3D RECONSTRUCTION OF POINT CLOUDS USING MULTI-VIEW ORTHOGRAPHIC PROJECTIONS

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND
ELECTRONICS ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCES
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Osman Topçu
June 2006

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Prof. Dr. Levent Onural (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Prof. Dr. Ergin Atalar

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Assistant Prof. Dr. Selim Aksoy

Approved for the Institute of Engineering and Sciences:

_____

Prof. Dr. Mehmet Baray

Director of Institute of Engineering and Sciences

ABSTRACT

3D RECONSTRUCTION OF POINT CLOUDS USING MULTI-VIEW ORTHOGRAPHIC PROJECTIONS

Osman Topçu
M.S. in Electrical and Electronics Engineering
Supervisor: Prof. Dr. Levent Onural
June 2006

A method to reconstruct 3D point clouds using multi-view orthographic projections is examined. Point clouds are generated by means of a stochastic process. This stochastic process is designed to generate point clouds that mimic microcalcification formation in breast tissue. Point clouds are generated using a Gibbs sampler algorithm. Orthographic projections of point clouds from any desired orientation are generated. Volumetric intersection method is employed to perform the reconstruction from these orthographic projections. The reconstruction may yield erroneous reconstructed points. The types of these erroneous points are analyzed along with their causes and a performance measure based on linear combination is devised. Experiments have been designed to investigate the effect of the number of projections and the number of points to the performance of reconstruction. Increasing the number of projections and decreasing the number of points resulted in better reconstructions that are more similar to the original point clouds. However, it is observed that reconstructions do not improve considerably upon increasing the number of projections after some number. This method of reconstruction serves well to find locations of original points.

Keywords: 3D reconstruction, visual hull, shape from silhouettes, volumetric intersection, point clouds, Gibbs sampler, orthographic projection

ÖZET

NOKTA BULUTLARININ ÇOK YÖNLÜ ORTOGRAFİK İZDÜŞÜMLERİNDEN GERİ ÇATMALARININ ELDE EDİLMESİ

Osman Topçu
Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Prof. Dr. Levent Onural
Haziran 2006

Üç-boyutlu nokta bulutlarının ortografik izdüşümlerinden geri çatmalarının elde edilmesine yönelik bir yöntem sınandı. Nokta bulutlarını oluşturmak için rasgele süreçler kullanıldı. Nokta bulutlarının meme dokusunda bulunan mikrokalsifikasyon oluşumunu modellemesi için rasgele süreçler tasarlandı. Nokta bulutları bir Gibbs örnekleme algoritması kullanılarak oluşturuldu. Nokta bulutlarının istenilen yönlerden ortografik izdüşümleri elde edildi. Elde edilen ortografik izdüşümlerden nokta bulutlarının geri çatmalarını elde etmek için hacimsel kesişim yöntemi uygulandı. Ortaya çıkan nokta bulutlarının geri çatmalarında hatalı noktaların oluşabildiği gözlendi. Bu hatalı noktalar, oluşma sebeplerine göre sınıflara ayrılarak, doğrusal birleşime dayanan bir performans değerlendirme ölçütü belirlendi. İzdüşüm ve toplam nokta sayısının performansa etkisinin araştırılması için deneyler tasarlandı. İzdüşüm sayısının arttırılması ve toplam nokta sayısını azaltılması gerçek nokta bulutlarına daha çok benzeyen geri çatmalar ortaya çıkmasını sağladı. Fakat izdüşüm sayısının arttırılmasının bir noktadan sonra performansa fazla bir etkisinin olmadığı anlaşıldı. Bu geri çatma yöntemi nokta bulutlarını oluşturan noktaların yerlerinin belirlenmesi için kullanılabilir.

Anahtar Kelimeler: Geri çatma, görsel zarf, silüetlerden şekillendirme, hacimsel kesişim, nokta bulutları, Gibbs örnekleyicisi, ortografik izdüşüm

ACKNOWLEDGEMENT

I would like to thank to my family for their constant support.

This work is partially supported by EC within FP6 under Grant 511568 with the acronym 3DTV.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

1.1 Statement of the Problem

Perspective projection maps points **A** and **B** to points **a** and **b** on the plane, respectively, as can be seen in Figure 1 where **O** represents the focal point.

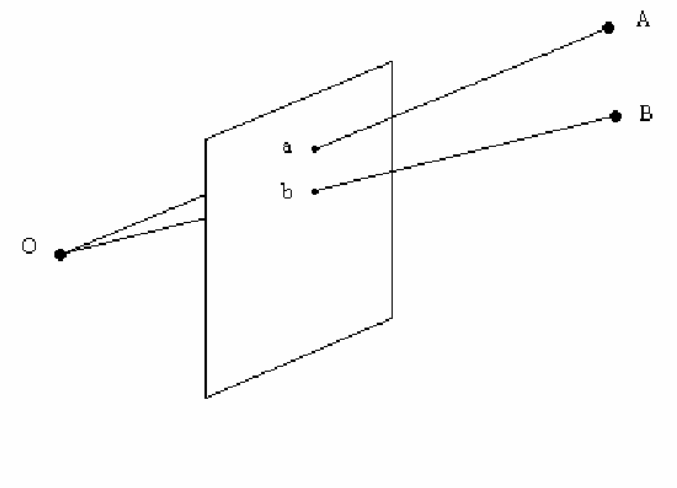

Figure 1 Perspective projection

Orthographic projection is a special case of perspective projection. When the focal point **O** in Figure 1 goes to infinity along the optical axis, those rays in Figure 1 become perpendicular to the image plane, thus, the projection becomes orthographic.

Orthographic projection has been used by architects and engineers for illustration purposes. More information on orthographic projection can be found in Chapter 3.

Finding shape from orthographic projections is usually studied to improve software for engineering drawings. Senda [18] reconstructs solids from three orthographic views. These views are top, side and front views. Senda uses ridgelines to construct surfaces and surfaces to construct the solid. Wang [19] studied the same problem and proposed a new method to perform reconstruction in a short time. Wang uses 2D vertices to get 3D candidate vertices, 3D candidate vertices to get 3D candidate edges. From 3D candidate edges, he gets candidate faces, and from this, a 3D solid model is constructed. The algorithm provides faster execution because he uses "boundary representation" and "constructive solid geometry" methods. Liu [20] comes up with a matrix-based approach to this problem. He represents conics in matrix form and constructs the 3D solid model using this matrix along with projection matrices.

Silhouettes provide us information about the object. It is possible to extract information about shape of an object by exploiting the silhouette information. Laurentini [12] who was inspired by Brady [16] proposed the idea of finding shape from silhouettes of an object from different viewpoints. He called the resulting shape found from silhouettes as "visual hull" of an object. In his paper [12], he implicitly claims that every object has a visual hull just like they have a convex hull. Visual hull of an object is defined as the closest approximation of an object obtained from silhouettes [12]. Volumetric intersection method is used to generate visual hull of an object using silhouettes from different viewpoints. Lines starting from the optical center of the image plane and passing from the pixels belonging to the silhouette are intersected. This method is called "volumetric intersection".

Srinivasan [29] used volumetric intersection method for 3D reconstruction along with a contour based strategy. He assumed that each object can be represented by parallel stacked contour planes that correspond to sampling in the third dimension. He developed a data structure to represent object contours lying on stacked parallel planes. Surface intersection method applied to the multi-view binary contour images was followed by a contour intersection step to generate a 3D contour representation of objects.

Matusik [13] used visual hull concept in a real-time virtualized reality application to catch up with timing requirements. His visual hull reconstruction method is image-based rather than silhouette-based. He computed visual hull of an object that is followed by texture mapping using images taken from reference views.

Point clouds are used to express the group of locations whose values are nonzero in a 3D lattice. Point clouds are chosen to be reconstructed because some kinds of lesions appearing in some medical imaging techniques can be modeled using them. An example of such lesions is grouping of microcalcifications visualized by a mammatome apparatus (see Figure 3). Those lesions are assumed to be grouped in the tissue. Therefore, point clouds stand for opaque regions that mimic those lesions. Locations of point clouds should be obtained exactly by the reconstruction process for subsequent operation. Carr [17] uses basic trigonometry and parallax shift to locate breast lesions.

The purpose of this thesis is to reconstruct point clouds from multi-view orthographic projections. Point clouds are generated through stochastic processes. Their orthographic projections are generated from any possible viewpoint. Point correspondences between these orthographic projections could not be found correctly. For this reason, volumetric intersection method is carried out to reconstruct point clouds from multi-view orthographic projections. A performance measure is devised to assess reconstructions upon observing erroneous reconstructed points. A series of experiments are performed to investigate the effect of number of projections and number of points to reconstruction.

1.2 Motivation

At the beginning, the motivation was to devise a method that automates the localization of breast lesions. Current mammatome technology require a physician to mark the lesion locations on mammography images. Triangulation using the corresponding marks on mammography images is applied along with basic trigonometry to find the 3D locations of breast lesions in a mammatome apparatus. Breast biopsy is done using a needle following the localization of breast lesions. In this work, the desirable course of study was to devise a method that automates 3D localization of breast lesions from actual mammatome images. Lesion positions were to be extracted from the

input mammatome images using image processing techniques. Moreover, volumetric intersection method was to be applied to find candidate locations of the lesions in the desirable course of study. Proper error correction methods are to be applied to improve the presicion and accuracy of lesion locations. However, we could not followed this course of study: we were not able to receive sufficient amount of real mammatome data. We made some attempts to collaborate with research centers studying breast cancer so that they could provide us with mammatome images. The attempts failed and we chose to proceed without using real data. Indeed, there was not enough time or resources to seek further for sources of mammatome data. So, we changed the purpose of this thesis. The main purpose was redefined to be the reconstruction of 3D point clouds from their orthographic projections.

Therefore, we artificially generated data using orthographic projections of point clouds. They are generated inside a discrete 3D lattice using stochastic methods. The stochastic methods are designed in such a way that points are distributed inside a discrete 3D lattice according to a desired distribution. Point clouds may be designed to represent lesions in breast tissue. Indeed, a thorough study should have carried out to derive statistics of distribution of lesions in breast tissue so that point clouds can statistically model lesions. However, there was not enough mammatome images to derive statistics of distribution of breast lesions and a study that derives the statistical distribution of breast lesions was not encountered in the literature. For this reason, we generated point clouds such that we believe that they mimic lesions in breast tissue to the best of our judgement. Orthographic projections of point clouds are generated and these orthographic projections are used as input data to this method.

1.3 Scope and Outline of This Thesis

This thesis is about 3D reconstruction of point clouds using multi-view orthographic projections. The proposed reconstruction method is developed using the visual hull concept and the volumetric intersection method. There are six chapters in this thesis. Chapter 1 is the introduction with background information. Chapter 2 explains the statistical methods used in point cloud generation. Gibbs sampler algorithm is explained

in this Chapter as well. 3D reconstruction using multi-view orthographic projections is the subject of Chapter 3. Visual hull concept is explained in Chapter 4. Results of the proposed reconstruction method as well as performance evaluation and comparison of the reconstructions are included in Chapter 5. Conclusions and future work are the subjects of Chapter 6.

Chapter 2

Point Cloud Generation Using 3D Stochastic Methods

Let L denote the set of all locations in a finite discrete 3D lattice and let s denote a location on the lattice. The finite discrete 3D lattice has the shape of a cube for the sake of simplicity. Discrete locations inside the finite discrete lattice are elements of the set L. These locations, denoted by $s \in L$, are composed of three components as $s = [x \quad y \quad z]^T$ where x, y, and z are integers representing the cartesian coordinate variables and T stands for transpose operator. Now, let f be the function that maps L into B where $B = \{0,1\}$. A point is defined as a single location, s, inside the finite discrete 3D lattice such that $f(s) = 1$. A point corresponds to a sample of an opaque region. It is used in the same meaning as opacity in this thesis. According to the same argument, binary 0 corresponds to transparency whereas binary 1 corresponds to opacity. Therefore, point clouds can be defined as the set $PC, PC \subseteq L$, that is composed of s such that $f(s) = 1$. Point clouds can contain any integer number of points in the closed interval $[0,N^3]$ where N is one side-length of the cubic finite discrete 3D lattice.

Figure 2 illustrates a finite discrete 3D lattice structure. In this Chapter, point clouds are generated using a stochastic model. Before going into the details of the stochastic model, the motivation of utilizing point clouds is explained below.

Microcalcifications are tiny spots of calcium in the breast. They are a sign of harmless cancer type that can turn into harmful type in a decade. They are formed when calcium ions start to accumulate on cell membranes in breast tissue. Coordinates of microcalcifications are used in localization of such lesions for biopsy. The distribution of microcalcifications in breast tissue differs from patient to patient. The point clouds are to

mimic the opaque regions in X-ray tomography like microcalcifications. They correspond to regions whose intensity is distinguished from their neighborhoods.

