazer tarayıcı, iç ve dış mekanları taramak için kullanılabilecek en doğru ve hızlı tarayıcı olarak seçilmiştir. Mevcut 3B Lazer tarayıcıların sınırlamaları göz önüne alınarak, ticari bir 1B lazer tarayıcının 3 eksenli kontrolünü yapan bir pan-tilt mekanizması tasarlanıp üretilmiş ve sonuçta, 1B tarayıcı, 3B tarayıcıya dönüştürülmüştür. Yeni 3B lazer tarayıcı, bu projede tasarlanan ve

üretilen, uzaktan kumandalı ve göstergeli araç'a (UKGA) kolaylıkla adapte edilebilen basit ve hafif bir yapıdadır. Yeni 3B Lazer tarayıcı, düzgün çalışmakla birlikte, tarama sonuçlarının doğruluğu ve çözünürlüğü henüz beklendiği gibi değildir. 3B modelleme sürecini, makul bir doğrulukla tamamlayabilmek için, Stevens Teknoloji Enstitüsünden sağlanan 3B Lazer tarayıcı (Cade Eye Tarayıcı) kullanılmıştır. Bu sistem yalnızca iç ve dış mekanları, kabul edilebilir bir doğrulukla taramakla kalmayıp, aynı zamanda, taranan bölgedeki her noktanın RGB verilerini de toplayabilmektedir.

3B lazer tarayıcı tarafından toplanan verilerin yerleştirilmesi veya kaydedilmesi işleminde, Meshlab isimli bir akademik yazılım aracı tarafından üretilen, yarı-otomatik bir yöntem kullanılmıştır. Teknopark binasının nokta bulutu modeli, Pointools adlı bir profesyonel yazılım kullanılarak oluşturulmuştur. Nokta bulutu modelinden, kesintisiz ve düzgün yüzeyler elde edebilmek için, yüzey düzeltme yöntemleri araştırılmıştır. Mevcut yöntemlerin, gürültülü veriler altında, gerçekçi yüzeyler üretmekte başarısız olduğu anlaşılmış ve, keskin özelliklerin gerçek görüntülerine daha yakın temsil edilebilmesi için, izotropik tabanlı fonksiyonlar kullanan örtülü yüzey düzeltme yöntemini esas alan yeni bir yöntem geliştirilmiştir. Bu yöntemin bazı başlangıç düzeyindeki basit uygulamaları bu tezde sunulmuştur, ancak, bu yöntemin bir binanın tamamının 3B modeline uygulanabilmesi için daha ileri çalışmalara ihtiyaç vardır.

Anahtar Kelimeler: 3D Lazer tarayıcı, yüzey geriçatımı yerleştirilme, radial taban fonksiyonları, genelleme.

ACKNOWLEDGMENTS

It is a pleasure to express my gratitude to those who made this thesis possible such as my supervisor Prof.Dr. N. Suha Bayindir for his excellent guidance, caring and patience and his encouragement, supervision and support from preliminary to the concluding level enabled me to do this research.

I would like to thank Prof. Dr. M.Hashemipour from Mechanical department for his encouragement and support during my research on this project.

I would like to thank my friends Mr.Maziar Movahedi who and Miss. Nastaran Shirgiri and others who were always willing to help and encouraged me in this report.

At last, I would like to thank my parents who were always supporting me and encouraging me with their best wishes.

I wish to dedicate this thesis to my dear parents (Mrs. Farzaneh Ebrahiminia and Mr. Mohammad Hossein Bakhshi) who have always supported and encouraged me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

1D	One dimension
2D	Two dimensions
3D	Three dimensions
ASIC	Application Specific Integrated Circuit
BASCOM	Basic Complier
CCD	Charged Coupled Device
CT	Computed Tomography
GIS	Geographic Information System
GPS	Global Positioning System
IDE	Integrated Development Environment
LB	Left Backward
LCD	liquid Crystal Display
LF	left Forward
LIDAR	light Detection And Ranging
LV	Low Voltage
NN	Neural Network
NI-MH	Nickel Metal Hydride cell
N-Size	Neighborhood Size
OCU	Operating Control Unit
OVF1	Timer1 over Flow
PC	Personal Computer
PR	Point Region

PWM	Pulse-Width Modulation
ROMV	Remote Operating and Monitoring Vehicle
TTL	Transistor-Transistor logic
UART	Universal Asynchronous Receiver/Transmitter
WLAN	Wireless Local Area Network

LIST OF SYMBOLS

θ	Polar angle
ρ	Smoothness parameter
ϕ	Azimuthal angle
T	Time variable
r	Ranged data in spherical coordinate
C	Speed of light in free space
φ_i	Hidden layers Functions
d_i	Signed- Distance function at point (x_i, y_i, z_i)
f_i	density scalar filed at point (x_i, y_i, z_i)
$f(x_i)$	Actual surface Function
$d(x_i)$	Desire Surface function
$\|.\|$	The second order derivative of the function
$\varphi_i(x)$	Hidden space basis
$E_s(\ )$	Interpolation cost function
$E_c(\ )$	Approximation cost function
E ( )	Regularization cost function
D_{ij}	Geodesic distance parameter
Dist ( )	Clustering distance parameter
W_i	Hidden layer to output layer Linear transformation coefficient
ε	Laser scanner error term

Chapter 1

INTRODUCTION

The quest for accuracy and improvement in the field of science and technology has been the driving forces for almost all scientists to remain develop novel scanners and reconstruction techniques. In the past several decades, 2-D scanners were used to produce 2-D maps or models which were not easy to anticipate by ordinary users. An upgrade to the 3-D scanner made it not only possible to find paths, but also to scan point by point and efficiently trace and locate any point on objects or environment`s surfaces with a high degree of accuracy and reliability.

A 3-D scanner is a device, which examines a real object's form to collect data related to its shape and its appearance (i.e. colour or texture). The data which is collected by the scanner is used to produce digital 3-dimensional models, which are useful for a wide variety of applications. The common applications of this instrument include industrial design, reverse engineering, prototyping, prosthetics, quality control, robotics, geographic information system (GIS) and also in the entertainment industry.

Fundamentally creating a 3-D model from the scanned data requires a sequence of pre-processing steps. These steps include data acquisition using a 3-D scanner and a microcontroller driver pan-tilt system, converting the 3-D coordinate information into a 3D point clod model and finally obtaining a 3-D model with the require

resolution using surface reconstruction method. The past decades has seen the evolution of many different algorithms and hardware designed to accommodate the various techniques and steps required for their efficient operations. The improvements of these algorithms were mostly triggered by the fact that the limitations of the previous algorithms were evident and therefore the new algorithms aim at overcoming the limitations of the system. This thesis will illustrate the problems inherent in earlier traditional scanning devices and how powerfully intuitive techniques will be used to create a more reliable and accurate 3-D scanning system.

1.1 Problem Statement

In order to undertake the task of creating a new and more efficient 3-D laser scanner system for scanning and reconstruction, and in accordance to the purpose of any modeling project, with the available budget, 3-D data acquisition techniques which accurately acquires coordinate data to obtain the most precise and reliable outcomes.

Formerly, 2-D colorful or monochromic images were popular in extracting the range data [27]. This method described stereoscopy in which the purpose was to capture several images of the same object or environment in a specified manner to produce a 3-D range data, which includes the coordinate points of the captured scene. Another method explains how a 2D laser scanner was mounted over robotic tank shape chase to scan and determine the topology of a tunnel. As seen in Figure 1.a the system starts to scan the inner boundaries of the walls and the obstacles inside the tunnel at each point [24]. The resolution of the scanned data depends on the speed of the Tank whereby the resolution decreased as the tank speed increase. Figure 1.b shows the 2D laser probe, which have used by this project. In addition using an airborne Lidar to

obtain spin images, and fuseing with ground-based Lidar data has porposed to improve classification of building infrastructure [8]. Figure 2.a shows the system principle diagram and figure 2.b shows the fused point cloud of the airborne Lidar and ground-based Lidar form scanned environment.

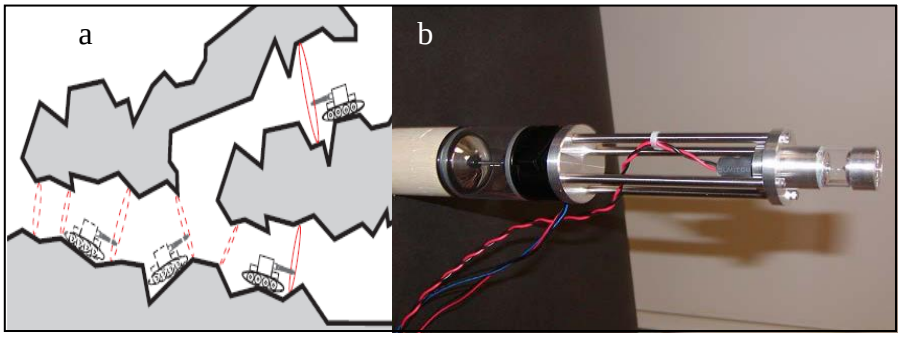


Figure 1.1: (a) Constructing a 3D map of rubble and (b) laser probe, [24].

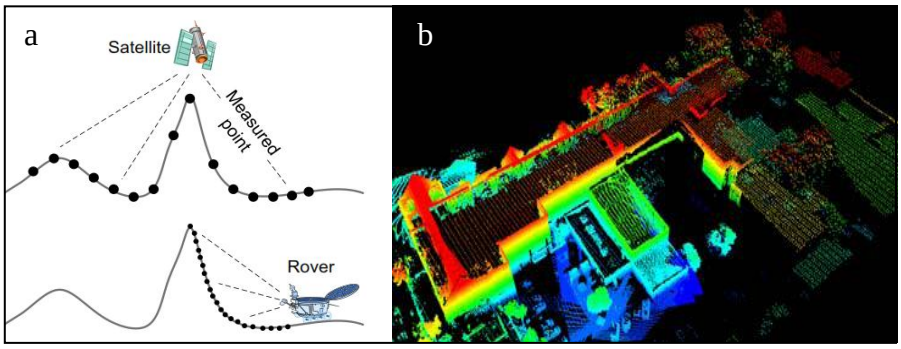


Figure 1.2: (a) Fusing Airborne and ground Lidar Principle and (b) Point cloud of fused result, [8].

3-D models have more advantages than 2-D models in that they give better points of view based on human perception, give a general idea about the scene and enable the user to rapidly recognize and visualize structures and obstacles such as trees, walls, windows and stairs, which cannot be seen in 2-D models. Therefore, 3-D range data models are adapted to a wide variety of applications and tasks such as rescue and security applications, self-localization and mapping systems or as a background model for tracing and detecting of people and cars [8].

Modeling of outdoor environments is complicated due to the large size of the objects and the long distances involved. As a result, the resolution of the shape of the objects becomes very low and some of the features may not be represented well. Outdoor environments usually have a wide variety of features to be represented such as buildings, trees, and cars. A relevant factor worth noting in outdoors is the scale of the environment, which varies from a few millimeters to several kilometers. Most approaches for indoor mapping deal with rooms and corridors while outdoor maps need to scale to kilometer squares. For a 3-D representation, one more dimension is added to the map, creating serious scaling limitations for practical use [48].

Finally, the terrain is normally flat indoors unlike the case of outdoors. Irregular terrain with depressions and small rocks make the task of mapping more challenging since they make the robot bump and change its direction thus inducing errors in proximity sensors and corrupting Odometric information. Outdoor 3D mapping have proven to address the 2-D shortfalls in the computer vision community and more recently by the robotics community [34].

There is some evidence to show the uses and importance of using 2-D systems for laser scanning and surface reconstruction. The wide range of shortcomings inherent in this system will make us focus on improving and eradicating these shortfalls to produce more accurate, precise and more realistic images both indoors and outdoors by using a 3-D system. This research will go a long way to elucidate the fact that 3-D models are able to represent more detailed information than classical 2-D systems, which are typical used in mobile robotic applications. The environment i.e. either indoors or outdoors, will be taken into consideration as well as terrain to enhance the merits of using 3-D systems.

1.2 Purpose of Study

Accuracy and correctness in science and technology are paramount in the production of any scientific device in order to make it a reliable use to man. Since in the past 2-D systems were used for laser scanning and reconstruction, their inaccuracies and the inability to scan and reconstruct indoor and especially outdoor environment prompted the creation of a 3-D systems.

This project aims at making a 3-D scanner, data acquisition and modeling system which is able to support robotics and geographic information system (GIS) used for modeling indoor and outdoor environments with reasonable accuracy. This 3-D scanner system has a higher resolution and speed than other traditional scanners and it is adapted to flexible software to produce the range-images appropriately. The device produces correct and accurate 3-dimensional maps that are essential for applications that need visual and geometric information of the environment.

3-D scanners are more robust systems than traditional coordinate measuring systems, which make the user able to not only make the virtual 3-D digital model or digitize free form of the objects, but also to allow for the possibility to change the shape of components. The importance of using 3-D systems is many folds. For instance in a product line of modern cars, many parts have to be merged and assembled to form the body of the car. The geometry of the parts has to be checked and dimension's accuracy should be ensured. Some widespread applications of this instrument relate to the fields of industrial design, reverse engineering, prototyping, orthotics and prosthetics, quality control, robotics, geographic information system (GIS) and entertainment.

1.3 Methodology

In this research, a practical and realistic methodology will be used to achieve the objective. The common pipeline in forming a 3-D model from a 3-D scanning result is shown in the figure 1.3 below. This pipeline includes four main processes, namely: data acquisition, alignment or registration, surface reconstruction and texture mapping.

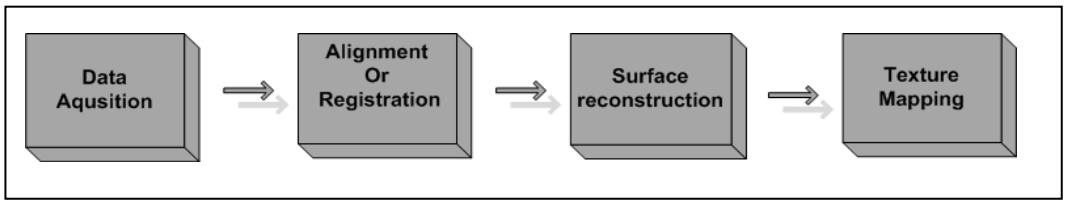


Figure 1.3: 3D modeling pipeline

Data Acquisition

Data acquisition mainly refers to the way the system collects 3-D coordinates and the RGB colour information from the environment or object. For this purpose, this research uses a 1-D laser scanner and a combination of hardware and software, to collect 3-D range data more accurately and easier than the previous methods.

In the first step, the designed pan tilt attached to the 1-D laser scanner makes it able to scan in 3-D space and cover the entire sphere. In addition, in order to pan and tilt the device in all directions, the system uses two servomotors as actuators. The second step requires the 3-D scanner to be installed over the designed Remote Operating and Monitoring Vehicle (ROMV) in order to increase the performance. This improved performance allows the system to perform both in indoor and outdoor environments. Although this allows the system to collect the 3-D range data, the system developed in this project takes one-step further and collects the RGB colour data, by attaching an RGB wide-angle camera and a GPS to the system. The camera collects the RGB

data corresponding to each measured range data and the GPS covers the position information of the system within the operation. As a result of these processes, a 3D model of the environment is obtained is obtained in the form of a point cloud.

Alignment or Registration

In most circumstances, one scan shot will be insufficient to cover a whole object or environment [43], [25], [16], [30], [29]. Several scans must be taken to from different angles and directions for optimal results. Usually there is a need to collect information and data about all sides of the item. All these scans have to be assembled in a unique and common reference coordinate system. This procedure is commonly named Alignment or Registration. Typically, the noise redundant and filtering methods are performed in this step,. Figure 1.4 illustrates this state.

Figure 1.4: Taking scan from different angels, [15].

Surface Reconstruction

This refers to the technique, which tries to estimate and reconstruct an arbitrary surface topology from the point cloud model [6], [50], [38], [28], [22]. This project uses a generalized implicit surface reconstruction method based on radial basis functions [39]. The ability to reconstruct a continuous and seamless surface from a disorganized sample points make this method very popular. The proposed method improves the performance accuracy of the surface reconstruction method by adding extra information about the construction of disorganized sample points. This additional information is produced through the sharp feature extraction method. Figure 1.5 below shows the main diagram of this state.

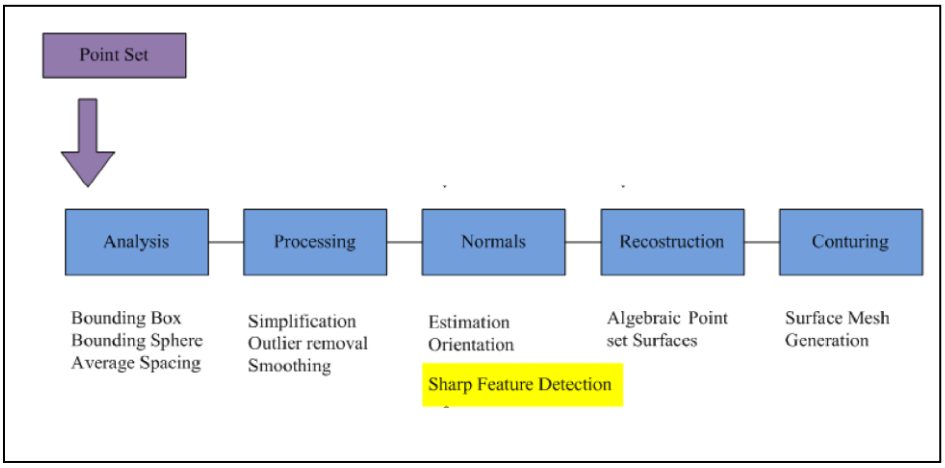


Figure 1.5: Surface reconstructions pipeline, [1].

Texture Mapping

After the reconstructing the surfaces of the model, the RGB colour data mentioned in the Data Acquisition step is added to the surface to create a realistic visual image of the 3-D model.

Although this project focuses mainly on data acquisition and surface reconstruction methods, it partially covers many aspects of 3-D modelling starting from scanned

data. Therefore, for better understanding and to comprehensively attain the objectives of this research, the complete project is separated into two main parts. The first part covers the Data Acquisition and Alignment phases and the second part discusses and describes the Surface reconstruction process. The four phases of the 3D modelling process shown in Figure1.3 exhibits the merits of the 3-D systems and demonstrates how these phases are implemented during the process of laser scanning and surface reconstruction.

Chapter 2

THEORY AND PRINCIPLES OF LASER SCANNING

2.1 Overview of 3-D Laser Scanning

The main purpose behind 3-D laser scanners is to capture and sample the geometric shape and appearance of an object's surface. This is done by creating the range data or 3-D point cloud model. Post-processing methods use this range data to reconstruct the shape of the sampled subject. This process is termed as surface reconstruction. A 3-D model is able to represent more detailed information than a traditional or classical 2-D map, which typically uses mobile robotic applications. These systems combine the laser range data by the RGB colour data (or vision data) obtained from a video camera in a single representation to form a digital textured 3-D model. 3-D captured models give better visualization as per human perception and provide a general idea of the scene under investigation by enabling the user to quickly recognize and visualize structures and obstacles such as trees, walls, windows and stairs, which cannot be seen in 2-D models [8].

By combining vision and laser range-finder data in a single representation, a textured 3-D model can provide remote human observations with a rapid overview of the scene. An added advantage is that in image processing applications, it enables a clear view of structures such as windows and stairs, which are invisible if a 2-D model, were used. This makes the 3-D models well suited for a diversity of tasks including

surveillance for security and rescue applications, self-localization, or as a background model for detection, tracking of people of indoor and outdoor environments and mapping.

3-D scanners are very analogous to cameras since they have a cone-like field-of-view like cameras and can simply capture and collect data about object`s surface which are not masked by obstacles. Meanwhile cameras collect colour information about the object`s surface within their file-of-view, 3-D scanners collect measured distance information about the object's surface. The 3-D range data created by the scanner describes the distance to the object`s surface at each particular point as a point cloud or range data. An additional beauty of the 3-D scanner is that the direction of the range finder can be changed either by rotating the device itself, or by using a rotary mirror, which is attached to the laser range finder system. The latter method is frequently used because the mirror is much lighter than the whole system in weight. Therefore, it is possible to rotate it faster with a higher degree of accuracy. Frequently used laser range finders are able to measure up to 10,000-100,000 points in each second [7].

In many situations, a one-scan shot will not suffice to process an entire model of a given object. To accurately capture sufficient information and data on all the sides and different angles of an item, up to several hundreds of scans may be taken. All these scans must be assigned in a unique and common reference coordinate system. This procedure is commonly termed *alignment or registration*. The scans are then merged to create and form an entire 3-D model. The entire process from starting to from single range data to model the complete object is normally known as a 3-D scanning pipeline.

2.2 Types of 3-D Scanning Techniques

Mapping indoor environments using mobile robots is a well-known problem which has poised scientists for the last two decades. However, most approaches used to map indoor environments cannot directly be used in outdoor environments. The three basic reasons for such challenges when doing outdoor mapping or scanning are mainly caused by the environment representation, scale, and rough terrain. In this respect, it is obvious that the type of scanner used in the process plays a vital role in data acquisition sequence.

There are wide range of technologies, which are able to capture range data to construct the 3-D models of items and objects. They can be categorized into two main types: Contact and Non-Contact 3-D scanners. Non-Contact category can be further divided into two sub categories; Active and Passive scanners. There are many systems and technologies, which can fall under each of these groups. A vivid discussion on these will follow from this section.

2.2.1 Contact

As the name specifies, contact 3-D scanners explore the surface of the subject by actually touching the item. A coordinate measuring machine (CMM) can be a good example of contact 3-D scanner device. This device in general is used in manufacturing and it is a very accurate useful and machine. The disadvantage of CMM or contact scanning technologies is that they have to be in direct contact with the object during the scanning process. This aspect is disadvantageous because the scanning device might change or damage some part of the subject. This fact is very realistic and visible especially when the system is scanning fragile or expensive object such as historical artefacts.

Another shortfall of this type of 3-D scanning technology is that they are comparatively slower than the other 3-D scanning techniques. The reason is that they have to physically change the position of the arm to which the probe is attached. This prohibits quick movement of the system and also reduces its operating speed to only a few hundred hertz. However, an optical system such as laser scanners operates from 10 to 500 kHz [17].

2.2.2 Non-Contact Active Scanners

Active scanners transmit a kind of radiation light and detect its reflection by their receivers to probe and explorer a real-world object or environment. These types of scanners usually emit light, ultrasound or X-ray. Conversely, non-contact active scanners are used in a variety of application including indoor and outdoor environments. They form a tiny object inside the human body to scan a huge item such as rocks formation and buildings in order to produce a 3-d Model. Some prominent methods of 3-D scanning techniques are;

Light Detection and Ranging (Lidar) Scanner: A Lidar scanner can emit laser beam in different angels, direction and at different ranges. The head or main body of system can rotate horizontally and a mirror inside the device can flip vertically or vice versa. The laser beam is typically used to measure the distance to the first object on its path. The figure below is a concise view of a Lidar scanner.

Triangulation Scanner: A triangulation 3-D laser scanner is a non-contact active scanner, which emits light beam to probe the environment and surface of objects. This type of laser scanner comprises of a combination of a camera and a laser emitter system. The laser system shines or projects the visible laser beam on the subject's

surface. It then uses the camera to detect and estimate the position of the laser beam on the subject or laser dot. Depending on the distance between the surface and the laser system, the laser dot appears and detects at different places in the camera's field of view. This method is denoted as triangulation because of the laser emitter and the laser dot and camera form three corners of a triangle. Figure 2.1 shows the triangulation scanner principle.

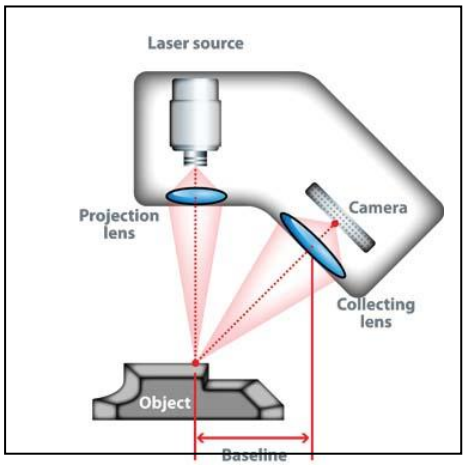


Figure 2.1: Principle of a laser triangulation sensor, [40].

Conoscopic Holography: In this system, a laser beam is projected onto the surface and then the immediate reflection along the same ray-path are put through a conoscopic crystal and projected onto a CCD (Charge-coupled device). The result depicts a diffraction pattern that can frequently be analyzed to determine the distance to the measured surface. The main advantage with Conoscopic Holography is that only a single ray-path is needed for measuring .Figure 2.2 shows the Conoscopic Holography principle.

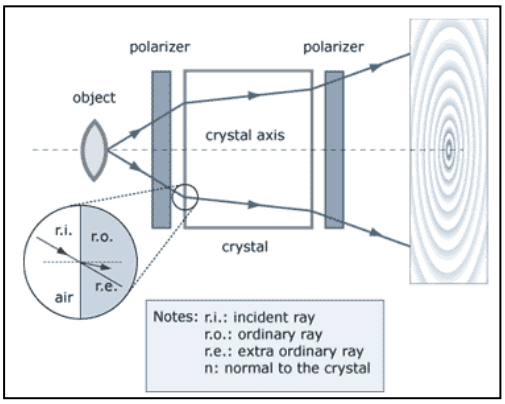


Figure 2.2: Conoscopic Holography Principle, [21].

Hand-held Laser: Hand held laser scanners generate a 3-D image by using the triangulation method described above. The laser dot is emitted to the subject's surface by a hand-held device. The system then measures the distance to the surface by using a sensor such as charge-coupled device or position sensitive device. Each particular point in the range data is collected based on the internal coordinate system of the device. The captured data can be store by a computer and be formed as a range data in a three-dimensional coordinate system. Hand-held laser scanners could also merge the range data with captured surface colours and textures to create the full 3-D model. The figure 2.3 below is an illustration of a hand-held laser scanner.

