

1 INTRODUCTION

1.1 ABSTRACT

A 3D scanner is a device that analyses a real-world object or environment to collect data on its shape and possibly its appearance. The collected data can then be used to construct digital, three dimensional models. Many different technologies can be used to build these 3D scanning devices; each technology comes with its own limitations, advantages and costs.

The project involves creation of a 3D scanner which scans the object to get 2D images and then processes these images to reconstruct the 3D object model using surface rendering. The objective is to reconstruct the 3D replica of the scanned object using Ball Pivoting Algorithm for surface reconstruction and to retain the strengths of previous interpolating techniques in a method that exhibits linear time complexity and robustness on real scanned data. Also, generating the correct topology for the object along with maintaining efficient usage of time and memory and ensuring numerical stability are some of the goals of the project.

1.2 INTRODUCTION AND MOTIVATION

3D modelling is well known in entertainment and medical fields but its applications reach out to many design and manufacturing processes too. The 3D scanner developed during the course of this project is a non-contact one since it uses a line laser to capture the geometry of the object. Also, the entire system includes the scanning as well as the reconstruction process. This type of scanning system develops the 3D replica by capturing and processing the 2D images of the object.

Collected 3D data is useful for a wide variety of applications. These devices are used extensively by the entertainment industry in the production of movies and video games. Other common applications of this technology include industrial design, orthotics and prosthetics, reverse engineering and prototyping, quality control/inspection and documentation of cultural artifacts.

1.3 PROBLEM STATEMENT

The project involves creation of a 3D scanner which scans the object to get 2D images and then processes these images to reconstruct the 3D object model using surface rendering. The objective is to reconstruct the 3D replica of the scanned object using Ball Pivoting Algorithm for surface reconstruction and to retain the strengths of previous interpolating techniques in a method that exhibits linear time complexity and robustness on real scanned data. Also, generating the correct topology for the object along with maintaining efficient usage of time and memory and ensuring numerical stability are some of the goals of the project.

1.4 SCOPE OF THE PROJECT

The project (3D Scanner) would include hardware as well as software implementation. The hardware setup consists of a camera collecting real time frames of the subject using line laser. These frames will be fed to the software module where point cloud, mesh model and the final rendered replica of the subject will be the output. It will give the 3D replica as the output for only simple objects, not having too many sudden changes in the geometry of the object. The 3D replica will not reflect the color of the object but will only reflect the geometry of the object.

1.5 ORGANIZATION OF THE PROJECT

The project report is divided into five major topics. They are as follows.

Chapter 2: Review of Literature

This chapter explains in detail the domain of this project, methods available for implementation, existing systems and their limitations and techniques used to implement the project. It also mentions the hardware and software requirements of the project.

Chapter 3: Analysis

This chapter includes a detailed analysis of the project with its functional and non-functional requirements. The chapter also gives an overview of the proposed system.

Chapter 4: Design

This chapter explains the entire design process in brief. It also mentions the various issues regarding the system and the challenges faced during the course of time.

Chapter 5: Implementation and Results

This chapter includes all the implementation details regarding the project and the proposed solutions for the major issues encountered during the project. It also explains the results achieved and their significance.

Chapter 6: Testing

This chapter focusses on the various test cases of the project and their relevance. It also includes the results achieved along with their detailed analysis.

Chapter 7: Conclusion and Further Work

This chapter includes the conclusion by detailing the outcome of the project. A brief about the possible enhancements on the project is also given in the chapter.

2 REVIEW OF LITERATURE

2.1 CURRENT METHODOLOGY AND TECHNOLOGY USED

2.1.1 Scanner Technology

There are a variety of technologies for digitally acquiring the shape of a 3D object. A well-established classification divides them into two types: contact and non-contact 3D scanners. Non-contact 3D scanners can be further divided into two main categories, active scanners and passive scanners. There are a variety of technologies that fall under each of these categories.

Contact 3D scanners probe the subject through physical touch, while the object is in contact with or resting on a precision flat surface plate, ground and polished to a specific maximum of surface roughness.

Active non-contact scanners emit some kind of radiation or light and detect its reflection in order to probe an object or environment. Possible types of emissions used include light, ultrasound or x-ray.

Passive non-contact scanners do not emit any kind of radiation themselves, but instead rely on detecting reflected ambient radiation. Most scanners of this type detect visible light because it is a readily available ambient radiation. Other types of radiation, such as infrared could also be used. Passive methods can be very cheap, because in most cases they do not need particular hardware but simple digital cameras.

2.1.2 Reconstruction Algorithms

Existing systems use two classes of methods for estimating surface points. The first method is the volumetric method which constructs the mesh model by applying the marching cubes algorithm to the volumetric structure of the object, generated using a distance function. This mesh model is then rendered to get the original object.

The second method is the mesh stitching method where disjoint meshes computed individually from separate scans are stitched into a single surface. Interpolation of meshes is an important step towards construction of the mesh model.

The existing techniques fall into two categories, sculpting based and region-growing based. In sculpting based methods, a volume tetrahedralization is computed from the data points, typically using the 3D Delaunay Triangulation. Region-growing method starts with a seed triangle, considers a new point and joins it to the existing region boundary, and continues until all points have been considered.

2.1.3 Limitations of existing methods

The sculpting methods guarantee the correct topology but computing the required 3D Delaunay triangulation can be prohibitively expensive in terms of time and memory required, and can lead to numerical instability when dealing with datasets of millions of points. Also since these algorithms work in single pass, the probability of them arriving to maximum accuracy is low as they might not be able to process the ‘out of core extensions’.

2.2 METHODOLOGY AND TECHNOLOGY USED

The project involved developing a 3D scanning and reconstruction system. The scanner developed is a non-contact one as it captures the reflected light of the laser line incident on the object. The algorithm used for reconstruction is the Ball Pivoting algorithm which follows an advancing front paradigm. This algorithm is suitable for objects with non-uniform surfaces and disoriented normals. Also, the functioning of the algorithm allows for optimization of time and memory. Also, due to the fact that it can be executed in multiple passes to solve ‘out of core extensions’, it poses an advantage as compared to other algorithms like Marching Cubes, Delaunay triangulation, etc.

2.3 PROJECT OVERVIEW

The proposed System will aim to create a three dimensional replica of a real world object using a line laser. The camera will record the images of the object as the laser line falls on it. Using this sequence of images, the system will generate a structure of points that would represent the object in 3D space. The next step converts the point cloud to get the mesh model which in turn is rendered to generate the digital 3D replica of the original object.

As the system relies on the object being placed on the turntable it will be able to scan only smaller objects but it can be expanded to create larger versions without many changes in design. The system will also not be very sensitive to minute details in the object as the angle of rotation is a limitation and so is the quality of the laser and camera, so lot of sudden changes might be missed by the scanner.

These factors can easily be changed by using a high quality camera and laser with a turning mechanism that would give very smooth turning of the object and allow a video feed to be used to collect a very large number of images giving us high detailing. The ball rolling technique used for processing the image is a faster and more accurate technique than those that have been used in the past .It will reduce the processing time for the images giving us the 3D image faster and more accuracy.

3 ANALYSIS AND DESIGN

3.1 REQUIREMENT ANALYSIS

3.1.1 Functional Requirements

The experimental setup would scan the object to be modelled in frames with the help of laser and camera to give a sequence of images. This information is fed to the software module which initially does some pre-processing on these images. These images are further manipulated to create the point cloud of the object. Point cloud is given as input to the BPA algorithm which generates the mesh model. Finally, mesh model will be rendered (surfacing) to obtain the object's replica in three dimensions. The use case diagram representing the functionalities of the system is shown in Fig 3.1.

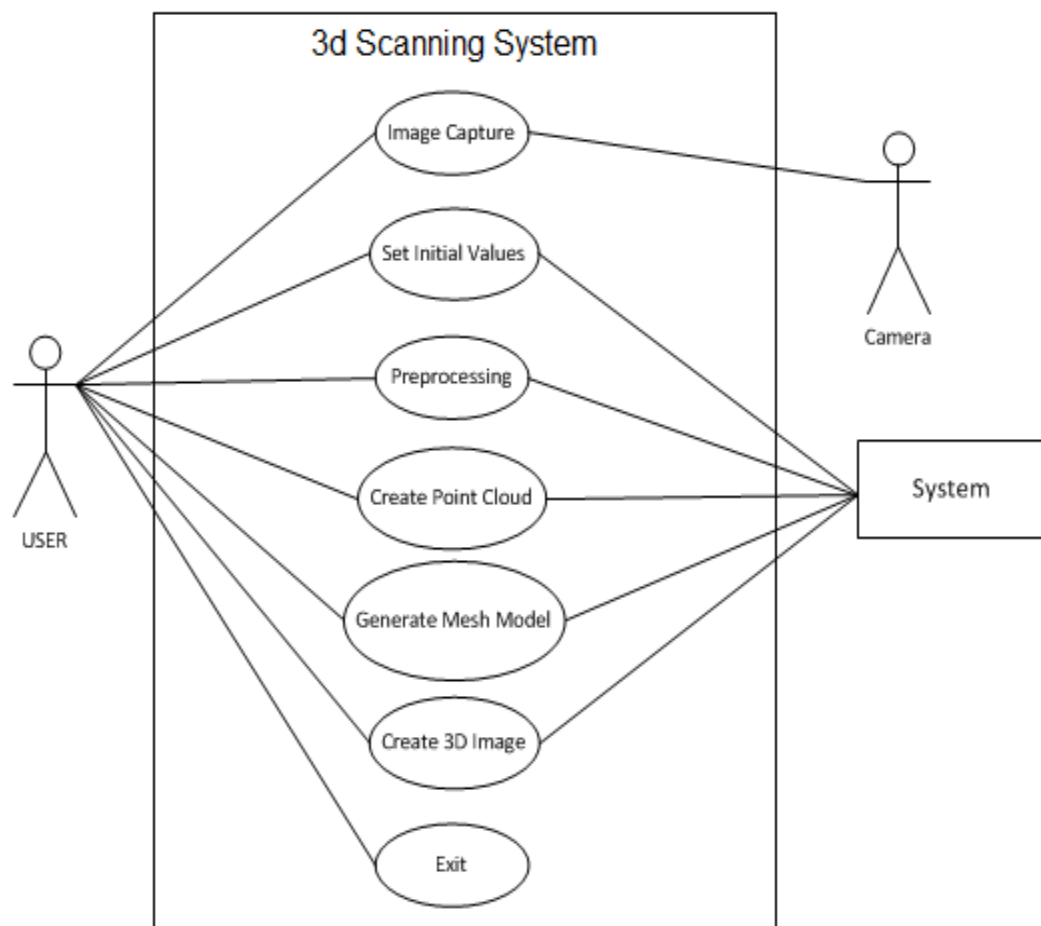


Figure 3.1 Use Case Diagram

3.1.2 Non-Functional Requirements

Robustness: Robustness is the ability of an algorithm to continue to operate despite abnormalities in input, calculations, etc. The objects may differ but the algorithm should be able to produce satisfactory results for each object.

Efficiency: Efficiency in general describes the extent to which time or effort is well used for the intended task or purpose. With the help of Ball-pivoting Algorithm (BPA) used for surface reconstruction, the amount of time taken in reconstruction is reduced by a substantial amount making the system more efficient.

3.1.3 Software & Hardware Requirements

Software

MATLAB 5.3.1 or higher (Licensed copy)

Hardware

RAM: 2048 MB

Processor: i3 or higher, any AMD X86 processor

Disk space: 2GB minimum

Camera resolution: 3.2MP or higher

Line laser: Red (3.5V~4.5V 16mm 5mW)

Operating System

Windows: Windows Server 2003 R2 Service Pack 2 or higher

Linux: Debian 5.x or higher

Mac: Mac OS X 10.6.4 (Snow Leopard) or higher

3.2 PROJECT DESIGN

3.2.1 Design Process

The block diagram of the system is shown in Fig. 3.2 below.