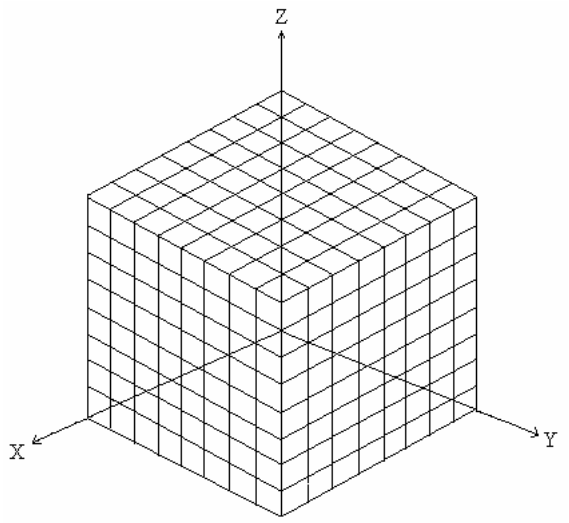

Figure 2 3D discrete lattice structure

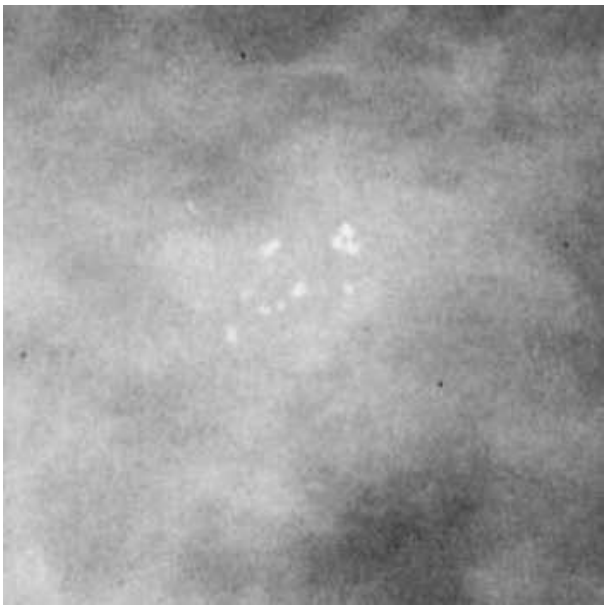

Figure 3 Digital Mammography image illustrating microcalcifications in breast tissue

2.1 The Stochastic Process

Finite discrete 3D lattice structure has been defined and illustrated in Figure 2. Point clouds are going to be generated inside a finite discrete 3D lattice, *L*. The stochastic process generates point clouds in three steps: (1) choosing cluster centers according to uniform distribution, (2) determining cluster size, shape and orientation, and (3) generating texture of clusters using Gibbs sampler. A cluster is a group of points gathered around some center that is called as "cluster center".

2.1.1 Cluster Centers

The cluster centers are initially chosen to be random vectors of three elements, namely, $p = [a \quad b \quad c]^T$ where $\{a, b, c\}$ are uniform random variables in the interval $[0, \quad 1)$ and $p \in L$. The random vector p is scaled to lattice dimension by using a linear scale factor so that the cluster centers are distributed inside the discrete 3D lattice in a uniformly distributed fashion. The elements of p are rounded to the nearest integer after scaling operation. The cluster centers do not have to indicate opacity. Random processes are used because lesion distribution changes from patient to patient in an indeterministic way. Point clouds are generated from points clustered around the cluster centers. These points that are clustered around a cluster center are generated by stochastically processing cubes.

2.1.2 Cluster Size, Shape and Orientation

Cluster size, shape and orientation are determined by means of operations on cubes involving random variables. Before explaining the operations, cube is defined in the following. A cube is a region in a discrete 3D lattice defined by the equality, $f(s) = 1$ where $s = [x \quad y \quad z]^T$ in the region with the constraint; $x_o \le x < x_o + d$, $y_o \le y < y_o + d$, $z_o \le z < z_o + d$. The variables x, y, z, x_0, y_0, z_0, and d are integers. The variable d is used

to denote the side-length of the cube, x_0, y_0, and z_0 stand for the reference corner coordinate variables of the cube.

Cubes with random side-lengths and random reference corner locations are generated around each cluster center. We choose $d \in \{1,2,3\}$ according to probability density given in Equation (1).

$$P(d = i) = \begin{cases} \frac{1}{3} & if \quad i \in \{1,2,3\} \\ 0 & otherwise \end{cases} \tag{1}$$

The reference corner coordinates of each cube are generated using a sequence of operations involving random variables. The purpose of these operations is to generate clusters distributed in an ellipsoidal region with correlated coordinates. A Gaussian random vector, $x = [x_0 \quad y_0 \quad z_0]^T$, is generated where x_0, y_0, z_0 represent the reference corner coordinates. Those random variables representing reference corner coordinates are zero mean but their variances are random in the interval [1, 3) according to uniform distribution. Therefore, cubes are created inside arbitrary ellipsoidal regions created by varying variances in each cartesian coordinate. Later, Gaussian random vector, *x*, is multiplied by a rotation matrix that is illustrated in Equation (2). The purpose of this operation is to correlate coordinates of cubes and thus clusters. The rotation matrix, *A*, in Equation (2) is created by multiplying three rotation matrices each rotating around one of x, y and z axes with random rotation angles, α, β, γ, respectively. The corresponding equation of this matrix multiplication is also given in Equation (2). The Gaussian random variables, α, β, and γ, that denote rotation angles in radians have zero mean and unit variance. The rotation operation is performed with respect to the corresponding cluster center. Each element of the resulting random vector representing reference corner coordinates of each cube, *x*', is scaled by a factor proportional to a side-length of the cubic discrete 3D lattice, *L*. This scaling operation has no effect on the means of reference corner coordinates whereas it scales their variances by a factor proportional to the square of the smallest side-length of *L*. The resulting reference corner coordinates after all of the described operations are floating point numbers with respect to their

cluster center. These floating point numbers are rounded to the nearest integer so that they correspond to elements of *L*.

$$\begin{bmatrix} x'_0 \\ y'_0 \\ z'_0 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} \qquad A = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$$

$$x'=Ax \quad \text{where} \quad A=R_Z R_Y R_X \tag{2}$$

User defined number of cubes of random dimension are distributed around all those randomly obtained cluster centers. The uniform pseudo-random number generator called Mersenne Twister (MT) [1] generates all the random numbers. The Gaussian random numbers that are derived from uniformly distributed random variables are zero mean and unit variance [2].

2.1.3 Cluster Texture - Gibbs Sampler

The purpose is to process cubes described in section 2.1.2 such that the resulting point clouds resemble lesion distribution like that of microcalcifications. Gibbs sampler is employed in converting floating small cubes into point clouds whose texture simulates breast microcalcifications.

Let *D* denote the region containing a cube and its surrounding voxels inside the discrete 3D lattice. *D* consists of coordinates δ, ε, ζ where δ, ε, ζ denote integers satisfying $0 \le \delta \le d+1$, $0 \le \varepsilon \le d+1$, $0 \le \zeta \le d+1$ and *d* is the side-length of the cube. The range of δ, ε, ζ extend from 0 to $d+1$ to contain the surrounding voxels where the coordinate (1, 1, 1) represent the reference corner of the cube. There is a mapping from domain *D* to a binary pattern as denoted by $w(D)$. Each possible outcome of this mapping, is given as,

$$w = \left(\chi_{000}, \chi_{001}, \chi_{010}, \ldots, \chi_{\delta\varepsilon\zeta}, \ldots\right) : \chi_{\delta\varepsilon\zeta} \in B$$
$$0 \le \delta \le d+1, \;\; 0 \le \varepsilon \le d+1, \;\; 0 \le \zeta \le d+1. \tag{3}$$

$\chi_{\delta\varepsilon\zeta}$ represents the elements of binary number set, B, at the lattice coordinate (δ, ε, ζ).

For simplicity, locations on D are represented by s_i and s_j and formulated as $D = \{s_i\}$. So, w can also be defined in terms of lattice locations as $w(s_i)$. Similarly, w' is also a binary pattern defined on D as $w'(D)$ and it can be defined in terms of lattice locations as $w'(s_i)$. Let Ω represent the set of all realizations of this mapping. Thus, $\Omega = \{w\}$ where w is a binary pattern over the lattice denoting realizations of the mapping and d+2 is the side-length of the region D. The Gibbs distribution that is related to $\chi_{\delta\varepsilon\zeta}$ is a probability measure denoted by P on Ω as,

$$P(w) = \frac{1}{Z} e^{-U(w)}. \tag{4}$$

In Equation (4), Z stands for the normalization constant, and $U(w)$ is the energy function associated with each sample function [24,25].

Before expanding the energy function, cliques and clique potentials are going to be defined. Given a 3D lattice, a clique is a subset of the lattice. The set of all cliques that is denoted by Q is the same as all the subsets of the lattice. Examples of one–voxel subsets and two–voxel subsets in a 2D lattice can be seen in Figure 4. The set of all one–voxel cliques are denoted by Q_1 and that of two–voxel cliques are represented by Q_2. There are $2^{(d+2)^3}$ cliques defined in a cubic 3D lattice with a side-length of d+2. Clique potential is a function defined on each such clique. Clique potentials are chosen to be position independent in our model.

Since cliques are defined on lattice locations, notation for lattice locations are going to be used in place of cliques as well. Therefore, If $s_1 = (\delta_1 \quad \varepsilon_1 \quad \zeta_1)$ and $s_2 = (\delta_2 \quad \varepsilon_2 \quad \zeta_2)$, $w(s_1) \in \{0,1\}$ and $s_1 \in Q_1$. Similarly, $w(s_1, s_2) \in \{(0,0),(0,1),(1,0),(1,1)\}$ and $(s_1, s_2) \in Q_2$.

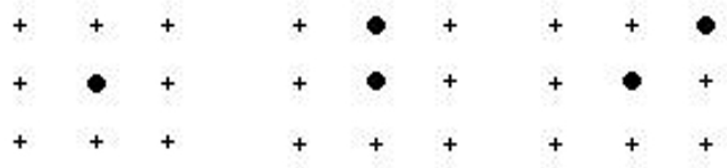

Figure 4 Single-voxel clique and examples of two-voxel cliques

The energy function $U(w)$ is given by,

$$U(w)=\sum_{q\in Q}V_q(w) \tag{5}$$

where q denotes individual cliques, Q denotes the set of all cliques, and V_q represents the clique potential in Equation (5). Illustrations about cliques can be found in [7] and in Figure 4. In this interpretation, majority of two-voxel cliques and cliques that have more than two voxels are found out to have zero clique potential after a series of calculations given in Equation (8). Only those two-voxel cliques that are in the 26-neighborhood of each other have position independent non-zero clique potentials. Therefore, the sum of clique potentials, $\sum_{q\in Q}V_q(w)$, becomes; $\sum_{q\in Q}V_q(w)=\sum_{s_i\in Q_1}V_1(w(s_i))+\sum_{\substack{(s_i,s_j)\in Q_2\\ s_i\neq s_j}}V_2(w(s_i,s_j))$, where $\sum_{\substack{(s_i,s_j)\in Q_2\\ s_i\neq s_j}}V_2(w(s_1,s_2))$ represent the sum of two-voxel clique potentials and $\sum_{s_i\in Q_1}V_1(w(s_i))$ represent the sum of one–voxel clique potentials. The clique potentials are defined as,

$$V_2\left(w(s_i,s_j)\right)=\begin{cases}-a & if \quad w(s_i,s_j)\in\{(0,0),(1,1)\} \quad where \quad s_j\in\eta_{s_i}\\ 0 & else\end{cases}$$

$$V_1\left(w(s_i)\right)=\begin{cases}k & if \quad w(s_i)=1\\ 0 & if \quad w(s_i)=0\end{cases}. \tag{6}$$

In Equation (6), η_{s_i} represents the 26-neighborhood of s_i in D. 26-neighborhood is the neighborhood scheme involving 26 voxels around the primary voxel. Rubik's cube is used to illustrate this neighborhood scheme in Figure 5.

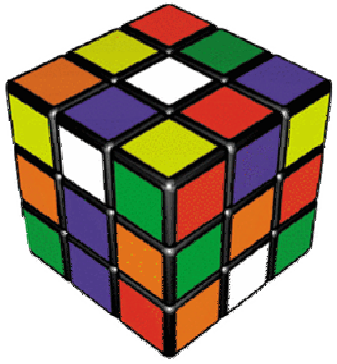

Figure 5 Rubik's cube to illustrate 26-Neighborhood

Gibbs distributed probability, $P(w)$, has been explained until now. The Gibbs sampler generates sample outcomes from this distribution. Each sample is an element of Ω. Samples are generated as a result of an iterative algorithm. Gibbs sampler starts with the initial pattern over the region D where D is defined in Subsection 2.1.3. The basic idea behind Gibbs sampler is to compare Gibbs probability (or energy) of the pattern containing all voxels inside the region D with Gibbs probability (or energy) of the modified pattern. The modification involves inversion of the binary value of a randomly picked voxel inside D. Therefore, binary value of a randomly chosen voxel of the current pattern is inverted resulting in a new pattern. And, Gibbs probability (or energy) of the current pattern is compared with that of the new pattern. At each iteration, if the new pattern yields more probable pattern (or lower energy) with respect to the Gibbs distributed probability (or energy) of the current pattern, then the new pattern is adopted as the current pattern. If, however, the new pattern has less probability, then the new pattern is adopted with a probability equal to the ratio of probability of the new pattern to the probability of the current pattern. If we denote the current pattern with w and the new pattern with w', the Gibbs sampler becomes,

$$
\begin{aligned}
&If \quad r = \frac{P(w')}{P(w)} = \frac{\frac{1}{Z}e^{-U(w')}}{\frac{1}{Z}e^{-U(w)}} = \frac{e^{-U(w')}}{e^{-U(w)}} = e^{-[U(w')-U(w)]} \geq 1, \quad w \rightarrow w' \\
&If \quad r < 1, \quad w \overset{with \ \ probability \ \ r}{\rightarrow} w'
\end{aligned}
\tag{7}
$$

and this is iterated four times the number of all voxels in w for randomly selected voxels.

The Gibbs sampler algorithm starts with the same pattern in each case. It is proved in [7] that the iterative algorithm yields a Gibbs sample for every starting configuration if there is enough number of iterations. It is empirically found that the iteration converges to a Gibbs sample when number of iterations is more than four times number of all voxels. This empirical derivation is supported by [7] where convergence properties of Gibbs sampler algorithm are established.