Figure 2.3: Hand Held laser scanner, [42].

Volumetric Techniques: Computed tomography is a medical imaging method, which generates a 3-dimensional image of the inside of an object by a large series of

two-dimensional X-ray images. Magnetic resonance imaging is another medical imaging technique that provides much greater contrast between the different soft tissues of the body than computed tomography (CT) does. This makes it especially useful in neurological (brain), musculoskeletal, cardiovascular, and ontological (cancer) imaging. These techniques produce a discrete 3-D volumetric representation that can be directly visualized, manipulated or converted to traditional 3-D surface by mean of an iso-surface extraction algorithm.

2.2.3 Non-Contact Passive Scanners:

Passive scanners manifest substantial differences with active scanners in that they do not propagate or emit any kind of light beams themselves. Instead, they work base on reflected ambient radiation and detect and gather information about a surface or object that is being scanned. Scanners of this type mostly rely on detecting visible light because ambient radiation is readily available in many cases and operating fields. However, they are also able to use other light source such as infrared radiations. The specification that they can operate just by using a digital camera and minimum dependency on any particular light source makes them very cheap.

Stereoscopy: Stereoscopy is a technique, which uses two video cameras located slightly away from each other. Both cameras are placed to be looking at the same object and scene. By analyzing the difference between the captured imaged by each camera, the system is able to find out and measure the distance of each particular point in the images. This technique relays on human stereoscopic vision principles [24].

2.3 Scanning Principles

The “time of flight” laser scanner or LIDAR scanner is a type of active scanners, which use the light beam of the laser to explore the subject. The heart of this type of device is a time of flight laser range finder. The laser range finder is a device, which measures the distance of object’s surface by timing the round trip of pulse of light. The laser used to propagate a pulse of light and the amount of time before the receiver detects the reflected light is calculated.

By knowing the speed of light “*c*”, and the round trip time, the travel distance of light can be achieved which is twice the distance between the device and the object’s surface. If “*t*” denotes the round trip time made by the light, then consequently the distance will be equal to (*c*. *t*)/2. The accuracy of this type of 3-D laser scanners depends on how well the device can measure the time “*t*” (Light approximately travel 1 millimetre in 3.3 Picoseconds).The laser range finder just measures the distance of one particular point in its direction of view. Hence, the laser scanner scans its entire surrounding environment one point at a time by changing the range finder’s line of sight or view direction in order to scan different points. As described above, the direction of the range finder can be changed either by rotating the device itself, or by using a rotary mirror, which is attached to the laser range finder system. Figure 2.4 shows the functionality of the simple laser scanner [17].

Generally, scanners capture the range data based on spherical coordinates. If the scanner position is assumed as the origin and the spherical angles of the vector or light beam out from the front of the system are taken as $\varphi=0$ and $\theta=0$ then the spherical coordinates of each point on the surface of the object are defined with the

distance from the origin and the angles of φ and θ. Using these three spherical coordinates, the position of the point can be clearly defined. However, in many cases, spherical coordinates is insufficient to full-fill the requirements of the modelling system and it is better to convert these coordinates into Cartesian coordinates. This transformation is typically performed by solving the linear transformation equations given in Eqn 1. Figure 2.5 shows the transformation principle from spherical coordinate to the Cartesian coordinate system.

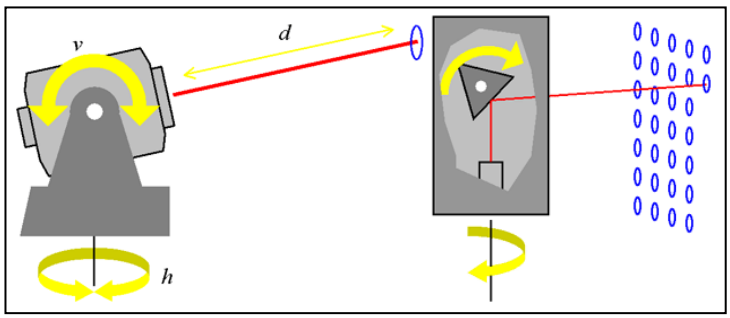


Figure 2.4: 3D Lidar or 3D laser scanner Principle

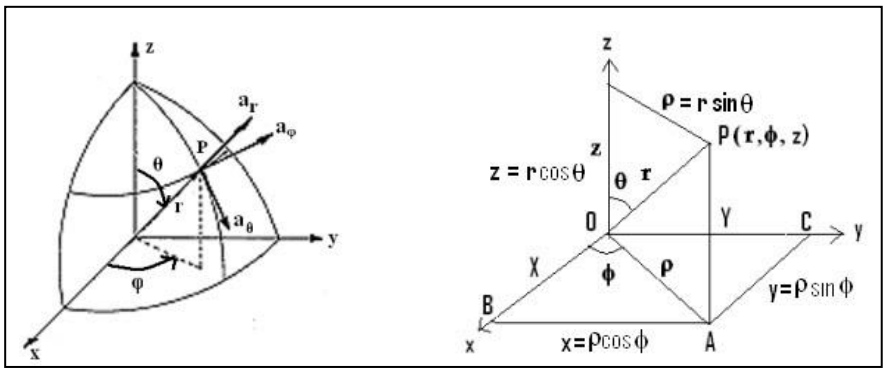


Figure 2.5: Transformation principle from spherical coordinate to the Cartesian coordinate system, [32].

$$\begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix} = \begin{bmatrix} \sin\theta\sin\phi & \cos\theta\cos\phi & -\sin\phi \\ \sin\theta\sin\phi & \cos\theta\cos\phi & \cos\phi \\ \sin\theta & -\sin\theta & 0 \end{bmatrix} \begin{bmatrix} A_r \\ A_\theta \\ A_\phi \end{bmatrix} \quad (2.1),[41]$$

2.4 Operation of The Pan-Tilt And Data Acquisition

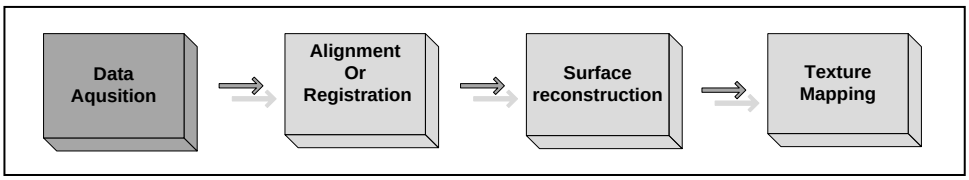


Figure 2.6: Data acquisition step in 3D modelling pipeline

As mentioned above, the direction of the range finder can change either by rotating the device itself, or by using a rotary mirror, which is attached to the laser range finder system. The latter method is more commonly used because the mirror is much lighter than the whole system in weight and hence, it is possible to rotate it faster with greater accuracy. Figure2.7 shows a sample of the device.

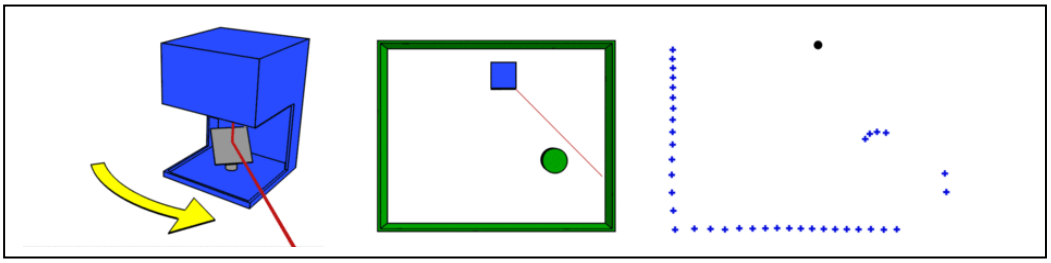


Figure 2.7: Principle of laser scanner with rotary mirror, [33].

This project however uses a commercially available 1-D Lidar (Noptel CMP3-30) for the scanning process to generate 3-D range data. This device does not contain a rotary mirror inside and has to rotate by using an external device called the Pan-Tilt to cover the whole spherical environment. This helps to overcome the limitations of a 1D scanner, which is able to scan only a particular point in its path, Figure 2.8.a and 2.8.b.

1-D Lidar (Noptel CMP3-30) uses distance measurement sensors, pulsed time-of-flight technology and integrated modules together with its own application-specific

integrated circuit (ASICs) for time calculation and signal processing. This technology allows high-speed measurement of distances from poor reflecting surfaces and has excellent resolution, Figure 2.8.c. The units are small in size, light in weight and have low power consumption. The technological solutions make the LIDAR sensors very small, reliable, and suitable to be navigated by a pan-tilt device.

Figure 2.8: (a) Servo motor based Pan-tilt`s rotation principle, (b) servo motor based Pan-tilt, (c) 1D Lidar (Noptel CMP3-30,) [35], [26].

Pan-tilt has to be very accurate i.e. it must have a high resolution, high-speed andmust be strong enough to satisfy the following three main requirements.

It must have the ability to hold the laser range finder with stability during the scanning process otherwise; it might produce the unwanted noise in scanning result.

The 1-D laser range finder is just able to scan one point at a time, however, the aim is to make 3-D scanning of the environment. Therefore, the device must be able to rotate simultaneously, both vertically and horizontally to cover the whole sphere or part of it during the process. The resolution of the pan-tilt determines the accuracy of the angles φ and θ.

In order to make a continuous scan of the environment, the laser scanner is going to be mounted at the top of a mobile robot. Although the robot has its own navigation sensors, it also needs to use the Lidar as a pathfinder and help the operator to drive and steer the system during an exploratory or rescue mission which would have low quality or inaccurate view points if a simple camera were used alone.

The actuators of the pan-tilt must be very accurate because the accuracy of the actuators has a direct impact on the accuracy of the scan result. Intuitively, a highly accurate actuator causes dense and high-resolution coordinate data, while a low accuracy actuator causes low resolution and sparse scan result or noisy point cloud. Figure 2.9 and Figure 2.10 are shown two point cloud models; the first is the result of a low-resolution scanner which is captured by servo motor pan-tilt base 3D laser scanner and the second one shows the high-resolution scan result, which is captured by Stevens University 3D laser scanner device (CAD Eye). RGB data are taken within the scanning process.

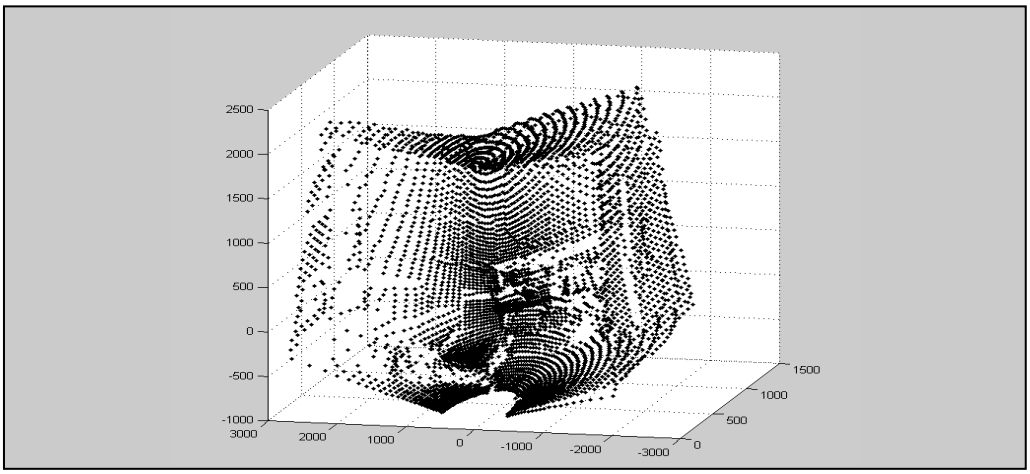


Figure 2.9: 3D model captured by servo motor pan-tilt base 3D laser scanner. Model captured from Room EE115, EMU

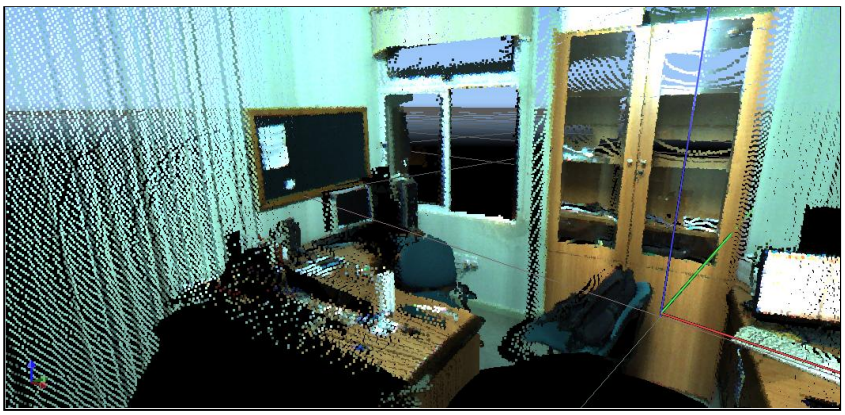


Figure 2.10: 3D model captured by “CAD eye” 3D laser scanner via tilting a 2D laser scanner device “SICK LMS”, (room A12, Techno Park, EMU)

The speed of actuator is an effective parameter in laser scanners. As earlier mentioned in this section the actuator is designed to rotate the 1-D Lidar in two orthogonal directions during the coordinate data collection process, in order to convert it into a multifunctional 3D laser scanner to be used in a wide variety of environments. During indoor scanning process, speed is not much of a challenge as in the outdoor environments. The speed of rotation plays an important role. In outdoor sites, the objects, like pedestrians, leaf of trees or cars, may have movements. The system must therefore have an appropriate speed to react and scan the whole environment, by taking the movement of the objects into account to reduce the effects of those movements in scan result. On the other hand, all the navigation system’s transmitters and controller units are active during the scanning process and they consume the power. Hence, by using the faster actuator, the system can work for longer periods and thereby decrease the power consumption.

To hold 1-D Lidar and rotate it in two directions, a light and accurate two-degree freedom of Pan and tilt has been made by using two servomotors. This system consists of the necessary brackets and hardware to allow the system to pan and tilt

the 1-D Lidar approximately over half of the sphere. Two degree of freedom pan-tilt uses two digital servomotors as its actuators. By considering the specifications of each motor, the pan-tilt is able to rotate to cover 180 degrees both horizontally and vertically, i.e. approximately half of the sphere. Figure 2.11shows the 3-D laser scanner by using a 1-D range finder and a servo motor-based Pan-tilt.

Figure 2.11: 3-D laser scanner by using a 1-D range finder and a servo motor-based Pan-tilt

The two degree of freedom pan-tilt and Lidar is mounted on top of the Remote Operating and monitoring Vehicle (ROMV) to provide a near 360° unobstructed 3-D scans. ROMV is outfitted with sensors and long-range wireless communications systems enabling Remote Operation and Monitoring with a Software-based Operator Control Unit (OCU). The robotic vehicle can also be used as a test platform for conducting studies and real-time experiments for autonomous operations. The purpose behind the ROMV will be described in more detail in the following section.

2.5 Remote Operating and Monitoring Vehicle (ROMV)

As earlier discussed, a 3-D model presents more detailed and information than a classic 2-D map, which is used in typical mobile robotic applications. This system combines the laser range data by camera output or vision data in a single model and

forms a digital textured 3-D model. However, in several situations, a one- scan shot will not suffice to form and create a whole model of the object. Mapping indoor environment systems by using mobile robots is a popular method of scanning which has been studied in the last couple of decades. However, many of these approaches are not convertible to be used in outdoor environments directly.

A relevant factor to be considered when making outdoor scanning is the scale of the environment. Most approaches for indoor mapping deal with rooms and corridors while outdoor maps need to scale to square kilometres. The user is obliged to travel hundred meters or kilometres to take enough range data sets appropriate for post processing steps and to build a 3-D model. In addition, the terrain is normally flat indoors as opposed to the case outdoors. Irregular terrain, depressions and small rocks make the task of mapping more difficult. Outdoor 3-D mapping has addressed this problem in the computer vision community for a long time now.

To overcome the difficulties in outdoor scanning, this project has purposed to use an additional device known as the Remote Operating and Monitoring Vehicle System (ROMV). ROMV utilizes the mechanical structure of a small size tracking robot platform as the frame for the Vehicle. The mobile robot is designed to carry the laser scanner during the indoor and outdoor scanning process and it can be controlled by the operator from a distance of about 100 meters from the system. Figure 2.12 shows the structure of ROMV.

During Remote operation of ROMV, multiple sensors are employed to provide the remote operator with a 360° scan range. This includes forward looking wide-angle

video cameras. The forward-looking video camera has a fixed position in front of the vehicle. In addition, it is equipped with an embedded ad-hoc wireless system, which directly transfers the video signal to the base station. This camera also has an embedded digital signal-processing unit, which allows the further image processing applications. ROMV chase also contains a very strong lifter at the centre from which its height is controllable by the user during the scanning process. The 3-D scanner is mounted over the lifter in the outdoor scanning mode. In the case that obstacles mask the line of sight of scanner or when it needs to scan the top of an object indoors, this system becomes very practical. The structure of vehicle's body will be explained in detail in the next section.

The ROMV is equipped with the C# based user interface, which enables the operator to communicate by each part of the design through the wireless system. Section [4.2] will describe the electronic hardware structure of design in detail and as follow. The structure of the C# based user interface remote operating and monitoring (ROAM) vehicle system is going to be explained in detail in the latter chapters

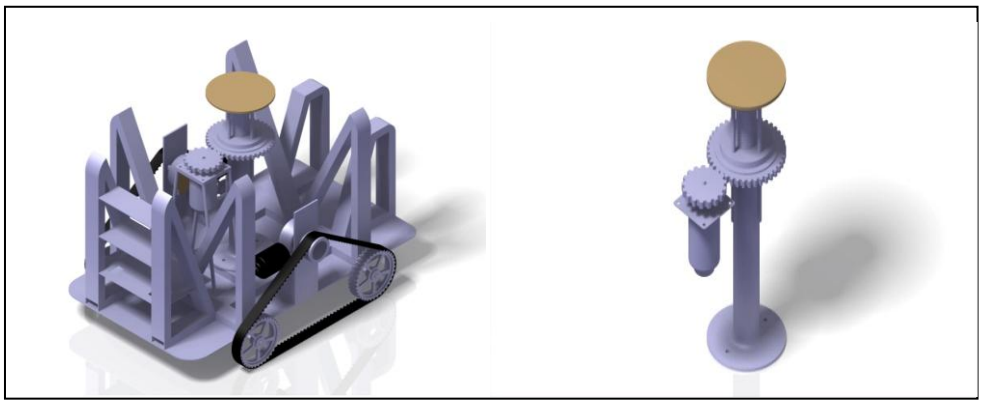


Figure 2.12: ROMV and lifter scheme

2.6 Obtaining Coordinates from Scanned Data

Generally scanners capture their range data based on spherical coordinate system. If this coordinate system is defined, the scanner becomes the origin and the vector or light beam out from the front of the system has the coordinates of φ=0 and θ=0 and each point in the cloud point is associated with φ and θ. In addition, the measured distance corresponds to the *"r"* component. With this therefore, the spherical coordinate system is completely ready for use. The positions of each particular point in the point cloud are defined in a local coordinate system with reference to the device coordinates. When the number of sample points in each vertical scan line is known, then the total number of vertical scanned lines can be specified. Systematically the number of sample points can be divided by the vertically rotation angel of the laser scanner. By this it becomes easy to find out the exact inclination (or polar angle θ which is the angle between the zenith direction and the line segment OP) of each point. In addition, by dividing the number of vertical scan lines horizontally, the rotational angel of the pan-tilt or the azimuth angle φ (is the signed angle measured from the azimuth reference direction to the orthogonal projection of the line segment OP on the reference plane) of each point can easily be obtained. The Figure 2.13 shows a practical view of the system described here.

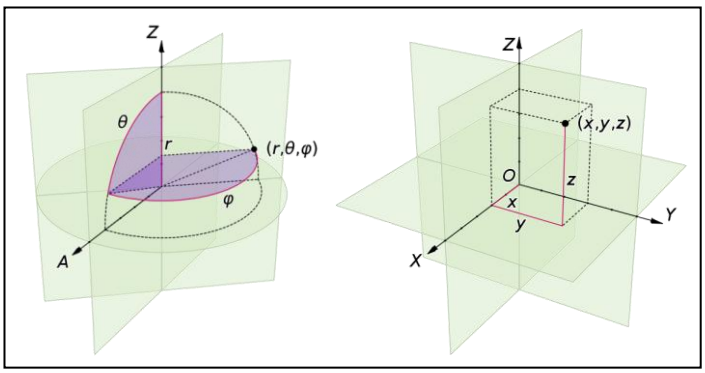


Figure 2.13: Convert spherical coordinates to Cartesian coordinates, [23].

Knowledge of these three spherical coefficients (r, θ, φ) for each point makes it possible to draw a graph or form the point cloud model in a spherical domain. However, these requirements cannot sufficiently satisfy the objectives of this project. The point cloud information is used in the further post processing phases for surface reconstruction and alignment. Many surface reconstruction applications have been published in the past and most of them have applied their methods on Cartesian based range data instead of spherical coordinates. Therefore, in this project we also convert all spherical coordinates to Cartesian coordinates to make them compatible with other readily available methods.

The conversion can be considered as two sequential rectangular to polar conversions. The first is the Cartesian x–y plane from (x, y) to (r, φ); where "r" is the projection of r onto the x–y plane, and the second in the Cartesian z-r plane from (z, r) to (r, θ). The correct quadrants for φ and θ are implied by the correctness of the planar rectangular to polar conversions.

The basic assumption considered in these formulae is that the two systems have the same origin. The spherical reference plane is the same as the Cartesian x–y plane, i.e., θ is inclined from the z direction and that the azimuth angles are measured from the Cartesian x-axis (so that the y-axis has φ=+90°). If θ measures elevation from the reference plane instead of the inclination from the zenith, the arc-cos above becomes an arc-sin, and the cos θ and sin θ below become switched.

$$r = \sqrt{x^2 + y^2 + z^2}$$
$$\theta = \cos^{-1}\left(\frac{z}{r}\right) \qquad (2.2)$$
$$\varphi = \tan^{-1}\left(\frac{y}{x}\right)$$

Conversely, the Cartesian coordinates may be retrieved from the spherical coordinates (r, θ, φ), as follows:

Where $r \in [0, \infty)$, $\theta \in [0, \pi]$, $\varphi \in [0, 2\pi)$,

$$x = r\sin\theta\cos\varphi$$
$$y = r\sin\theta\cos\varphi \qquad (2.3)$$
$$z = r\cos\theta$$

2.7 Alignment or Registration

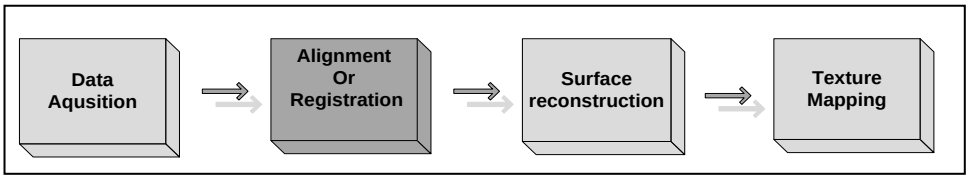


Figure 2.14: Forming the Point Cloud Model (Alignment or Registrations)

The output of the laser scanner device contains x, y, z parameters of Cartesian domain for each scanned point. However, there has been no model formed until now and there are just raw data in hand. To form a 3D model from the x, y, z coordinate data, several methods have been produced to graphically form and present the range data sets. One of the well-known and widely used methods in this step is the point cloud presentation. This method forms the 3-D graphic model of the object or scanned environment by arranging all sample data corresponding to their Cartesian coordinates in 3-D graphic space. In this model each sampled data appears as a particular dot in 3-D space and collection of all these dots or points formed a 3-D point cloud model of the scanned object or environment. Hence, forming a point

cloud model or a range data is not in fact a challenging step. Based on the platform or environment in which the graphical process is made, the user can load and plot the point cloud model easily. This project uses the Mat-lab environment as the graphical environment and point cloud model is formed by using the "***scatted3*** and ***plot3***" commands.

In many cases, a one-scan shot will not be enough to process a whole model or object. Many scans have to be taken from several different angels and directions to capture information and data about all sides of the item. All these scans have to be assigned to a unique and common reference coordinate system. This process is not only used to form a unique and complete model of the scanned object or environment but also to approximately create a seamless point cloud model. This procedure is commonly called alignment or registration.

However, in most cases the created point cloud model may suffer from the sparse noisy data sets or overlapping problem, which may be caused by an insufficient or inaccurate merging process. In order to eliminate this problem, many techniques and algorithms have been proposed. Such techniques use different types of noise redundant and merging techniques where two closed point algorithms are mostly used. Hence, this step plays an important role as a pre-processing step through the surface reconstruction processing and modelling sequences.

Alignment methods are categorized in to three main categories: manually, fully automatic and semi automatic.

- In the manual method, the user tries to adjust and align two or more different range data taken from the same environment or object but from

different angel or points of view. The common features in cloud points is that the user is able to rotate, shift, scale or change the coordinates of each data set and find the best position to align, merge or stitch the cloud points.

- Full automatic methods are methods, which do not need any contribution by the user during the registration process. These methods try to find the common features and attributes in each cloud points separately, like sharp features, cones, edges and curves. Subsequently, they use one of the prominent estimation and detection methods to find the common attributes between them. By applying the corresponding transformation and translation methods, they finalized the alignment procedure. These types of algorithms and methods still have a problem with speed and accuracy and the number of vertices or points. Poisson alignment method is a good sample for field work, [44].