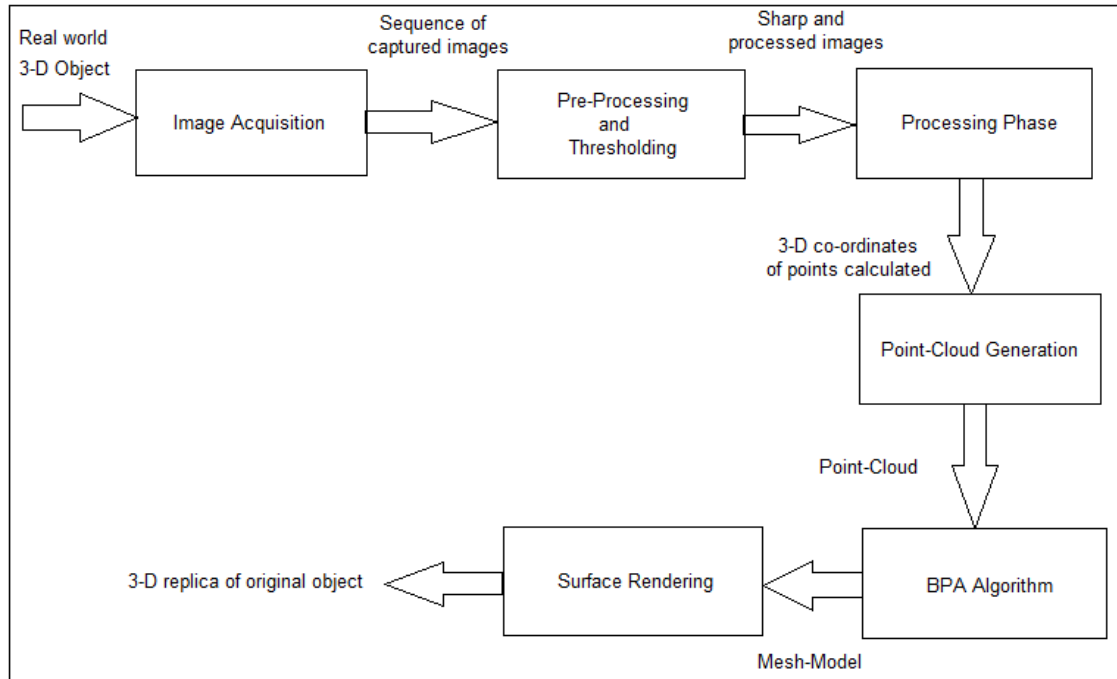


Figure 3.2 Block Diagram of the System

The following issues need to be addressed in the design of the system.

Initial Setup

The initial setup involves aligning the object, laser and the camera correctly in order to avoid errors in image acquisition. This phase forms the base of the image acquisition process since alignment of these elements has a large effect on the quality of image captured and also on the results generated by the software application which takes these captured images as inputs. Also, the environments in which the images are captured play an important role in this phase. Since a red line laser is used in the image acquisition process, the images have to be captured in dark so that the laser line impression on the object is clearly visible and not lost in the background light. Other factors that have to be considered in this phase are that the whole setup (object, camera, laser and the platform) should be stiff and immovable except the turn table on which the object is mounted. This is necessary to help the system produce accurate results.

Image acquisition

This phase is the most important phase of this project. It has to be carried out with great precision and accuracy so that the images generated are suitable for the software application without much pre-processing.

Pre-processing phase

The images captured from the image acquisition phase should ideally have a black background with red impressions indicating the laser points in the image. But some noise is bound to arise due to reflective surface of objects or due to objects with sharp surface changes. This phase includes extracting the RED component of the RGB images where only the red pixels are highlighted and rest everything is black. Since, there will always be some noise and discontinuous intensity patches, they will have to be processed to yield sharp and continuous intensity patterns.

Point Cloud generation

The result of this phase serves as the input to the BPA algorithm. In this phase pixels from the pre-processed images are extracted and their 3D co-ordinates are calculated using the Triangulation process. Points from several images are combined and plotted to give the point-cloud model for the object.

BPA algorithm

This algorithm is the main focus of this project. The input for this phase is the point-cloud of the object. Here the BPA algorithm is applied where the point-cloud is converted into a mesh model.

Surface rendering

The mesh model is then put through a surface rendering process which gives surface to the mesh model and renders it to produce a 3D replica of the original object. Since capturing the colour of the object was never the aim of this project, so the final 3D replica generated will be of mono-colour or any random color scheme.

3.2.2 Architecture

The organization of all the above stages and the communication between them architecturally looks as shown in Fig. 3.3

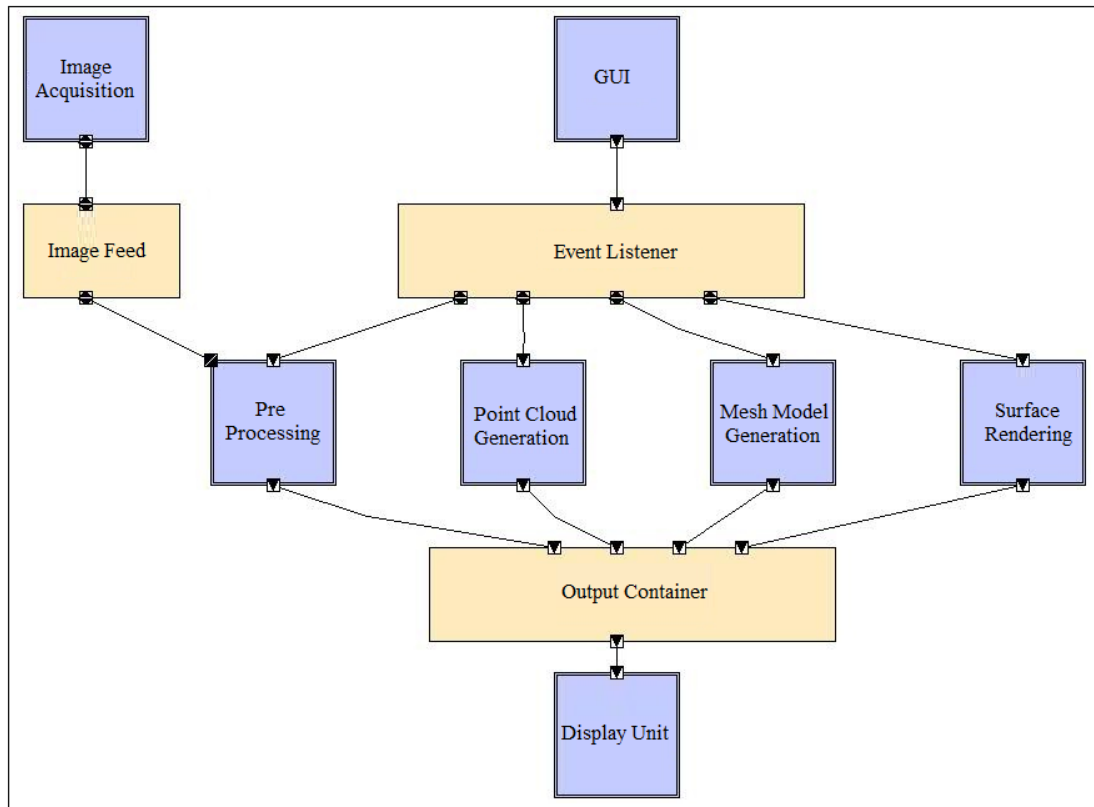


Figure 3.3 Architectural design of the system showing components and connectors

Part Catalog

- The GUI interacts with the user i.e. takes in inputs and displays results.
- The Image Acquisition unit takes 2D images of real world object.
- The Pre-Processing stage performs thresholding and noise removal on those images.
- Point Cloud Generation module uses the filtered 2D images and generates the point cloud for the object.
- Mesh Model Generation module applies Ball Pivoting Algorithm to generate the mesh model for the object.
- Surface rendering component renders the mesh model with triangular surfaces.
- The Display unit displays the output from each stage.

Interaction Description

- Image Feed stores the image from the image acquisition stage. The pre-processing stage takes input from Image Feed.
- Event Listener manages communication between the various components of the system and the user/GUI.
- The output container holds all the results from all the stages and provides them to the Display unit.

Architectural Description

- The architectural style adopted is Event Based.
- The GUI is connected to all the components of the system except the Image acquisition stage.
- The components are interdependent on each other since the output from one stage serves as the input to the other stage.
- The filters are independent of each other.

4 IMPLEMENTATION AND RESULTS

4.1 IMPLEMENTATION DETAILS

1. Initial setup
2. Image acquisition
3. GUI creation
4. Pre-processing and thresholding
5. Calculation of 3D coordinates
6. Point cloud generation
7. BPA algorithm
8. Test cases
9. Surface rendering

The implementation includes the above mentioned stages. Out of these, the initial 6 stages were completed in the last semester and the remaining was completed in the current semester. Below is a Gantt chart indicating the implementation durations for each stage.

ID	Task Name	Start	Finish	Duration	Q3 11	Q4 11				Q1 12			Q2 12
					Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	
1	INITIAL SETUP	22-08-2011	04-11-2011	10w 5d									
2	IMAGE ACQUISITION	05-09-2011	11-09-2011	1w									
3	GUI CREATION	12-09-2011	10-11-2011	8w 4d									
4	PRE-PROCESSING AND THRESHOLDING	15-09-2011	30-09-2011	2w 2d									
5	CALCULATION OF 3D COORDINATES	30-09-2011	19-10-2011	2w 6d									
6	POINT CLOUD GENERATION	21-10-2011	24-10-2011	4d									
7	BALL PIVOTING ALGORITHM	15-11-2011	30-03-2012	19w 4d									
8	SURFACE RENDERING	02-04-2012	03-04-2012	2d									
9	TEST CASES	25-03-2012	12-04-2012	2w 5d									

Figure 4.1 Timeline for the project

4.1.1 Initial Setup

A tentative image of the initial setup is shown in Fig. 4.2.

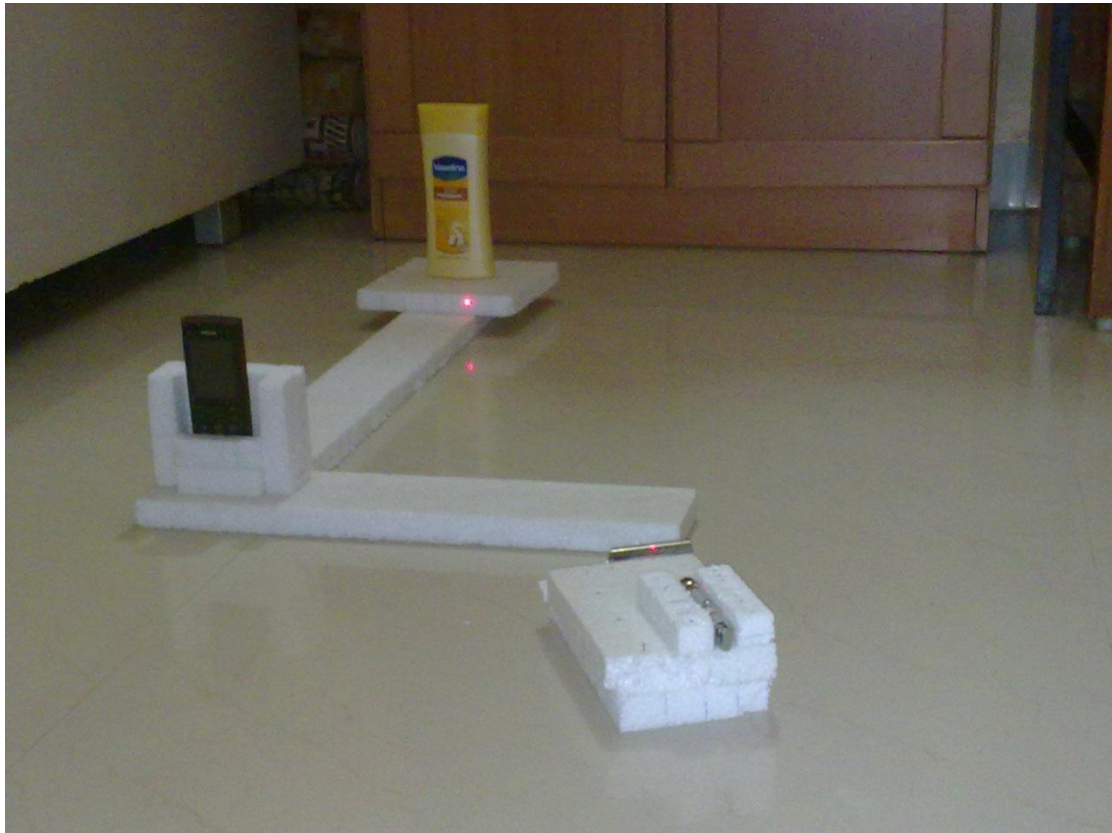


Figure 4.2 Initial Setup

For the system to be accurate, this stage is of utmost importance since any error in this stage will propagate and may magnify in the later stages. The entire setup has to be steady along with the fact that the images need to be taken in proper light so that the laser impressions on the object can be accurately captured by the camera. One of the issues faced in this stage is the unavailability of line lasers. To overcome this, a special technique was used to generate a laser line from a point laser. This was done by reflecting the point laser on a curved surface which produces a laser line impression.