Equation (7) is further simplified by canceling all one-voxel clique potentials except for the clique potential of randomly picked voxel and by canceling all two-voxel clique potentials formulated in Equation (6) except for clique potentials composed of two neighboring voxels involving the randomly picked voxel. The cancellation is because that only a single voxel is changed when going from w to w'. All the other voxels have the same values as before. Therefore, clique potentials with respect to these unchanged voxels are the same and they yield further simplification. The cancellation and simplification is given in Equations (9) and (10). Before applying simplification and cancellation, difference of Gibbs energies of any two patterns w and w' are given as,

$$
\begin{aligned}
&U(w') - U(w) = \sum_{q \in Q} V_q(w') - \sum_{q \in Q} V_q(w) = \\
&\sum_{s_i \in Q_1} V_1(w'(s_i)) + \sum_{\substack{(s_i, s_j) \in Q_2 \\ s_i \neq s_j}} V_2(w'(s_i, s_j)) - \sum_{s_i \in Q_1} V_1(w(s_i)) - \sum_{\substack{(s_i, s_j) \in Q_2 \\ s_i \neq s_j}} V_2(w(s_i, s_j)) .
\end{aligned}
\tag{8}
$$

The simplification is given in Equation (9) where the pattern w' is same as w except for the value of a single voxel, s_v. Starting with Equation (8), $\sum_{s_i \in Q_1} V_1(w'(s_i))$ can be

expanded as; $\sum_{s_i \in Q_1} V_1(w'(s_i)) = \sum_{s_i \neq s_v} V_1(w(s_i)) + V_1(w'(s_v))$. Similarly, $\sum_{\substack{(s_i,s_j) \in Q_2 \\ s_i \neq s_j}} V_2(w'(s_i,s_j))$ can be expanded as; $\sum_{\substack{(s_i,s_j) \in Q_2 \\ s_i \neq s_j}} V_2(w'(s_i,s_j)) = \sum_{\substack{(s_i,s_j) \in Q_2 \\ s_i \neq s_j \\ s_i \neq s_v}} V_2(w(s_i,s_j)) + \sum_{\substack{(s_v,s_j) \in Q_2 \\ s_v \neq s_j}} V_2(w'(s_v,s_j))$ since w' is same as w except for the value of s_v. Therefore, Equation (8) can be written as;

$$\begin{aligned}
&\sum_{s_i \in Q_1} V_1(w'(s_i)) + \sum_{\substack{(s_i,s_j) \in Q_2 \\ s_i \neq s_j}} V_2(w'(s_i,s_j)) - \sum_{s_i \in Q_1} V_1(w(s_i)) - \sum_{\substack{(s_i,s_j) \in Q_2 \\ s_i \neq s_j}} V_2(w(s_i,s_j)) = \sum_{s_i \neq s_v} V_1(w(s_i)) + \\
&V_1(w'(s_v)) + \sum_{\substack{(s_i,s_j) \in Q_2 \\ s_i \neq s_j \\ s_i \neq s_v}} V_2(w(s_i,s_j)) + \sum_{\substack{(s_v,s_j) \in Q_2 \\ s_v \neq s_j}} V_2(w'(s_v,s_j)) - \sum_{s_i \neq s_v} V_1(w(s_i)) - V_1(w(s_v)) - \\
&\sum_{\substack{(s_i,s_j) \in Q_2 \\ s_i \neq s_j \\ s_i \neq s_v}} V_2(w(s_i,s_j)) - \sum_{\substack{(s_v,s_j) \in Q_2 \\ s_v \neq s_j}} V_2(w(s_v,s_j))
\end{aligned} \quad . \quad (9)$$

After canceling similar terms in Equation (9) and applying the clique potentials given in Equation (6), simplification in Equation (10) is obtained.

$$\begin{aligned}
&V_1(w'(s_v)) + \sum_{\substack{(s_v,s_j) \in Q_2 \\ s_v \neq s_j}} V_2(w'(s_v,s_j)) - V_1(w(s_v)) - \sum_{\substack{(s_v,s_j) \in Q_2 \\ s_v \neq s_j}} V_2(w(s_v,s_j)) = \\
&V_1(w'(s_v)) - V_1(w(s_v)) + \sum_{\substack{(s_v,s_j) \in Q_2 \\ s_v \neq s_j}} \left(V_2(w'(s_v,s_j)) - V_2(w(s_v,s_j))\right)
\end{aligned}$$

$$V_2(w'(s_v,s_j)) - V_2(w(s_v,s_j)) = \begin{cases} a \;\; if \;\; w(s_v,s_j) \in \{(0,0),(1,1)\} \;\; where \;\; s_j \in \eta_{s_v} \\ -a \;\; if \;\; w(s_v,s_j) \in \{(0,1),(1,0)\} \;\; where \;\; s_j \in \eta_{s_v} \\ 0 \;\; else \end{cases} \quad (10)$$

$$V_1(w'(s_v)) - V_1(w(s_v)) = \begin{cases} -k \;\; if \;\; w(s_v)=1 \\ k \;\; if \;\; w(s_v)=0 \end{cases}$$

The algorithm is iterated four times the number of all voxels in the entire region. The resulting pattern is a Gibbs sample. The algorithm is illustrated in Figure 6 in 2D using a square in place of a cube. The initial pattern involves a 2D 3 by 3 square white region surrounded by black region as illustrated in Figure 6.a. The dimension of patterns in Figure 6 is 5 by 5. After one step of the iteration, the pattern shown in Figure 6.b is generated. After five steps, the pattern in Figure 6.e is generated. After 100 steps that is 4 times the total number of pixels, the pattern in Figure 6.f is generated.

Each cluster is processed as described and the Gibbs sample is placed inside the 3D lattice. If two overlapping clusters are encountered while placing a Gibbs sample inside the discrete 3D lattice, then values of the latter cluster overwrites the previous one. The described three step random process in subsections 2.1.1, 2.1.2 and 2.1.3 completes the generation of the point cloud.

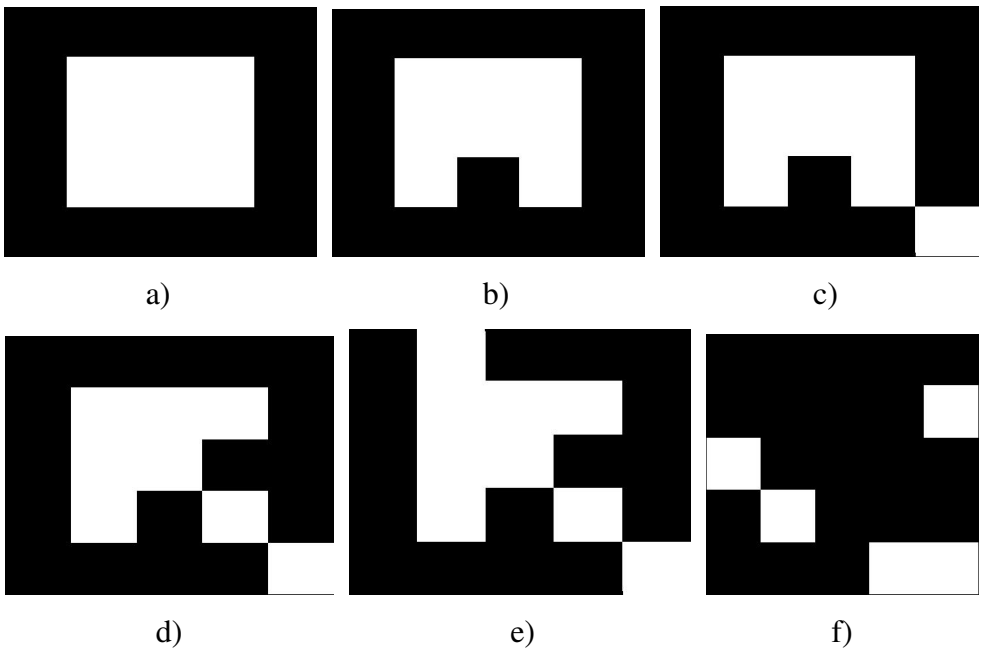

Figure 6 Illustration of Gibbs sampler algorithm in 2D using a 3 by 3 square white region surrounded by black region as an initial pattern. a, b, c, d, e, f indicate evolutions with the iteration.

The algorithm can be summarized as follows:

For the designated cube, start with the pattern over the region D that contains a cube and its surrounding voxels as described in subsection 2.1.3. Although there is no obligation for the initial pattern, this pattern is chosen for easy implemention.

1. Randomly pick a voxel, s_v, inside the current pattern, according to uniform distribution. Generate a new pattern by inverting binary value of s_v.
2. Compute the difference of energy functions, $U(w')$-$U(w)$, as given in Equations (9) and (10).
3. Compute r that is given in Equation (7) using the result of Step 2.
4. If $r \geq 1$, then adopt the new pattern as the current pattern and jump to Step 1.
5. If smaller, $r < 1$, then adopt the new pattern as the current pattern with probability equal to r and jump to Step 1.

The loop is stopped when the number of iterations reaches four times the number of all voxels inside D.

Chapter 3

Reconstruction from Multi-View Orthographic Projections

3.1 How to Generate Orthographic Projections of Point Clouds

Point clouds are generated on a discrete 3D lattice as described in Chapter 2. After generating point clouds, their projections are going to be computed. Orthographic projection is used in this work rather than perspective projection because it models parallelized rays utilized in some medical imaging techniques.

Orthographic projection is a projection without scaling. The parallel rays incident on the projection plane are perpendicular to the plane. Orthographic projection of three points is illustrated in Figure 7.

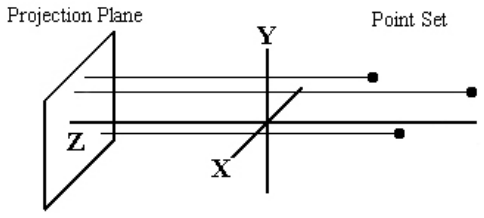

Figure 7 Orthographic projections of three points in space

The X and Y coordinates are preserved in this projection but the depth information (Z coordinate) is lost. During orthographic projection operation, the Z value is zeroed out as can be seen from the matrix operation,

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \underline{\lambda} = \Omega \underline{\Lambda}\,, \tag{11}$$

where λ denotes the projection plane coordinates, Ω denotes the orthographic projection matrix, and Λ denotes the world coordinates of the points in homogenous coordinates.

Homogenous coordinates stem from homogenous equation of the form $ax + by + cz + dw = 0$ where a, b, c, d are scalars and x, y, z, w are homogenous coordinate variables [27]. The w variable represents the scale factor. The scale is d in the equation above. Variables a, b, and c are divided by the scale d to convert into cartesian coordinates. Therefore, homogenous $[a\ b\ c\ d]^T$ is equivalent to $[a/d\ b/d\ c/d]^T$ in cartesian coordinates. Homogenous coordinates are used to represent conic equations correctly in matrix and/or vector form. The difference between homogenous and cartesian coordinates is the addition of a new dimension, w. The scale in the matrix equations above is 1.

The operation of taking orthographic projection of point clouds is illustrated in Figure 8. Suppose there are three projection planes with normal vectors and points scattered in a 3D lattice as shown in Figure 8. There is a center of rotation where all normal vectors of projection planes intersect if they are not parallel. Orthographic projections of those points are taken using imaginary parallel rays retrograde to normal vectors of projection planes. The projection planes can have any possible orientation always facing the discrete 3D lattice. Imaginary parallel rays are constantly chosen to be perpendicular to the target projection plane.

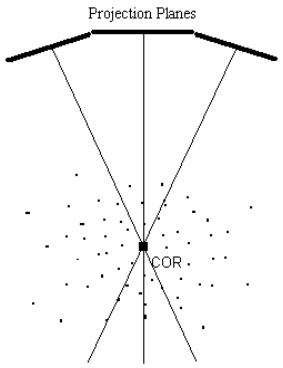

Figure 8 Cross-section of projection planes and point clouds

Imaginary parallel rays pass through the 3D lattice in the opposite direction of the normal vectors of the projection planes. If they confront points during their travel, they hit the point; therefore, that is the end of their travel. Thus, binary 1s appear in the corresponding pixels of the projection planes when they encounter a point during their travel. If they do not confront anything, they reach the projection plane and a binary zero appears in the corresponding pixels of the projection planes.

These parallel rays are designed to have any orientation. However, if continuous line model is adopted, then those continuous rays will pass through intervoxel space (see Figure 11). Detection of those lines running into points during their travel in the intervoxel space is experimented to be error-prone. For this reason, discrete line model is adopted in place of continuous line model for those parallel rays. Discrete parallel rays do not pass through intervoxel space unlike continuous ones. They pass through lattice elements along their paths. In other words, discrete rays travel in a direction by visiting nearest lattice elements as can be seen in Figure 9. In Figure 9, there are three discrete parallel rays shown by gray color on a screen composed of pixels. There are two more identical white-colored rays in between those three rays. It can be seen from Figure 9 that discrete parallel rays span the discrete 2D lattice and they do not intersect. We can generalize this conclusion and claim that discrete 3D parallel rays also span discrete 3D lattice and they do not intersect as well. Furthermore, notice that, each pixel on digital rays in Figure 9 is connected to another pixel in its 8-neighborhood. 8-neighborhood scheme is used in generating those rays so that parallel rays do not intersect with each

other whereas they do intersect if 4-neighborhood scheme is adopted. Neighborhood schemes in the plane are illustrated in Figure 10. The crosses are in the 4-neighborhood, the dots and crosses together are in the 8-neighborhood of the large point at the center.