Semi automatic methods are the combination of manual and full automatic methods. In the first step, the user selects and specifies the most similar points in both range data manually and then software matches the remaining points automatically based on common feature estimation and detection methods. Point-tools software is a good example in this field. This project has merged the range data sets by using the Mesh Lab tools software [44] and with the semi-automatics method. Figure 2.15 shows a good example of point clouds merging sequence. This figure shows three different point clouds captured form different angels of the entrance of EMU Techno park building. As it is appearin Figure 2.15 each of these point clouds is just able to cover a part of this building. Hence, in order to cover the entire building, these point clouds

aligned and merged to gather in a unique coordinate system, a) front captured range data. b) Left corner captured range data c) right corner captured data .d) the merged data sets.

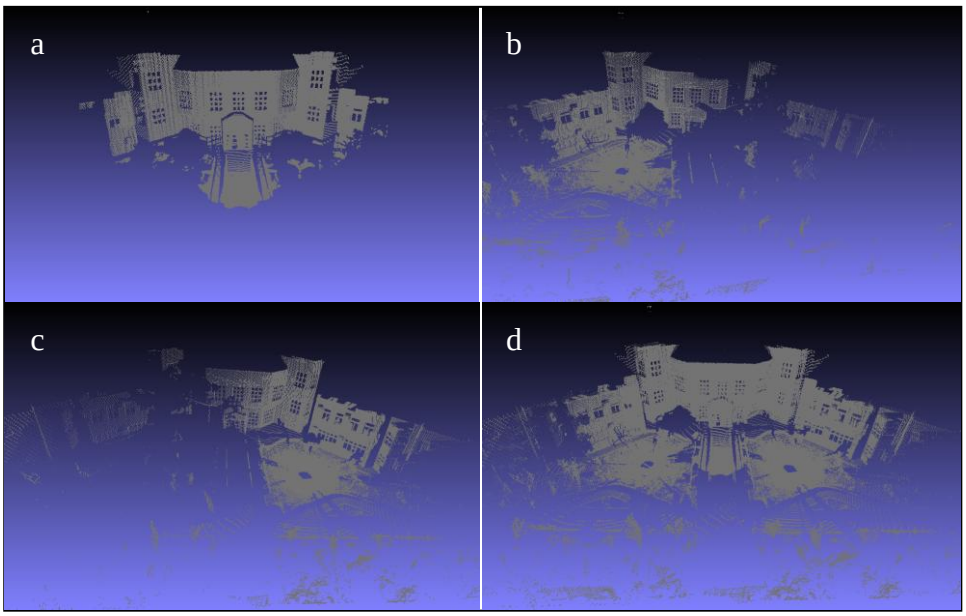


Figure 2.15: Point clouds merging sequence, (EMU, Techno Park, front entrance)

2.8 Surface Reconstruction

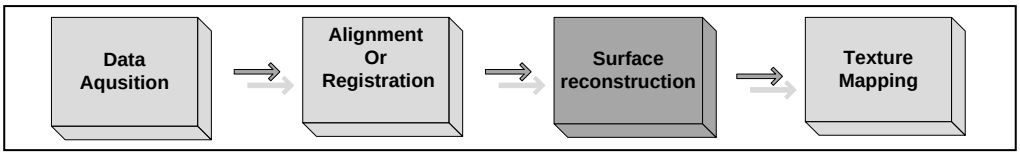


Figure 2.16: Surface reconstruction step in 3D modelling pipeline

Surface reconstruction mainly refers to the technique, which tries to estimate and reconstruct the arbitrary surface topology from the point cloud model. In the past decades, there have been numerous algorithms published for the purpose of reconstructing the surfaces from the cloud points. Their goal is to accommodate smooth, seamless and dense representation of the surface. In most cases, there is no visualization or pre-knowledge available about the structure of these points except their coordinates. Hence, before starting to reconstruct a surface, it is necessary to analyze the range data in order to extract more information about their orientation,

smoothness and the connectivity or relation of points in space. This information then will be applied to surface reconstruction method as a data to perform a smooth, seamless and reliable surface. The structure, accuracy and types of information, which can be extracted by these methods, are different depending on the type of the reconstruction method. Figure 2.17 shows the main sequence of a 3-D surface reconstruction based on unorganized point cloud information.

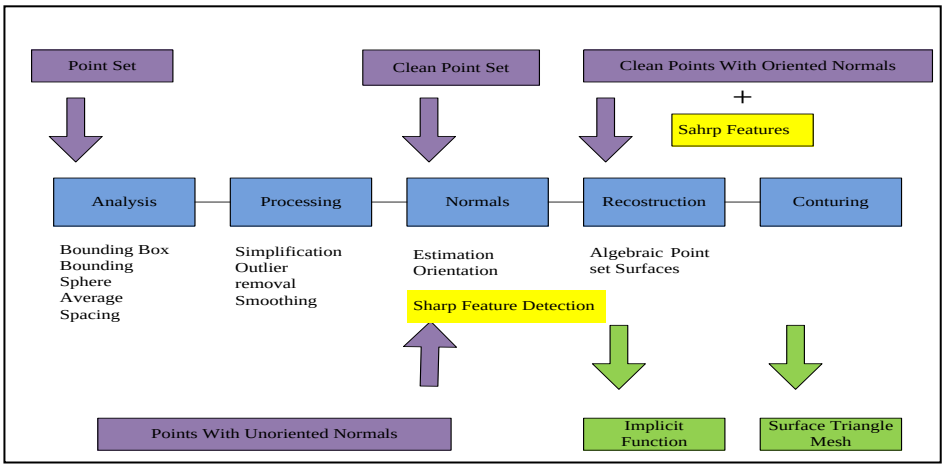


Figure 2.17: Main sequences of point cloud based 3-D surface reconstruction [1].

The next section will briefly explain the different types of surface reconstruction methods and review the proposed algorithms in this field. The following chapter [3] will then clarify the aim of this project by showing the type of point cloud model, which is available for the preferred reconstruction method. Finally, the method and the algorithms behind our method will be explained in detail in section [3.3]

Chapter 3

DESIGN AND IMPLEMENTATION OF A 3-D PAN-TILT CONTROL SYSTEM

3.1 Design Principles

Specific hardware and software has been designed and constructed in order to create a multifunctional 3D laser scanner with appropriate speed and accuracy. This design uses a commercially available 1-D Lidar (Noptel CMP3-30) to generate detailed 3-D scans. The 1-D Lidar is mounted onto a two degree of freedom pan-tilt on top of the robotic platform thus enabling it to rotate the Lidar in two orthogonal directions , Figure 3.1. This two degree of freedom pan-tilt allows for ample flexibility in the orientation of the 1-D Lidar thereby enabling a single 1-D Lidar scanner to be used for a variety of 3D scan applications. By acquiring the simultaneous position information of the actuators during each 1-D Lidar scan, and controlling the actuators to change the orientation of the Lidar, 3-D scans of an environment can be generated. The scanning system mounted on top of the ROMV provides a near 360° unobstructed 3-D scan of the environment. ROMV is out-fitted with several sensors and a long-range wireless communications system enabling remote operation and monitoring with a software-based Operator Control Unit (OCU). The OCU enables mapping of large scale or human inaccessible environments by a single operator from a remote location. The robotic vehicle, which is also used as a test platform for conducting research and real-time experiments on autonomous operations in

manufacturing labs. The development of ROMV will be described in more detail in the following sections.

3.2 Construction of the Pan-Tilt Mechanical Parts

As explained above, a very simple, accurate and light two-degree of freedom of Pan and tilt is designed to hold a Lidar and rotate it in two directions by two servomotors. The pan-tilt base panel is made of a 3- millimeter aluminum sheet. This system consists of the necessary brackets and hardware to allow system to pan and tilt the 1-D Lidar proximally over half of the sphere. The two degree of freedom pan-tilt uses two digital servomotors as its actuators. It appears that the pan-tilt is able to rotate and cover 180 degree both horizontally and vertically approximately half of the sphere considering the specifications of each motor. Given the pulse width, range of each motor (800-2200usec) and dead bandwidth value (3usec), the system can collect 466 points horizontally and 466 point vertically in each complete process. The 466 samples per 180 degree (0.28 degree) is the maximum possible resolution of this pan-tilt. This pan-tilt can collect 217156 points in each scan process. This also provides very dense, accurate and low-noise point cloud information. Figure 3.1 shows the pan-tilt, 1-D Lidar and the Digital servomotors.

The scanning process is controlled by a micro-controller based operating and control unit (OCU). This system controls the laser range finder and pan-tilt during the scanning process.

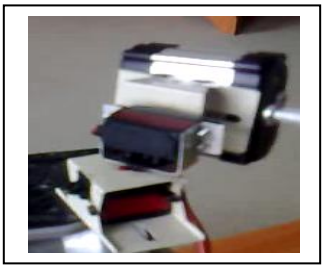


Figure 3.1: Designed servomotor base 3D laser scanner by using 1D Lidar (noptel)

3.3 Microcontroller based Scanner Operating and Controlling Unit (OCU)

This section introduces the 3-D scanner controller system which contains four main parts. Figure 3.2 shows the detailed Block Diagram of the desired controller system.

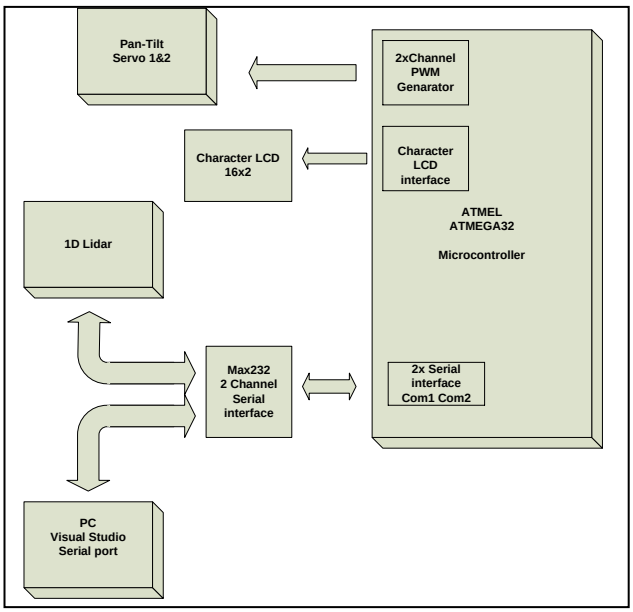


Figure 3.2: Microcontroller based OCU(Operating and Controling Unit) main diagram

This system has been designed based on an ATmega32 AVR microcontroller, which works as a data acquisition and control system. The microcontroller initiates the pan-tilt actuators to stand at home position by producing the PWM pulses for each Servomotor, which is directly connected to the microcontroller's pins. The 1-D Lidar that works with serial protocol and the 9 pin RS232 connector is connected to the microcontroller's serial port by using the max232 interface. Through this channel the microcontroller is able to trigger the Lidar and is able to read the distance values which have been detected by the Lidar. The microcontroller saves the lidar-measured result in a serial buffer and the current coordinates of the pan-tilt can be monitored on the LCD. The user can check and track the results and send them to the PC

through the serial port. The user interface software is developed with Visual C# and it is able to read the data from the microcontroller and save them in the related folder. At the end of each measurement, the PC sends an acknowledgment signal to the micro controller and becomes ready for next measurement phase. After getting acknowledgment from the PC, the microcontroller triggers the pan-tilt for the next movement. It thus starts to produce the PWM (pulse width modulation) pulses accordingly and this sequence is iteratively continued as long as the scanning process carry's on repeatedly.

3.3.1 Driving Servo Motors (PWM Generator)

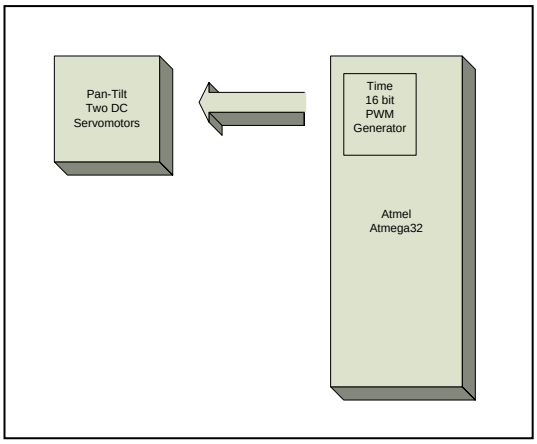


Figure 3.3 : OCU main pan-tilt dirving block diagram

A servomotor consists of a small DC motor, a small set of gears, a potentiometer and an electronic circuit for controlling the position of the motor. Each servo has three wires where two wires are for the power supply input and the third is used to receive the PWM control signal. A series of comparative control pulses are sent to the servo from the OCU through the signal wire.

A servo receives the pulse from the OCU each 20ms. The servo must hold that position for the next 20ms until the next command recived. Even if the forces from

the 1D Lidar are trying to make it move, the servo must resist to keep its position. . Figure 3.4 shows a DC digitl servo-motor timing diagram.

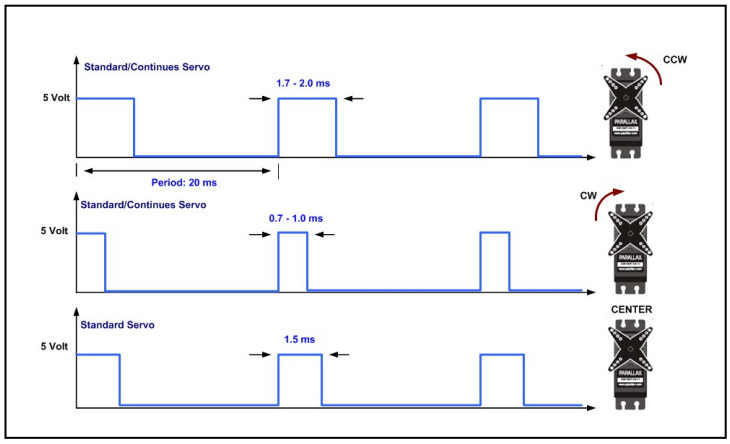


Figure 3.4: DC digitl servo-motor timing diagram, [5].

The PWM control pulses have to be generated without any conflict from other peripherals of the micro-controller that are parallel to the main routine. Hence timer1 has been used to produce a stable and accurate PWM waveform. An internal crystal, which is set to 8 MHz, directly seeds Timer1 and it has a 16-bit register. Once he program starts, this register starts to count from 0 to 2^{16} (0 -65535). Each step takes 1/Clock pulse per second and when the register is full, it will overflow. After $65536 \cdot (\frac{1}{ClockPuls})$seconds, the overflow flag OVF1 is raised. OVF1 register has its own subroutine which will be active whenever the over flow has occurred. This sequence will continue concurrently with the main routine until the end of program without any conflict. Using the timer/counter structure makes the user able to provide very accurate timing for different applications like PWM generators and delay. Figure 3.5 shows the flow-chart, which represents how micro-controller produces two PWM signals by using timer1.

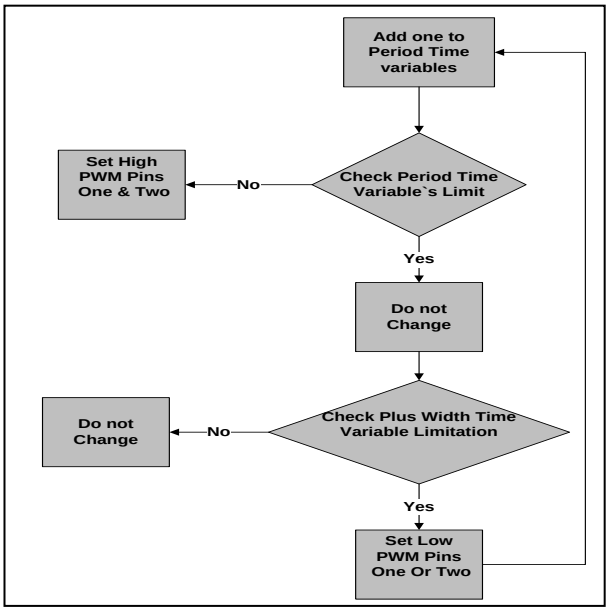


Figure 3.5: OCU, PWM generator flow chart

In this routine timer1's initial value is 65500. This means that after only 36 steps, this register becomes full and the OVF1 flag becomes active. Knowing that the internal clock is set to 8 MHz, 36steps makes 0.000549 sec or approximately $5.5.10^{-4}$ sec. When the timer1's subroutine becomes active, the timer-counter increases once in each occurrence. The maximum value of the timer-counter variable is 806. This implies that after 806 times that overflow occurs at a row, approximately 20 mille seconds delay is provided ($806 \times 5.5 \times 10^{-4} = 0.46$ msec) around 50Hz. It appears from the above case that two variables, timer1 register value and timer-counter value are used for controlling the Fixed 50 Hz frequency of the PWM generators. When the timer-counter has zero value, both PWM outputs (portA.0 and portA.1) become one (5Volt) and when the timer counter attains the maximum value of 806, both outputs become one again (5Volt).

For making the PWM generator, the subroutine needs another variable to control the width of rectangular waveforms. As such, additional variables (pwm, plm1, plm2) have been defined accordingly. The variable pwm is increased each time when the

timer1's subroutine is executed and it acts like timer-counter variable. Whenever the pwm becomes more than the specific value, (plm1 for PWM channel1 and plm2 for PWM channel 2), the relative port becomes zero and stays zero till the timer-counter becomes zero again. By changing the plm1 and pml2 during the main routine, it is possible to achieve the different pulse width. An issue worth mentioning is that the pwm variable and the timer-counter make same period time and by increasing or decreasing plm1 and plm2 variables one step, the Pulse width will be changed only by 5.5^{-4} seconds. Figure 3.6 clarifies the whole picture.Appendix A contains PWM generator`s Bascom command lines.

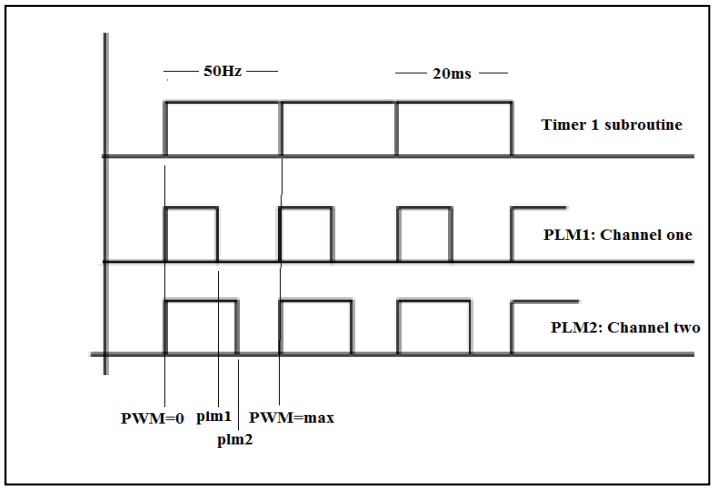


Figure 3.6: PWM generators timing diagram

3.3.2 Universal Asynchronous Receiver/Transmitter (UART)

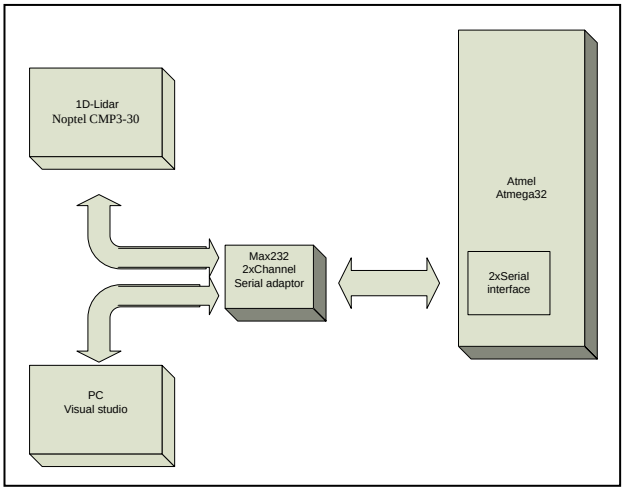


Figure 3.7: OCU main serial communication block diagram

The previous section has explained the functions of the controller system. This section presents the way in which the main controller makes an interaction by the personal computer and the 1D Lidar.

The 1-D Lidar (Noptel) and the Personal computer have a DB-9 connector as the serial input/output interface. This enables the user to send the necessary commands like the trigger and calibration commands, and also to read the corresponding information and acknowledge or send the measured distance values information through it to the connected device. A Universal Asynchronous Receiver and Transmitter (UART) can be used to send and receive data between two devices. More especially, these devices can be PC-to-PC, PC-to-micro controller and micro controller-to-micro controller. The UART communicates using TTL voltages +5V and 0V or LV TTL depending on micro controllers VCC voltage.

For connecting a microcontroller to the PC and the Lidar, it needs to use the RS232 protocol specifications. This means that the hardware communication is made with specific voltage levels, such as +15V and -15V. This can be achieved by using a MAX232 level shifter. The hardware circuit layout is shown in Figure 2.8.

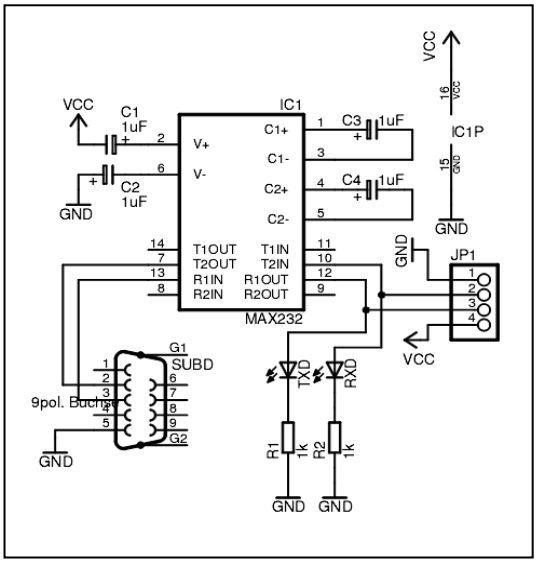


Figure 3.8: MAX232 PC to Microcontroller connection Schematic, [2].

The DB-9 connector has 9 pins but only 3 of them are used in serial communication, namely; GND, TX and RX. Through these hardware channels, micro-controller starts to send the trigger command to the Lidar (with the letter A). After detecting the trig command from the microcontroller, the Lidar becomes active and starts to measure the distance as explained earlier. The results of the scanner are then sent to the receiver port of the microcontroller as an acknowledgement. The Micro-controller displays the result to the user via a 16 x2 LCD display. The received information by the microcontroller is also sent to the PC through the other serial channel. These information or measured values then appear on the screen and are used by the C# software in the post processing steps. C# program structure and it`s attributes are going to be discussed later in chapter 4. During this procedure, after each step, the microcontroller changes the pan-tilt's direction accordingly, figure 3.9 shows the OCU main flow-chart.

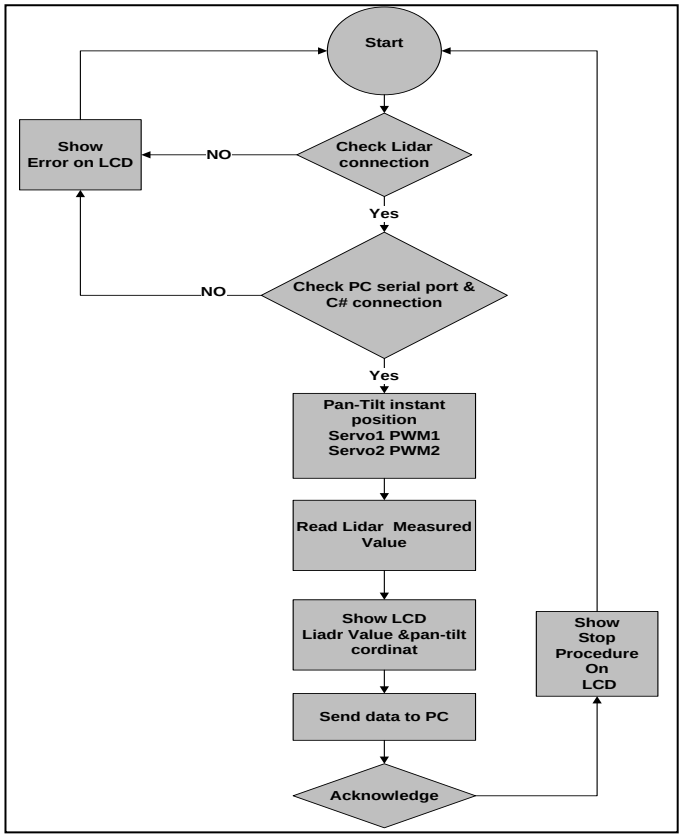


Figure 3.9: OCU main flow chart

For serial communication, devices need to follow not only the hardware rules but also the software protocols. Therefore, the band rate, parity, stop and start bit have to be set according to the specification of each device and the manner of communication. These settings are easily made using the Bascom-AVR compiler. If the controller has no UART or an additional UART is needed, the software UART can be an alternative, which makes the operator able to connect the Microcontroller to more than one pc or device. The software UART is not as robust as the hardware UART thus it might cause timing problems if the main routine has many interrupts in its body. This project uses both Hardware and software UART. The hardware UART is used for connecting the micro-controller to the PC and the Software UART is used for connecting the micro-controller to the1-D Lidar scanner [4].

3.3.3 Collecting and Measured Data From Scanner Through the C# User Interface

In this section, the user interface window, which enables the operator to communicate with several parts of the system to detect and save the measured distance values by the 1-D Lidar, is explained.

The user interface software has a very close and comprehensive interaction with the Micro-controller based pan-tilt control system to handle whole 3D scanning processes like triggering the Lidar, changing the pan-tilt direction, collecting the measured data and saving the results during the scanning process. It is therefore evident that all these processes should be controlled and monitored with sophisticated software for the system to perform properly and accurately. This software environment is produced by using the abilities of C# language running on the PC, which is linked, to the system through the wireless communication unit. This

section explains the design of the window and the fundamental communication structure of the user interface.