4.1.2 Image Acquisition

After the initial setup and alignment of the camera, the laser and the object mounted on the turntable, the image acquisition procedure begins. The turntable on which the object is mounted is rotated by a fixed angle after every image capture until the entire 360 degree of the object is traversed. The step-size and the time-delay of the rotation

are fixed to a particular angle ' θ ' and time period ' t ' respectively. After every time-period ' t ', the camera captures an image and then the turntable rotates by an angle ' θ '. The image capturing speed of the camera and the rotating speed of the turntable are synchronized accurately. The result of this stage is a sequence of images captured which goes as input to the pre-processing stage.

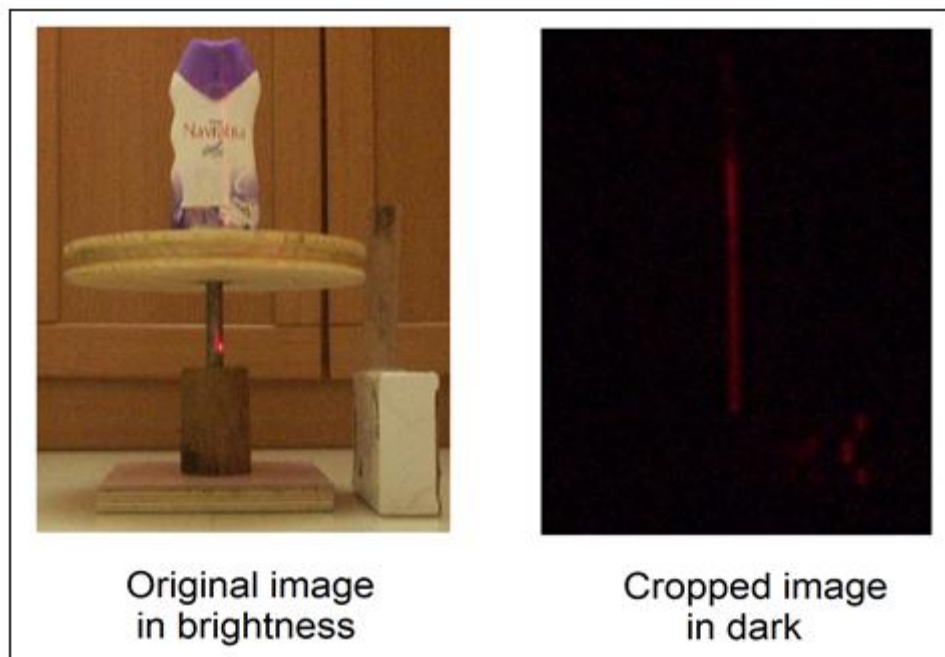


Figure 4.3 Image in light and dark

4.1.3 Pre-Processing stage

This stage takes the sequence of images generated from the image acquisition stage as input. Here initially, the images are cropped to remove the unwanted portions of the images. This step not only reduces the processing time of future stages but also reduces unwanted computations. The next step involves extracting the RED component from the RGB images thus focussing on only the desired pixels. Thresholding is then applied to these images so that only the pixels that have a larger contribution are present and others are eliminated.

This is done in the following manner:

- Once an image is captured, using the details specified by the user, like the start coordinates for the crop action and the height and width of the cropped section, the crop action is performed thus allowing the further stages to

concentrate on the business area of the image which has the laser line impression.

- After the image is cropped, the red component is extracted from the image which transforms a 24 bit RGB image into an 8 bit double image. This is followed by thresholding wherein the entire image is traversed pixel by pixel to filter out and highlight only those pixels that have intensities greater than the threshold intensity specified by the user. The filtered pixels are marked as white and the rest as black.
- Even after this amount of pre-processing, we still have large number of points for the algorithm to work with. To reduce this noise removal has to be done. In this step, we consider only one point per row as valid out of the available points. So the number of points generated by each image would be equal to the number of rows in that image.
- Further there is a provision for refined filtering where we need not consider all the points that an image generates; we have the option of skipping certain amount of fixed points after every point selected. This fixed amount is decided by the user and will vary with the object; more complex is the object, more would be the points required for its reconstruction and thus the fixed value would be less.

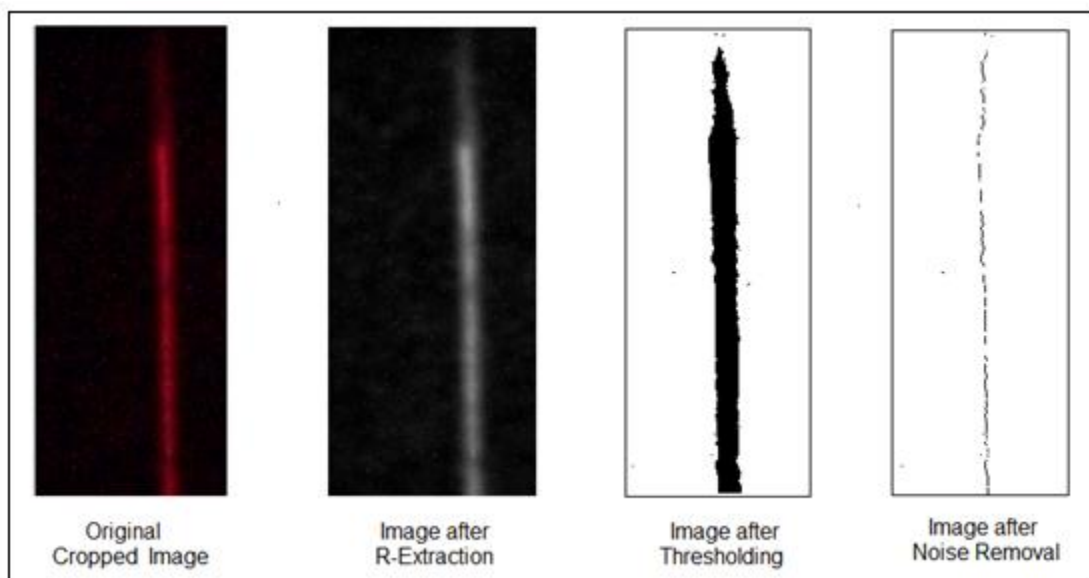


Figure 4.4 Pre-Processing on acquired images

4.1.4 Processing stage

Using these processed images, this step calculates the 3-D co-ordinates of the surface points. The position of the pixels from a fixed axis in the images will give the x coordinates of the pixels. The y position of the pixel will be its real-world z coordinate.

Using the process of triangulation, given one side and angle of a triangle, the side that represents the y coordinate of the object can be found out as shown in figure 4.5

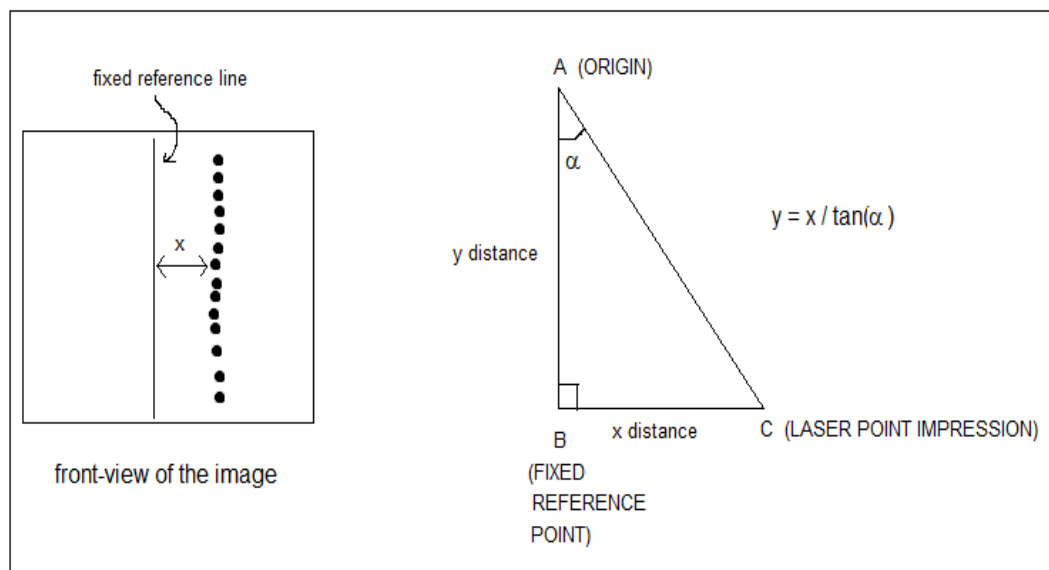


Figure 4.5 Triangulation procedure (calculation of 3D coordinates)

4.1.5 Point Cloud Generation

The 3D coordinates of the points calculated from the previous step acts as inputs to this stage. The points currently are stored differently for different images. But these are not the actual coordinates but are relatively calculated. For calculating their actual positions in 3D space, they have to undergo 3D rotation. Thus, the actual positions of each point is calculated by applying 3D rotation by an angle 'a' corresponding to each image when it was captured (i.e. $a=0$ degree for 1st image, $a=\theta$ for 2nd image, $a=2*\theta$ for 3rd image and so on). After calculating the actual coordinates of the points, all the points are plotted in 3D space to get a point-cloud of the object.

4.1.6 BPA Algorithm

The main concept underlying the Ball-Pivoting Algorithm is quite simple. Let the manifold M be the surface of a three dimensional object and S be a point-sampling of M . Let us assume for now that S is dense enough that a ρ -ball (a ball of radius ρ) cannot pass through the surface without touching sample points. We start by placing a ρ -ball in contact with three sample points. Keeping the ball in contact with two of these initial points, we “pivot” the ball until it touches another point, as illustrated in Fig 4.6. We pivot around each edge of the current mesh boundary. Triplets of points that the ball touches form new triangles. The set of triangles formed while the ball “walks” on the surface constitutes the interpolating mesh.

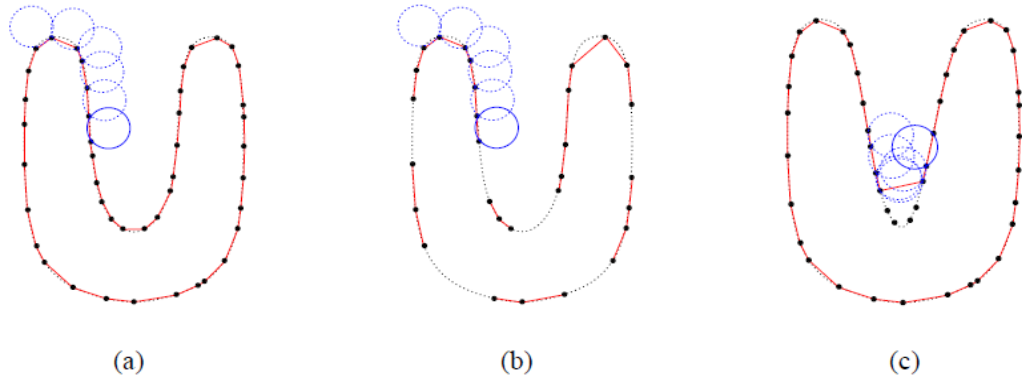


Figure 4.6 Progression of BPA

The BPA follows the advancing-front paradigm to incrementally build an interpolating triangulation. BPA takes as input a list of surface-sample data points σ_i , and a ball radius ρ . The basic algorithm works by finding a seed triangle (i.e., three data points $(\sigma_i, \sigma_j, \sigma_k)$ such that a ball of radius ρ touching them contains no other data point), and adding one triangle at a time by performing the ball pivoting operation until all data points have been covered. A detailed explanation of the algorithm is given below.

Input

List ‘S’ of m surface points ‘ a_i ’ and a ball radius ‘ r ’.

Output:

The mesh model of the object represented by an m -by-3 array.

Algorithm BPA(S,r)

```

1. while (true)
2.     while ( e(i,j)) = get_active_edge(F))
3.         if (ak = ball_pivot( e(i,j) ) &&(not_used(ak ) || on_front( ak)))
4.             output_triangle (ai , aj , ak)
5.             join ( e( i ,j ), ak , F )
6.             if (e( k , i) belongs to F)
7.                 glue(e(i , k), e(k , i), F)
8.             if (e(j , k) belongs to F)
9.                 glue(e(k , j), e(j , k), F)
10.            else
11.                mark as boundary(e(i , j))
12.            if ( ( ai, aj, ak) = find_seed_triangle())
13.                output triangle( ai, aj, ak)
14.                insert edge(e(i , j), F)
15.                insert edge(e( j , k), F)
16.                insert edge(e( k , i), F)
17.            else
18.                return

```

Both `ball_pivot` and `find_seed_triangle` (lines 3 and 12) require efficient lookup of the subset of points contained in a small spatial neighbourhood. This spatial query was implemented using a regular grid of cubic cells, or voxels. Each voxel has sides of size $2r$. Data points are stored in a list, and the list is organized using bucket-sort so that points lying in the same voxel form a contiguous sub list. Given a point p , we can easily find the voxel V it lies in by dividing its coordinates by $2r$. We usually need to look up all points within $2r$ distance from p , which are a subset of all points contained in the 27 voxels adjacent to V (including V itself). The grid allows constant-time access to the points.