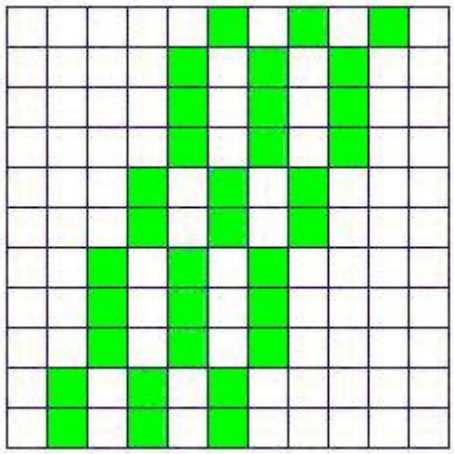

Figure 9 Parallel discrete lines

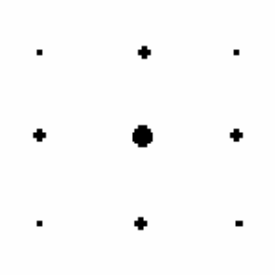

Figure 10 Figure illustrating 4-Neighborhood and 8-Neighborhood

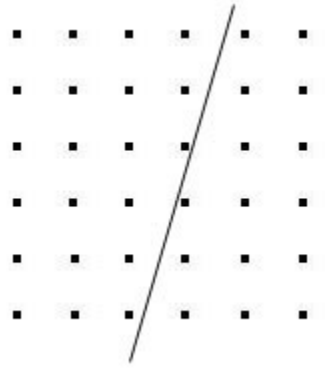

Figure 11 Continuous line in a 2D discrete lattice

3.2 How to Extract Depth Information from Orthographic Projections

There are numerous works in the literature that are dedicated to finding the lost information from projections in many ways and for various purposes [4, 5, 6, 15, 17]. In this thesis, finding the depth information of points forming point clouds inside a 3D lattice is one of the goals. One of the ways to do this is to devise search algorithms to find point correspondences between projection images. After finding point correspondences, rays emanating from those locations at different images are intersected using triangularization. The resulting intersection gives the coordinates of the reconstructed point. If point correspondences are not found correctly, then the reconstruction will not be the same as the original point clouds.

Figure 12 and 13 are orthographic projection images of point clouds taken from different viewpoints. The point clouds are generated as described in Chapter 2 and their orthographic projection is taken as explained in Chapter 3 Section a. In this Section, experiments that were carried out to find the depth information of point clouds are explained.

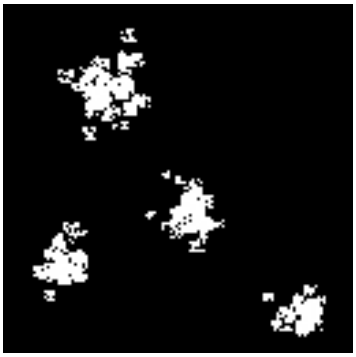

Figure 12 Orthographic projection of point clouds

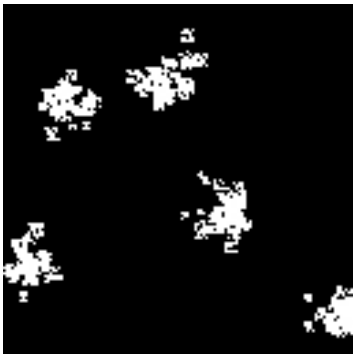

Figure 13 Orthographic projection of point clouds from different viewpoint

We are going to investigate projection images to devise methods for reconstruction. The white regions in Figures 12 and 13 indicate regions having a value of binary 1 and the black regions indicate regions having a value of binary 0. Those binary images are scaled for illustrative purposes. The two clusters at top left in Figure 13 are overlapped in Figure 12 as a result of projections from different viewpoints. The remaining clusters in Figure 13 have minor differences when compared to corresponding clusters in Figure 12 if observed carefully. Distinctive pixels in both figures are those that have different value than surrounding pixels. White pixels surrounded by black ones and black pixels surrounded by white pixels form those distinctive pixels.

The first approach was to apply block matching algorithm to find point correspondences. However, this algorithm did not yield satisfactory results due to merged clusters and some of the blocks could not be matched. Hierarchical block matching algorithm was applied to improve results of block matching algorithm and to allow matching algorithm to have smaller blocks when necessary. This also did not result in satisfactory results. The last experiment was to apply Lowe's scale invariant feature transform [30] into those images. Only a small number of feature points could be extracted. The number of those feature points was not enough to calculate the angle between normal vectors of the projection planes. Although the experimented algorithms yield satisfactory results with grayscale images, they failed to function properly in binary projection images. One of the reasons is that amplitude information is not present in binary images.

If correspondences were found correctly, then rays emanating from matched points would be intersected. The coordinates of the intersection of those rays would be claimed to belong to the original point. However, when the correspondences are not found correctly, then reconstructed object will be different from the original one. Therefore, if correct point correspondences cannot be found, then algorithms involving point correspondences should be given up. An alternative method is to intersect all the rays emanating from white pixels of all projection images instead of intersecting them one at a time. It is experienced from the previous method that there will be intersection points that do not belong to the original point clouds. These points will be handled by further processing. Figures 20 and 21 illustrate intersection of rays. These rays travel in the direction of normal vectors of their planes. This method is called volumetric intersection and explained in Chapter 4. The intersection points form clouds of points called visual hull of point clouds. Visual hull is also explained in Chapter 4. Obviously, the resulting reconstruction has more points than the original point clouds. Therefore, this thesis continues with methods devised to improve reconstructions of point clouds.

Chapter 4

Visual Hull

4.1 Convex Hull

Suppose we have a set *S* in multi-dimensional space. *S* is defined to be convex if any line segment joining any two points, $m \in S$ and $n \in S$, is inside the set. In Figure 14, the second set is not convex whereas the first is because the line joining *m* and *n* contains points outside the set *S* [8, 9, 10, 11].

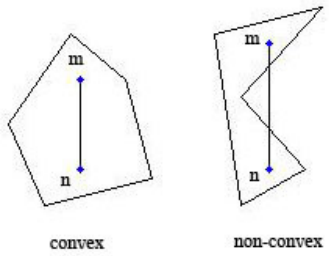

Figure 14 Convex and non-convex sets

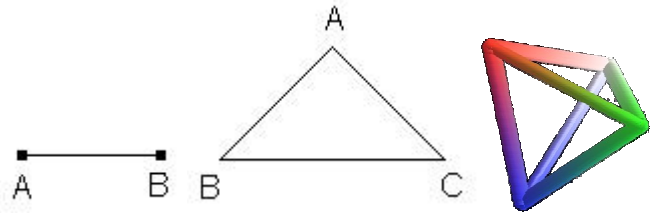

Figure 15 A line segment, a triangle and a tetrahedron

A line segment is an example of one-dimensional convex set. A triangle is the example of two-dimensional convex set with minimum number of vertices. Likewise, an example of three-dimensional convex set is a tetrahedron (Figure 15). Notice that, the points on a line segment can be computed using linear combination of its endpoints. Moreover, the points inside a triangle can be computed from the linear combination of its

vertices. Likewise, the linear combination of corners of a tetrahedron gives the points inside it. These linear combinations should be such that the coefficients must be nonnegative and they must add up to 1. This is called the convex combination [8, 9, 10, 11]. Suppose there are k points that form a convex set namely x_1, x_2,…, x_k. Then, convex combination is formulized as:

$$\alpha_1 x_1 + \alpha_2 x_2 + ... + \alpha_k x_k \quad with \quad \alpha_i \geq 0 \quad \forall i \quad and \quad \alpha_1 + \alpha_2 + ... + \alpha_k = 1 \qquad (12)$$

Convex hull is also a convex set. Convex hull of S is the smallest convex set that contains S. In other words, it is the intersection of all convex sets containing S. All the points contained in the convex hull of S can be calculated using convex combinations of points of S. The proof is trivial.

Before going into the convex hull in a plane, I need to define the extreme points. A point in S is not an extreme point if it lies on the open line segment formed by any two points in S. Moreover, a point is not an extreme point if it lies inside the triangle formed by three points that are element of S, not on the triangle. Those points that are not one of the above are extreme points. Therefore, the convex hull of extreme points is the same as the convex hull of the point set. So, in order to find the convex hull of S, all I need to do is to find the extreme points and connect them in some order. There are numerous algorithms to compute the convex hull in plane. A few of them are "quickhull" [8, 10], "Graham scan" [8, 10, 11] and "incremental algorithm" [10, 11]. An illustration of a convex hull in a plane is given in Figure 16 where extreme points are the ones connected with lines.

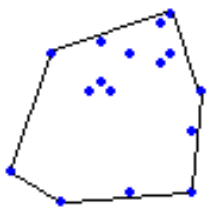

Figure 16 A set of points in plane and their convex hull

Convex hull is a polyhedron in space which looks like the one given in Figure 17. Convex hull in space can be treated in the same way convex hull in plane is treated. Some of the 3D convex hull computation algorithms are extensions of 2D algorithms into 3D. Convex hull in space may be computed using the "incremental algorithm" [8, 10], "gift wrapping" [10] or the "randomized incremental algorithm" [10].

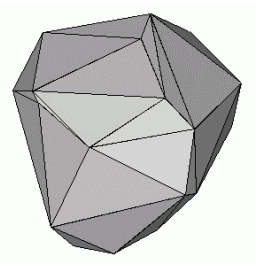

Figure 17 Convex hull of a polyhedron in space

4.2 Visual Hull

Consider the traffic signs in Figure 18.

Figure 18 Traffic signs

There is a silhouette of a man walking: a pedestrian. From this silhouette, we understand that there is a regulation related to pedestrians. Similarly, the sign with a silhouette of falling rocks tells us that rocks may fall or have fallen. Those signs with silhouettes of a bus and a bike give us useful information as well. Indeed, our brains recognize objects from the silhouette information provided. The information provided by

silhouettes is not detailed and can be deceptive as well. However, there exist situations where detail or ambiguity is not critical. The concept of visual hull emerged from the idea of using silhouette information in 3D reconstruction.

The visual hull is a 3D concept. The aim is to recognize or to recover the shape of an object. Constructing visual hull of an object requires volumetric intersection method that is explained in Chapter 4, Subsection c. Briefly, volumetric intersection method intersects cones from all the silhouettes to get the visual hull. The tops of these cones are the optical centers of the images. In this thesis, orthogonally projected images are used, therefore, cones become cylinders. Visual hull that is constructed using the volumetric intersection method with respect to proper viewpoints approximates an object. The closeness of this approximation depends on the number of silhouettes, convexity of the object and the width of the viewing region. If the object is equal to its visual hull, then the volumetric reconstruction is perfect [12]. Convex hull contains the visual hull and together they contain the object. In mathematical notation: $CH(\Theta) \supseteq VH(\Theta) \supseteq \Theta$ where Θ denotes the object [12]. Taking projections of an object and its visual hull with respect to same viewing regions result in equivalent silhouettes. Moreover, two objects can be distinguished from their silhouettes according to the same viewpoint if and only if their visual hulls are different [12]. Otherwise, they yield the same silhouettes from the same viewpoints.

4.3 Volumetric Intersection Method

The volumetric intersection method is applied to build the visual hull of point clouds. Let V_i denote the cylindrical volume constructed by sweeping the i^{th} silhouette; then the volumetric intersection method provides the visual hull, VH, built from n silhouettes according to $VH_n = \bigcap_{i=1}^{n} V_i$ [13]. As n becomes infinitely large and all possible viewpoints are included, VH_n converges to a shape called VH_∞ of the object of interest. Figure 19 illustrates this method using two silhouettes.

There are two image planes in Figure 19 each with square silhouettes in them. Those square silhouettes are swept in the normal directions of their planes and the volume

emerged by their intersection is the visual hull. This volume is highlighted using white color in Figure 19. If the silhouettes were taken by perspective projection, then cones were to be intersected [12, 13].

The volumetric intersection algorithm developed in this thesis emanates parallel discrete lines from silhouettes in the normal direction of the projection planes. Considering those 3D lattice locations as bins, each line adds 1 to the bin as they pass through. Therefore, the intersections of all silhouettes are those bins that have a value equal to the number of silhouettes. The coordinates of those bins are gathered to construct the visual hull of point clouds.

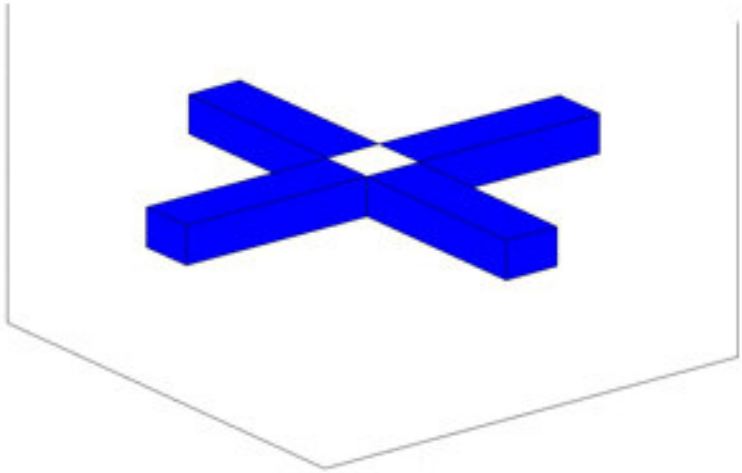

Figure 19 Illustration of volumetric intersection method. Squares are swept to generate the cube

4.4 Removal of Undesired Reconstructions

The visual hull of point clouds is not equal to the original point clouds for most cases. In order to always have perfect reconstruction; the number of projection planes has to be larger than the number of points. The reason for this is illustrated in Figures 20 and 21.