The main window of the user interface has been split into two parts. The right hand side contains the control keys: Start communication, Stop communication and Save to Excel. The left hand side of the form contains a dialog-box, which appears as a rectangular window on which the measured values are monitored, in Figure 3.10, which will be explained in detail in the following section.

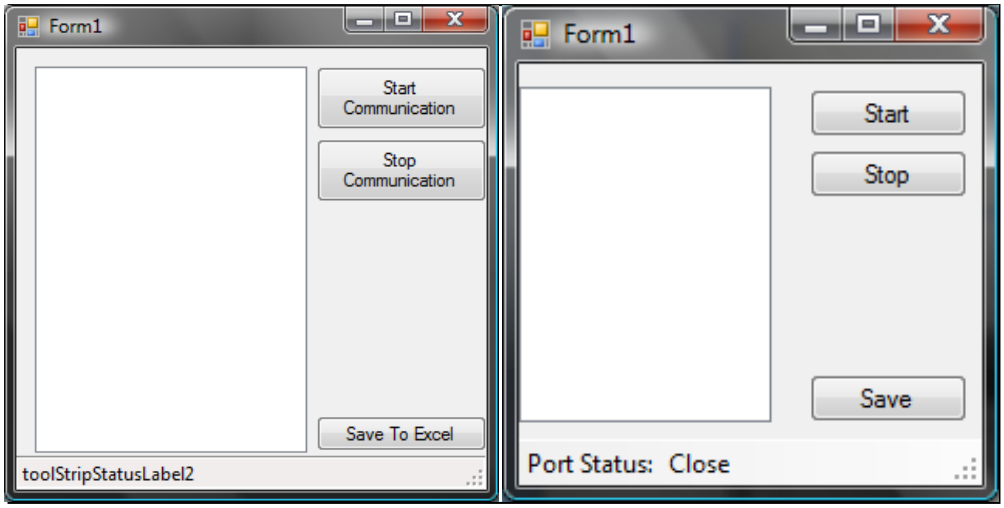


Figure 3.10: Pan-tilt C# based user interface contoller

By pressing the Start Communication button, the user interface opens the serial com port1 of the PC, which is set as a defaults RS232 communication port. In order to ensure that the port is opened, the other dialog-box at the bottom of the page should show the Port Status. By opening the Asynchronous serial port, the user interface is connected to the Micro controller based control system through the wired or wireless channel. It has to be mentioned that the type of serial channel has no effect on the software and hardware design. The RS232 wire based channel can simply be replaced by a serial wireless system. In this project, the RS232 cable is used during

the development stage, after which it was replaced by the Handy Port HPS–120 in the final design [18].

When two system starts to communicate and interact with each other, the user interface sends a ready command to the micro-controller by sending a pre-defined 8-bit number and waits for the Micro controller's acknowledgment. After sending these commands, two devices become synchronized with each other. The control system then starts to adjust the pan-tilt's direction after capturing the first distance value. Subsequently, it triggers the Lidar to take the distance measurement. This measured value is then received and buffered by the micro controller. At this time, the C# user interface program starts to log the received data by using the Excel software and simultaneously monitors the Excel file at the dialog window at the left hand side of the main page. During the login time, microcontroller stays in the stand-by mode until the software sends the confirmation message (which means that all data has been received successfully) and detected by the micro-controller. This sequence continues until the last distance value is captured and micro controller sends the stop process command.

The operator can save the excel file at the pre-defined path on the PC by pressing the Save Excel button. These saved results are used by Matlab platform for checking the results during the implementation time or drawing the point cloud model and surface reconstruction in following steps. The user can cancel the whole procedure by pressing the Stop Communication button if something wrong happens during the process. The Software will close the Asynchronous serial channel and then

subsequently, the control system will detect this occurrence and automatically stops the scanning procedure.

3.3.4 Linking Matlab Platform and Excel File to the User Interface

The previous sections explained the whole user interface software that controls the ROAMV by the specified arrow keys and joystick, the way of Scanning and saving the data in excel file. Up to now, the system has just developed the Excel file and does not portray any further meaning of the complete result for human perceptions, like a graph or a point cloud model.

The rest of this section explains the mathematical procedure on Matlab environment, to produce the point cloud model from the measured coordinates and linking the M.files to the user interface software.

By pressing the Save to Excel button, the software automatically starts to save the measured data to an Excel file in specific path on PC. The related Matlab environment is called by the program in C# language and a very simple M-file routine starts to retrieve the pre-measured values from the excel file and plot them on a point cloud model. In The first step, Matlab starts to extract the polar angle θ and azimuthally angle φ row by row as explained before by knowing the number of samples at each column and the number of columns.

The second step converts the spherical coordinate to the Cartesians coordinate by a very simple pre-defined Matlab command (spher2cart). In this state, the Cartesian coordinates are available and it is possible to draw a 3-D graph of the ranges data. There are two different positions which the user interface can draw the point cloud model; first, it can plot the points whose coordinates are obtained and then update the

graph for each new point. Second, it can draw the whole graph at the end of the procedure.

Figure 3.11.b shows the result of systematic point cloud drawing within the procedure and Figure 3.11.a shows the whole graph at the end of converting procedure. Since drawing the graph within the converting sequences and updating it for each point is a very time consuming process, this project has adopted a way in which the whole procedure is completed before the graph is drawn.

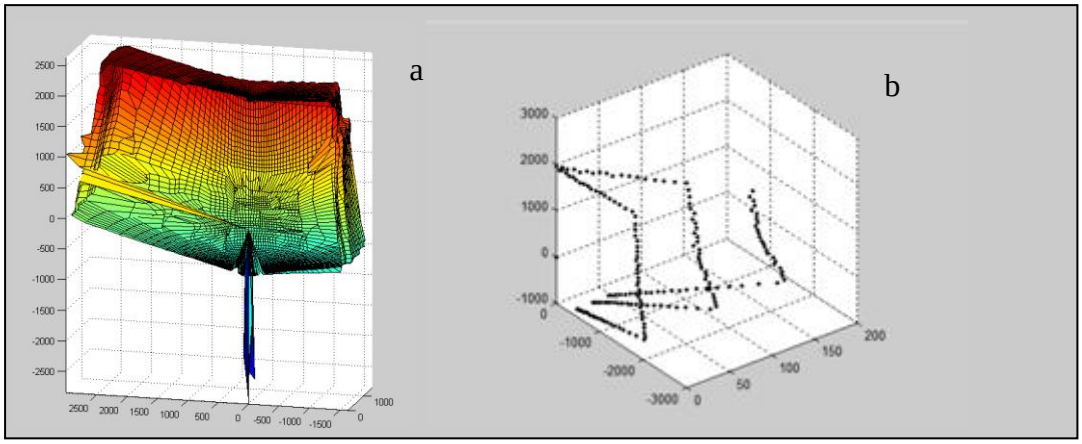


Figure 3.11: The user interface data monitor window, 3D scanning has performed on room EE115, EMU. (a) Forming point cloud at the end of converting procedure, (b) forming point cloud within data acquisition procedure

Chapter 4

DESIGN AND IMPLEMENTATION OF REMOTE OPERATING AND MONITORING VEHICLE BODY (ROMV)

4.1 Design Principles of ROMV

As mentioned before, the mobile robot is designed to carry the laser scanner during indoor and outdoor scanning process. It is generally controlled by the operator who may stand about 100 meters away from the system. During remote operation of ROMV, multiple sensors are employed to provide the remote operator with a 360° situational awareness. There is a one wide-angle video camera (forward looking), which is fixed in front of the vehicle. In addition, it is equipped with an embedded ad hoc wireless system, which directly transfers video signal to the base station. This camera also has an embedded digital signal-processing unit, which allows for further image processing applications.

ROMV chase also contains a very strong lifter at the centre with its height being controlled by the user during the scanning process. The 3-D scanner is mounted over the lifter in the outdoor scanning mode. In case where the line of sight of the scanner is masked by an obstacle, or when scanning the top of object indoors, the device becomes very useful. The structure of vehicle body is going to be explained in detail in section 4.2. The ROMV is equipped with the C# based user interface, which

makes the operator able to communicate with each part of the design. This then collects the measured distance values by Lidar, shows the final point cloud model, controls the rover by Joystick, keyboard's arrow keys and watches the video streams of bird-eye wireless camera during the rover driving process. This will ensure users to have a better point of view in the case of path finding. The structure of the C# based user interface remote operating and monitoring vehicle system is going to be explained in detail in section 4.4

4.2 Construction ROMV Vehicle Body`S Mechanical Part

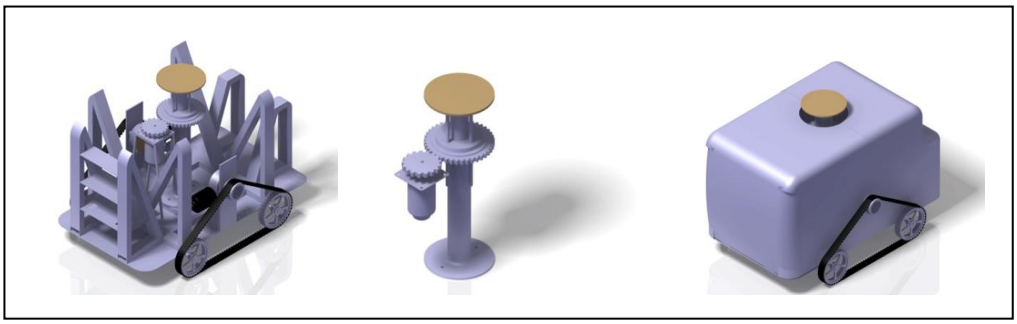


Figure 4.1: (a) ROMV chase, (b) lifter structure, (c) ROMV and body cover schem

ROMV utilizes the mechanical structure of a small size threaded robot platform as the frame for the Vehicle. Threads are used to reduce the slip and more evenly distribute the weight of the robot, thus making it useful for loose surfaces such as sand and gravel. In addition, a thread system with some flexibility can better conform to a bumpy surface. Continuous contact with the ground prevents slipping that may have occurred if wheels were used. The evenly distributed weight helps the robot to tackle a variety of surfaces easily. In addition, it will significantly increase the robot's ground clearance without incorporating a larger drive wheel.

Using the mini tank chase robot makes the system able to explorer the indoor and outdoor environment without a need for any change in the structure. Although it

looks heavy when considering its size, the vehicle platform is completely made of steel. This makes it very stable and safe for carrying the very sensitive hi-tech scanner and the embedded electronic control devices. Figure 4.1 shows the ROMV vehicle body.

The vehicle is driven using two wiper motors, which do not oscillate back and forth but instead rotate continuously in one direction. This type of motor is called a "gear head" or "gear motor" and has the advantage of producing a high torque. Tests using one wiper motor and a torque wrench have shown that at 12 volts on high speed, the motor produces a torque 13.5 pound-feet and on low speed, it has a 17.5 pound-feet of torque. The output shaft of each gear is directly connected to the threaded wheels [45].

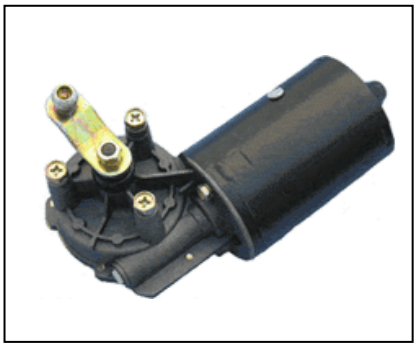


Figure 4.2: Wiper motor, [45].

4.3 Construction of Electronic Parts

4.3.1 Motor drive system

For driving purposes, the motors have to be connected to the system which enables them to control the speed and direction of each motor. TReX Dual channel DC motor controller is used for this purpose, Figure 4.3. The TReX Dual-Motor Controller is a

adaptable, high-power DC motor controller designed to seamlessly merge autonomous and human control of small and medium-sized robots.

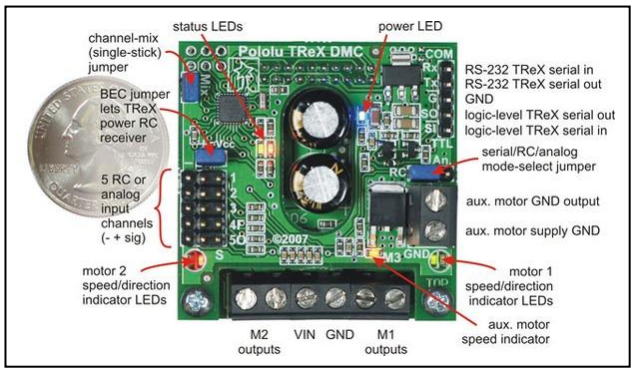


Figure 4.3: TReX motor control, [37].

The TReX receives its power through the VIN/GND terminals. The input voltage has to be in arrange 6 to 16 V and the power source must be able to supply the current motors. This controller can supply peaks of 30A and up to a continuous 13A to each of its two-bidirectional motors [37].Figure 4.4 shows the TreX single battery motor connections.

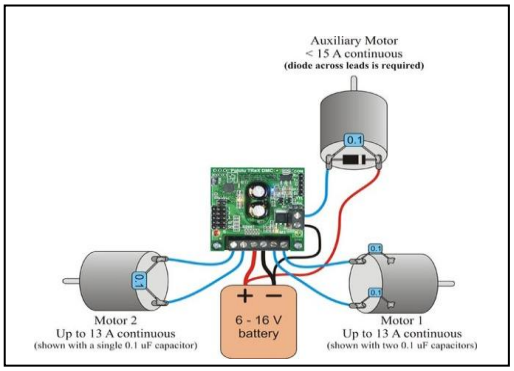


Figure 4.4: TReX motor connections (single battery), [34].

4.3.2 Wireless Communication System

Handy-Port HPS–120 with a medium range communication capability (100 meter) is a 2.4GHz ISM wireless cable replacement for RS–232 devices. Plugging one Handy-Port into RS–232 device and the other into the PC and it becomes a wireless replacement for the serial cable with a range of up to 100m. Power for the Handy Port attached to the PC is supplied by PCs spare USB ports. Power for the other Handy-Port (attached to RS-232 device) can be supplied in one of two ways: From an external-VDC power adapter or via the DB9 connector from the RS-232 device's own power supply.

Figure4.5 shows the Handy-Port HPS–120 .

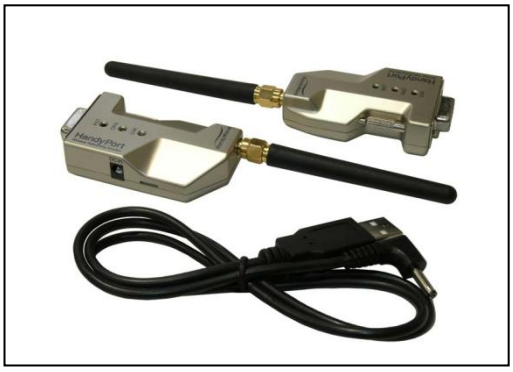


Figure 4.5: Wireless transmitter Handy-Port HPS–120, [18].

At the PC part, the motor control system and the User interface, which is controlled by the operator, communicate through the wireless system to operate the robot. In the following section, the software and the specifications of the Graphic User Interface, which is written by C #language, will be explained.

4.3.3 Forward Looking Wireless Video Camera

During Remote operation of ROMV, multiple sensors are employed to provide the remote operator with a 360° position information.

A commercial wireless SRV-1 Blackfin Camera is used as the forward-looking camera, Figure 4.6. This is a bundle of Two-board sets which include the Blackfin Camera with 500MHz Analog Devices, Blackfin BF537 processor, 32MB SDRAM, 4MB Flash, an Omni vision OV7725 VGA low-light camera, along with combo radio/motor board, Lantronix Match port b/g radio module and an antenna. These are essentially all of the electronics of a survivor robot.

Figure 4.6: The SRV-1 Blackfin Camera, [9].

The SRV-1 Blackfin Camera transmits the video signals through the wireless channel to the PC at the base station. The operator can monitor the video stream on the screen. The C# program has a window, which enables the user to connect to the robot's camera and watch the video stream. The software section will introduce the structure and specifications of the C# program.

4.3.4 Battery Power Supply

two 12 NI-MH , 4000mAh battery pack are used (connected as two 12V parallel battery banks) to power the scanner , Pan-tilt Servo motors , rover motors ,motor driver , wireless system, forward looking camera , its transmission system and other onboard electronics.

DC-DC converters are used for stepping down the 12V to lower voltages that are needed for low power electronics. Each part of the structure has its own voltage regulator, which can tolerate more than 15V DC input voltage. So it is unnecessary to regulate the voltage for each of them. The vehicle can operate for approximately 3 hours on a single charge with this battery configuration.

4.4 Software User Interface Structure

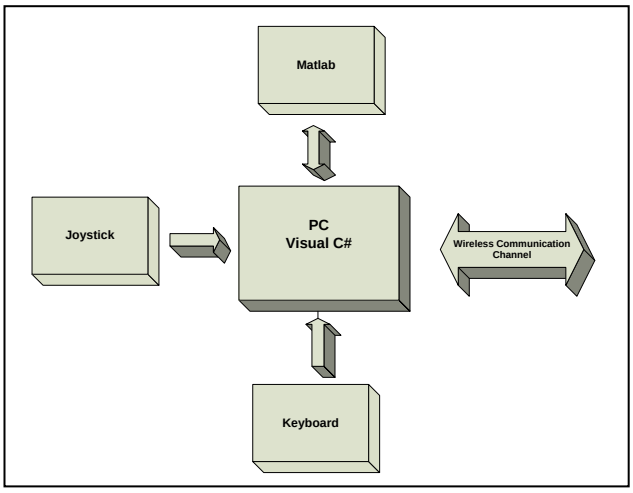


Figure 4.7: Software user interface main block diagram

This section presents the C# based user interface, which makes the operators able to communicate with each part of the design, collect the measured distance values from the Lidar, construct the final point cloud model, control the rover by and monitor the video streams of the bird-eye wireless camera during the motion of the rover.

Visual C# Express is an easy-to-use, free and integrated development environment (IDE) designed for beginning developers. Students As discussed earlier, each part of the User interface software is created in different stages of the process. Finally, all the parts merged to form a unique package. The following section is going to

introduce each sequence of design in detail according to their order. Figure 4.7 shows the whole schematic of C# based user interface by block diagram.

4.4.1 Motor Control by Arrow Keys

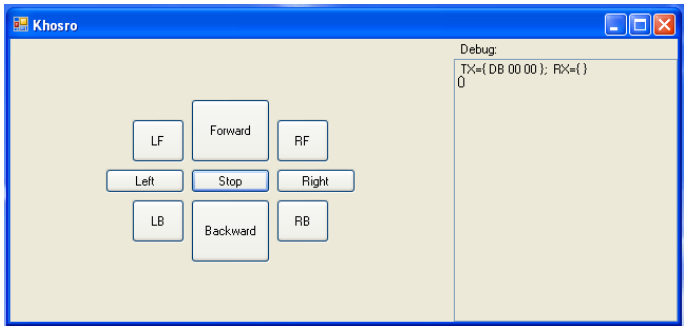


Figure 4.8: Motor controller by arrow key interface

After running the program, it automatically selects the Com Port1 of the PC to transmit the driving commands through the Asynchronous serial (RS232) port to the motor controller device. Figure 4.8 shows the basic motor controller interface design by C#. This form is split in to two parts: Debug part on the right hand side and control buttons part on the right hand side. The control part has nine buttons, which are shown in this figure. The user is able to drive the mobile robot by selecting the proper buttons in the control section. By pressing each of the buttons, the corresponding command related to the speed and direction of each motor are sent to the TReX Dual-Motor Controller through the Asynchronous serial (RS232) port. Serial ports can be connected directly to the motor controller's RS232 port by using a wire or by using the Handy-Port HPS–120 through the wireless channel.

The stop, forward and backward buttons send corresponding commands to each motor by defining the same speed and direction for both motors. Forward and backward buttons set the motors speed to a maximum value and the stop button sets

the speed of wheels to the minimum. For rotating the system body around its centre without any deviation to the sides, the right and left buttons are available. By selecting each of them, the opposite side wheels have to move forward by a maximum speed while the other should stop without any rotation.

For instance, if the right button is pressed, the left motor produces maximum power in the forward direction and the control system applies the maximum voltage to both input poles of the other motor. It prevents a rotation that causes unwanted steering force from other motors during the process. Figure 4.9.a shows the top view of mobile body when it rotates around the centre by pressing the Left button.

LF, RF, LB and RF are the control buttons to let the mobile body to turn to the left and right in two directions; backward and forward. The function of these buttons is the same as that of the left and right buttons but instead both wheels move at different speed. For instance by pressing LF button the power toward forward direction is applied to both motors, but in this case, the right motor rotates faster than the left one hence the whole mobile body slowly turns to the left. Figure 4.9.b shows the top view of the mobile body when it turns toward left by pressing the LF button.

The right hand side part is named as the Debug window. This part is designed for checking the command, which are sent to the motor controller through the RS232 connection and has been used to detect bugs during the test process. In the final version of the user interface, this part has been removed. As it appears in figure [40], the debug part displays the corresponding command for the stop button (DB 00 00,

DB= both motors move forward and each follows two digits that represent the speed of motor one and two respectively).

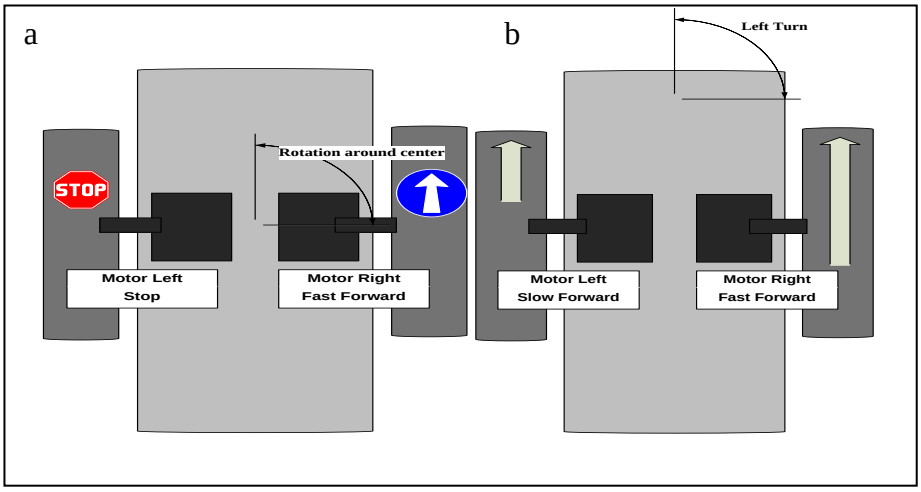


Figure 4.9: (a) Top view of mobile body when it rotates around the centre, (b) top view of the mobile body when it turns toward left

4.4.2 Video and Arrow Keys Controllers

Figure 4.10: Video and arrokey user interface

For developing the design and increasing the ability of user to remotely control the robot, it seems useful to use the video SRV-1 Blackfin Camera`s WiFi 802.11b/g radio module during the navigation process. Therefore, a new partition is attached to the main body of the user interface. The renewed design which has shown by figure4.10 has two parts; the left part acts like the previous case for controlling the motors' speed and direction. The newly attached part at the right side of the screen

monitors the camera picture on the screen. This part has two buttons, namely: the connect button and the disconnect button. The connect button allows the user to connect to the SRV-1 Blackfin Camera through the wireless system. For making this connection, there is no need to attach an external wireless system to the PC. SRV-1 This ability therefore makes the control relatively easier. Before pressing the connect button, it is necessary to connect the PC to the SRV-1 Blackfin Camera. After running the system there is no more setting needed and the software can stream the camera result properly. The disconnect button is used to close the screen window and disconnect the PC from the SRV-1.

4.4.3 Joystick

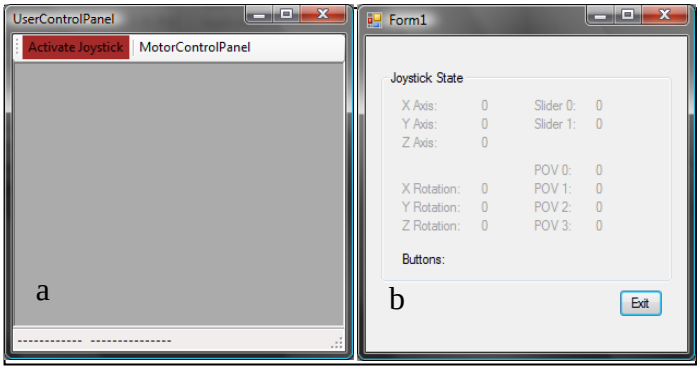


Figure 4.11: User controller interface includes Joystick and arrow keys

The motor control user interface design operates through specified control buttons: left, right, forward, backward, LF, LB, RF and RB. Each control key has its own pre-defined speed and direction. During the testing and debugging procedure, some drawbacks of this design and disabilities of the user to steer and drive the mobile chase may be noticed. For example, the operator cannot change the speed of the rover in each direction. Therefore, with little changes in the architecture and by attaching the Joystick to the system, the problem is solved in the most practical way as possible. Figure 4.11 shows the new chase control user interface design.

The new user interface window is split in to two main pages; the Active Joystick and the Motor Control Panel. By selecting the Motor control panel in the same window as in figure4.10, the panel will be opened and if the operator selects the Active Joystick user, he will have access to the Motor control window by using the Joystick. In this window,which shown by Figure4.11.b, the user can steer the rover in all directions by moving the joystick. The operator can therefore change the speed of the motors by shifting the Joystick back and forth. The Joystick is connected to the PC via the USB port.

The new voltage and current monitoring modules have also been added to the new design. The bottom part of this window has a dialog box, which makes the user able to find out the value of voltage and currents, which is used by each motor during the process. For extracting the value of the voltage and current at every 500msec, the C# program sends the corresponding command request to the motor controller board (TReX Dual-Motor controller) through the serial port and displays the returned values at the screen.