The BPA algorithm takes the list S of surface points a_i along with a ball radius r . It uses a collection of linked list F which at any point stores the edges which have been used for pivoting. Initially it consists of the edges of the seed triangle.

Line 12 finds the seed triangle from S . The seed triangle is the collection of 3 edges randomly selected. Then these 3 edges are added to F indicated by line 14, 15 and 16. Line 2 gets the active edge from F which is assigned to $e(i, j)$.

The function `ball_pivot` at line 3 finds a point a_k such that a_i, a_j and a_k form a triangle whose vertices pass through a circle of radius r without enclosing any other point. The point a_k is selected only if it satisfies the above condition and if it is not present in F . This point a_k is then added to F . Then using the glue operation, these three edges of the triangle are joined in F . This process then continues till S is deprived of any points.

One major issue that this system had to deal with is that it might not be possible to interpolate the point cloud and generate a mesh model by simply using a single radius. To overcome this problem, BPA was applied onto the same object using a set of radii. Using this technique, the entire object would be interpolated in multiple passes. In each pass, the algorithm worked on those sections of the point cloud which were not interpolated in the previous passes. Thus pass by pass the entire mesh model was generated. The mesh model is stored in an m -by-3 array where each row of the array corresponded to the three vertices of a triangle. The mesh model is thus a large set of triangular surfaces. For more accurate reconstruction of objects or for complex objects, greater number of images will be captured leading to larger sample space of points S and thus a larger set of triangles representing the mesh model.

The BPA involves certain methods/functions that play an important role in implementing the algorithm. Some of them are explained below.

Ball_pivot()

Given an active edge or 2 points, this algorithm is responsible for finding a 3rd point which satisfies the all-important ball pivoting condition that the 3 points should lie on

the circumference of a sphere of radius r such that no other point lies inside the sphere. This condition is implemented by taking help from complex 3D geometry.

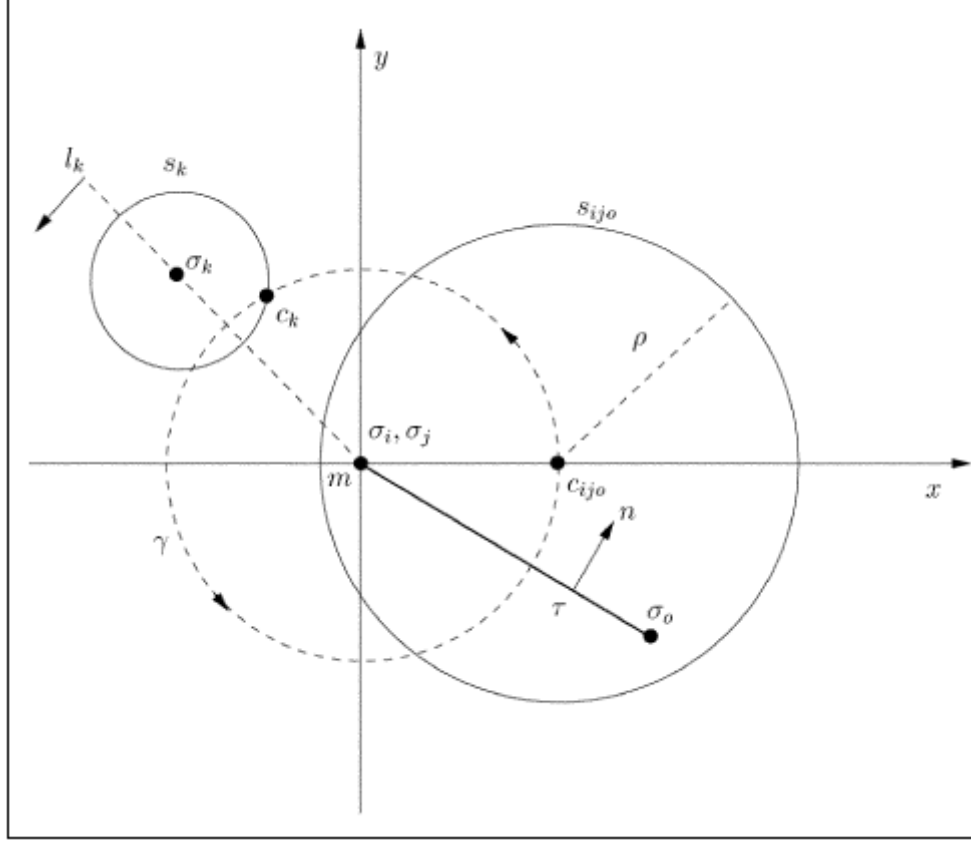


Figure 4.7 3D geometry top view depicting ball pivoting operation

Assuming the active edge is $\sigma_i\sigma_j$ such that $\sigma_i\sigma_j$ is perpendicular to the page and m i.e. the midpoint of $\sigma_i\sigma_j$ is the origin. Also $\sigma_i\sigma_j$ and σ_o are 3 vertices of an already found triangle such that sphere of radius ρ passes through all these points. Now, the active edge is $\sigma_i\sigma_j$ and the algorithm is in search of a new point σ_k that satisfies the ball pivoting condition. The top view of the sphere passing through $\sigma_i\sigma_j$ and σ_o is a circle S_{ij0} with center C_{ij0} having radius ρ . This circle lies on the x - y plane such that C_{ij0} is on the x -axis. Now, the first step towards finding σ_k is to draw a sphere with m as the center and $\|m-C_{ij0}\|$ as the radius. Also, draw a sphere of radius ρ at the point under consideration σ_k . If the sphere at m with radius $\|m-C_{ij0}\|$ and the ρ sphere at σ_k intersect, then the triangle $\sigma_i\sigma_j\sigma_k$ satisfy the ball pivoting condition and the ball pivot operation will return σ_k as the new point. The ball pivot operation is applied for every point in the same as well as neighbouring voxels until a new point is found. If no point is found then the active edge is marked as a boundary edge.

Find_seed_triangle ()

This method is responsible for finding 3 points in 3D space such that they satisfy the ball pivoting condition which states that there has to be a sphere of radius ρ passing through these 3 points without including any other point. It considers every point in the present voxel and finds another point within 2ρ distance i.e. within same as well as neighbouring voxels. After finding the second point, it applies the ball pivoting operation to find the third point. This process continues for all voxels until a valid triangle (3 vertices/points) is found. Once the 3 points are discovered, the 3 edges of the triangle so formed are added to the Front queue. If the operation does not discover any valid triangle, then the algorithm proceeds to the next pass with a new radius or halts if the entire radius set has been tried.

Join () and Glue ()

The join operation adds new edges to the Front queue and the glue operation is responsible for removing duplicate edges.

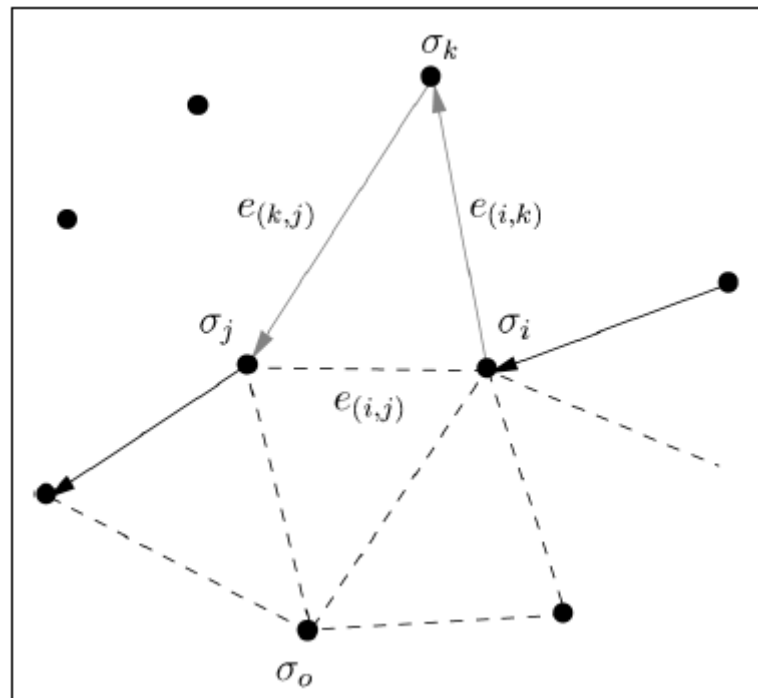


Figure 4.8 Join operation

A join operation simply adds a new triangle, removing edge $e(i, j)$ from the front and adding the two new edges $e(i, k)$ and $e(k, j)$.

4.1.7 Surface Rendering

This stage uses the mesh model generated using the BPA algorithm as an input. It uses appropriate surface rendering algorithm to render the mesh model and generate the 3D replica of the original object. An example of surface rendering on a mesh model is shown in Fig. 4.9.

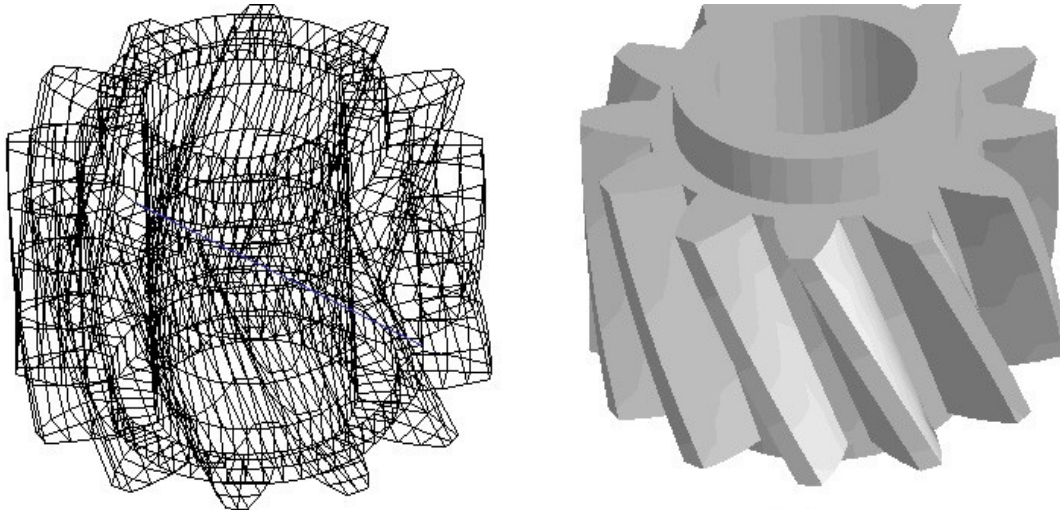


Figure 4.9 Rendering applied on a mesh model

4.1.8 GUI Design

The GUI is developed using Matlab R2010a. The figures 4.10 and 4.11 indicate the two GUI windows explained.

The first window named 'Scan3D' is the primary window. It has the different command buttons for different intermediate operations which enable to apply different operations on the images. All the operations behind the buttons are interdependent. Results of each stage/operation are displayed using figure component of matlab.