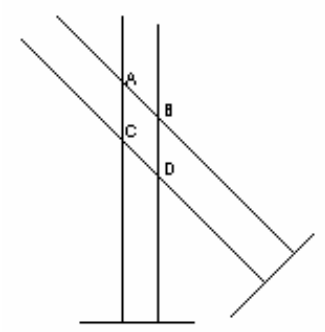

Figure 20 Reconstruction of 2 points using 2 image planes

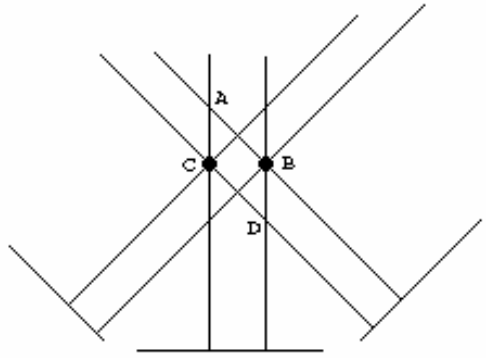

Figure 21 Reconstruction of 2 points using 3 projection planes

Figure 20 illustrates volumetric reconstruction method using 2 image planes each consisting of 2 "white" pixels. The resulting reconstructed points are A, B, C, and D. Points A and D alone gives the same silhouettes on both projection planes. Likewise, C and B give the same silhouettes. Even A, B, C, and D all together do so. There is an ambiguity in this reconstruction. There are three candidates for original points. This ambiguity is eliminated by introducing a new projection plane as in Figure 21. Points B and C remain the only solution of this reconstruction. From the above discussion, we can conclude that total number of points should be less than the number of distinct projection

planes for perfect reconstruction from silhouettes of any arrangement of points. Although there are cases where less number of silhouettes than total number of points yield perfect reconstructions, this statement offers perfect reconstruction for majority of cases. Of course, there are exceptions to this statement. Any closed volume whose interior voxels are empty, i.e. 0, is an exception to this statement. No matter how many different projections are used, they always yield a reconstruction with occupied interior voxels.

When the total number of points exceeds the number of distinct projection planes, aliases or incorrect reconstructions are likely to occur as in Figure 20. Larger number of silhouettes than the total number of points in 3D discrete lattice has to be used to avoid this kind of error in reconstructions. However, this is not possible in reality. When total number of points gets reasonably large, it is not possible to design required number of distinct projections. Therefore, this kind of error is likely to occur in volumetric reconstruction. The approach adopted to deal with this error is to adjust number of projection planes and their orientation so that this kind of error is minimized.

Another source of undesired reconstructions is that digital lines having smaller angle between them do not intersect at a single voxel; instead they intersect over several consecutive voxels due to the discrete nature of those lines. See Figure 22 for an illustration.

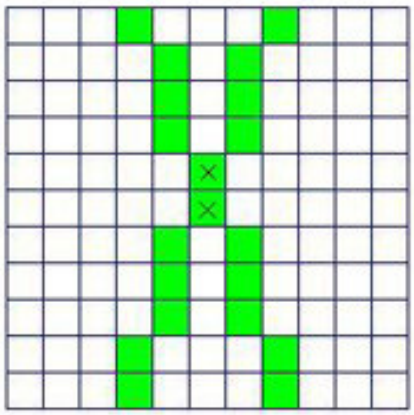

Figure 22 Two digital lines intersecting in more than one pixel

Therefore, several consecutive voxels are reconstructed representing a single voxel. Projection planes whose normals have dot product close to 0 should be used in taking projections to avoid multiple voxels representing a single voxel. This is illustrated in Figure 23.

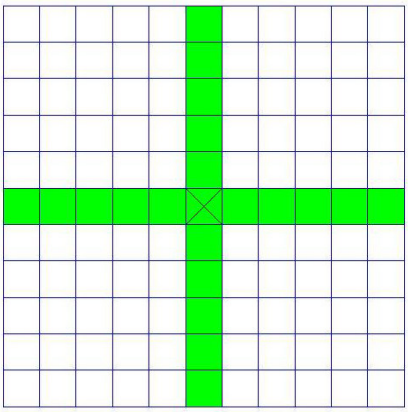

Figure 23 Intersection of right angled lines

Chapter 5

Experiments, Performance Assessment and Comparison

Simulation software is developed that takes orthographic projections of point clouds as input and generates a video file as output. The video file is created to display both original and reconstructed point clouds side by side. Original point clouds lies on left of the frame whereas reconstructed point clouds on the right. The time axis of the generated video is the same as the z axis of discrete 3D lattice. Another software is also developed to generate point clouds and their orthographic projections. This software is pipelined into the simulation software.

The reconstructed point clouds that are yielded by simulation software contain erroneous reconstructions in most of the cases. The reasons for these erroneous reconstructions are mentioned in Subsection 4.4 along with the ways to avoid them. Since those ways are unrealizable in most practical cases, a performance measure is designed to determine the best reconstruction within the limitations. The designed performance measure is chosen to be a linear combination. It is based on penalizing erroneous reconstructed points according to their types. The penalties are in the form of weights. Heavy penalty corresponds to a large weight. If a point that appears in the original data is not reconstructed, then this error is heavily penalized. This kind of error is labeled as 'type 3' and it is denoted by subscript 3. If multiple consecutive voxels are reconstructed for a single voxel as illustrated in Figure 22, then this kind of error is labeled as 'type 1' and denoted by subscript 1. Type 1 error is lightly penalized. Rest of the errors are labeled as 'type 2' and denoted by subscript 2. Type 2 error is moderately penalized. The proposed measure is given as: $M = N_1 \times W_1 + N_2 \times W_2 + N_3 \times W_3$ where M denotes performance measure, N_1 and W_1 denote number and weight of type 1 error, N_2 and W_2 denote those of type 2 error, N_3 and W_3 do those of type 3 error. The weights are fixed for

each type of error in every performance measure computations; the numbers of each type of erroneous voxels are computed in each run.

A point cloud consisting of a single point at a known location inside the 3D lattice is generated. Projections of this single point are generated for detecting type 1 errors. These projections are exactly the same as those projections used in reconstructing the original point clouds. Visual hull of this single point is constructed from projection images of this point as described in Chapter 4. The visual of a single point yields either a single reconstructed point as shown in Figure 23 or multiple consecutive points as illustrated in Figure 22 depending on the orientation of projection planes. If there are multiple consecutive voxels, then their number and orientation with respect to the original point is determined. The number and orientation of multiple consecutive voxels are used to detect type 1 errors. Therefore, for each correctly reconstructed voxel, multiple consecutive voxels are searched according to the number and orientation determined. If, however, visual hull of a single point at a known coordinate yields a single point at the same coordinate, then type 1 error does not exist in the reconstructed point clouds with the current orientation of projection planes.

Type 3 errors are detected simply by checking if reconstructed data does not have any of the points of the original data. The author has scarcely experienced type 3 error in the experiments.

Type 2 errors are detected after type 1 and type 3 errors. Simply, those reconstructed points that do not belong to the original data and that are not type 1 and type 3 errors are picked as type 2 error.

The purposes of this Chapter are to reconstruct point clouds, assess their reconstructions and study parameters affecting the reconstruction. If possible, conditions for perfect reconstruction of point clouds are going to be investigated, as well. A simple experiment involving a point cloud of several points is going to be performed for the reconstruction. Moreover, another experiment using point clouds containing spread points is going to be carried out to bring more insight into the results of the first experiment. Error types are planned to be analyzed with these two experiments as well. Furthermore, reconstructions that are performed using increasing number of projections are going to be rated according to the performance measures devised above. This

assessment is supposed to reveal the relation of performance of reconstruction to increasing number of projections. With the observation that number of points also affects reconstruction, reconstructions are planned to be assessed also by varying the number of points.

A series of experiments has been designed to investigate the reconstructed point clouds. There are two degrees of freedom in this method of reconstruction, namely, the number and the orientations of projections. Total number of points cannot be directly manipulated by the user because it is defined as a result of stochastic processes. In order to observe its effect to performance of reconstruction, total number of points is fixed by deleting points after total number of points reaches a predetermined value. Experiments are carried out to investigate the effects of number and orientations of projections on reconstruction of point clouds as well as the effect of total number of points. The first experiment is performed to demonstrate the effect of number of reconstructions on a single point cloud. Indeed, this experiment gives readers an idea about the relation of total number of points and number of projections that was introduced in Figures 20 and 21. The first simulation of this experiment is carried out using 3 projections of the point cloud. Figure 24 illustrates used rays. These rays are perpendicular to each other. Volumetric intersection method is applied to those projections as described in Chapter 4. A video file is generated to display both reconstructed and original point cloud. A chosen frame of this video file is shown in Figure 25. There are more points in the reconstructed data than the original data as seen in Figure 25. The hypothesis for this sort of reconstruction is that the number of projections is not sufficient. The simulation continues with the same point cloud using 4 projections. A fourth projection with a different orientation is added to the previous projections. A video is generated to display contents of both original and reconstructed point clouds. The ray directions used in this reconstruction are illustrated in Figure 26. The same frame shown in Figure 25 with the same data points is displayed in Figure 27. The additional projection image provides improvement in reconstruction as can be seen upon comparing Figure 25 and Figure 27. But still, reconstructed and original pattern are not identical.

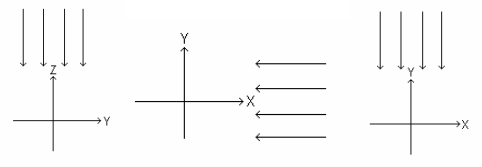

Figure 24 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 25

Figure 25 A frame of video file displaying the original data on the left and the reconstructed data on the right side

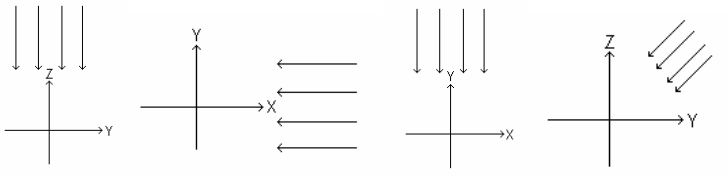

Figure 26 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 27

Figure 27 Another frame displaying both the original and the reconstructed data

Therefore, another projection is added to the simulation to observe its effect. This additional projection is chosen to be perpendicular to the previous one to reduce the volume of intersection. The ray directions are illustrated in Figure 28 and the corresponding video frame is displayed in Figure 29. Reconstruction from 5 projection images resulted in the same pattern in both reconstructed and original frame. Upon comparing Figures 25, 27, and 29, it can be concluded that increasing the number of projection images increases performance of reconstruction in general, as expected.

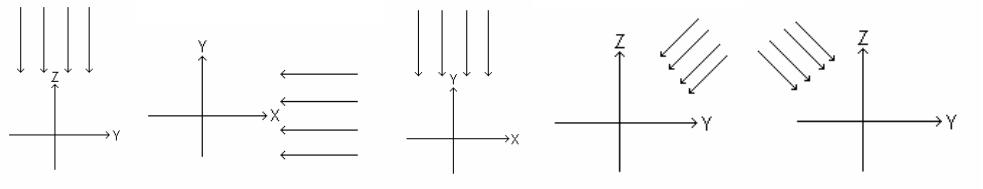

Figure 28 Ray directions indicating the orientation of the projection planes that are used to generate the reconstruction displayed in Figure 29

Figure 29 Video frame showing original data and its reconstruction

Now, we have an idea of the effect of number of projections on the reconstruction. However, the point cloud used in the previous experiment contained a small number of points clustered around some region. Reconstruction of such data is simple and successfully carried out in most cases. The second experiment aims to bring more insight into the observation made in the first experiment and aims to make conclusion derived from the first experiment more concrete. The point cloud data is composed of points that are spread over the finite discrete 3D lattice. Moreover, improvement in the reconstruction of several frames is observed as the number of projections is increased. Five frames of output video files are analyzed with respect to increasing number of projections. Figure 30 illustrates the results of the first simulation of this experiment. The orientations of projection planes used in this simulation are shown in Figure 24. Three projection planes are used in this simulation.

Figure 30 has five cross-sections that display frames 76, 77, 78, 79, 83 of output video file. These frames are chosen by the author so as to provide the reader the reconstruction of cross-sections with various point distribution. Moreover, the first four frames are consecutive so reconstruction in the third dimension can be observed as well. In the cross-section at the top of Figure 30, there are erroneous reconstructions that belong to error type 2. Those erroneous points appear both in the neighborhood of the original points and in the inter-space of point clouds. The same arguments are valid for the other cross-sections as well. We conclude that the number of projections, three, is not sufficient to yield satisfactory reconstructions with this point clouds. Therefore, we keep on simulating by adding another projection to those of previous simulation. Ray directions of this simulation are illustrated in Figure 26. The resultant frames are shown in Figure 31. This time erroneous reconstructions in the inter-space of clouds of points are reduced significantly. However, erroneous reconstructions in the neighborhood of the original points dominate in this simulation. Therefore, simulation continues by adding another projection. Ray orientations are shown in Figure 28. The results are displayed in Figure 32.