Chapter 5

SURFACE RECONSTRUCTION

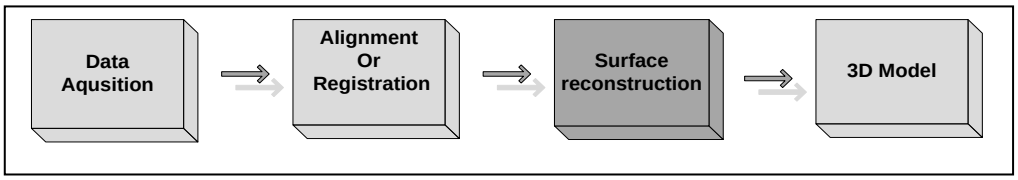


Figure 5.1: Surface reconstruction step in 3D modelling pipeline

Surface reconstruction mainly refers to the method, by which, seamless and smooth surfaces can be obtained from the point cloud models, by using interpolation and approximation. Within these methods, regularized implicit surface reconstruction method based on radial basis functions is more powerful than the other methods in surface reconstruction. Generally, this method reconstructs a unique smooth isotropic (radial) surface function by minimizing an energy function to achieve a desired surface function (regularization) [39]. This method can approximate rather than interpolate surface points and able to implicitly handle complex shapes of arbitrary topologies. This approximation passes a surface over the points and increases the ability of the method to reconstruct a smooth surface. Smoothness parameter is defined as global or local by the user within the procedure. However, interpolation means passing the surface exactly through the surface points. This approach constructs a smooth model in one-step and force the implicit function to be locally similar everywhere. However, this method is not successful in non-planar regions of the surface [13]. For instance, in the neighbourhood around a point on an edge, the surface is smooth along the edge but not across it. Besides, the global smoothness

parameter, used in this method, is set manually by the user according to the human perception.

As a case study, this research has searched for a solution to the problem by extracting the distinct features including edges, corners and planar regions of the point cloud model and locally reconstructs them by using implicit surface reconstruction based on isotropic basis functions. This enforces the surface to be sharper across the edges than along them. The orientation of the isotropy is determined by categorizing points as being rooted in a planar region on an edge or at a corner by applying the novel method on sharp features detection by [46]. Also, the proposed method replaces the manual smoothness parameter setting process by an automated process. This is determined by locally and adoptively varying the smoothness parameter according to the attribute of the surface at planar, edges and acute angel regions.

Our proposed method utilizes the implicit surface using isotropic (radial) basis functions, to reconstruct a small sample of a point cloud model. Moreover by considering the limitation of the previous surface reconstruction methods to represent the sharp features, the novel feature extraction method proposed by [46] has been implemented and its results are presented.

5.1 Definition of Radial Basis Functions and Surface Reconstruction Modelling

Concept of radial basis function is rooted from applications of neural networks. Generally, multi layer perceptions are well known for their ability to solve stochastic estimation approaches or pattern classification. However, we can look at them

completely from different perspectives as surface fitting consideration or interpolation problem. For instance, consider two groups of points, which are scattered in a 3-dimentional space. If we can pass the hyper-plain in which we can easily separate them to two classes, we consider them as linearly separable. In most cases however, it is impossible to pass a hyper-plain in the space of inputs data and we know this problem to be a non- linearly separable. Hence, instead of passing the hyper-plain, we have to pass the hyper-surface in the input space to separate completely the two classes of data. In 2-D input domain, instead of passing a line we have to pass a curve. According to the positions of the scattered points in the space, the type of curve or surface, which passes for classification of the patterns, is different. They can be linear, spherical or quadratic. Figure 5.2 shows examples of the separability problem in two-dimensional space. however Cover`s theorem proposed a solution for this problem.

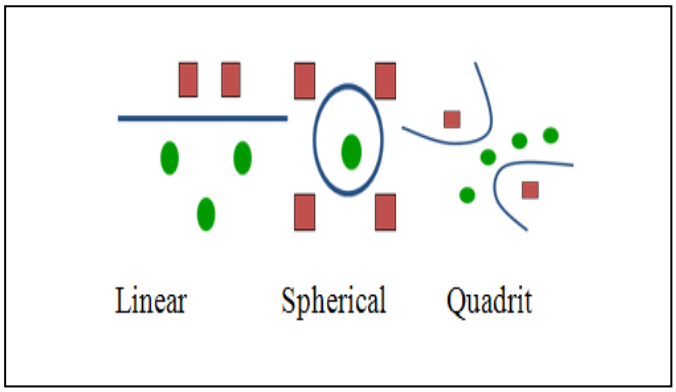


Figure 5.2: 2D examples of Seprabilty

Cover's theorem: "a complex pattern-classification problem cast in a high-dimensional space non-linearly is more likely to be linearly separable than in a low-dimensional space. Cover's theorem says if we were able to map the inputs pattern to the higher dimensional space, our chance to find the hyper plane in the mapping space would be increased [20]. This consept has shown by figure 5.3.

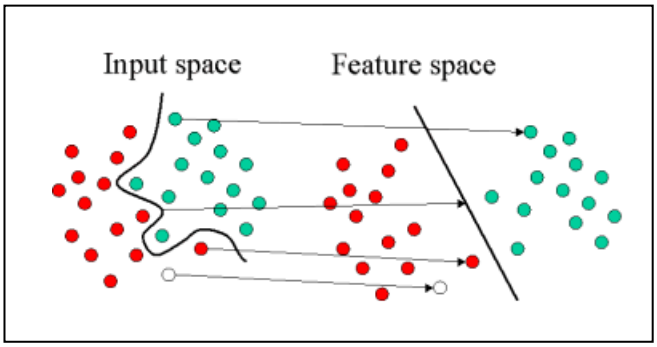


Figure 5.3: Mapping From input space to Hidden laye space

To solve this problem the hidden layers in hands can be looked at from another perspective. Hidden units in hidden layers provide the set of functions, which forms the basis to the hidden layer space. Basically, we are going to non-linearly map with the inputs data from the input space to the hidden layer's space. To do this mapping, we need some basis function as such the hidden units provide these basis functions.

$$\varphi(x) = < \varphi_1(x), \ldots, \varphi_{m_1}(x) >$$

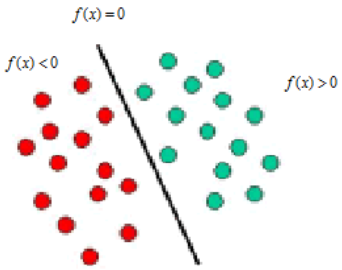
φ_i Hidden Functions (5.1)

$\{\varphi_i(x)\}_{i=1}^{m_1}$ Hidden space

On the other hand, the output of the output layers unit is produced by a linear combination of the mapped points in the hidden layer. The output function gives good views about the classification state of each mapped point.

$f(x) = w^T \varphi(x)$ Output function (5.2)

A (binary) partition of the sampel points is called dichotomy, (C_1,C_2) of the training set C is φ-separable if there is a vector W of dimension m_1 such that:



$$w^T \varphi(x) > 0 \qquad x \in C_1$$
$$w^T \varphi(x) < 0 \qquad x \in C_2$$

The combination of mapping functions (basis) and linear combination process is known as Radial Basis Functions (RBF). Figure 5.4 shows the basic RBF network structure.

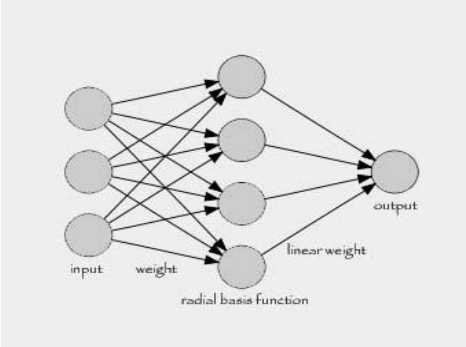


Figure 5.4: Radial Basis Function network main diagram, [14].

A basic Radial basis network structure is shown in this figure has three layers.

- Input layer: source of nodes that connect the NN to an environment.
- Hidden layer: performs a non-linear mapping from the input space to the hidden space by φ_i functions.
- Output layer: performs a linear mapping from the hidden space to the output space by W_i coefficients.

In order to model a impilicit surface reconstroction with radial basis function networks, we can consider an impilicit surface reconstruction method as a iso-surface of the function $\phi : R^3 \rightarrow R$. By another definition, it is the set of points that

satisfy the equation $\phi(i) = f_i$, where f_i can be define as the density scalar filed at point (x, y, z). This concept has shown by Figure 5.5.

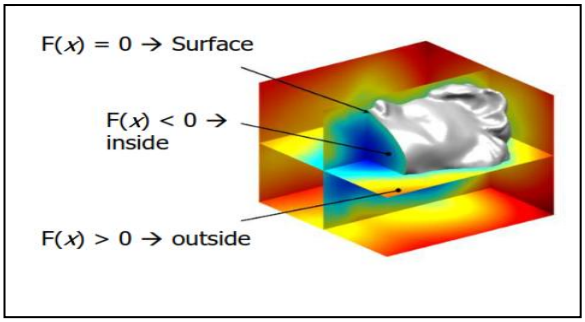


Figure 5.5: Implicit model of a surface using iso-surface presentation, [10].

Tipically the equation is of the form $f(x, y, z) = 0$

Where the following properties are defined for outside and inside detection:

- $f(x) = 0$ for points on the surface
- $f(x) < 0$ for points inside the surface
- $f(x) > 0$ for points inside the surface

The major benefit of implicit surfaces over other ones is their ability to classification the surface points from off-surface points.This technique aims not only to find the calasses but also attempts to find the seperator function or iso-surface funcnction , which separates the training patterens to two classes.This algorithm allows the system model the surface reconstruction problem in a proper manner.

The goal is to find a function *f*, which implicitly defines a surface and satisfies the equation:

$$f(x_i, y_i, z_i) = 0, \quad i = 1,.., n \tag{5.3}$$

Where $\{(x_i, y_i, z_i)\}_{i=1}^{n}$ are points lying on the surface.

In order to avoid the trivial solution, *f* set to zero everywhere. In the other hand, off-surface points are appended to the input data and are given non-zero values. This classification tends a useful interpolation problem [10]. Find *f* such that :

$$f(x_i, y_i, z_i) = 0 \qquad i = 1,2,...,n \qquad \text{(On-surface points, class one= C1)}$$
$$f(x_i, y_i, z_i) = d_i \neq 0 \quad i = n+1,...,N \qquad \text{(Off -surface points, class two=C2)} \tag{5.4}$$

The above equation still has the problem with generating the corresponding off-surface point's value "d_i". An obvious choice fo" *f*" is a signed-distance function, where d_i are chosen to be the distance to the closest on-surface point. Points inside the object are assigned negative values, while those outside are assigned positive distance values. These off-surface points are generated by projecting along surface normal. Off-surface points may be assigned to either side of the surface as illustrated in Figure 5.6

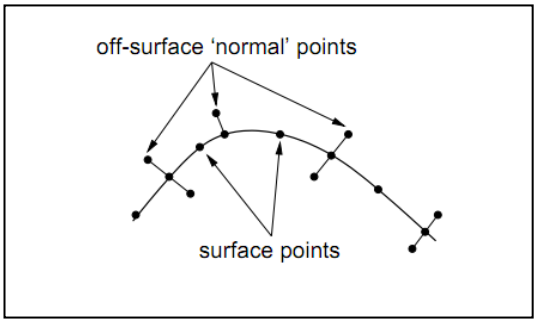


Figure 5.6: Asigned –distance function is form the surface data by speciying off-surface point along surface normal, [10].

Experience has shown that it is better to augment a data point with two off-surface points with one on either side of the surface. Hence, there are two problems to solve; estimating surface normal and determining the appropriate projection distance.

In the presence of a partial mesh, it becomes undemanding to define off- surface points, since normal are implied by the mesh connectivity at each vertex. In the case of un-organized point-cloud data, normal may be approximated from a local

neighbourhood of points. This requires estimating both the normal direction and determining the length.

5.1.1 General RBF Function

In general, an RBF is a function of the form

$$f(x) = p(x) + \sum_{i=1}^{N} w_i \phi(|x - x_i|) \qquad (5.5)$$

Where ρ is a polynomial of low degree and the basic function φ is a real valued function on [0,∞), typically unbounded and of non-compact support. We call the w_i `s the weights and the x_i 's canters of the RBF.

Solving the problem for the weights refers as fitting the RBF, and computing *f*(x) refers as evaluating the RBF. Popular choices for φ are Gaussian, Cubic and linear radial basis functions

Gaussian $$\varphi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \quad \sigma > 0 \qquad (5.6)$$

Linear $$\varphi(r) = -r \qquad (5.7)$$

Cubic $$\varphi(r) = r^3 \qquad 5.8)$$

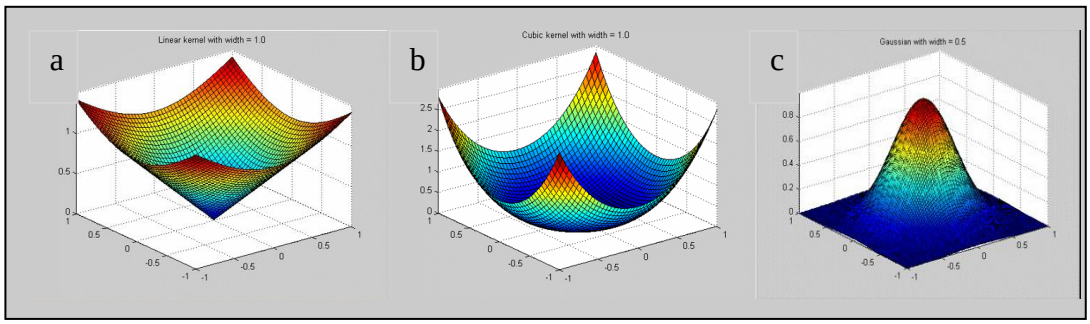


Figure 5.7: Example of radial basis function graph, (a) Linear, (b) Cubic, (c) Gaussian, [30].

5.1.2 RBF Interpolation

In many practical problems, while sample points are noisy, exact interpolation is neither required nor desired. One way to grant this is to solve the interpolation system only to certain accuracy. Thus, one approach corresponds to the Equation (5.5) could be approximate with low energy such that:

$$E_s(f) = \frac{1}{N}\sum_{i=1}^{N}(d(x_i) - f_i)^2 \tag{5.9}$$

Where $d(x_i)$ is a desire surface and f_i is an actual surface function. Equation 5.9 is known as standard error term.

Such a model can be applied to fitting signed distance data from a laser scanner.

$$| f_i - d(x_i) | \leq \varepsilon, i = 1,..., N \tag{5.10}$$

The fit would then be an estimate of the signed distance to surface and ε could be chosen as an estimate of the scanner error. It would suffice in many situations to assume that each measurement is the combination of a underlying signal and random noise.

5.1.3 RBF Approximation

Smoothness interpolant refers to the senses, which minimizes certain energy functional and interpolate the data. For example, given a set of nodes $X = \{x_i\}_{i=1}^{N} \in R^3$ and a set of function values $\{f_i\}_{i=1}^{N} \in R$, RBF has to satisfy the smoothness conditions by minizing the smoothness term $E_c(x)$.

$$E_c(f) = \| \vec{D}F \|^2 \tag{5.11}$$

$\| \vec{D}F \|^2$ is a measure of the energy in the second derivative of f.

$$\| \vec{D}F \|^2 = \int_{R^3} (\frac{\partial^2 f(x)}{\partial x^2})^2 + (\frac{\partial^2 f(y)}{\partial y^2})^2 + (\frac{\partial^2 f(z)}{\partial z^2})^2 + 2(\frac{\partial^2 f(x)}{\partial x \partial y})^2 + 2(\frac{\partial^2 f(x)}{\partial x \partial z})^2 + 2(\frac{\partial^2 f(x)}{\partial y \partial z})^2 dx \tag{5.12}$$

5.1.4 Generalization

Given the nodes $\{x_i\}_{i=1}^{N} \in R^d$

$$E(f) = E_s(f) + \rho E_c(f) \tag{5.13}$$

Minimize $\rho \| \vec{D}F \|^2 + \frac{1}{N}\sum_{i=1}^{N}(d(x_i) - f_i)^2$ (5.14)

Where, $\rho \geq 0$ and $\|.\|$ denotes the smoothness penalty defined by Equation (5.13)

This approach is identified as generalization function $E(f)$. The parameter ρ control smoothness against fidelity to the data .The solution to generalization Problem is also an RBF of the form equation 5.5 but it approximates the given values f_i at the nodes x_i .

Results of globally varying smoothness parameter are shown by figure5.9. In this example a smooth surface has been automatically fitted to the LIDAR scans of a fountain in Santa Barbara using an RBF approximation. The statue is approximately 2m×5m and was scanned using a CYRAX2400 scanner.

In the surface-modelling field of science, the aim is mainly to reduce the high frequencies and random variations in the surface data that related to the noise and passing through a smooth surface. The other requirement is to estimate the magnitude of the noise and this is related to the structure of the laser scanner. In this method there is no clear relation exists between the smooth constructed surface by RBF and the ρ. In addition, the deviation from the constraints or nodes is just identified by the product of RBF weights W_i and ρ for solving the generalization equation 5.14.

However, estimating ρ is computationally expensive to iteratively choose ρ since this involves fitting and constructing an RBF many times. As a result, many authors select their parameter based on experience. The following section will introduce a simple method for selecting and identifying the smoothing parameter. This will be done by categorizing points as being entrenched in a planar region, on an edge, or at a corner by performing the newly proposed method on sharp feature detection by [46].

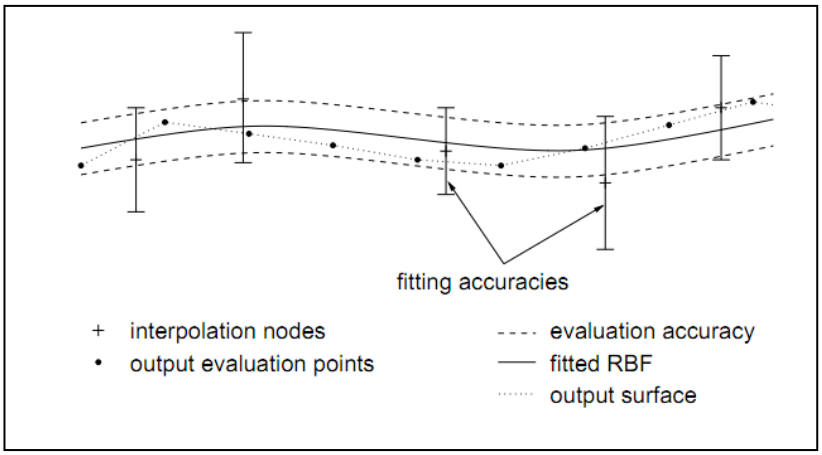


Figure 5.8: Illustration of the fitting and evaluation parameter, [11].

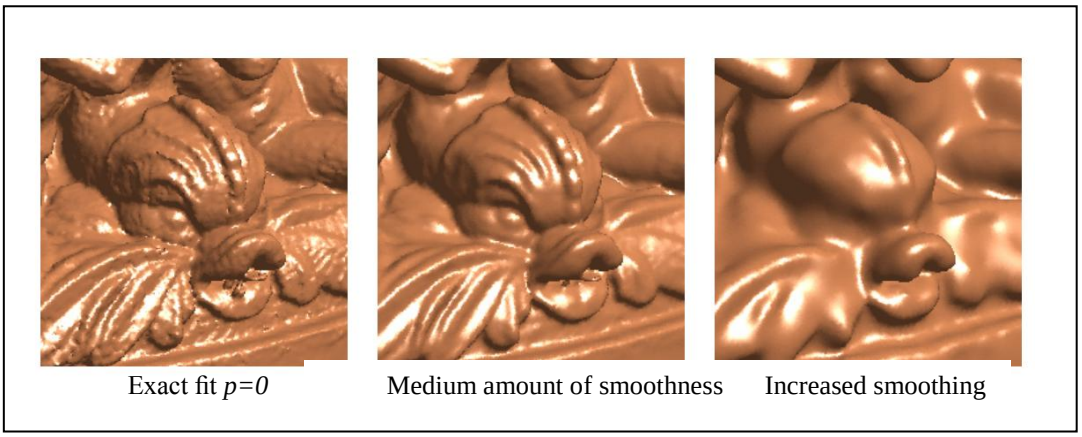


Figure 5.9: Globally varying the smoothness parameter, [10].

5.2 Feature Extraction

5.2.1 Sharp Feature Detection

In this part, a brief explanation on the basic aspects of estimating features and detection method based on the new approach by [46] is given. This approach consists of three main steps: first is to create the data structure from the input point cloud. The second step locally classifies the data by making local neighbourhood clusters. The Third step analyzes the local neighbourhoods in order to extract the sharp features.

5.2.2 Data Structure

In order to detect the sharp features in point cloud model, firstly all points must be analyzed to see whether they are parts of the sharp feature or not. This analysis is made by checking the neighbourhood of each particular point. Neighbourhoods are extracted through the K-nearest space-partitioning algorithm. This method is able to approximate nearest neighbourhood search by using the Kd-tree. This step is known as pre- processing step for the method and is performed once. In addition, the result can be stored in a Riemannian graph. In this graph, the minimum spanning tree connecting the point is expanded to connect each point with its k-nearest neighbours.

5.2.3 Neighbourhood Analysis

Through the Kd-tree space-partitioning algorithm, the “n” neighbourhood of the K-nearest points $p \in P$ are constructed. The next step is to perform Gauss map clustering algorithm in order to check if the sample points belongs to a sharp feature or not.

5.2.4 Discrete Gauss Map

Let n_p be a normal vector of point p corresponding to n nearest neighbourhoods of it. The discrete Gauss map of the neighbourhood of p can be defined as the mapping of

n_p onto the unit sphere cantered at p .Figure 5.10 shows the Gauss map of the normal vectors corresponding to the neighbourhoods of point P.

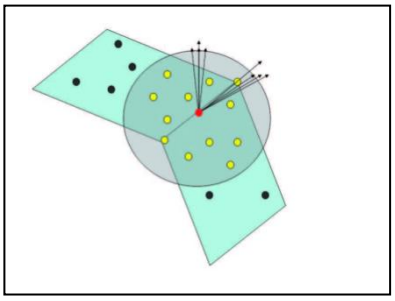


Figure 5.10: Gauss map clustering example

Feature detection is now performed, through an accurate selection process, known Gauss map clustering.

5.2.5 Gauss Map Clustering

Gauss map clustering is motivated by the fact that in the case of a smooth piecewise C^0 surfaces, the Gauss map of the neighbourhood of a surface point is different whether the point is lied on flat or discontinuous tangent plane.

If the points lied on nearly flat surface, the Gauss map of neighbour points will show one cluster of points on the sphere.

If the points lied on curved surface, the points on the sphere will be spread over a larger region.

In the case of a tangent plane discontinuity, the points of the sphere will represent two separate clusters; see Figure 5.11 for some 2-D examples.

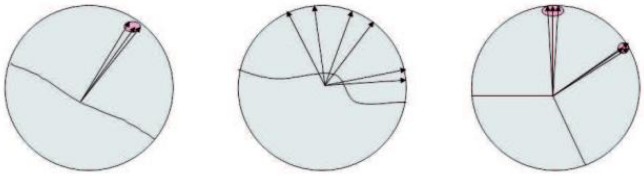


Figure 5.11: 2D example of possible sharp feature detection cases, [10].

5.2.6 Distance Measure:

Now an important step toward the clustering algorithm is properly choosing a distance measure or D_{ij}, which shows how two normal are close to each other. It logically appears that the angel between two normal vectors can be also measured by geodesic distance between two points on the gauss sphere

$$d(x_{ij}, x_{jk}) = \min\{d_g(x_{ij}, x_{kl}), d_g(x_{ij}, x_{lk})\}$$

Where $I, j, k, l \in I_p$ and

$$d(x_{ij}, x_{jk}) = \arccos(< n_{ij}, n_{kl} >) \qquad (5.15)$$

5.2.7 Clustering

Many clustering approaches are work based on a priori specified number of clusters. Since the goal of this project is just to distinguish some clusters from the other which are sparsely distributed in Gauss map sphere, there is no need to specify the value for number of clusters. Hence, this project uses the hierarchical bottom-up clustering method purposed by [19]. This method initially specifies a cluster number to each point. It subsequently merges them so as to create the larger clusters. On the other hand the mean distance can be a good approximation condition to stop this sequence. Therefore, D_c is defined as the mean distance between elements of each cluster. Figure 5.12 shows the results of clustering approach by varying the clustering parameter.