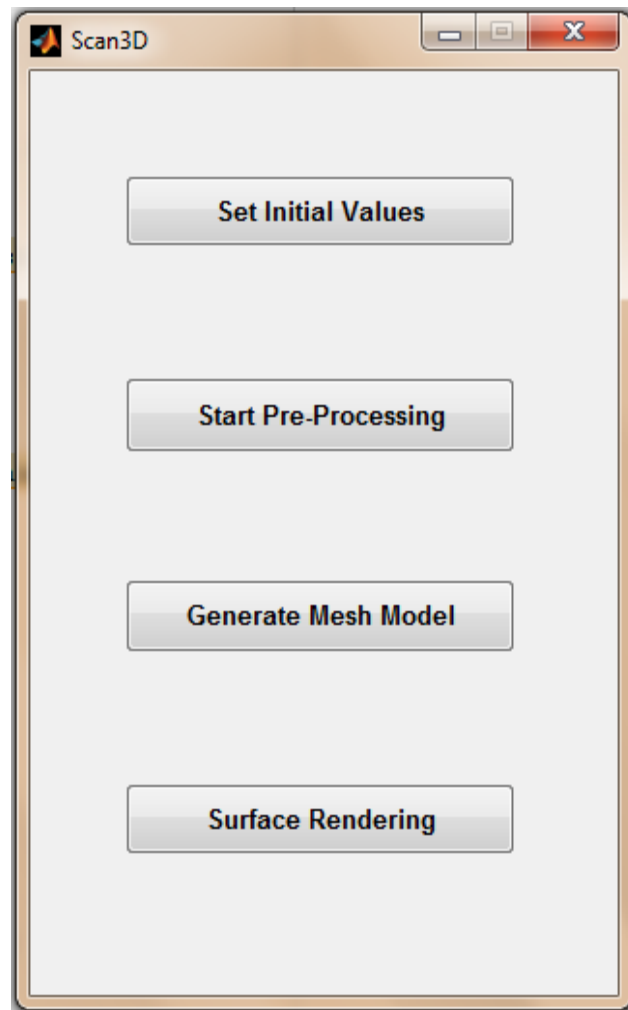
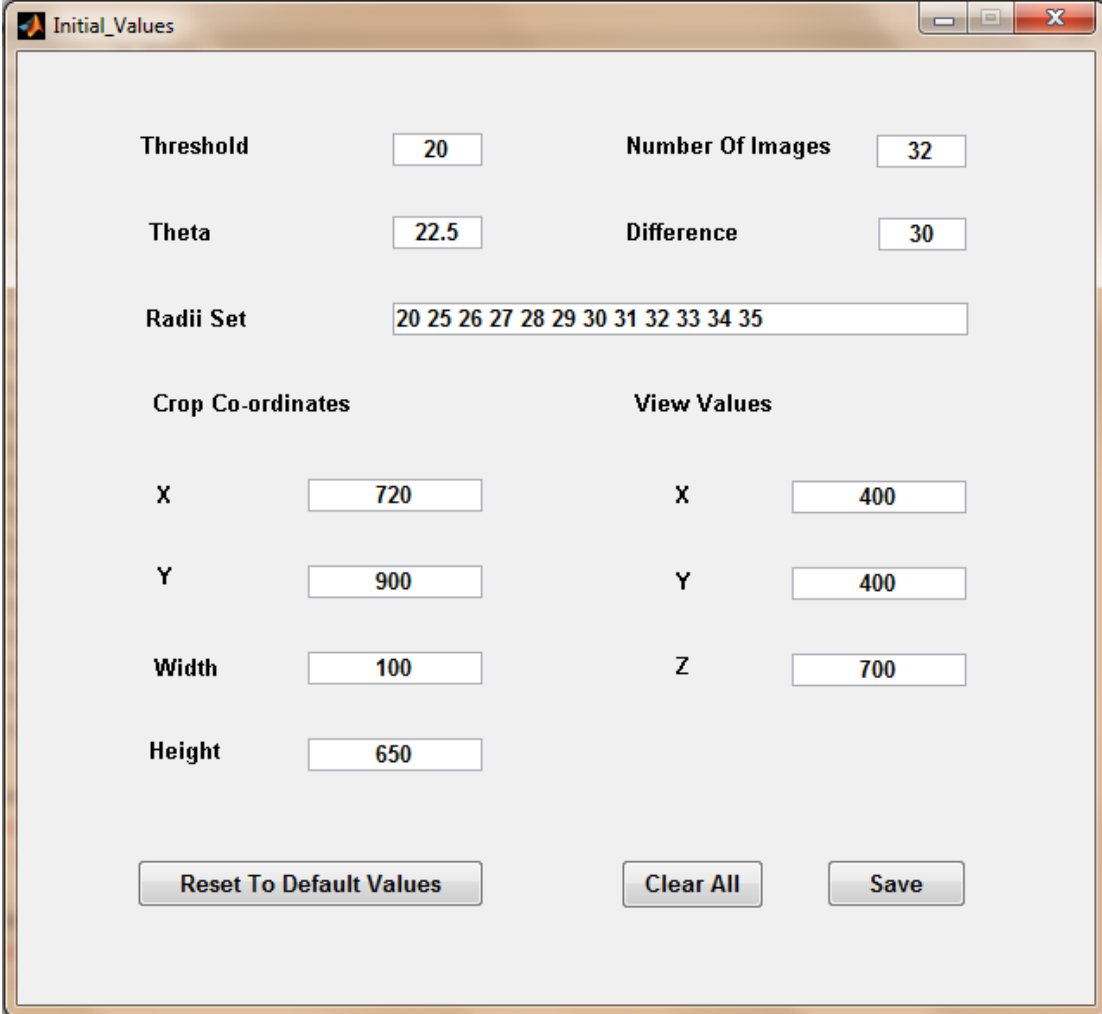


Figure 4.10 Primary Window

An important part of this GUI is the Set Initial Values section since this section may vary for every object. This is the secondary window which gets displayed on a click event on 'Set Initial Values' button. It has textboxes for setting the initial values for various variables. These initial values include the cropping coordinates, the intensity threshold for pre-processing, the radii set, the view coordinates, etc. It has buttons for clearing the textboxes and to reset them to default values.



The image shows a software window titled "Initial_Values". It contains several input fields and buttons. The fields are organized into two columns. The left column has "Threshold" (20), "Theta" (22.5), "Radii Set" (a list of numbers from 20 to 35), "Crop Co-ordinates" (a sub-header for "X", "Y", "Width", and "Height"), and their respective values (720, 900, 100, 650). The right column has "Number Of Images" (32), "Difference" (30), "View Values" (a sub-header for "X", "Y", and "Z"), and their respective values (400, 400, 700). At the bottom, there are three buttons: "Reset To Default Values", "Clear All", and "Save".

Parameter	Value	Parameter	Value
Threshold	20	Number Of Images	32
Theta	22.5	Difference	30
Radii Set	20 25 26 27 28 29 30 31 32 33 34 35		
Crop Co-ordinates		View Values	
X	720	X	400
Y	900	Y	400
Width	100	Z	700
Height	650		

Buttons: Reset To Default Values, Clear All, Save

Figure 4.11 Initial values Window

4.2 RESULTS

The intermediate results of the system are displayed in the following manner.

Below is a depiction of how the scanned images look after the red component is extracted from them.

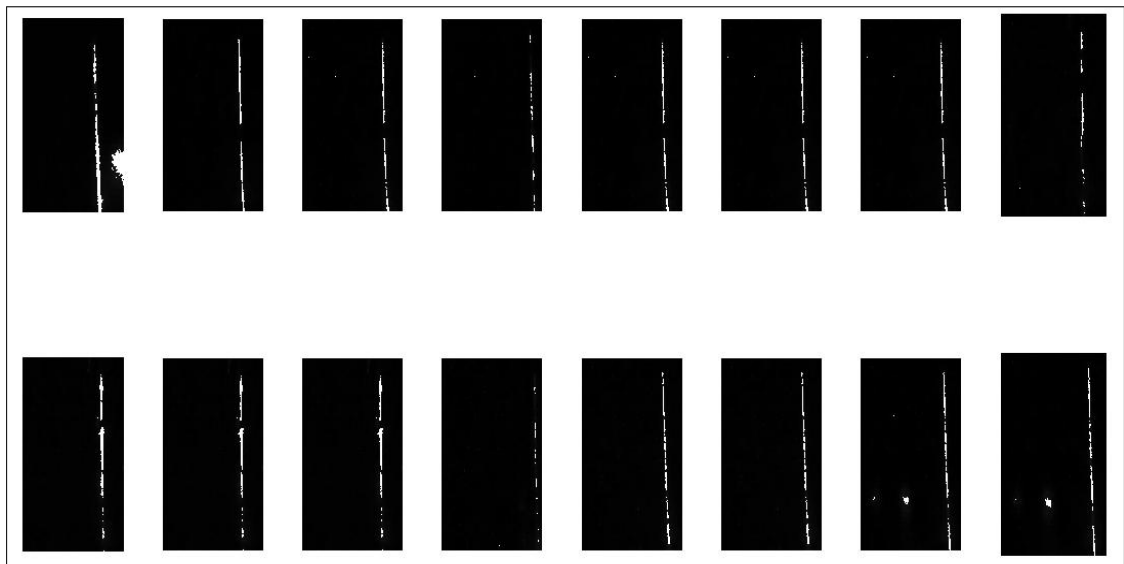


Figure 4.12 After 'R' extraction

Thresholding is the step where the pixels that contribute the most towards the image are picked and the others are discarded. The results after this step are as shown below.

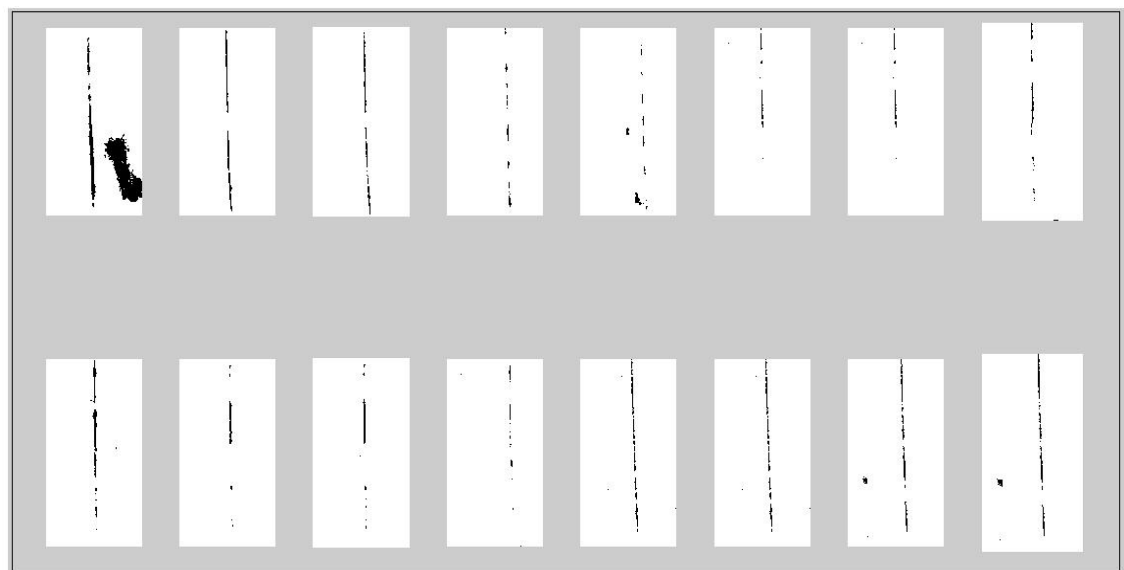


Figure 4.13 After thresholding

Noise removal is undertaken essentially to increase accuracy by reducing errors. The results from this step are shown below.



Figure 4.14 After noise removal

Using these images, the 3D coordinates of the points are calculated which when plotted in 3D space generates the point cloud for the object.

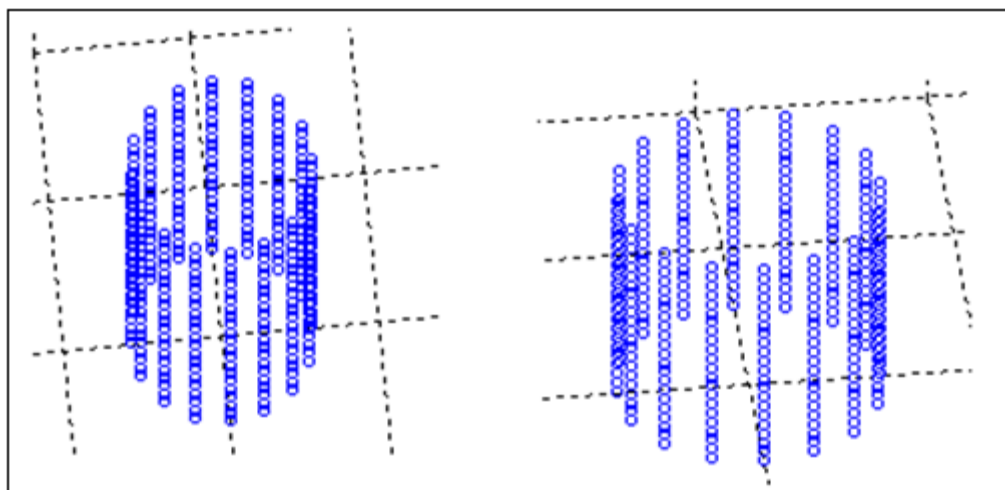


Figure 4.15 Point cloud of a cylindrical object

The point cloud serves as the input to the Ball Pivoting algorithm which generates the mesh model as shown.