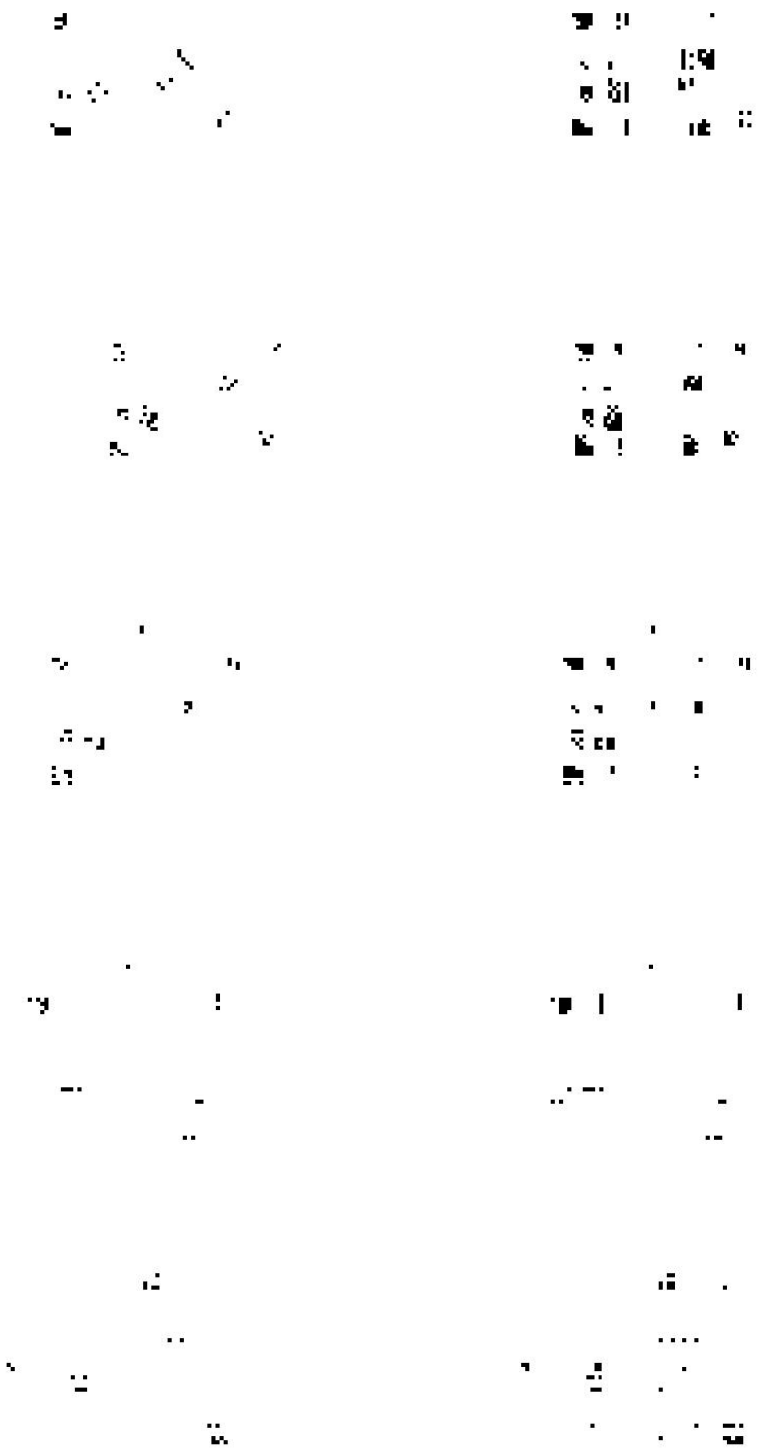

Figure 30 Frames 76, 77, 78, 79, 83 of output video. Three projections whose ray directions are shown in Figure 24 are used in this reconstruction.

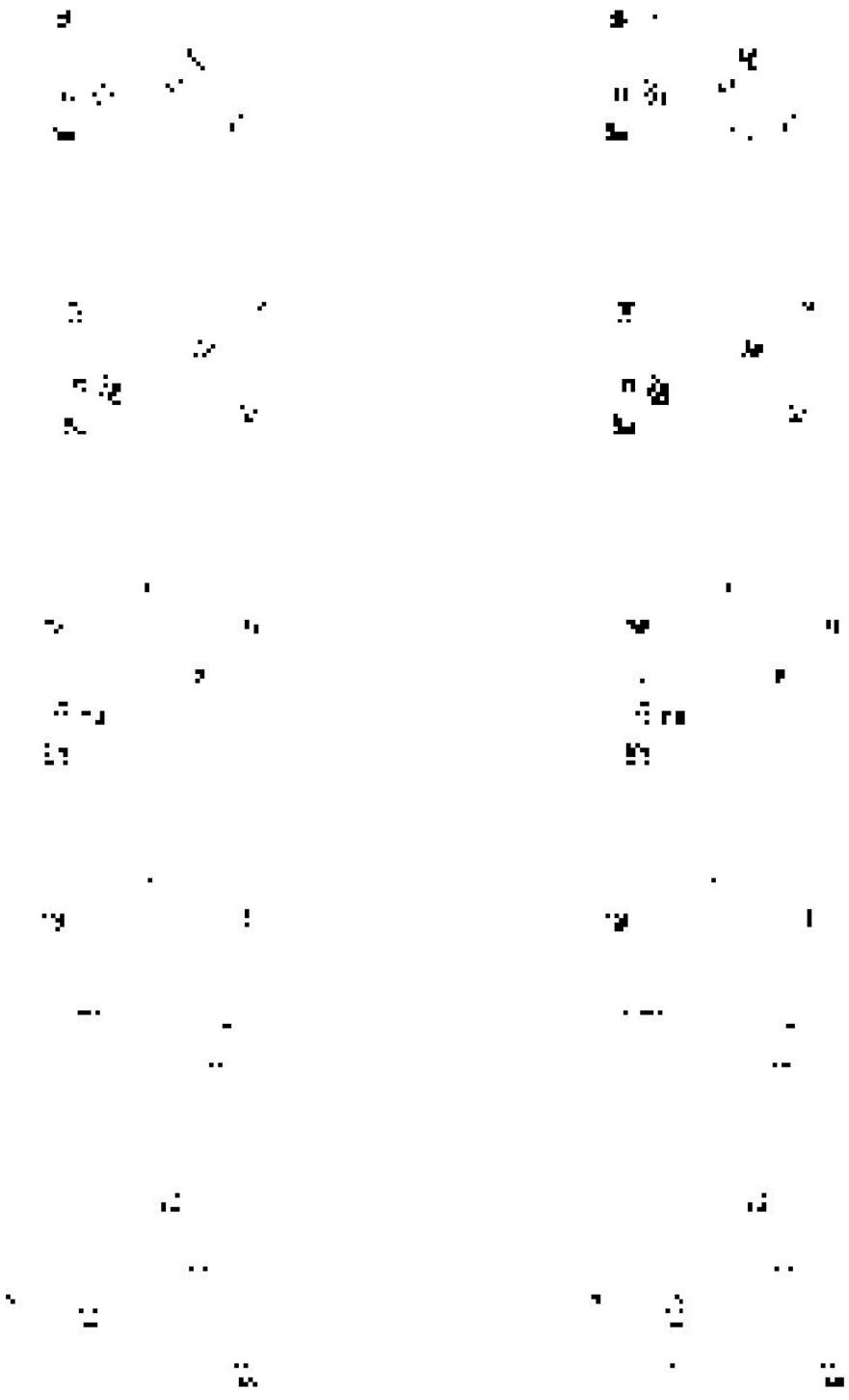

Figure 31 Frames 76, 77, 78, 79, 83 of output video. Four projections whose ray directions are shown in Figure 26 are used in this reconstruction.

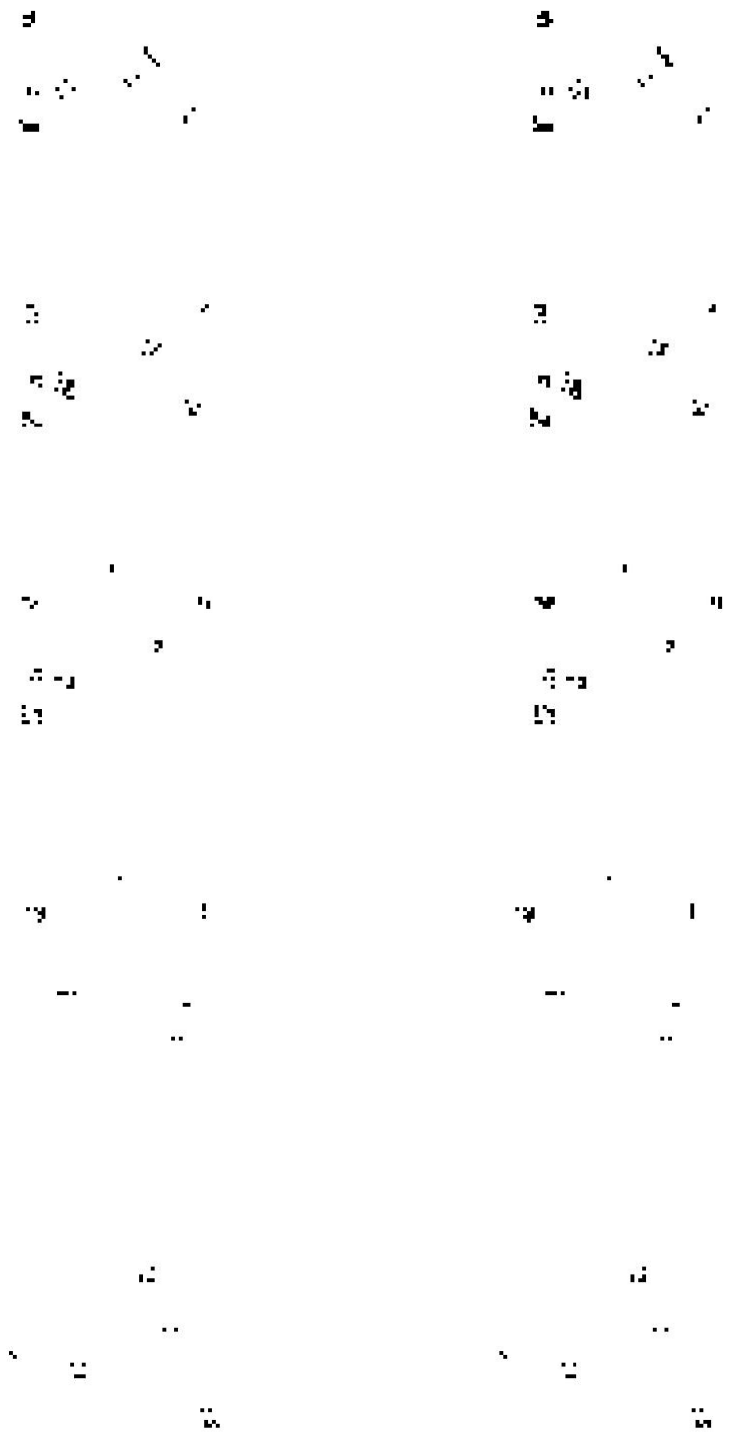

Figure 32 Frames 76, 77, 78, 79, 83 of output video. Five projections whose ray directions are shown in Figure 28 are used in this reconstruction.

The erroneous reconstructions in the inter-space of point clouds are completely disappeared in all frames in Figure 32. Those that are in the vicinity of original points are reduced but not completely gone. So, experiment continues with next simulation by adding an extra projection to see what can be achieved with this data. The ray directions of this simulation are shown in Figure 33. Figure 34 illustrates the resultant frames. Reconstruction is performed using six projections of point clouds. Reconstructed frames and original ones look more alike in this simulation when compared to previous ones. Although, there are a few extra reconstructed points that may be unimportant in some applications, experiment continues with the next simulation using seven projections.