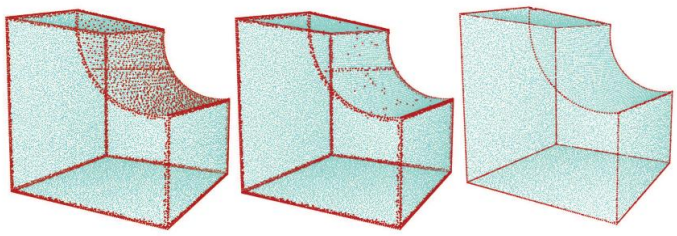


Figure 5.12: Result of clustering approach by varying neighbourhood size and clustering distance

5.3 Implementation and Results

5.3.1 Analysis

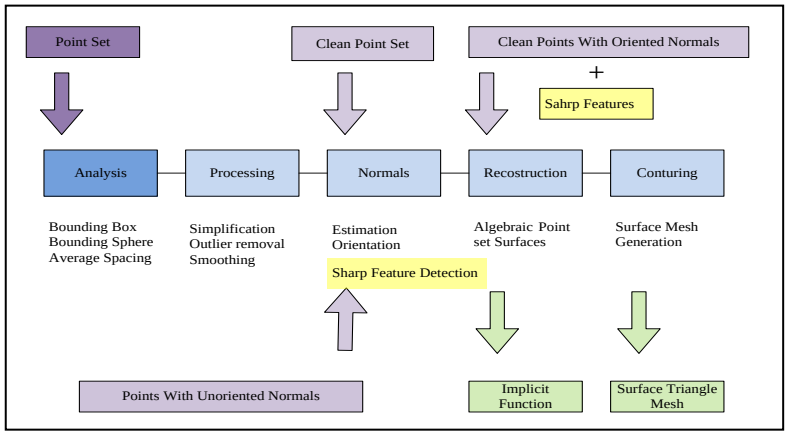


Figure 5.13: Analysis sequence through the surface reconstruction pipeline

As described in section 5.1 this method is focused on adaptive implicit surface reconstruction based on radial basis functions. This is adaptively performed on each part of the point cloud model. Hence as a first step, it is essential to partition the whole point cloud model to smaller sub- groups with fewer points. For this reason, this step has used one of the most well known space-partitioning algorithms “Octree”. This algorithm is not only famous for its ability to partition the point cloud model, but also for its capability of estimating the connectivity, bounding box, bounding sphere, average spacing and cost reduction in term of finding the k-nearest neighbourhoods in the next steps . An Octree is a tree data structure in which each internal node has exactly eight children. Octrees are most often used to partition a 3D space by recursively subdividing it into eight octants. Octrees are the three-dimensional analogue of Quadtrees. The name is formed from Oct + tree, and normally written "octree".

Each node in an octree subdivides the space it represents into eight octants. In a point region (PR) Octree, the node stores an explicit 3-D point, which is the "center" of the subdivision for that node; the point defines one of the corners for each of the eight children. The root node of a PR Octree can represent an infinite space. The Octree prevents the subdividing process when it reaches the specific pre-defined value. This value is mostly known as the depth of Octree. This method of subdividing the point's clouds is based on children's volume box. When the length of the box covering the nodes points exceeds the predefined value, the Octree stops subdividing that root. This value is selected based on the density of the point's samples. In the next step, the points, which are lying inside of each child's, will be extracted and will be passed to the next sequence of the 3-D modelling pipeline to perform individually.

Figure 5.14 shows the structure of the Octree algorithm. Figure5.15 shows the top and front view of performed Octree algorithm with different box length value. As it appears in this figure by decreasing the box-length value, the numbers of points, which lie in each box, are decreased.

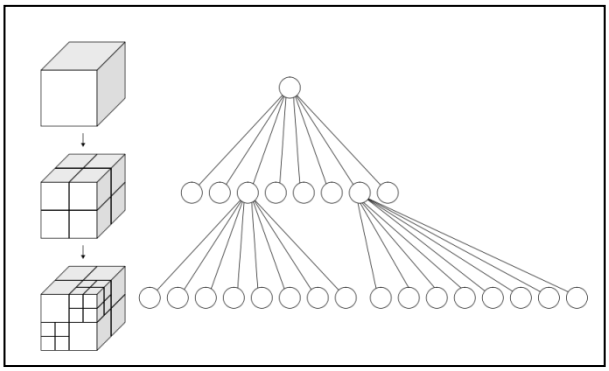


Figure 5.14: Octree structure

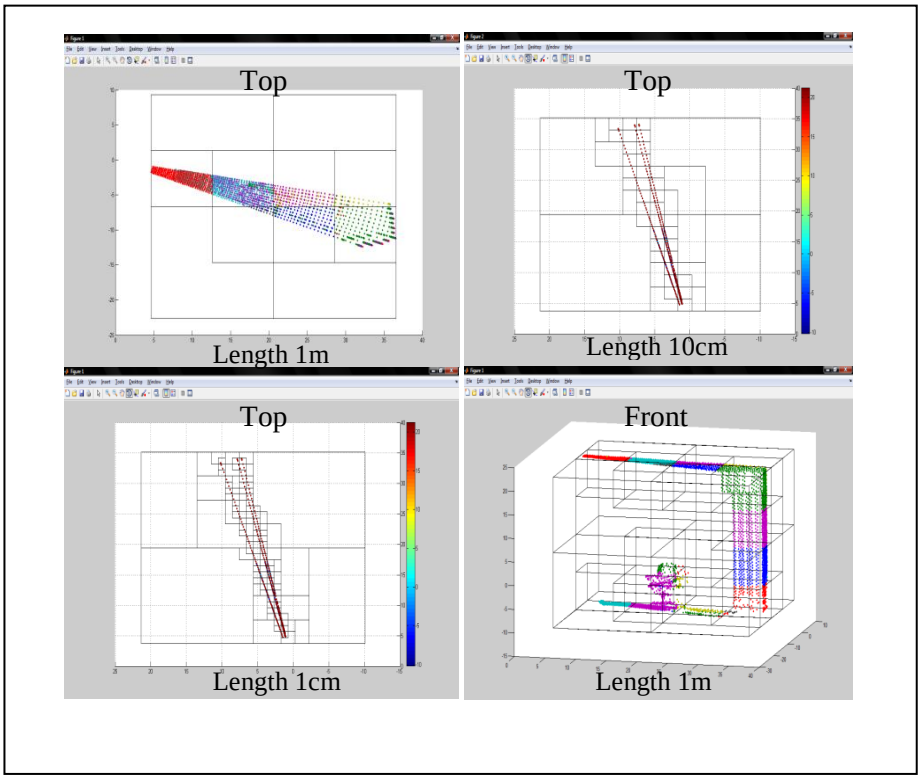


Figure 5.15: Applied Octree partitioning method on scanned room A12, Techno Park, EMU by "CAD eye" 3D laser scanner

5.3.2 Processing (Simplification and Outlier removal)

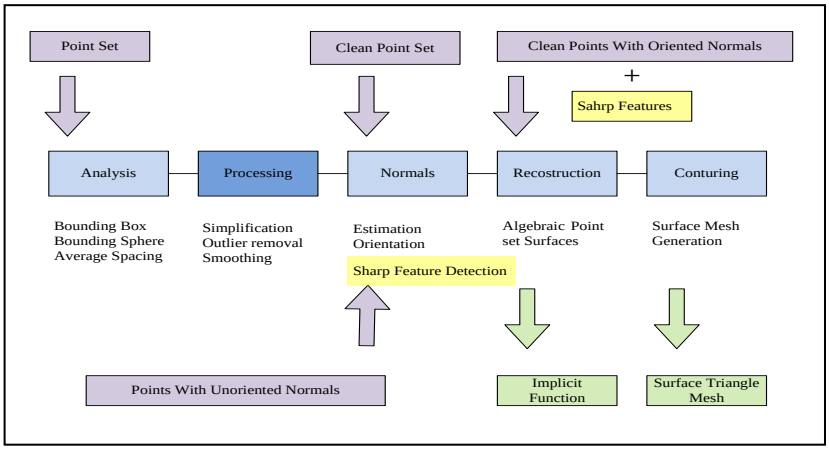


Figure 5.16: Processing sequence through the surface reconstruction pipeline

The main aim of the processing sequence is simplifying, removing and reducing the noisy sample points. Therefore, this project has used a very fast and useful method.

At first, by applying the K-nearest neighbourhood algorithm to detect the n-nearest neighbourhood of each sample point, and by merging the “n” closes points, it then replaces them with the new sample points corresponding to their average coordinate or centre. Where the “n” is related to the number of nearest points to each sample points and can be vary from one to the maximum number of points in each analysis sequence output. Figure 5.17 shows the tree type of nearest search algorithms. The figure to the utmost right is an example of finding the nearest points to the centre of a rectangle with specific length. The one at the middle is a 2-D example of finding the nearest points around a centre of a circle with specific radius, and the one at the utmost left is an example of finding the K nearest to the specific point.

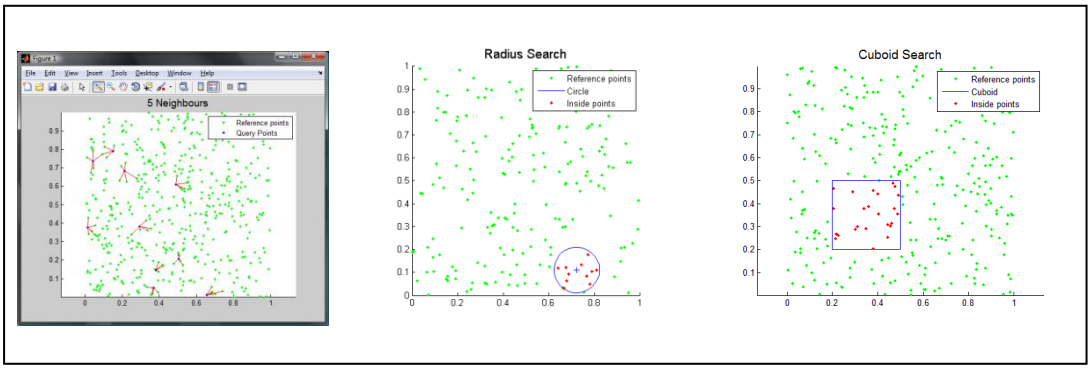


Figure 5.17: Methods of finding nearest neighbourhoods

The first step helps to reduce the number of unwanted sample points, which may lie on the surface through the alignment sequence. This provides a better point cloud model in terms of samplings uniformity. On the other hand, this method decreases the amount of noise, which may also lie on the surface points due to the scanner error term. Figure 5.18 shows the path of sample points after performing the processing sequence by different number of “n”. The black points on each figure correspond to the centre of the K-nearest search algorithm around colorized set of points, which represents different neighbourhood sizes.

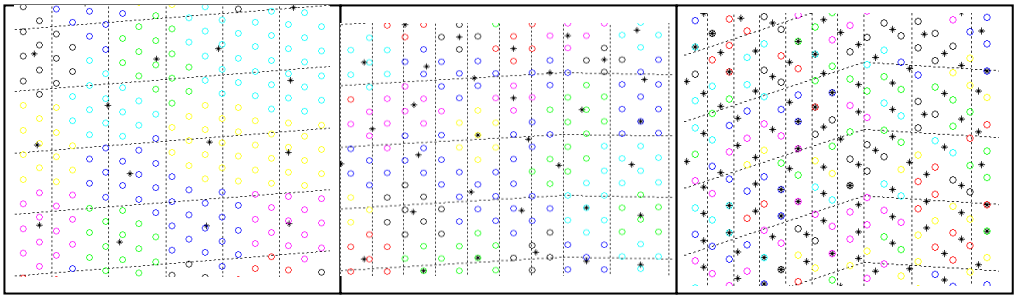


Figure 5.18: Example of finding k-nearest neighbourhood by different neighbourhood size

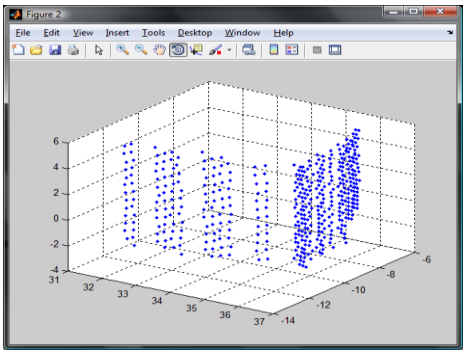


Figure 5.19: Noisy and real 3D sample point cloud

5.3.3 Normal Vector Extraction

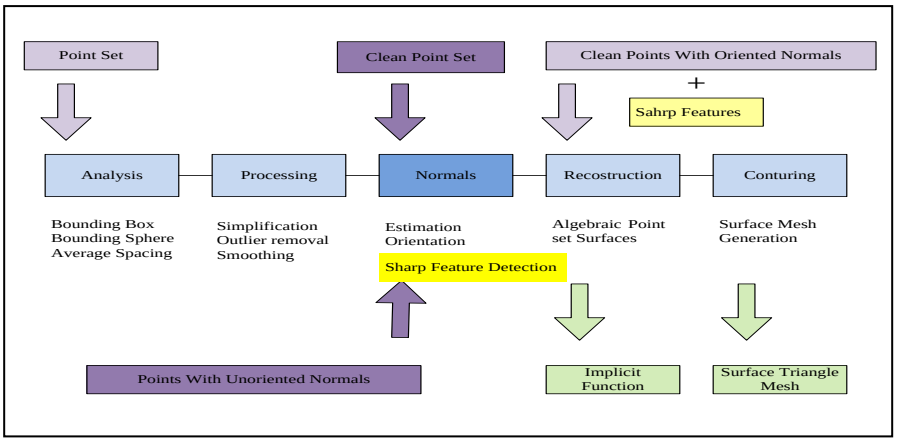


Figure 5.20: Normal vectors extraction sequence through the surface reconstruction pipeline

This part mainly estimates the direction of the global or local surface as a seed for the surface reconstruction method. Orientation of the surface typically is extracted according to the direction of the normal vectors of each particular sample points in the point cloud model.

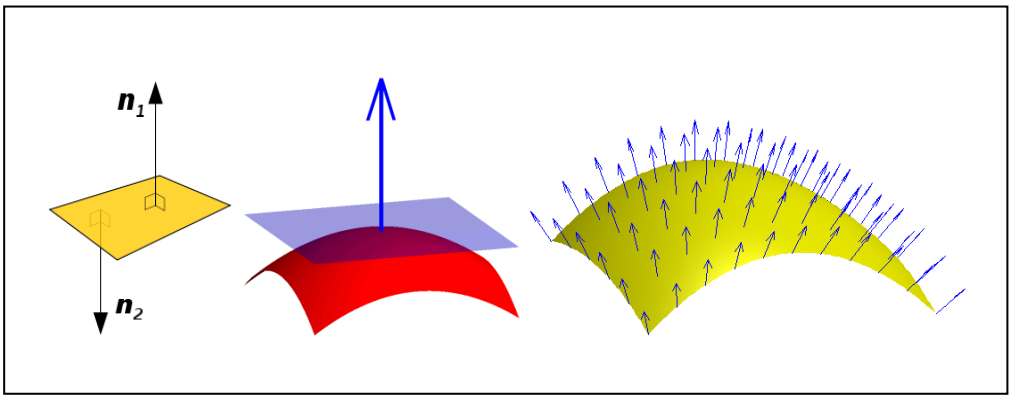


Figure 5.21: Normal vector principle

A normal surface or simply normal to a flat surface is a vector that is perpendicular to that surface. A normal to non-flat surface at a point 'P" on the surface, is a vector that is perpendicular to the tangent plan to that surface at "P". Therefore, extracting the normal surface plays a crucial role for estimating the direction of the surface. This project extracts the normal from the un-organized points and uses the method proposed by [49]. As described in this section, this method starts by finding the K-nearest points to each point. Where p is the point at which the normal will be found and $\{n_1,n_2,..,n_j,..,n_K\}$ are the k-nearest neighbourhoods points .(the number of neighbourhoods' points are change according to the described method at section 5.3). In the next step the normal vectors of each two points are calculated respect to the point p by taking the outer product of these two vectors :

$$(p - n_{pi}) \times (n_{pj} - p) \qquad (5.16)$$

Then the vectoral sum of these normal is calculated to find the normal vector of the point ‘p”. This part of the project can easily calculate the normal vector with respect to all sample points lying on the surface. Figure5.22 shows the normal vector of an arbitrary point with respect to the different number of nearest points. Figure 5.22.a shows the normal vector of an arbitrary point with respect to the 3 closest sample points and Figures5.22a and b sequentially show normal vectors of the same point “p” with respect to 8 and 32 closest sample points.

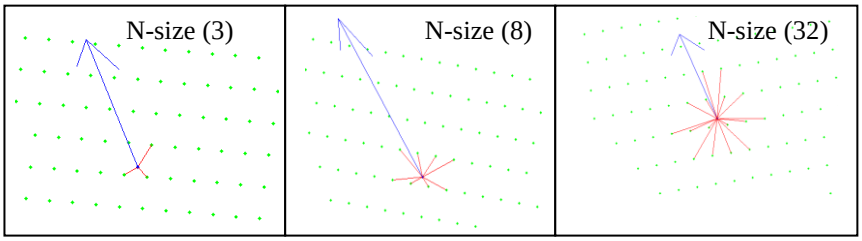


Figure 5.22: Finding normal vectors respect to different neighbourhood size

This method is performed on all sample points of a noisy and a clear sample points shown by Figure 5.23. This is made in order to analyze the effect of neighbourhood size on the direction of surfaces. The noisy sample point is selected randomly out of the Octree partitioning algorithm’s results, and the clear sample points (in sense of noise) are artificially produced as a case study. As explained before at section 5.3, the numbers of nearest points have a direct influence on direction of normal vectors and respectively on realizing the orientation of a surface at that point.

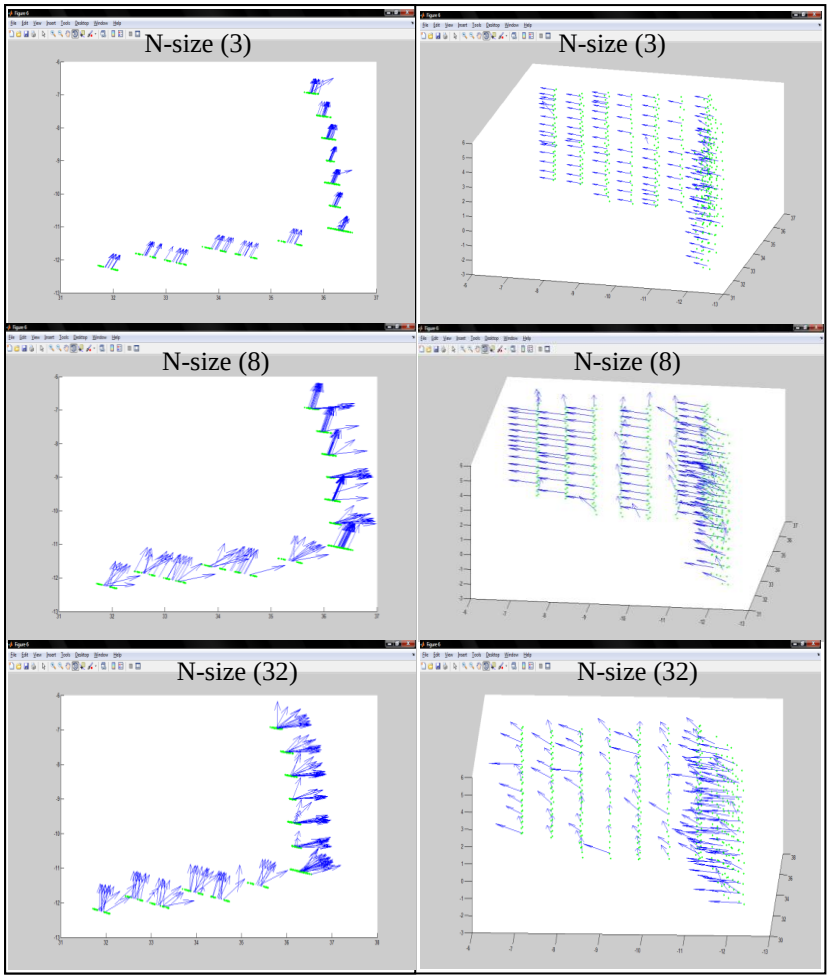


Figure 5.23: Normal vectors orientation analysis based on different neighbourhood size

As it is appear in Figure5.23 by increasing the size of neighbourhoods, the direction of the normal vectors tends more to the real surface orientation. However, after a specific number (32), the boundary point's normal vectors would change their direction under the influence of other surfaces and do not give a good view of the surface junction or edges.

5.3.4 Gauss Map

In this step, based on the explanation in chapter 5, normal vectors are mapped to the Gauss sphere, and the decision-making and clustering processes are done in order to distinguish the flat and sharp features. Figure 5.24 shows the side view of the Gauss map respectively to the different neighbourhood size of noisy sample point. The

expectation of this step for normal's direction and clustering approach are shown in Figure 5.11. As appear in Figure 5.24, although the sample points are noisy in this case, by increasing the neighbourhood size, the Gauss map results are more likely converge to the expectation and make two distinguishable clusters. However, by increasing the neighbourhood size too much to say 120 (as in this case), the sample points lying on adjacent surfaces are affected in the direction of the normal vectors and cause an undistinguishable clustering result.

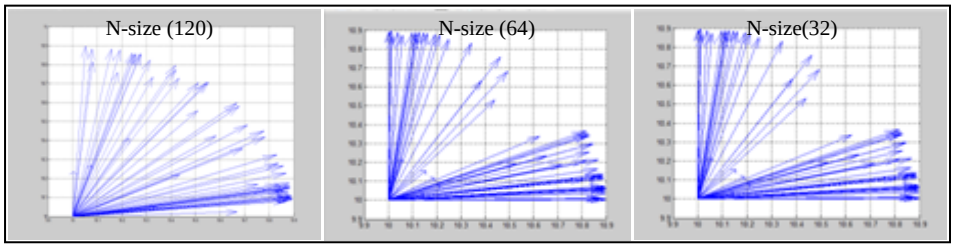


Figure 5.24: Analysis the normal vectors direction on Gauss map according to the neighbourhood size

5.3.5 Clustering

The aim of this step is to cluster the normal vectors. This step sets the neighbourhoods size to a fix value where the normal vectors give approximately good view of the original surface orientation. The distance value parameter has to be varied in order to change the cluster's size. Figure 5.25 shows the Dendrogram related to the distance of the mapped normal vectors on Gauss spheres. While the horizontal vector of the diagram shows the sample point's number in the points cloud, the vertical vector shows the distance value between normal vectors. As earlier described in section 5.2 the distance value plays a vital role in the clustering process. By selecting very small value for this parameter, many clusters may produce. Conversely, selecting very large value for this parameter may cause it to make only one cluster, which may be missing all sharp features that may lie on the

surface. Figure 5.26 shows different value for distance parameter and the direct effect on the partitioned results. This figure shows three different value for clustering and the distance parameters on clear sample points. The figures to the left at each column represent the results of the clustering approach by different parameter value on Gauss map of the normal vectors. The figures on right hand sides of each row respectively show the direct effect of this parameter on clear sample point. It has to be mentioned that points with same direction have been labelled with same colour in right column.

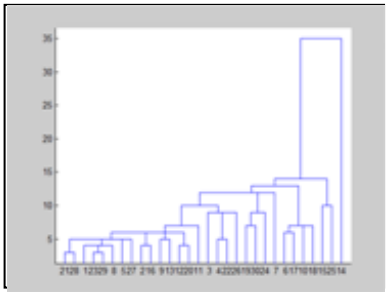


Figure 5.25: Dendrogram for neighbourhood size 48

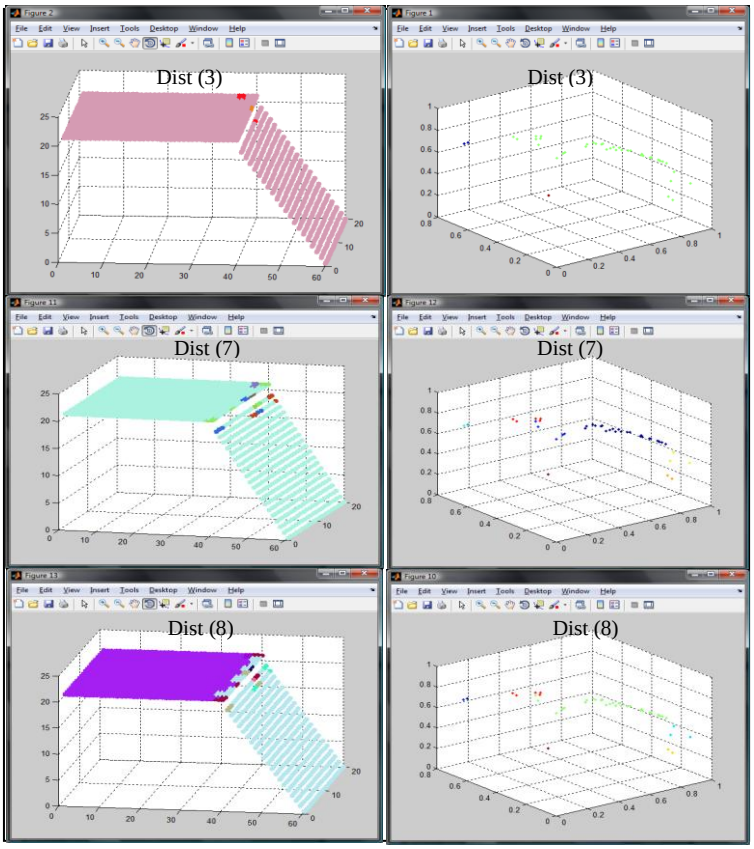


Figure 5.26: Clustering distance parameter analyzing on noise free point cloud

As shown above, by changing the clustering parameter or distance threshold value, this method is able to distinguish the direction of each surface and also the edges points. Since the clustering parametre is less than 8 (in this example), this method is just able to partially ditingush the edge points. It cannot distinguish the difference between surfaces and their direction. Therefore in the first 2rows, most of the points have same color and just the edge points are in diferent colors. By inceasing the clustering parameter to 8 and more, the clustering approach gives a better view of each surface oriantation. It is able to distinguish between surface directions and also between surfaces and edges points by showing them in different colors. In the following Figure5.27 the clustering approch has applied on real noisy sample data.