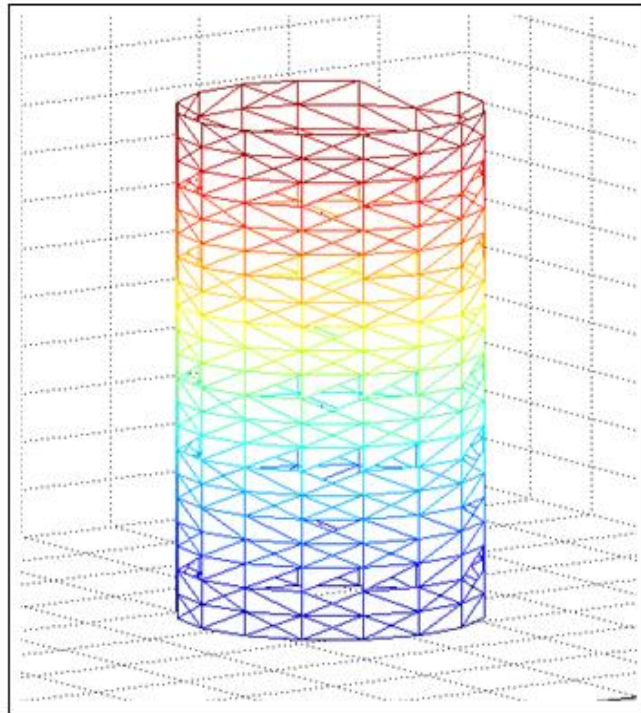


Figure 4.16 Mesh model of a cylindrical object

The mesh model is rendered to give the final replica of the object as shown.

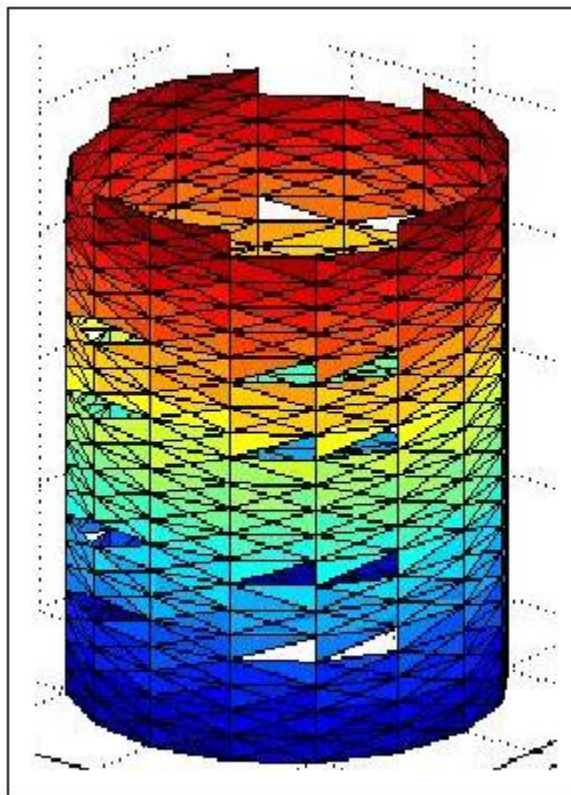


Figure 4.17 Mesh model after surface rendering

5 TESTING

5.1 TEST CASES

Test case 1:

The object tested here is a Vaseline bottle. The system will try to replicate the structure of the bottle which is wide at the bottom, tapers at the middle followed by wide again at the top.



Figure 5.1 Vaseline bottle

32 images were taken during the image acquisition process which means the step angle was $360/32=11.25$ degree. The original images taken were cropped and displayed by the algorithm as shown in Fig. 5.2.

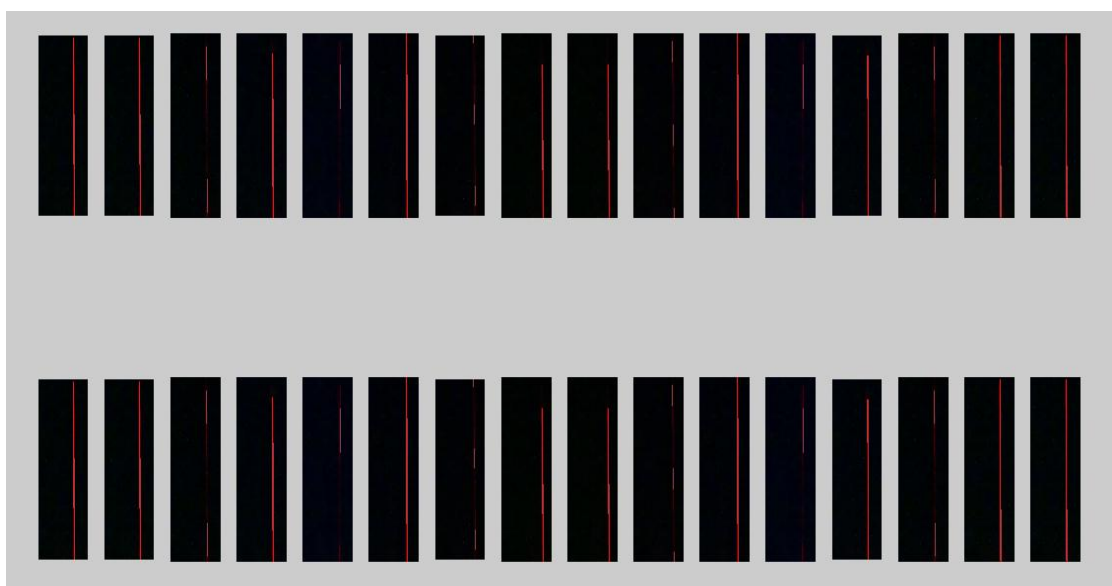


Figure 5.2 Original cropped images of the Vaseline bottle

Pre-processing is applied on these images. The pixels below a certain user-specified intensity are filtered out and the others are highlighted to give the images below.

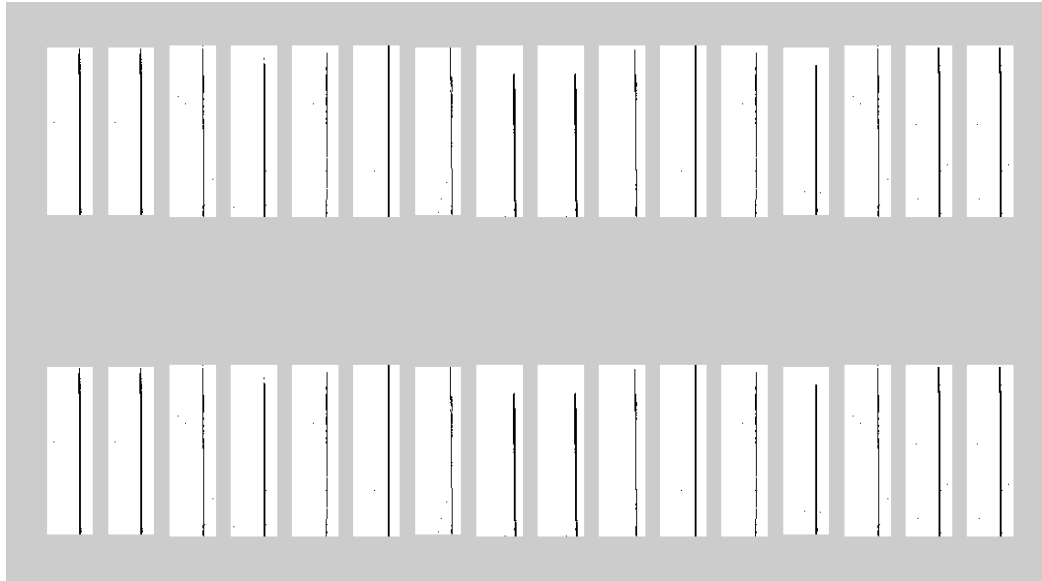


Figure 5.3 Images after the pre-processing operation

Using these filtered images, the 3D coordinates for all the points are generated which are then plotted in 3D space to give the point cloud for the Vaseline bottle. It is evident from the point cloud image that the initial algorithms were successful in capturing the geometry of the bottle. Since 22 points were captured from each image and the total number of images were 32, so $32 \times 22 = 704$ points were used to represent the point cloud for this object.

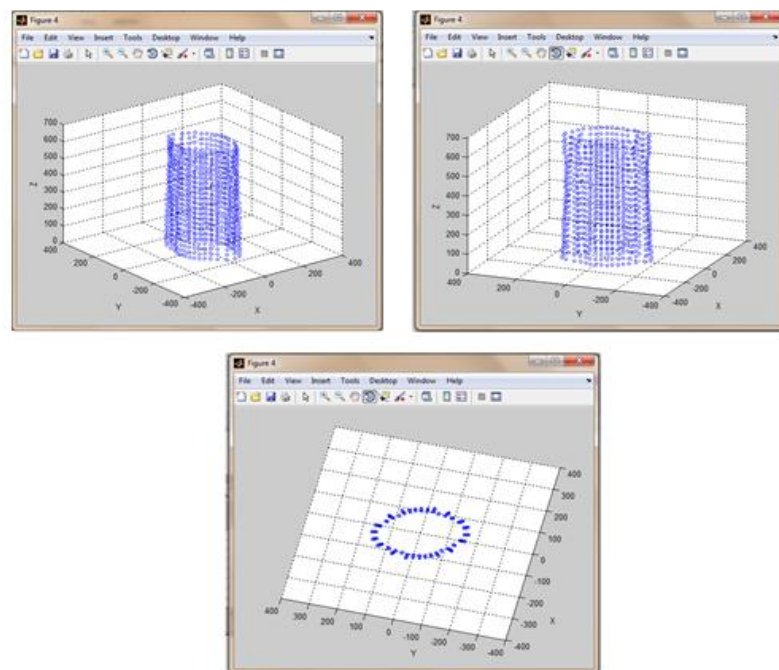


Figure 5.4 Point cloud of the Vaseline bottle from different angles

The mesh model was generated pass by pass using a radius set $rp=[20\ 25\ 26\ 27\ 28\ 29\ 30\ 31]$. Thus the entire mesh model was generated in 8 passes and the individual pass results are shown below.