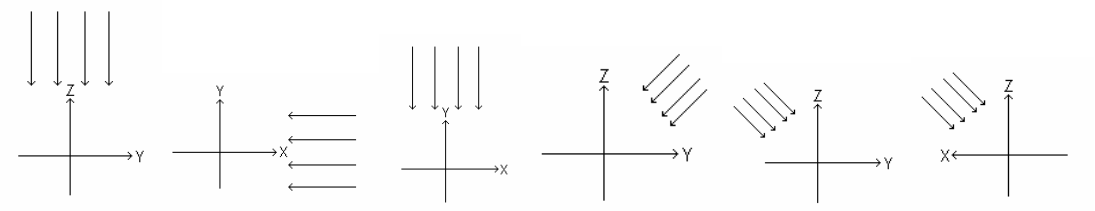

Figure 33 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 34

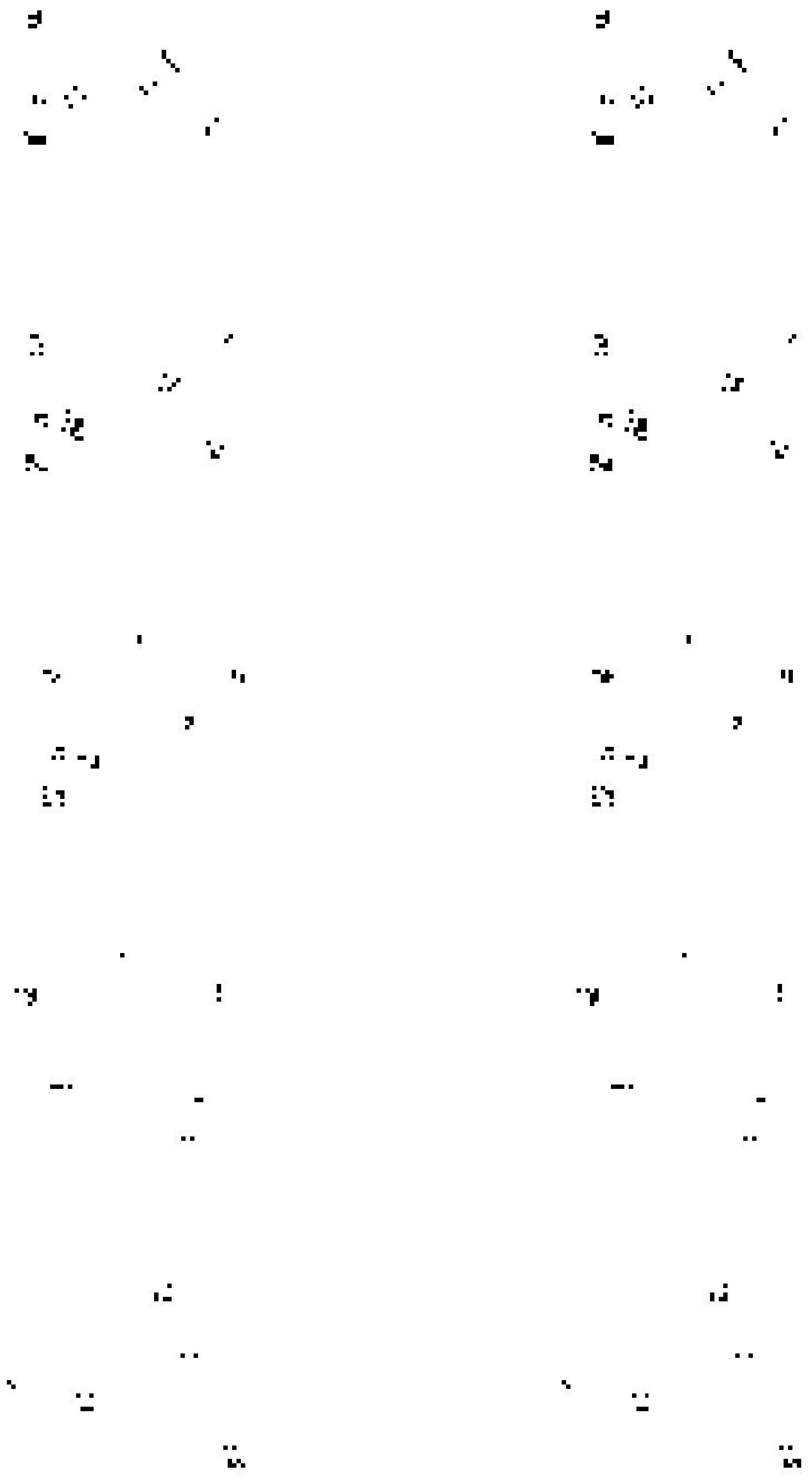

Figure 34 Frames 76, 77, 78, 79, 83 of output video. Six projections whose ray directions are shown in Figure 33 are used in this reconstruction.

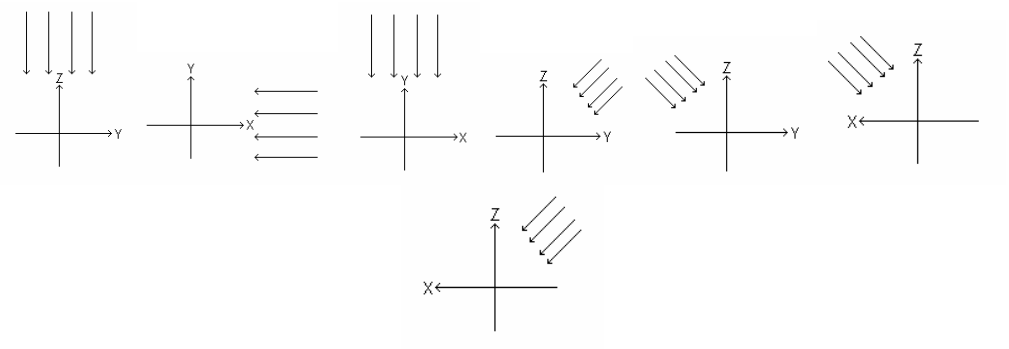

Figure 35 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 36

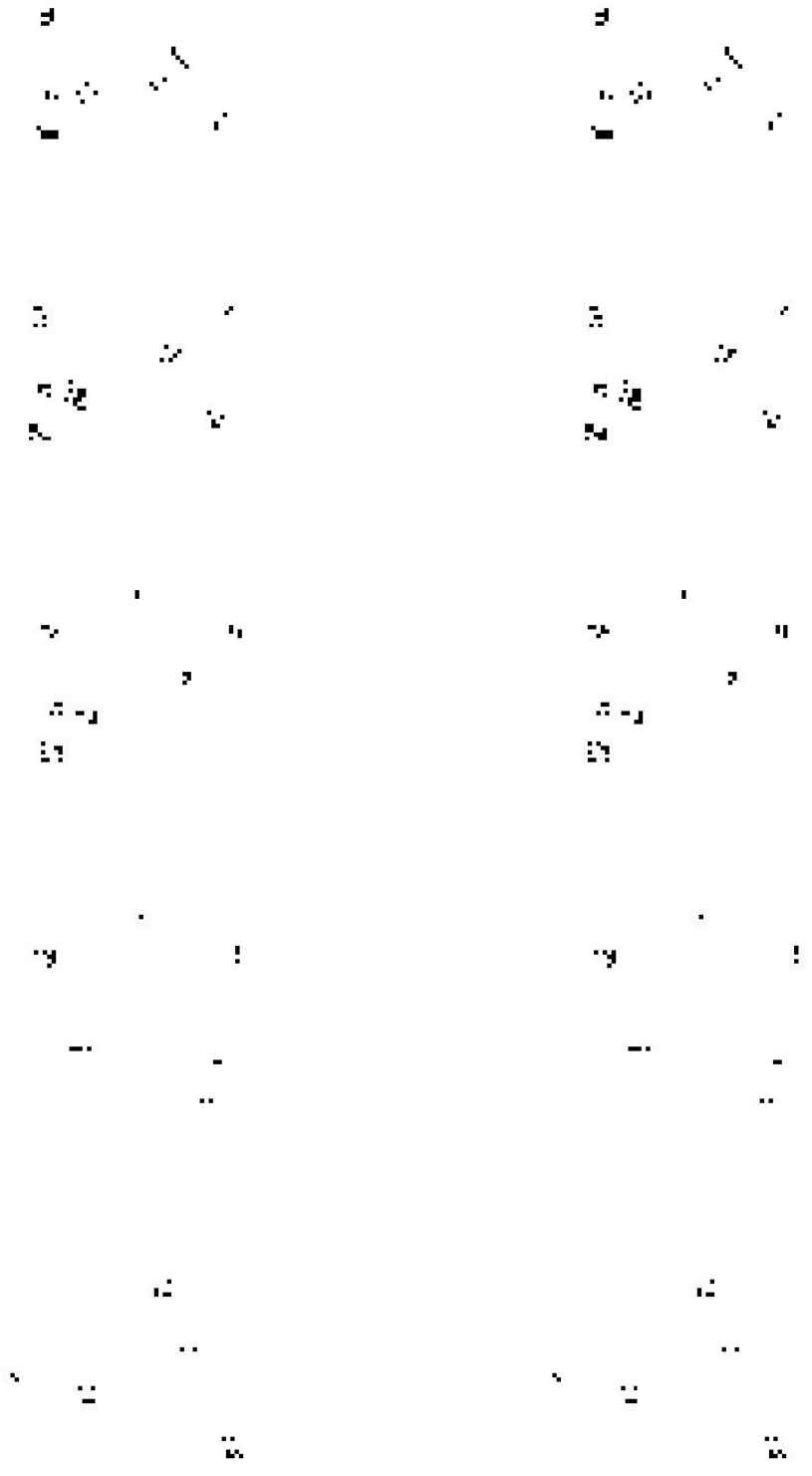

Figure 36 Frames 76, 77, 78, 79, 83 of output video. Seven projections whose ray directions are shown in Figure 35 are used in this reconstruction.

Figure 36 illustrates that frames 79 and 83 achieved perfect reconstruction. The other frames got improvements in their reconstructions. Seven projections have been enough to achieve this result. To investigate ways resulting in perfect reconstructions, experiment continues with simulations involving larger number projections that are formed by adding a new projection to previous ones each time.

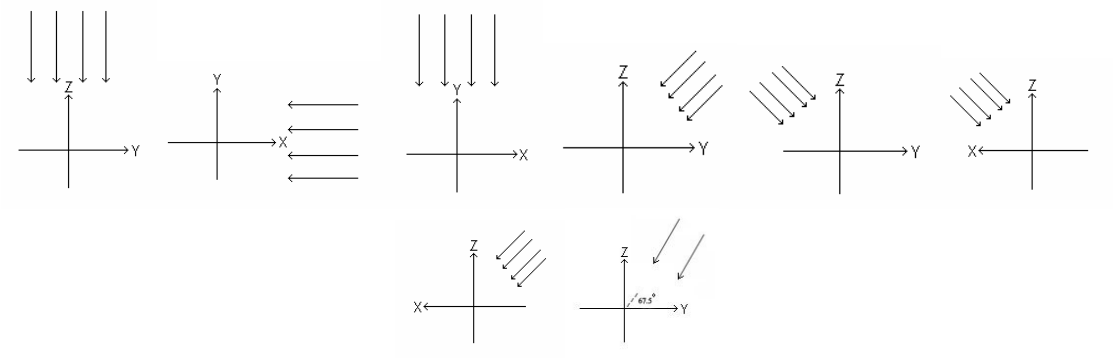

Figure 37 Ray directions indicating orientation projection planes that are used to generate the reconstruction displayed in Figure 38

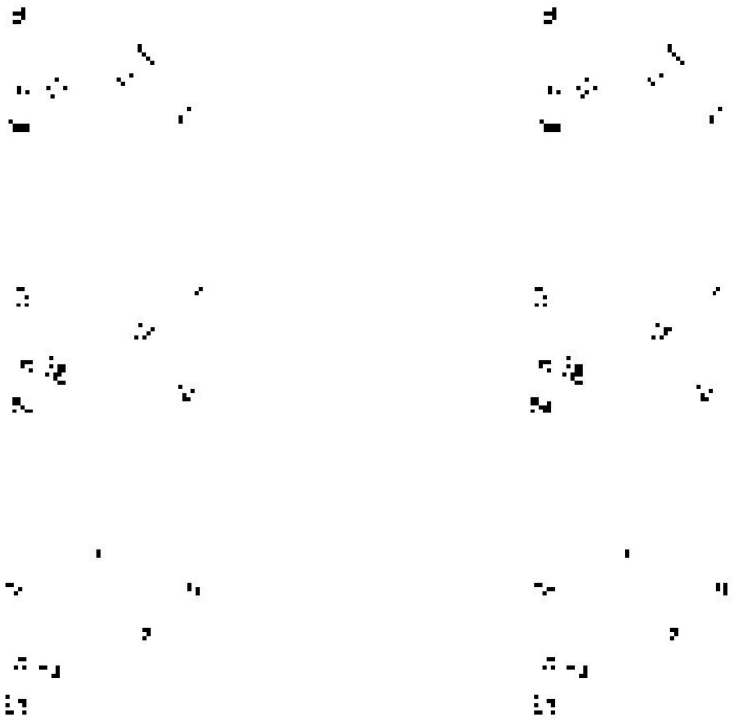

Figure 38 Frames 76, 77, 78 of output video. Eight projections whose ray directions are shown in Figure 37 are used in this reconstruction.

Frames 76, 77, and 78 indicate slight improvements in reconstruction in Figure 39 while frames 79 and 83 remain to illustrate perfect reconstruction. Frames 79 and 83 are not displayed in Figure 38 for this reason. Furthermore, we keep on simulating by adding another projection plane to the planes of previous reconstruction as shown in Figure 39. Frames 76 and 77 in Figure 40 do not provide any improvement when compared to those in Figure 38 whereas frame 78 provided slight improvement in reconstruction. The rest of the frames already got perfect reconstructions. Like before, we add a new projection plane as shown in Figure 41.

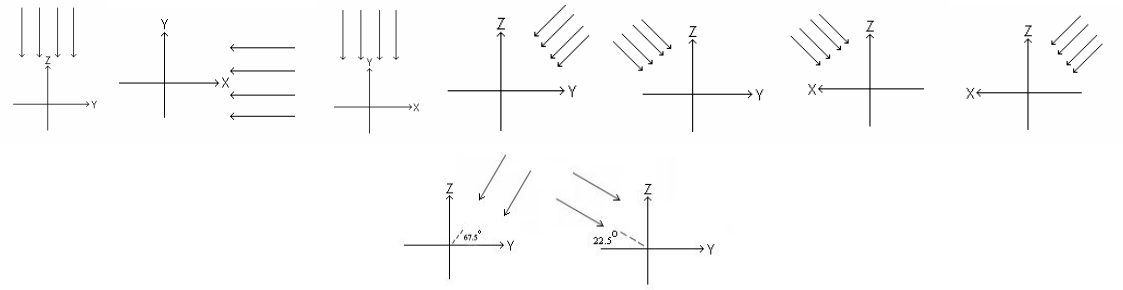

Figure 39 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 40

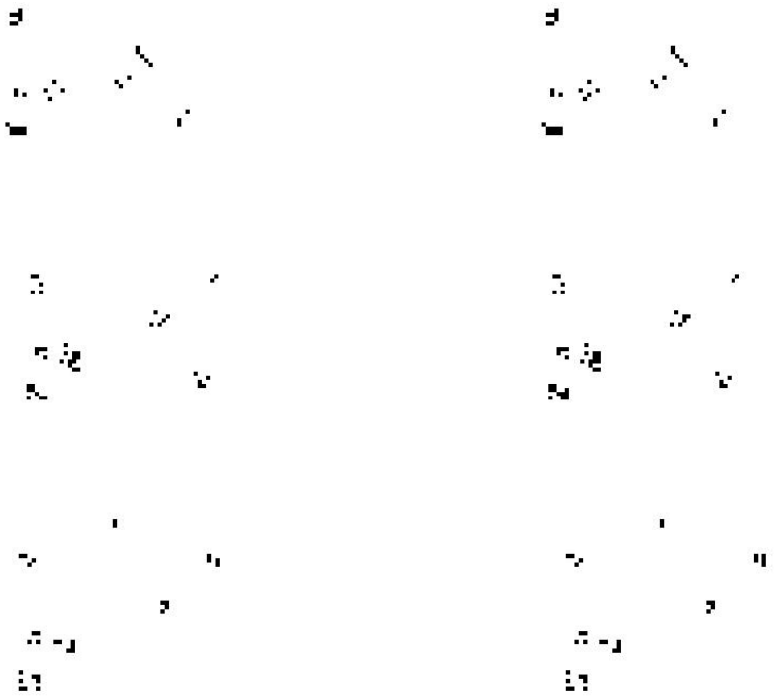

Figure 40 Frames 76, 77, 78 of output video. Nine projections whose ray directions are shown in Figure 39 are used in this reconstruction.