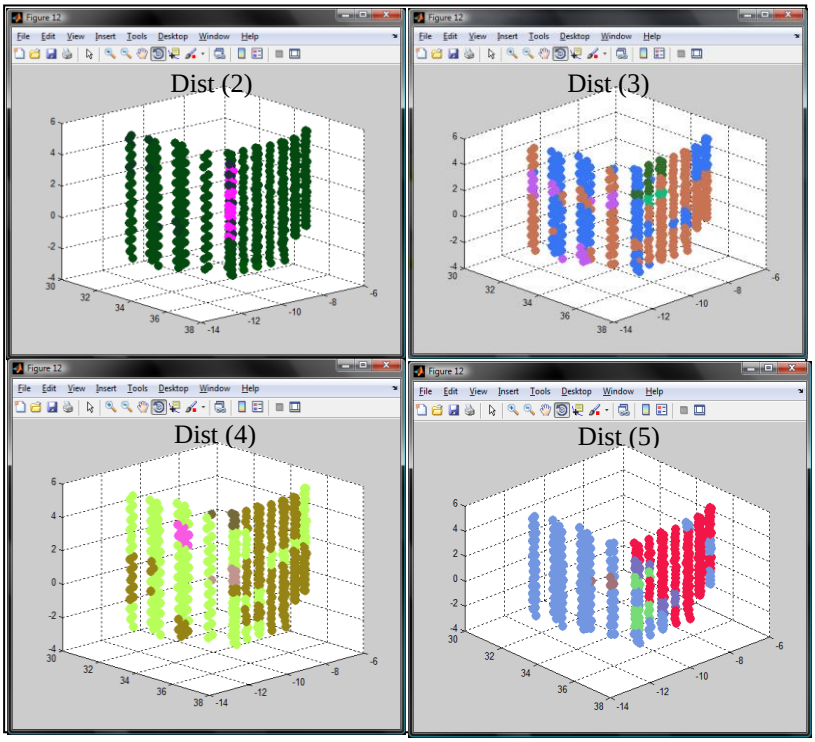


Figure 5.27: Example of the clustering approch on real noisy sample data.

As a result of applying the sharp feature detection method, the direction of each surface and the exact boundary points can be extracted. The figure 5.28 shows the final result of the method on artificial right angel sample points. As it is appear, two

flat surfaces and the boundary line points are determined. Figure 5.28.b shows a strip line related to the boundary points. Figure 5.28.c shows two surfaces with different colours except boundary sample points.

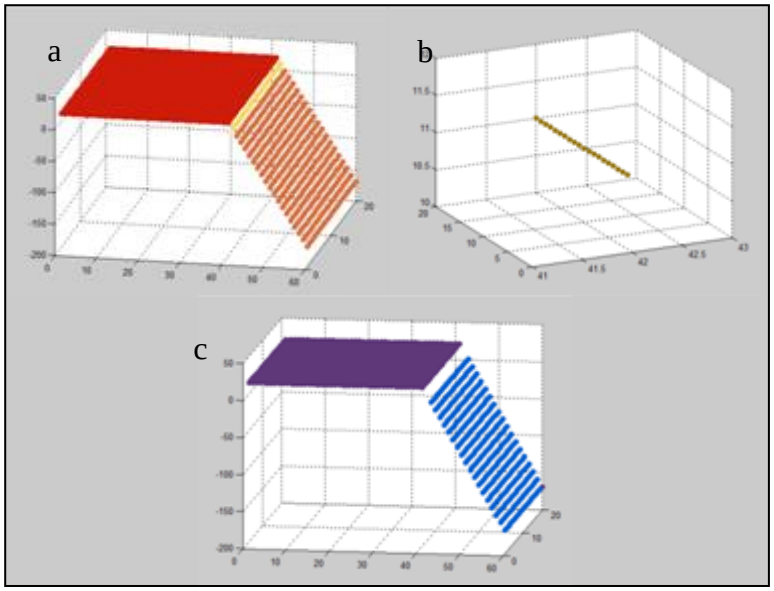


Figure 5.28: Result of applying sharp feature detection method on noise-free point cloud

Figure5.29 shows the final result of the sharp feature detection method on noisy and real sample points. Unfortunatly due to the noise which lies on the sample points, this method is not able to clasify the flat surfaces and edges points properly.

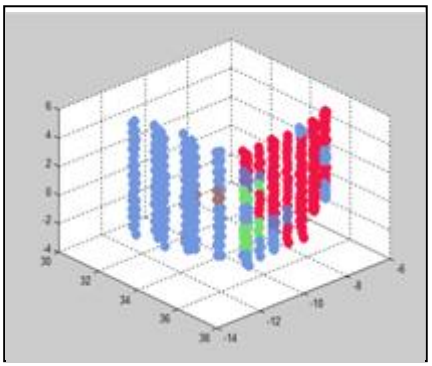


Figure 5.29: Result of applying sharp feature detection method on noise-free point cloud

5.3.6 Surface Reconstruction

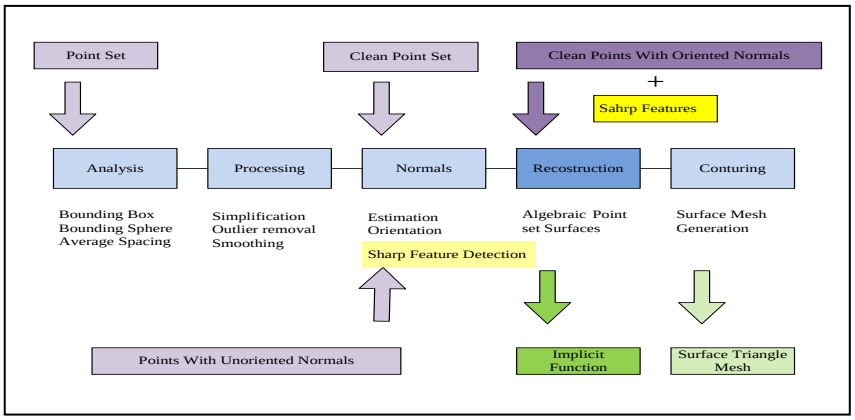


Figure 5.30: Surface reconstruction sequence through the surface reconstruction pipeline

A major point of this project is to use the captured range data and produce the reliable, smooth and seamless 3-D model by applying a proper surface reconstruction method. It has used the ready matlab code of conventional isotropic implicit representation [12]. This can smoothly interpolate the surface where there is little or no data that is compact when compared to discrete volumetric distance functions. It also can either approximate or interpolate the data. The resulting surfaces are inherently manifold, smooth, and seamless [23].

This approach works based on global isotropic implicit surface reconstructions methods. The chosen method is able to use different type of radial functions for reconstructing surfaces. Such a function i.e. Gaussian, cubic and linear functions are applicable by this ready method. This method allows the user to choose a function and the global smoothness parameter.

Using radially symmetric basis function inherently assumes however, that the surface to be reconstructed is everywhere locally symmetric. Such an assumption is true only at planar regions. Hence, surface reconstructing based on using isotropic basis is insufficient to recover objects that exhibit sharp features. In addition, the amount of smoothness may be different in each part of a surface. As mentioned before, the selected noisy sample point as a case study in this chapter does not have a same smoothness in each part. Therefore selecting a global smoothness parameter for reconstructing the surface does not sufficiently preserve the form and shape of the surface.

The following graph shows the outcome of applying the isotropic implicit surface reconstruction method, by choosing a cubic function and global smoothness parameter of ($\rho = 0.2$) on the noisy data set. It appears this method could not preserve the sharp corner. Figure 5.31 shows the result of applying the method mentioned with same global smoothness parameter on noise free artificial sample point. Wrinkles appearing on the surfaces are caused by non-uniform distribution of the points on the point cloud model. In addition, this method is not able to preserve the sharp corner even in noise- free sample point.

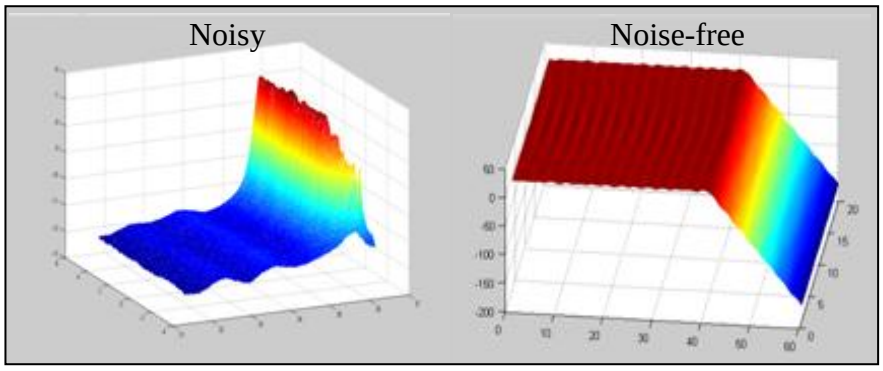


Figure 5.31: Result of applying the global implicit surface reconstruction method on noise free and noisy point cloud artificial sample point

5.3.7 Linking surface reconstruction and sharp feature detection

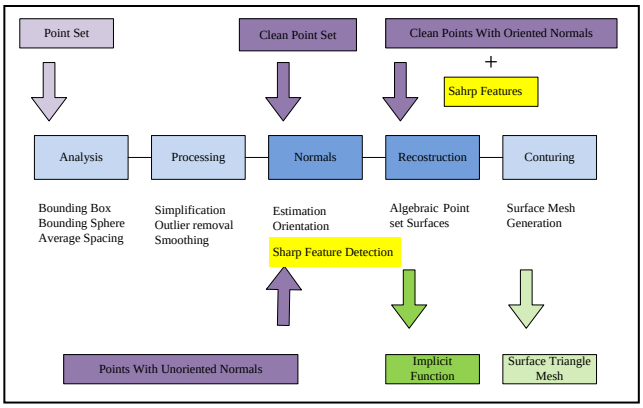


Figure 5.32: Adding sharp feature detection method on the surface reconstruction pipeline

As earlier explained, the conventional isotropic RBF implicit surface reconstruction method is unable to preserve the form or sharp features and it directly depends on the global smoothness parametre rho (ρ) . This method works based on the normal vectors direction alone. Hence this project has extracted additional information about the direction of surfaces and sharp feature just based on normal vectors information and as a seed and applied it to the reconstruction method. This additional information leads to the clustering of sample points to separate classes based on their smoothness attributes. It also enables the extraction of the sharp features which occurs on edges and surface boundries. Hence, this method is able to reconstruct each class of the sample points seprately based on their atributes with different smoothness parameter. As such the sharp features get a better result as compared to the conventional methods.

Figure 5.33 shows the result of locally adaptive surface reconstruction method on noise free sample points. As it is appears, each surface is reconstructed seperately by using different smoothness parameters.

Figure 5.33.a shows the reconstructed top surface of the sample points seperately with smoothness parameter ($\rho=0$) . figure5.33.b shows the side view of this surface. The reconstructed surface is exactly passed through all sample points and it gives an excellent view of this surface.

Figure 5.33.c shows the same surface by applying a smoothnesss parametre of ($\rho=0.2$). In case of comparison, the locally reconstucted surface in Figure 5.33.c with a smoothness parameter ($\rho=0.2$), and the globaly smoothness parameter result Figure 5.31.b with same smootheness parameter, it is obvious that the locally adaptive smoothness parameter choice proposed in this project efficently works properly in this case study.

Figure 5.33.e shows both surfaces in the same graph with different "ρ" value of the top surface reconstructed by $\rho=0$ and the second surface reconstructed by applying ($\rho=0.2$).

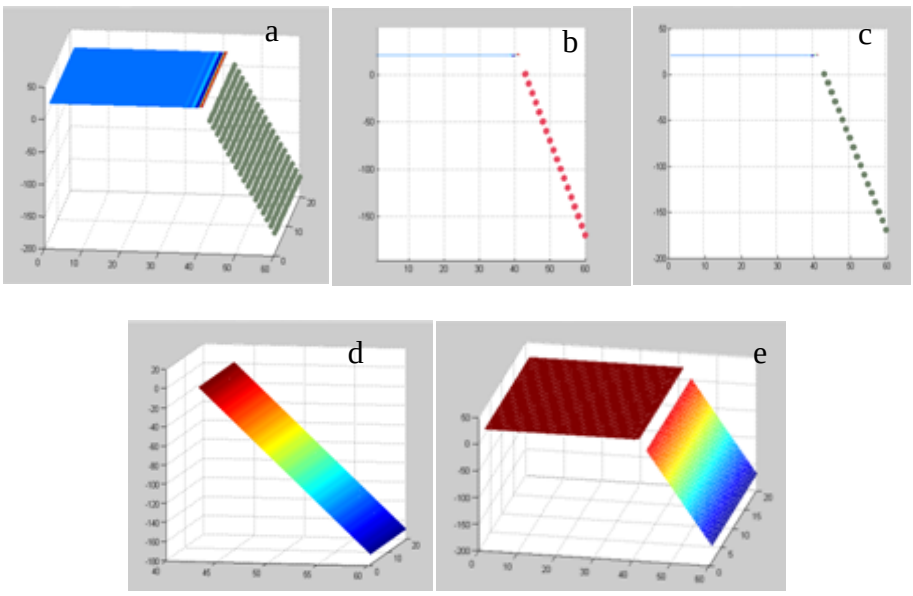


Figure 5.33: Result of locally adaptive surface reconstruction method

For connecting, the two separated reconstructed surfaces at the edge points, preserves the first, second and third order continuity and corner shape. This project has used the proposed method by [3], which connects the adjacent surfaces to each other by extending these surfaces toward the next surface. These methods initially find the boundary points of one surface by adding them to the sample points of the other surface and then doing same for other surfaces. These methods led each of the surfaces forming towards the next surface within the reconstruction procedure and make a continuous surface at the end. This project used the sharp feature points lying at the boundary of two surfaces, and found the closes points on each surface. The following figure 5.34 shows the basic structure that is explained by the method for preserving the continuity, which is applied on the boundary points of noise free sample points. Appendix B contains impilimentaion`s Matlab codes.

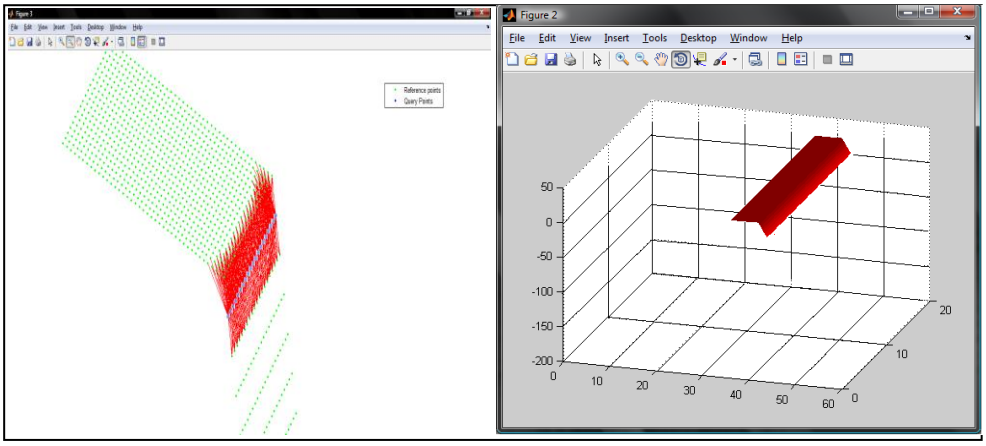


Figure 5.34: principle of merging locally reconstructed surfaces

CONCLUSION

3D modeling of indoor and outdoor environments and buildings are investigated where the main stages of Data Acquisition, Alignment or registration and Surface reconstruction are fully understood and hardware and software designs and implementations for each stage are accomplished with limited performance. Due to the numerous methods and applications available in each sequence, a comprehensive survey is practically impossible. Therefore, this research has attempted to mainly review and cover available methods and applications on data acquisition and surface reconstruction sequences; also, this research has partially covered many aspects of indoor and outdoor 3D modeling starting from scanned data.

In case of 3D data acquisition, this research has started to review and compare advantages and drawback of the systems available for 3D scanning .This research has chosen 3D laser scanner as the most accurate and fastest scanning system appropriate for scanning indoor and outdoor environments. In the second step, available 3D laser scanners have been analyzed and their drawback and limitations have been extracted. Such limitations as, the weight, complexity, disability to switch between indoor and outdoor environments in terms of resolution and mobility are highlighted.

A commercial 1D laser scanner has been converted into a 3D laser scanner with the addition of a 2 axis pan-tilt controller. The results of the 3D scanning process are sent to the PC through a wireless channel and the corresponding point cloud model is formed. In order to provide mobility to the device, a remote operating and

monitoring vehicle (ROMV) has been designed and the 3D laser scanner has been mounted on it to make the system able to scan indoor and outdoor environments automatically.

However, unfortunately the resolution of the designed 3D laser scanner could not fulfill the expectations due to the complexity of the system. Therefore, to fulfill the requirements the of data acquisition process, this project has utilized the 3D Laser Scanner provided by the Stevens University. This system is able to collect RGB data corresponding to each scanned point as well as the coordinate data. Moreover, this device is equipped with a GPS and capable to determine the position of scanner during the outdoor scan.

In case of Alignment or Registration this research reviewed, the principle of related methods in this field and classified them in three main categories: manual, automatic and semi-automatic methods. By considering this classification, this research has used the available semi- automatic method produced by the academic software tool Meshlab. Through the semi-automatic method, this software lets the user to merge and align several point cloud models of the same object or environment, which have been taken from different points of view to a with respect to unique and common reference coordinate system.

In case of surface reconstruction, the principle of main pervious methods on reconstructing surfaces based on point cloud models has been reviewed. The advantages and disadvantages of these methods are compared in terms of their ability to interpolate and produce smooth surfaces. Moreover, their ability to form close, open and complex surfaces has been analyzed. As a result, this project has

implemented the generalized implicit surface reconstruction method based on radial basis function appropriate for reconstructing seamless and smooth surface from the point cloud model.

However due to the fact that the smoothness parameter is selected globally in the previous method, the reconstructed surface is not able to present the local sharp features faithfully. A simple solution is proposed in this work, to represent the sharp features faithfully. The new method overcomes the limitation of the implicit surface reconstruction method by locally determining the surface attributes and varying the smoothness parameter of planar regions and sharp features. To demonstrate the performance of the new method, a simple point cloud model was used with two planar surfaces with a sharp edge. The performance of the method was tested under noise free and noisy point cloud models and reasonable results were obtained.

FUTRE WORK

Data acquisition: as described within the main body, the resolution of a captured point cloud strictly depends on the accuracy of the servo motors, which have used in this project as actuators for diving pan-tilt in two directions. The first drawback of using this type of motors is their inabilities to rotate continuously, which do not let the system to achieve a dense 3D scan result. The second disadvantage is since the scanning process depends on procedure of the pan-tilt, and interrupted in each step, the scanning time is remarkably increased.

For solving theses problems using brushless dc motors would be helpful to increase the resolution of the captured point cloud and reducing the time of procedure.

Alignment: as described in alignment section the key point through the registration process is finding the common features. So, most of the full automatic registration methods are working base on estimating the similar and common features such as corners, cones and sharp angels in different point clouds from the same object or environment. In the next step the translation and merging procedures perform accordingly on point clouds.

Hence, the sharp feature detection method which has used in this project could be appropriate method for extracting the common sharp features in point clouds and would help the system to perform the alignment sequence automatically.

Surface reconstruction: this project has subdivided the captured point cloud to the smaller sections and has performed the surface reconstruction method locally on each

of them. In future these sections or patches will be attached to gather to form a 3D model of an entire surface.

As described earlier the classification method which has used in this project is just able to classify the planner regions form the sharp features and does not able to distinguish between curve-linear surfaces, sharp features and planer regions. This method also does not able to recognize the amount of curvature in carve linear surfaces.

In future using a classification approach which could be able to clarify the local behavior of complex surfaces would be essential to fulfill the above requirements. In addition, categorizing the local behavior of a surface will be led to find a meaning full relation between the actual smoothness parameter and amount of desire smoothness of a surface within the locally surface reconstruction method.

REFERENCES

[1]Alliez, P., Saboret, L., & Guennebaud, G. (2009, October). *Surface Reconstruction from Point Sets. Re*trieved August 8, 2011, from CGAL: http://www.cgal.org/Manual/3.5/doc_html/cgal_manual/Surface_reconstruction_points_3/Chapter_main.html

[2]Alternatives of MAX232 in low budget projects. (2006, December 6). Retrieved August 5, 2011, from scienceprog: http://www.scienceprog.com/alternatives-of-max232-in-low-budget-projects/

[3]Bajaja, C. (1994). Free Frome Surface Design with A-patch. *Garphic interface*, (pp. 174-181).

[4]Bascom-Avr. (2011, August 10). *UART*. Retrieved August 10, 2011, from MCS electronics: http://www.mcselec.com/

[5]Bascom-AVR. (2011, August 10). *Uart*. Retrieved August 10, 2011, from MCS electronics: http://www.mcselec.com/

[6]*Basic Servo Tutorial*. (n.d.). Retrieved August 10, 2011, from cmama: http://www.rcmama.com/basic%20servo.html#Top7

[7]Bernardini, F., & Rushmeier, H. (2002). *The 3D Model Acquisition Pipeline.* Malden,USA: Blackwel.

[8]Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., & Taubin, G. (1999). The ball-pivoting algorithm for surface reconstruction. *Visualization and Computer Graphics* , 349 - 359 .

[9]Caceres, J. J., & Clint Slatton, K. (2007). *ImprovedClasssification of Building Infrastructure from Airborne Lidar Data Using Spin Images and Fusion with Ground-Based Lidar*. Folrida,USA: University of Florida.

[10]Camera, B. (2010, July 2). *Surveyor SRV-1 Blackfin Camera*. Retrieved August 10, 2011, from Surveyor SRV-1 Blackfin Camera: http://www.surveyor.com/blackfin/

[11]Carr, j., & Beaston, R. (2003). Smooth surface reconstruction from noisy range data. *Graphite* (pp. 119-126). Melborn: ACM.

[12]Chen, L. (2011). *Servo Motors – Information, Usage and Control*. Retrieved August 5, 2011, from Lirtex: http://www.lirtex.com/robotics/servo-motors-information-and-control/

[13]Chirokov, A. (2006, October 9). *Matlab Central*. Retrieved August 10, 2011, fromMathworks: http://www.mathworks.com/matlabcentral/fileexchange/10-056-scattered-data-interpolation-and-approximation-using-radial-base-functions

[14]Dinh, H. Q., Turk, G., & Slabaugh, G. (2002). Reconstructing surfaces using anisotropic basis functions. *Computer Vision* (pp. 606 - 613). IEEE.

[15]EM, B. (n.d.). *Elena Marchiori, Vrije University.*

[16]Evans, J. (2010, August). *From civil to inventor*. Retrieved August 05, 2011, fromCivil2invertor:http://civil2inventor.wordpress.com/category/autodesk/autocad-civil-3d/page/2/

[17]Fairfield, N., & Wettergreen, D. (2009). Fairfield, N.; Wettergreen, D. *Robotics and Automation*, (pp. 1637 - 1642). IEEE.

[18]François, B., Miche, l. P., & Guy, G. (2004). Accurate 3D acquisition of freely moving objects. *2nd International Symposium on 3D Data Processing, Visualization, and Transmission* (pp. 422-9). Los Alamitos: IEEE.

[19]HandyWave. (n.d.). *handy wave*. Retrieved August 10, 2011, from handy wave:http://www.handywave-usa.com/downloads/HPS_120_manual_v1.0-_english.pdf

[20]Hastiye, T. (2009, january 17). Making a step to stereo vision. Retrieved August 10, 2011, from aforgenet: http://www.aforgenet.com/articles/step-_to_stereo_vision/

[21]Haykin, S. S. (2009). RBFN. In S. S. Haykin, *Neural networks and learning machines* (p. 906). Prentice Hall.

[22]*Holography measurment*. (2007). Retrieved August 5, 2011, from Solarius: http://www.solarius-inc.com/html/holo.html

[23]Hoppe, H. (1994). *Surface Reconstruction from Unorganized Points.* Washington: University of Wshington.

[24]Jorge, i. (2009, May 3). *Coord system*. Retrieved 8 10, 2011, from wikipedia: http://en.wikipedia.org/wiki/File:Coord_system_SZ_0.svg

[25]Katsushi, L. (2001). Modeling from Reality. *3rd International Conference on 3-D Digital Imaging and Modeling* (pp. 117-124). Quebec: IEEE Computer Society.

[26]Keqiang, Z., Yan, Z., & Wei, W. (2009). Automated 3D scenes reconstruction for mobile robots using laser scanning. *Control and Decision Conference, 2009. CCDC '09. Chinese* (pp. 3007 - 3012). Dalian, China: IEEE.

[27]Kirillov, A. (2009, january 17). *Making a step to stereo vision*. Retrieved August 10, 2011, from aforgenet: http://www.aforgenet.com/articles/step-_to_stereo_vision/

[28]Kurisu, M., Yokokohji, Y., & Oosto, Y. (2005). Development of aLaser Range Finder for 3D map-Building in Rubble. *IEEE International Conference on Mechatronics and Automation,* (pp. 1842-1847). NiagaFalls,Canada: IEEE International Conference.

[29]Li, X., Wan, W., Cheng, X., & Cui, B. (2010). An improved Poisson Surface Reconstruction algorithm. *Audio Language and Image Processing (ICALIP)* (pp. 1134 - 1138). IEEE.

[30]Martinez, J., Mandow, A., Reina, A., & Morales, J. (2009). Outdoor Scene Registration from 3D Laser Range Data with Coarse Binary Cubes. *Industrial Electronics*, (pp. 2308 - 2313). IEEE.

[31]Martinez, J., Reina, A., & Mandow, A. (2007). Spherical Laser Point Sampling with Application to 3D Scene Genetic Registration. *Robotics and Automation* (pp. 1104 - 1109). IEEE.

[32]Matlab. (n.d.). matworkexchangefilecenter(2011)

[33]Mika, S., & Andreas, G. (n.d.). *Calc 3D*. Retrieved August 10, 2011, from calc3d: http://www.calc3d.com/help/ehelp.html

[34]Mike. (2008, March 9). *LIDAR-scanned-SICK-LMS-animation*. Retrieved August 10, 2011, from wikipedia: http://en.wikipedia.org/wiki/File:LIDAR-scanned-SICK-LMS-animation.gif

[35]Myoung-Jong, Y., Gu-Young, J., & Yu, K.-H. (2008). Generation of space grid map by 3D detection of obstacle distribution. *Control, Automation and Systems* (pp. 2054 - 2057). IEEE.