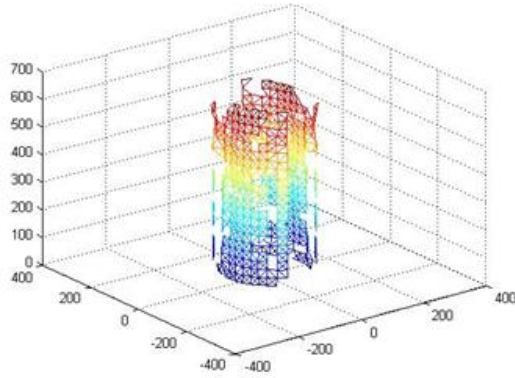


Figure 5.5 Pass 1, $r=20$

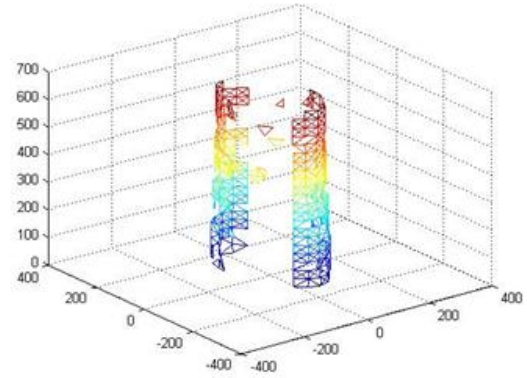


Figure 5.6 Pass 2, $r=25$

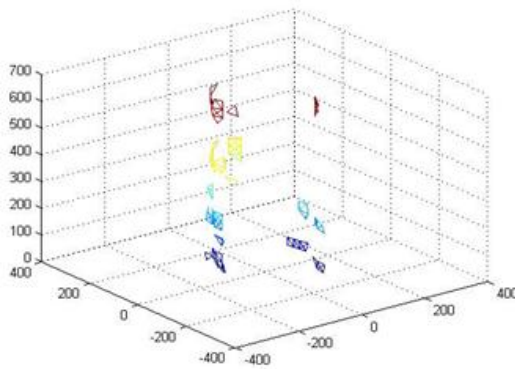


Figure 5.7 Pass 3, $r=26$

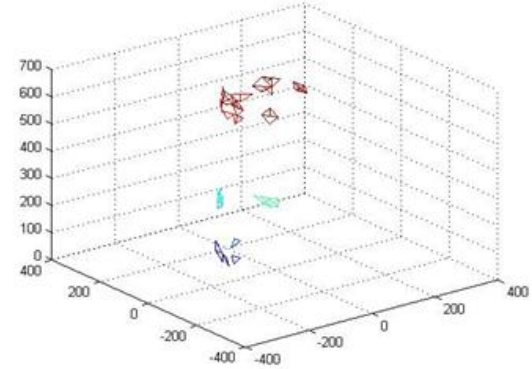


Figure 5.8 Pass 4, $r=27$

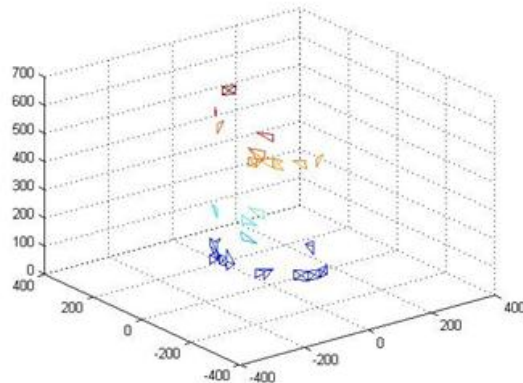


Figure 5.9 Pass 5, $r=28$

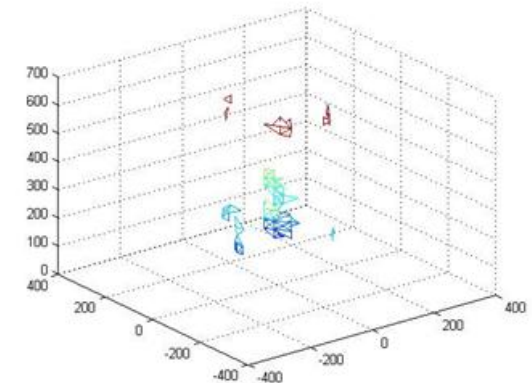


Figure 5.10 Pass 6, $r=29$

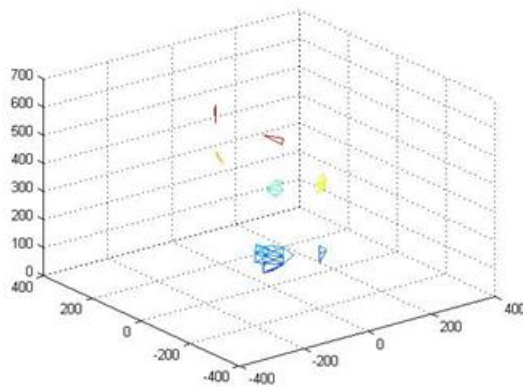


Figure 5.11 Pass 7, r=30

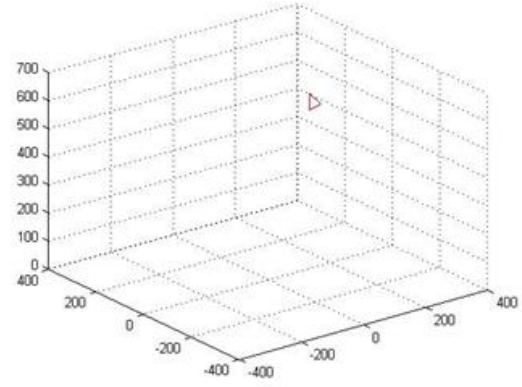


Figure 5.12 Pass 8, r=31

The final mesh formed from the union of the results from these passes is shown in Fig. 5.13. The entire mesh model is generated using about 2000 triangular surfaces. This object was reconstructed in about 3 hours.

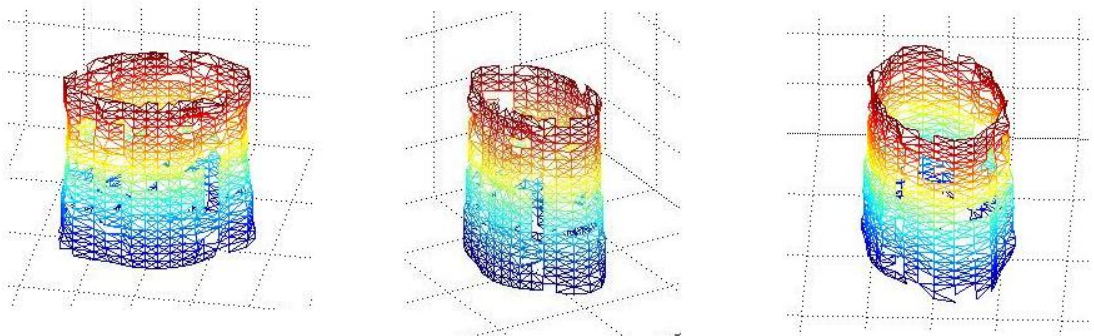


Figure 5.13 Final mesh models for the Vaseline bottle

This mesh model is then rendered to give the final replica of the Vaseline bottle as shown in Fig. 5.14. The accuracy of this model is about 85 %. After including the out of core extensions it will rise up to 92 %. 3D replicas will never be 100 % accurate but the accuracy can be increased up to 99 % by improving the image acquisition process and by capturing more images.

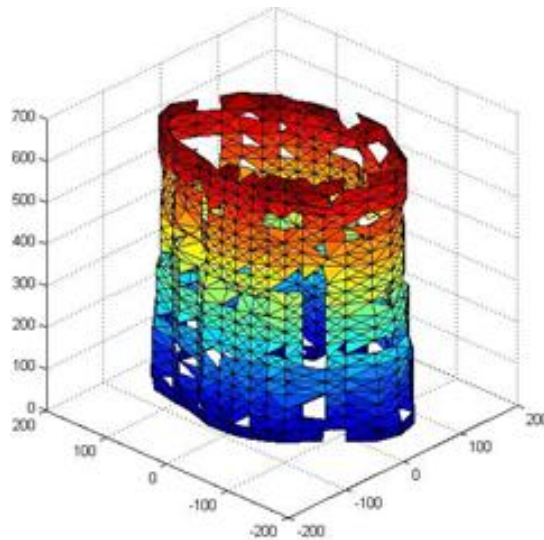


Figure 5.14 Final 3D replica of the Vaseline bottle

Test case 2:

The object tested here is a Dermicool bottle.



Figure 5.15 Dermicool bottle

32 images were taken during the image acquisition process which means the step angle was $360/32=11.25$ degree. The original images taken were cropped and displayed by the algorithm as shown in Fig. 5.16.

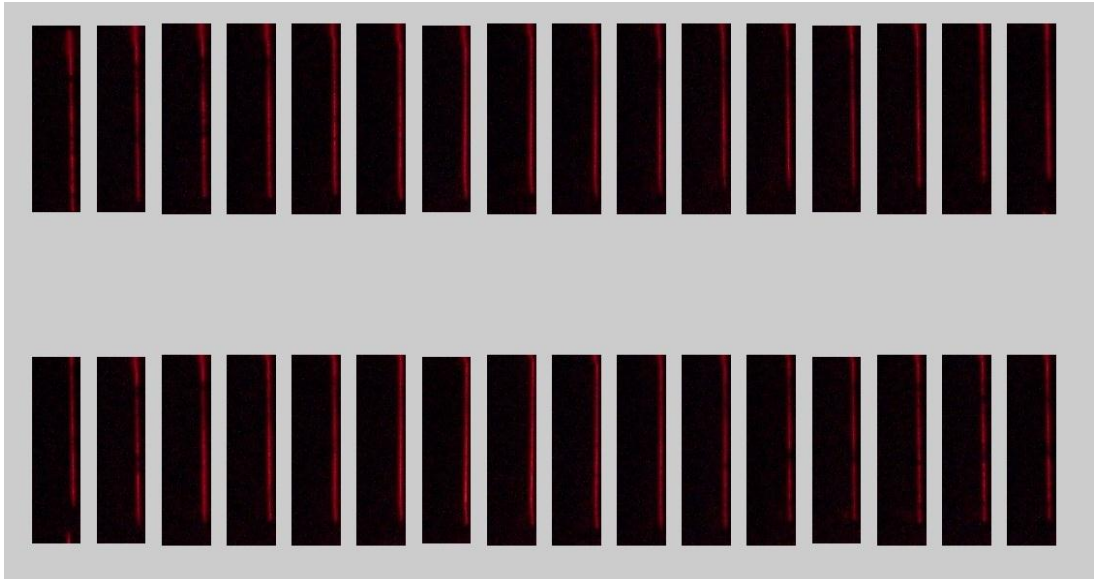


Figure 5.16 Original cropped images of the Dermicool bottle

Pre-processing is applied on these images. The pixels below a certain user-specified intensity are filtered out and the others are highlighted to give the images below.

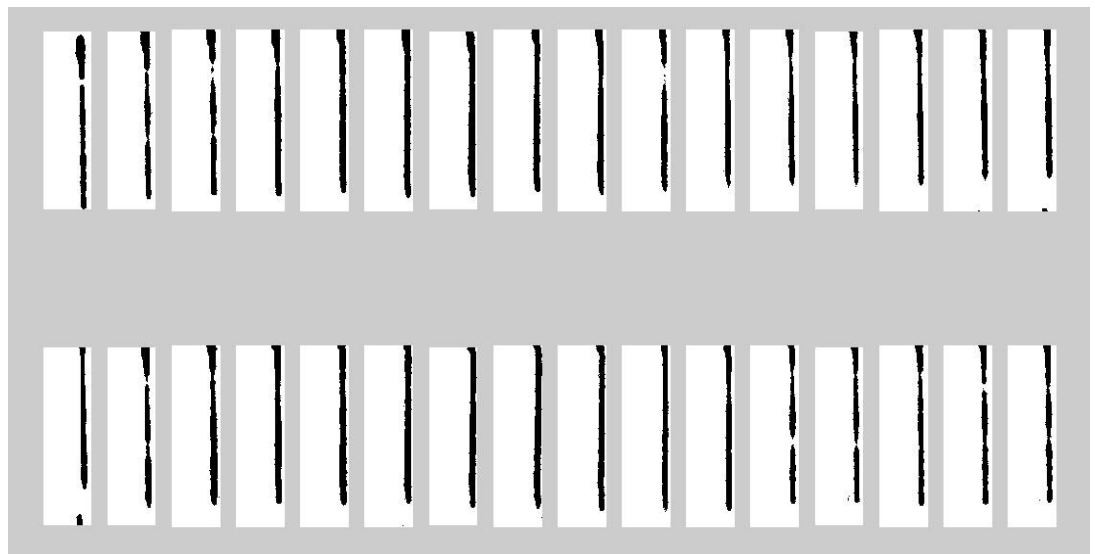


Figure 5.17 Images after the Thresholding operation

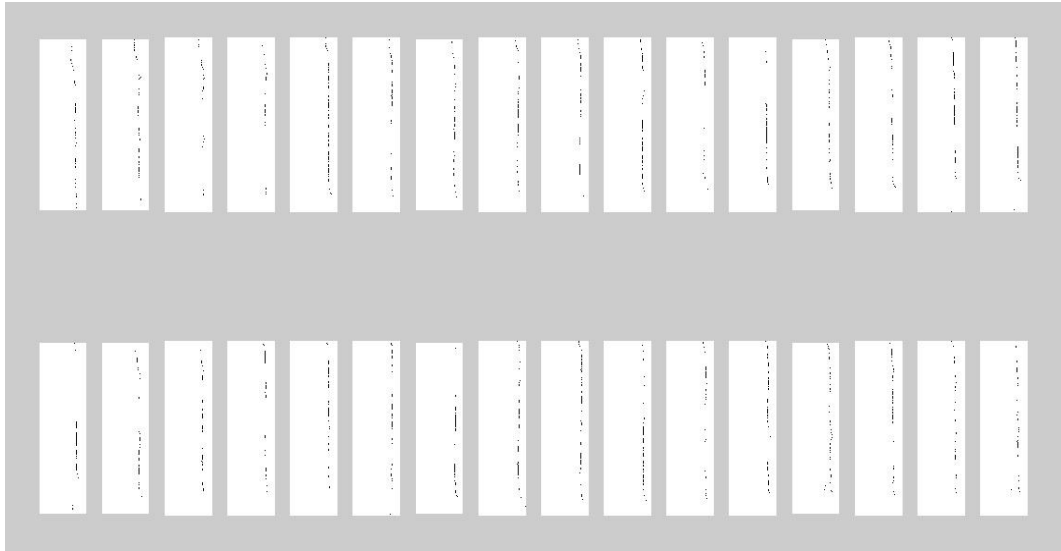


Figure 5.18 Images after the noise removal operation

Using these filtered images, the 3D coordinates for all the points are generated which are then plotted in 3D space to give the point cloud for the Dermicool bottle. It is evident from the point cloud image that the initial algorithms were successful in capturing the geometry of the bottle. Since 23 points were captured from each image and the total number of images were 32, so $32 \times 23 = 736$ points were used to represent the point cloud for this object.