[36]Noptel. (n.d.). *Noptel*. Retrieved August 10, 2011, from Noptel: http://www.noptel.fi/eng/company/index.php?doc=3_products/cataloque

[37]Plantininga, S., & Vegeter, g. (2006). Isotopic meshing of implicit surfaces. *Visual Compute,Springer* , 45-58.

[38]PololuTeam. (2011, August 10). *TRex controller*. Retrieved August 10, 2011, from Pololu roboric & electronics: http://www.pololu.com/

[39]Potentials, E. S. (2009). Efficient Surface Reconstruction From Noisy Data Using Regularized Membrane Potentials. *Image Processing,IEEE Transaction* , 1119 - 1134.

[40]Quan Dinh, H., & Turk, G. (2002). areconstructing Surfaces by Volumetric Regularization Using Radial Basis Functions. *IEEE* , 1358-1371.

[41]Sniderman, D. (2010, December 14). *3D scanning 101*. Retrieved August 5, 2011, from DE(Desctop Engineering): http://www.deskeng.com/articles/-aaazje.htm

[42]Spherical Co-ordinate System. (2009, November 27). Retrieved Agust 10, 2011, from emtmadeeasy.blogspot: http://emtmadeeasy.blogspot.com/2009/-11/spherical-coordinate-system.html

[43]Sze. (2011). *ZCorp ZScanner 600 Portable 3D Laser Scanner*. Retrieved August 5, 2011, from iTech News Net: http://www.itechnews.net/2009/09/-16/zcorp-zscanner-600-portable-3d-laser-scanner/

[44]Takeuchi, E., & Tsubouchi, T. (2007). A 3-D Scan Matching using Improved 3-D Normal Distributions Transform for Mobile Robotic Mapping. *Intelligent Robots and Systems* (pp. 3068 - 3073). IEEE.

[45]Team, M.-l. (2011, February 16). *meshlab*. Retrieved August 10, 2011, from mesh lab: http://meshlab.sourceforge.net/

[46]*Using a wiper motor in*. (2011, August). Retrieved August 10, 2011, from scary-terry: http://www.scary-terry.com/wipmtr/wipmtr2.htm

[47]Weber, C., & Hamilton, S. (2010). Sharp Feature Detection in Point Clouds. *Shape Model and applications* (pp. 1-12). TU Kaiseslauten: IEEE.

[48]Wiora, G. (2006, April 10). *wikipedia*. Retrieved August 2011, from en.wikipedia: http://en.wikipedia.org/wiki/File:Laserprofilometer_EN.svg

[49]Wolf, D., Howard, A., & Sukhatme, G. (2005). Towards geometric 3D mapping of outdoor environments using mobile robots. *Intelligent Robots and Systems* (pp. 1507 - 1512). Los Angeles, CA, USA: IEEE.

[50]Zhang, B., & Smith, W. (2011). 3D city site model extraction through point cloud generated from stereo images. *Computing for Geospatial Research & Application.* ew York, NY, USA: ACM New York, NY, USA ©2011.

[51]Zhou, K., Gong, M., Huang, X., & Guo, B. (2011). Data-Parallel Octrees for Surface Reconstruction. *Visualization and Computer Graphics* , 669 - 681.

APPENDICES

Appendix A: Bascom AVR

```
$regfile "m32def.dat"
$crystal = 8000000
Config Portb = Output
Config Porta.5 = Input
Config Porta.4 = Output
  Config Timer1 = Timer , Compare A = Clear , Compare B = Clear , Clear Timer = 0 ,
Prescale = 1
  Dim C As Byte
  Dim A As Word
  Dim B As Word
  Dim Status As Bit
  Config Portb.2 = Input
  Config Portb.1 = Output
  Config Portb.3 = Output
  Enable Interrupts
  Enable Timer1
  Enable Oc1a
  Enable Oc1b
  Enable Ovf1
  On Ovf1 Ts
  On Oc1a Tim
  On Oc1b Tim2
'---------------
Main:
Do
Start Timer1
Compare1b = 10000

 For B = 10000 To 20384 Step 100
    For A = 6190 To 20384 Step 25
      Waitms 40
      Compare1a = A
    Next A
   Compare1b = B
   Next

Loop

End                                                    'end program

'---------------------------------
Tim2:
Reset Portb.1
Return

Tim:
Reset Portb.0
Return

Ts:

C = C + 1
If C = 3 Then
 Timer1 = 0
 Set Portb.0
 Set Portb.1
 C = 0
Else

Timer1 = 20500
End If

Return
```

Appendix B: Matlab Codes

```
% ----------sharp feature detection and locally surface reconstruction-------

%load data

clear all
close all
clc
inputdata=load ('C:\backup10jun2011\scan\2010-9-17-16-49-41\outglob.xyz');
  x = inputdata(10000:15000,1);
  y = inputdata(10000:15000,2);
  z= inputdata(10000:15000,3);
p=[x,y,z];
X=p';

% --------------------------------Octree partitoning-------------------------
make octree by length
find the number of points in each octtee block

Octree = BuildOctree(x, y, z,8)
for i=(1:size(Octree,2))
     j(i)=size(Octree(1,i).group,2);
for k=1:j(i)
samp=Octree(1,i).group(1,k).child;
hold on


scatter3(x(samp),y(samp),z(samp),'.');
sampn=length(samp);
end
hold on
end


% --------------------------------select Otree Partition----------------------

%select one of the Octree blocks as a sample
%find the number of samples on it
%find the number of clusters, clustering and find the center of each of the %by k-
means command in 5 iterations.
samp=Octree(1,3).group(1,8).child;
figure
scatter3(x(samp),y(samp),z(samp),'.');
p=[x(samp),y(samp),z(samp)];
sampn=length(samp);

 %number of samples in each octree box
clustn=round(sampn/16);

 %Cluster number. Number of partitions mean of them is equal to the defined %number

cmat=[x(samp),y(samp),z(samp)]; %matrix of samples
x1=x(samp);
y1=y(samp);
z1=z(samp);




i=0;
[f,Cent] = kmeans(cmat,clustn,'replicates',3);
qp=Cent;
%f=vector contains clustered points
figure
scatterd(cmat',f);
hold on
scatter3(Cent(:,1),Cent(:,2), Cent(:,3),'k*');
figure
for i=1:clustn
  [r,c] = find(f==i);
  sortmat(i)=length(r); %number of points in each cluster
end
```

```
%---------------------------K-nearest point search-------------------------
Find k nearest points to each center and make the n clusters
Parameter k specifies the number of neighborhoods to each center or qp or quarry
points
k=max(sortmat);
[KNNG,distances,dis1vec,dis2vec,dis3vec,p2x,p2y,p2z]= myKN(p,qp,k);

%---------------------------Normal vectors extraction-----------------------
%find the normal vectors respect to the number of neighbors at each center
%point and ploting process
pnormalvec=[0,0,0];
pnormalvec1=[0,0,0];
 for i=1:clustn
    for j=1:k-1
normalvec=cross([dis1vec(j,i),dis2vec(j,i),dis3vec(j,i)],[dis1vec(j+1,i),dis2
vec(j+1,i),dis3vec(j+1,i)]);
pnormalvec1=pnormalvec1+normalvec;
    end
     pnormalvec(i,:)=normr(pnormalvec1);   %normalized normal vector
     pnormalvec1=[0,0,0];
 end

%------------------------- plot normal vectors ----------------------------

quiver3(qp(:,1),qp(:,2),qp(:,3),abs(pnormalvec(:,1)),abs(pnormalvec(:,2)),a(pnormalve
c(:,3)));
hold off
figure
h1=plot3(p(:,1),p(:,2),p(:,3),'g.');
hold on

%------------------------------ plot Gauss map ----------------------

quiver3(cent1(:,1),cent1(:,2),cent1(:,3),abs(pnormalvec(:,1)),abs(pnormalvec(:,2)),ab
s(pnormalvec(:,3)));







%------------------- Gauss map clustering base on Geodesic distance --------

new=[abs(pnormalvec(:,1)),abs(pnormalvec(:,2)),abs(pnormalvec(:,3))];
  D = pdist(new,@spdisf);
  L = linkage(D,'single');
  T = cluster(L,'maxclust',4);

  figure
  scatter3(new(:,1),new(:,2),new(:,3),20,T,'filled')



%---------------------------- Select planner regions and plot ------------------

 C=rand(1,3);
 figure
 for cc1=1:max(T)
   c1=find(T==cc1);
   for cc2=1:size(c1,1)
   c2=find(f==c1(cc2));
   scatter3(x1(c2),y1(c2),z1(c2),90,C,'filled')
   hold on
   end
   C=rand(1,3);
   hold on
 end

end


%---------------------------- Select edge points and plot ------------------

edgesp=[];
C=rand(1,3);
figure
```

```
c1=find(edges<(max(clustn)/5))
 for cc2=1:length(c1)
     c2=(find(T==c1(cc2)))
     C=rand(1,3);
      for cc1=1:length(c2)
      c3= find(f==c2(cc1))
      edgesp=[c3;edgesp]
      scatter3(x1(c3),y1(c3),z1(c3),50,C,'filled')
      hold on
      end
 end
```

```
%--------- Implicit surface reconstruction using Radial basis Functions -----

ti1 = min(x1(edgesp)):0.2:max(x1(edgesp));
ti2= min(y1(edgesp)):0.2:max(y1(edgesp));
[XI,YI] = meshgrid(ti1,ti2);
op=rbfcreate([x1(:)'; y1(:)'], z1(:)','RBFFunction', 'cubic');
rbfcheck(op);
rbfcheck(op);
ZI = rbfinterp([XI(:)'; YI(:)'], op);
ZI = reshape(ZI, size(YI));
mesh(XI,YI,ZI), hold
edgesp=[];

 end  %program


%-------------------------------------------------------------------------
%-------------------------------------------------------------------------
%---------------------------------Functions-------------------------------



%---------------------------------myKN-------------------------------------


function [KNNG,distances,dis1vec,dis2vec,dis3vec,p2x,p2y,p2z]= myKN(p,qp,k)

% N=reference points
%  Nq=query points
% k= neighborhood size



tic
ptrtree=BuildGLTree3D(p');
fprintf('\tGLTree built in %4.4f s\n\treturned pointer %4.0f:\n\n',toc,ptrtree);

fprintf('START K NEAREST NEIGHBOR SEARCH:\n')
tic
[KNNG,distances]=KNNSearch3D(p',qp',ptrtree,k);
 fprintf('\t  NNsearch of%4.0f neighbors in %4.0f reference points and %4.0f query
points took: %4.4f s\n\n',k,length(p),length(qp),toc);

[KNNG2, distances2]=BruteSearchMex(p',qp','k',k);

fprintf('DELETING THE TREE\n\n')
DeleteGLTree3D(ptrtree);

fprintf('TEST SUCCESFULLY COMPLETED !!!\n\n')

figure
title([num2str(k),' Neighbours'],'fontsize',14);
axis equal
 hold on
h1=plot3(p(:,1),p(:,2),p(:,3),'g.');
h2=plot3(qp(:,1),qp(:,2),qp(:,3),'b.');
legend([h1,h2],'Reference points','Query Points');
p1x=qp(:,1);
p1y=qp(:,2);
```

```
p1z=qp(:,3);

for j=1:k
 p2x=p(KNNG(:,j),1);
 p2y=p(KNNG(:,j),2);
 p2z=p(KNNG(:,j),3);
 plot3([p1x,p2x]',[p1y,p2y]',[p1z,p2z]','r-')
 dis1vec(j,:)=[(p1x-p2x)];
 dis2vec(j,:)=[(p1y-p2y)];
 dis3vec(j,:)=[(p1z-p2z)];

end

%--------------------------------------------------------------------------


%------------------------------Build Octree----------------------------

function Octree = BuildOctree(x, y, z, MinCubeSide)
%BUILDOCTREE Build an Octree representation from a set of points in space.
%   OCTREE = BUILDOCTREE(X, Y, Z, MINCUBESIDE) will create a structured variable
%   containing the octree representation of (X, Y, Z).  MINCUBESIDE is the
%   smallest allowable side length of a cube in the tree.  Only non-empty
%   groups will be represented in the tree.
%
%   The structure, OCTREE, will have the following fields,
%
%   OCTREE(1:NLEVEL+1).GROUP(1:NGROUP).CHILD(1:NCHILD)
%   OCTREE(1:NLEVEL+1).GROUP(1:NGROUP).GROUPCENTER
%   OCTREE(1:NLEVEL+1).GROUP(1:NGROUP).CUBELENGTH
%
%   At every level, NLEVEL, of the tree, there are NGROUPS.  For each group,
%   there are NCHILD children.  GROUPCENTER is the (x0(k), y0(k), z0(k)) coordinate
%   at the center of GROUP(k). CUBELENGTH is the side length of the cube whose
%   center is at (x0(k), y0(k), z0(k)).
%
%   NOTE: A typical octree representation is numbered 0:NLEVEL. The output OCTREE in
%         BUILDOCTREE is numbered 1:NLEVEL+1 since a Matlab array can not be indexed
%         by 0.
%
%   David Sall
%   david.sall@gmail.com
%   03/01/2011
%

%%%%%
%%%%%   EQUATIONS NEEDED TO SET UP THE OCTREE
%%%%%

% ... The following equations* are used to generate the Octree.  At every level,
except NLEVEL,
% ... of the tree the parents and children can be identified by a unique ID.  At
level, NLEVEL,
% ... the children are identified by their integer value K, i.e., (x(K), y(K), z(K))
K = 1,
% ... length(x).

% ... 1) Compute the unique indicies for each element at the finest level, l =
Nlevel.
% ...
% ...    i = (x - xmin) / dl
% ...    j = (y - ymin) / dl
% ...    k = (z - zmin) / dl
% ...
% ... 2) Compute a unique ID for a given (i, j, k) index at any level, ilevel.
% ...
% ...    ID = 2^(2*ilevel) * k + 2^(ilevel) * j + i
% ...
% ... 3) Compute the indicies (i, j, k) given the unique ID at any level, ilevel.
% ...
% ...    k = ID / 2^(2*ilevel)
% ...    j = ( ID - 2^(2*ilevel) * k ) / 2^(ilevel)
% ...    i = ID = 2^(2*ilevel) * k - 2^(ilevel) * j
% ...
% ... 4) Compute the PARENT (i, j, k) indicies given the CHILD (i, j, k) indicies.
% ...
% ...    i_parent = i_child / 2
```

```
% ...    j_parent = j_child / 2
% ...    k_parent = k_child / 2
% ...
% ... 5) Compute the center (x, y, z) location of a group at any level using the
minimum
% ...    location of the box at the root level and the (i, j, k) indicies.
% ...
% ...    x_center = xmin + (i + .5) * dl
% ...    y_center = ymin + (j + .5) * dl
% ...    z_center = zmin + (k + .5) * dl
% ...
% ... * "Integral Equation Methods for Computational Electromagnetics", Prof. Stephen
Gedney,

%%%%%
%%%%% SET UP THE BOUNDING CUBE, INITIALIZE VARIABLES, AND NUMBER OF LEVELS IN THE
TREE
%%%%%

% ... determine the point set extrema.
xmax = max([x y z], [], 1);
xmin = min([x y z], [], 1);

% ... expand the extrema by a small amount to make sure that, numerically, all points
in the data set will
% ... be included in the tree.
xdel = (xmax-xmin)/2*.0005;
xmax = xmax+xdel;
xmin = xmin-xdel;

% ... side length of a cube that will completely enclose all points in the data set.
Lc = max(xmax-xmin);

% ... shift the center of the bounding cube.
XYZCenter  = (xmax+xmin)/2;
CubeCenter = xmin+Lc/2;
xmin       = xmin-(CubeCenter-XYZCenter);

% ... number of levels in the Octree.  Level number NLEVEL is the finest level, i.e.,
the smallest size
% ... cube.  Level number 0 is the root level and is the largest side, i.e.,
bounding, cube.
% ... NOTE: Matlab does not allow idexing an array with 0 so the Octree index is
shifted by 1 (so that
% ...       Octree(0:Nlevel) is indexed, Octree(1:Nlevel+1)).
Nlevel = round(log10(Lc/MinCubeSide)/log10(2));

% ... integer constants 2^l  and 2^(2l).
twol  = 2^Nlevel;
two2l = twol^2;
% ... side length of each cube at the finest level.
dl    = Lc/twol;


%%%%%
%%%%% SET UP ARRAYS AT THE FINEST LEVEL
%%%%%

% ... NOTE: at the finest level, the CHILD ID's are actually the index of each point
in the data set.  The
% ...       associated PARENT ID's are the ID number of the cube where each point
resides.

% ... compute the parent ID for each data point ...
i      = floor((x-xmin(1))/dl);
j      = floor((y-xmin(2))/dl);
k      = floor((z-xmin(3))/dl);
parent = two2l*k+twol*j+i;

% ... and the center of each cube containing each point.
groupcenter = [xmin(1)+(i+.5)*dl xmin(2)+(j+.5)*dl xmin(3)+(k+.5)*dl];


%%%%%
%%%%% BUILD THE OCTREE
%%%%%
```

```
% Loop from the finest level to the root level.
for ilevel = Nlevel+1:-1:1

    % ... sort the PARENT ID's into ascending order.  The index sort vector are the
associated CHILD
    % ... group numbers.
    [parent, child] = sort(parent);

    % ... find the index of the last parent in each group.
    group = find(diff([parent; parent(end)+1]));

    % ... number of groups.
    Ngroups = length(group);

    % ... load the Octree at ILEVEL.  The PARENT ID is replaced by an integer counter
(the unique ID
    % ... at each level is only used to determine the groups at each level).
    ib = 1;
    for igroup = 1:Ngroups

        % ... update the ending index number of the current group.
        ie = group(igroup);

        % ... children.
        Octree(ilevel).group(igroup).child = child(ib:ie);

        % ... center of the cube containing all of the children (i.e., the cube
center associated
        % ... with the PARENT ID).
        Octree(ilevel).group(igroup).groupcenter = groupcenter(child(ib),:);

        % ... side length of the cube whose center is given above.
        Octree(ilevel).group(igroup).cubelength  = dl;

        % ... update the starting index number of the next group.
        ib = ie+1;

    %---------------------------------
    center=Octree(ilevel).group(igroup).groupcenter;
lenght=dl;
a=[0 1 1 0 0 0;1 1 0 0 1 1;1 1 0 0 1 1;0 1 1 0 0 0]*lenght;
b=[0 0 1 1 0 0;0 1 1 0 0 0;0 1 1 0 1 1;0 0 1 1 1 1]*lenght;
c=[0 0 0 0 0 1;0 0 0 0 0 1;1 1 1 1 0 1;1 1 1 1 0 1]*lenght;
for f=1:6
  % scatter3(1,2,3)

    h=patch(a(:,f)-(lenght/2)+center(1),b(:,f)-(lenght/2)+center(2),c(:,f)-
(lenght/2)+center(3),'w');
    set(h,'EdgeColor','k','LineWidth',1,'FaceColor','none')
end

    %---------------------------------



    end

    % ... extract the unique PARENT ID's along with their associated indicies.
    [parent, igroup] = unique(parent);
    i = i(igroup);
    j = j(igroup);
    k = k(igroup);

    % ... exit the loop if at the root level.  All work is done at this point and the
following
    % ... computations are unnecessary.
    if ilevel == 1, break;, end

    %%%%%
    %%%%% COMPUTE THE PARENT ID'S FOR THE NEXT LEVEL UP
    %%%%%

    % ... update the constants used to convert (i,j,k) indicies to the PARENT ID's.
    twol  = twol/2;
    two2l = two2l/4;
    dl = dl*2;
```

```
    % ... compute the PARENT ID's for the next level using the CHILD indicies ...
    i = floor(i/2);
    j = floor(j/2);
    k = floor(k/2);
    parent = two2l*k+twol*j+i;
    % ... and the associated center of each group.
    groupcenter = [xmin(1)+(i+.5)*dl xmin(2)+(j+.5)*dl xmin(3)+(k+.5)*dl];

end


% End of function BuildOctree
return
```

```
 %--------------------------------------------------------------------
```

```
------------------implicit surface reconstruction (rbfcreat)----------
```

```
 function options = rbfcreate(x, y, varargin)
%RBFCREATE Creates an RBF interpolation
%   OPTIONS = RBFSET(X, Y, 'NAME1',VALUE1,'NAME2',VALUE2,...) creates an
%   radial base function interpolation
%
%   RBFCREATE with no input arguments displays all property names and their
%   possible values.
%
%RBFCREATE PROPERTIES
%

%
% Alex Chirokov, alex.chirokov@gmail.com
% 16 Feb 2006
tic;
% Print out possible values of properties.
if (nargin == 0) & (nargout == 0)
  fprintf('               x: [ dim by n matrix of coordinates for the nodes ]\n');
  fprintf('               y: [   1 by n vector of values at nodes ]\n');
  fprintf('     RBFFunction: [ gaussian  | thinplate | cubic | multiquadrics |
{linear} ]\n');
  fprintf('     RBFConstant: [ positive scalar     ]\n');
  fprintf('       RBFSmooth: [ positive scalar {0} ]\n');
  fprintf('           Stats: [ on | {off} ]\n');
  fprintf('\n');
  return;
end
Names = [
    'RBFFunction      '
    'RBFConstant      '
    'RBFSmooth        '
    'Stats            '
];
[m,n] = size(Names);
names = lower(Names);

options = [];
for j = 1:m
  options.(deblank(Names(j,:))) = [];
end

%*************************************************************************
%Check input arrays
%*************************************************************************
[nXDim nXCount]=size(x);
[nYDim nYCount]=size(y);

if (nXCount~=nYCount)
  error(sprintf('x and y should have the same number of rows'));
end;
```

```
if (nYDim~=1)
  error(sprintf('y should be n by 1 vector'));
end;

options.('x')           = x;
options.('y')           = y;
%************************************************************************
%Default values
%************************************************************************
options.('RBFFunction') = 'linear';
options.('RBFConstant') = (prod(max(x')-min(x'))/nXCount)^(1/nXDim); %approx. average
distance between the nodes
options.('RBFSmooth')   = 0.0;
options.('Stats')       = 'off';

%************************************************************************
% Argument parsing code: similar to ODESET.m
%************************************************************************

i = 1;
% A finite state machine to parse name-value pairs.
if rem(nargin-2,2) ~= 0
  error('Arguments must occur in name-value pairs.');
end
expectval = 0;                          % start expecting a name, not a value
while i <= nargin-2
  arg = varargin{i};

  if ~expectval
    if ~isstr(arg)
      error(sprintf('Expected argument %d to be a string property name.', i));
    end

    lowArg = lower(arg);
    j = strmatch(lowArg,names);
    if isempty(j)                       % if no matches
      error(sprintf('Unrecognized property name ''%s''.', arg));
    elseif length(j) > 1                % if more than one match
      % Check for any exact matches (in case any names are subsets of others)
      k = strmatch(lowArg,names,'exact');
      if length(k) == 1
        j = k;
      else
        msg = sprintf('Ambiguous property name ''%s'' ', arg);
        msg = [msg '(' deblank(Names(j(1),:))];
        for k = j(2:length(j))'
          msg = [msg ', ' deblank(Names(k,:))];
        end
        msg = sprintf('%s).', msg);
        error(msg);
      end
    end
    expectval = 1;                      % we expect a value next

  else
    options.(deblank(Names(j,:))) = arg;
    expectval = 0;
  end
  i = i + 1;
end

if expectval
  error(sprintf('Expected value for property ''%s''.', arg));
end


%************************************************************************
% Creating RBF Interpolatin
%************************************************************************

switch lower(options.('RBFFunction'))
      case 'linear'
        options.('rbfphi')    = @rbfphi_linear;
      case 'cubic'
        options.('rbfphi')    = @rbfphi_cubic;
      case 'multiquadric'
        options.('rbfphi')    = @rbfphi_multiquadrics;
```

```
      case 'thinplate'
        options.('rbfphi')    = @rbfphi_thinplate;
      case 'gaussian'
        options.('rbfphi')    = @rbfphi_gaussian;
    otherwise
        options.('rbfphi')    = @rbfphi_linear;
end

phi       = options.('rbfphi');

A=rbfAssemble(x, phi, options.('RBFConstant'), options.('RBFSmooth'));

b=[y'; zeros(nXDim+1, 1)];

%inverse
rbfcoeff=A\b;

%SVD
% [U,S,V] = svd(A);
%
% for i=1:1:nXCount+1
%     if (S(i,i)>0) S(i,i)=1/S(i,i); end;
% end;
% rbfcoeff = V*S'*U*b;


options.('rbfcoeff') = rbfcoeff;


if (strcmp(options.('Stats'),'on'))
    fprintf('%d point RBF interpolation was created in %e sec\n', length(y), toc);
    fprintf('\n');
end;

function [A]=rbfAssemble(x, phi, const, smooth)
[dim n]=size(x);
A=zeros(n,n);
for i=1:n
    for j=1:i
        r=norm(x(:,i)-x(:,j));
        temp=feval(phi,r, const);
        A(i,j)=temp;
        A(j,i)=temp;
    end
    A(i,i) = A(i,i) - smooth;
end
% Polynomial part
P=[ones(n,1) x'];
A = [ A      P
      P' zeros(dim+1,dim+1)];

%******************************************************************************
% Radial Base Functions
%******************************************************************************
function u=rbfphi_linear(r, const)
u=r;

function u=rbfphi_cubic(r, const)
u=r.*r.*r;

function u=rbfphi_gaussian(r, const)
u=exp(-0.5*r.*r/(const*const));

function u=rbfphi_multiquadrics(r, const)
u=sqrt(1+r.*r/(const*const));

function u=rbfphi_thinplate(r, const)
u=r.*r.*log(r+1);

%******************************************************************************
%******************************************************************************
```