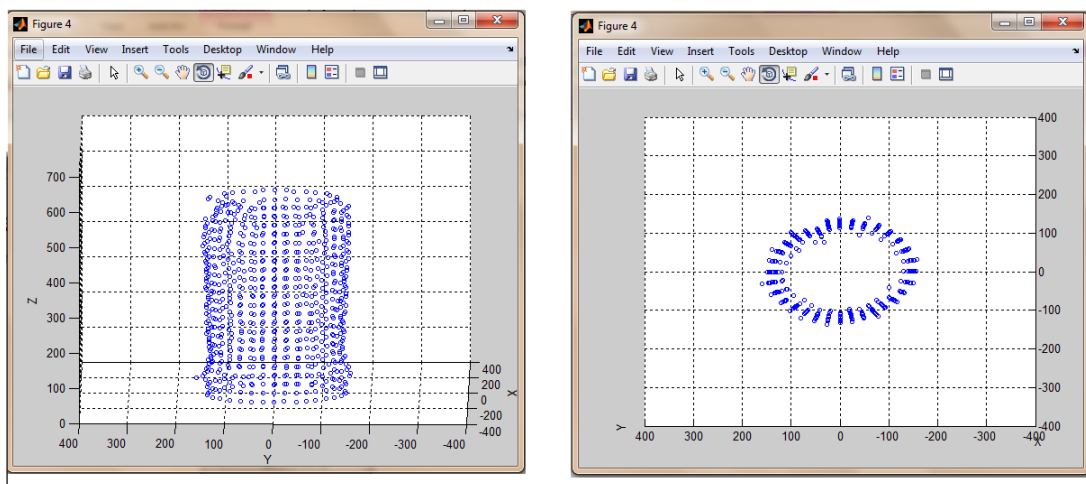
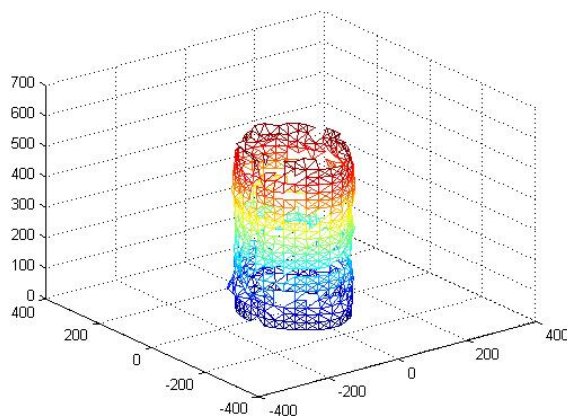
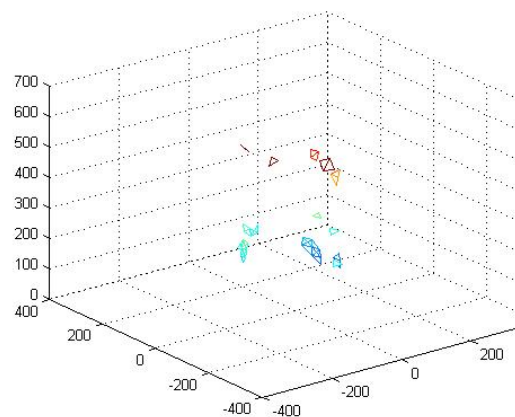
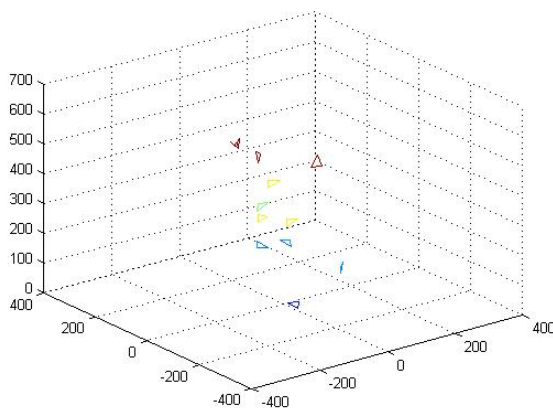
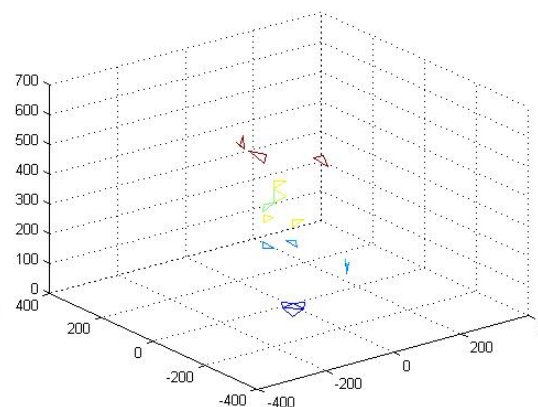
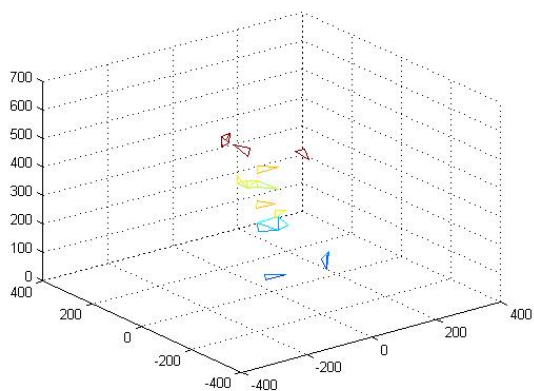
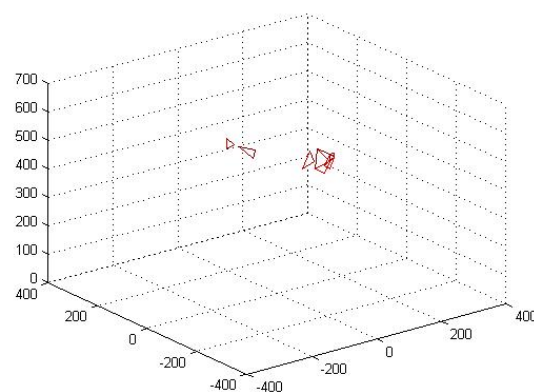


Figure 5.19 Point cloud of the Dermicool bottle from different angles

The mesh model was generated pass by pass using a radius set $rp = [20 \ 21 \ 22 \ 26 \ 29 \ 31]$. Thus the entire mesh model was generated in 6 passes and the individual pass results are shown below.

Figure 5.20 Pass 1, $r=20$ Figure 5.21 Pass 2, $r=21$ Figure 5.22 Pass 3, $r=22$ Figure 5.23 Pass 4, $r=26$ Figure 5.24 Pass 5, $r=29$ Figure 5.25 Pass 6, $r=31$

The final mesh formed from the union of the results from these passes is shown in Fig. 5.26. The entire mesh model is generated using 1508 triangular surfaces. This object was reconstructed in about 2 and 1/2 hours.

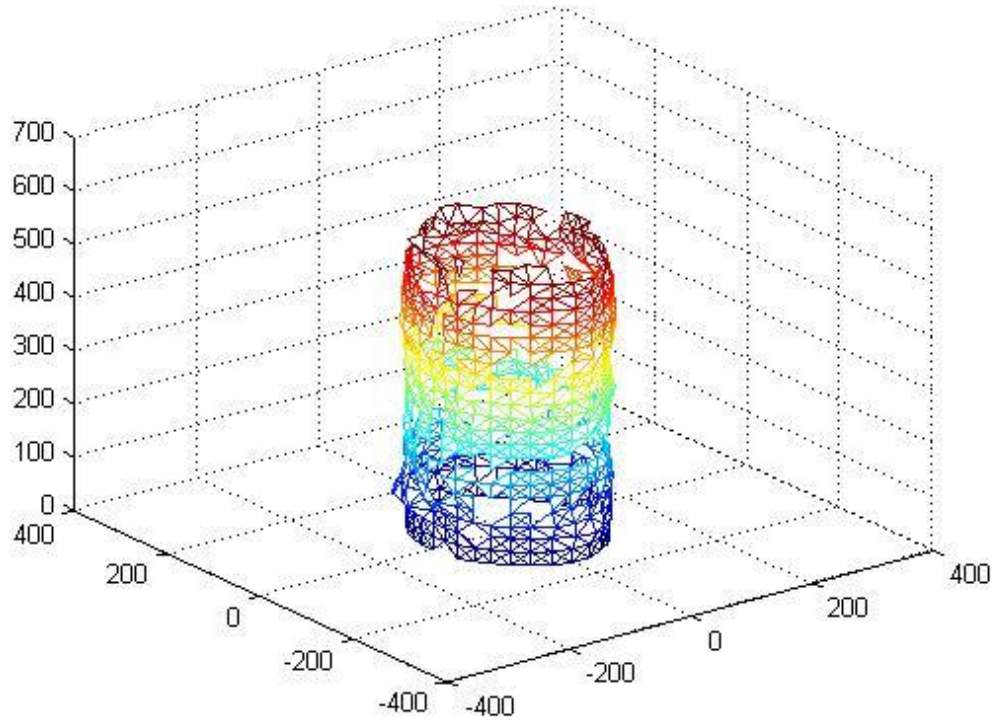


Figure 5.26 Final mesh model for the Dermicool bottle

This mesh model is then rendered to give the final replica of the Dermicool bottle as shown in Fig. 5.27.

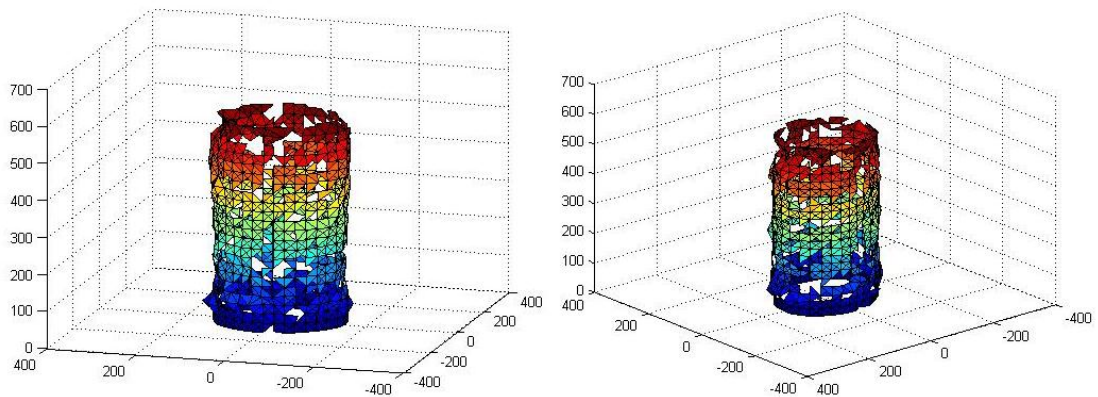


Figure 5.27 Final 3D replica of the Dermicool bottle

6 CONCLUSIONS AND FURTHER WORK

6.1 CONCLUSION

The 3D scanner was implemented using Ball Pivoting Algorithm to capture the geometry of the real world object which was depicted as a computer-aided 3D model. Furthermore, the algorithm was implemented successfully to run in linear time while maintaining the advantages of previous interpolating techniques.

The project had genuine limitations as far as the image acquisition process is concerned but since the objective of the project was the Ball Pivoting Algorithm, the results generated by the BPA for an ideal object (assuming accurate acquisition results) were completely accurate. The algorithm is a general one, so the final accuracy of the system will depend heavily on how accurate the image acquisition process is.

6.2 FURTHER WORK

The 3D scanning technique could be modified to capture not only the geometry but also the visual attributes of the object like the color, luminance, etc. Also, to simplify the image acquisition process and to increase its accuracy, the entire acquisition system can be automatized by using a stepper motor to rotate the turn table. In addition to that, instead of capturing images, the video of 360 degree rotation of the object can be captured. This can then be broken down into frames to be used as images. As far as the algorithm is concerned, the BPA can be enhanced to even consider the orientation of the surface normals to construct the mesh model. Also research has to be done to find out whether the time taken by this algorithm in case of smaller objects will change for larger objects given that both objects contain similar number of points in the point cloud.

REFERENCES

- [1] The Ball-Pivoting Algorithm for Surface Reconstruction-Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Claudio Silva, Member, IEEE, and Gabriel Taubin, Senior Member, IEEE OCTOBER-DECEMBER 1999.
- [2] Roger Muellerr, "3d laser scanner", "<http://www.muellerr.ch/engineering/laserscanner/default.htm>", 2011.
- [3] H. Edelsbrunner and E.P. Mücke, "Three-Dimensional Alpha Shapes," ACM Trans. Graph., vol. 13, no. 1, pp. 43-72, Jan. 1994.
- [4] Glenn Murray, "Rotation About an Arbitrary Axis in 3Dimensions", "<http://inside.mines.edu/~gmurray/ArbitraryAxisRotation/>", 11 July 2011.
- [5] TjioHokHoo & Mohd Rizal Arshad, "A Simple Surface Mapping Technique using Laser Triangulation Method", School of Electrical and Electronic Engineering ,Engineering Campus, Universiti Sains Malaysia (USM), 14300 Nibong Tebal, Penang.
- [6] 3D laser Scanning: Reverse Engineering, Surface Modelling, Photogrammetry-3D applications, "<http://www.applications3d.com/>", 2008.
- [7] 3D Digitizing, "Medical_Brochure_03.pdf".
- [8] 3D Scanner, http://en.wikipedia.org/wiki/3D_scanner.