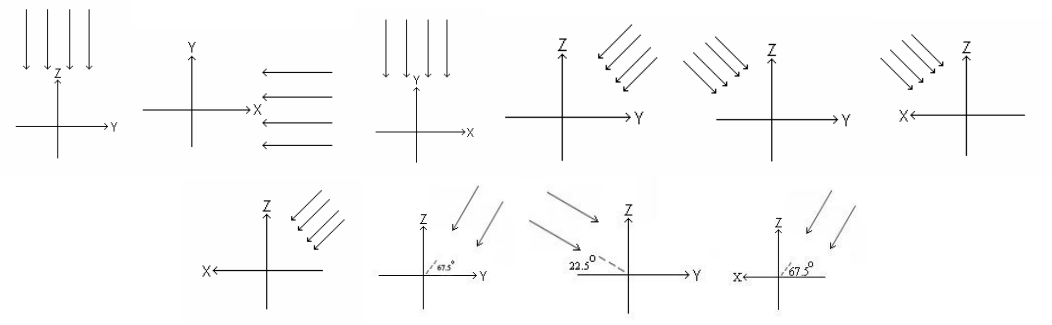

Figure 41 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 42

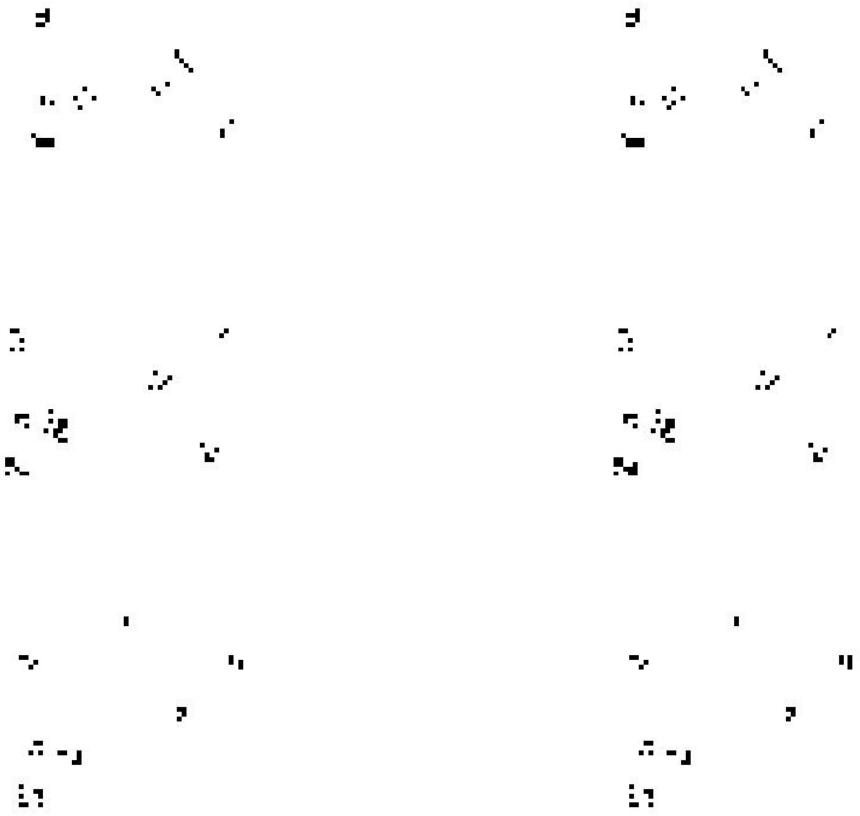

Figure 42 Frames 76, 77, 78 of output video. Ten projections whose ray directions are shown in Figure 41 are used in this reconstruction.

Frames 76 and 78 of this simulation do not demonstrate any improvement while frame 77 performed small improvement as seen in Figure 42. So, we keep adding new projections to already existing projection planes as illustrated in Figure 43. Frames 76 and 78 achieved perfect reconstruction in this simulation using 11 projections shown in Figure 44. Frame 77 kept improving monotonically.

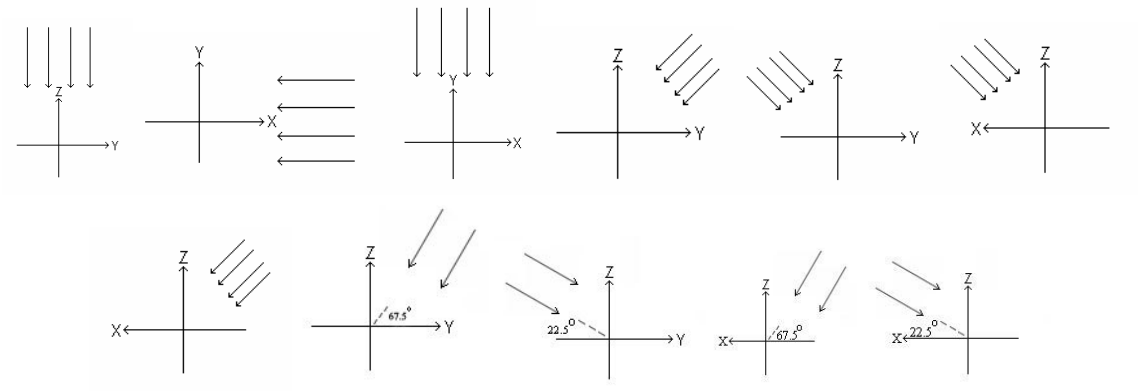

Figure 43 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 44

Figure 44 Frames 76, 77, 78 of output video. 11 projections whose ray directions are shown in Figure 43 are used in this reconstruction.

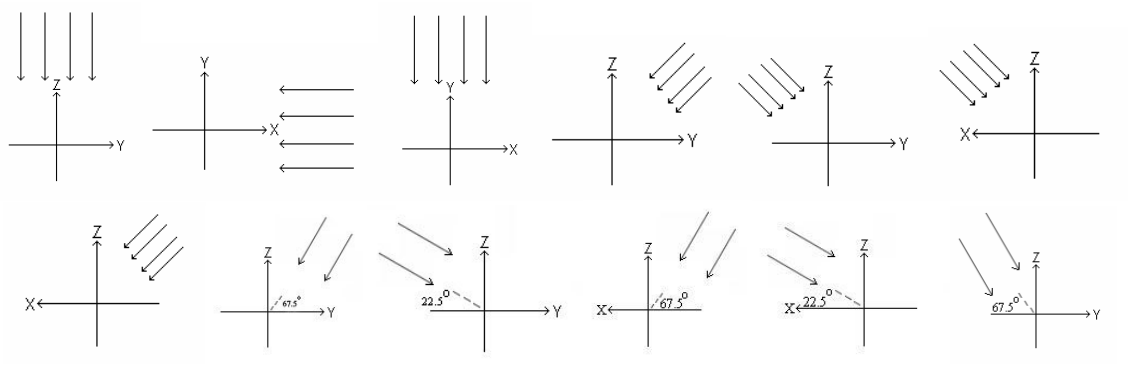

Figure 45 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 46

Figure 46 Frame 77 of output video. 12 Projections whose ray directions are shown in Figure 45 are used in this reconstruction.

By comparing Figures 44 and 46, it is seen that reconstruction of frame 77 does not improve however; the reconstruction improves by using 13 projections as shown in Figure 48.

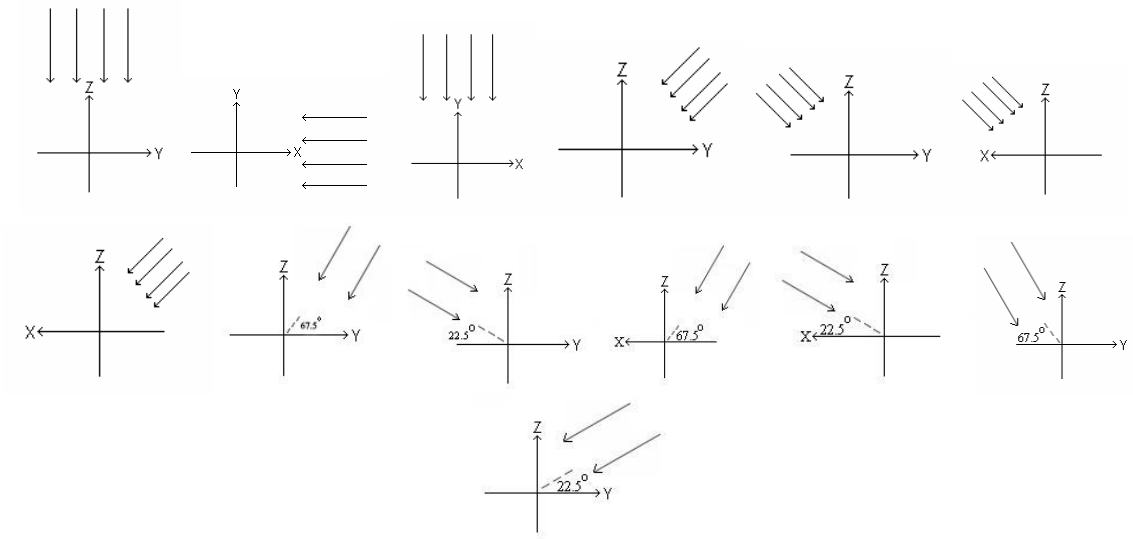

Figure 47 Ray directions indicating orientation projection planes that are used to generate the reconstruction displayed in Figure 48

Figure 48 Frame 77 of output video. 13 Projections whose ray directions are shown in Figure 47 are used in this reconstruction.

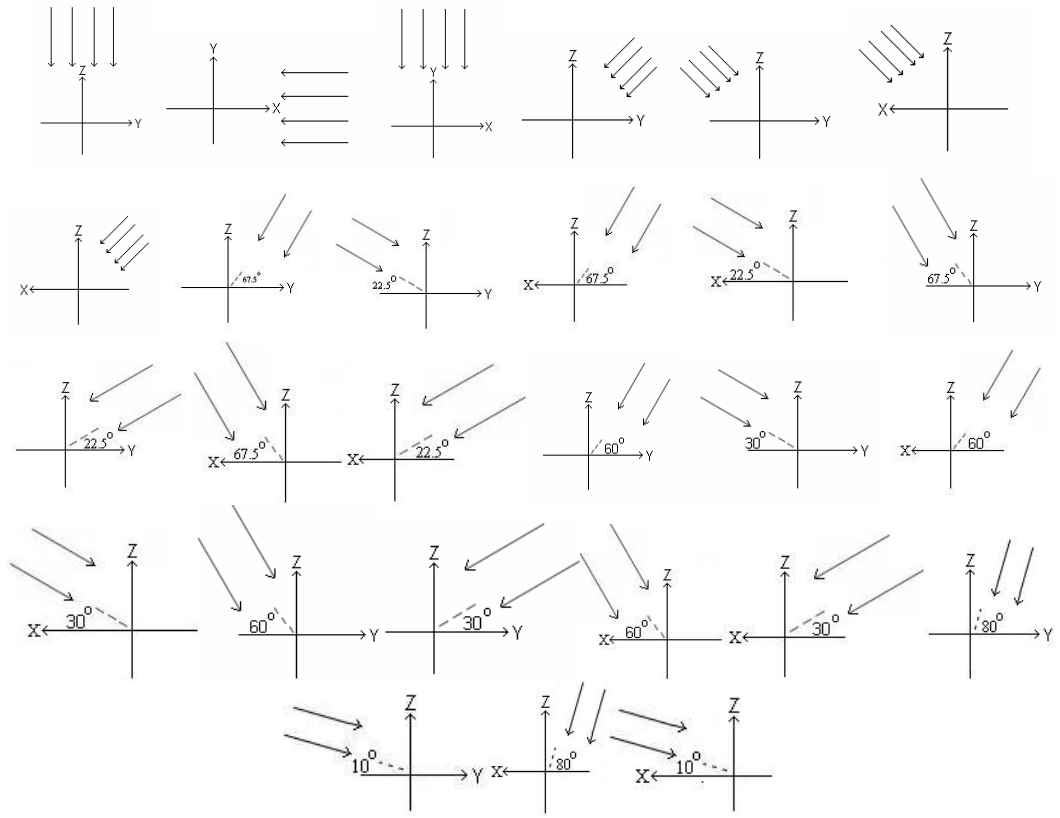

Figure 49 Ray directions indicating orientation of projection planes that are used to generate the reconstruction displayed in Figure 50

Figure 50 Frame 77 of output video. 27 Projections whose ray directions are shown in Figure 49 are used in this reconstruction.

After a sequence of simulations, it has been observed that reconstruction does not improve upon introducing larger number of projections each time. Simulations continued by adding a new projection each time until there are 27 projections shown in Figure 49. Figure 50 demonstrate that the reconstruction is perfect in this frame as well. Therefore, increasing number of projections improves reconstruction, as expected.

It has been observed until here that increasing the number of projections yields more satisfactory reconstructions. Instead of performing simulations using projections whose normal vectors are perpendicular to each other, a new simulation is carried out using projection planes whose ray directions are illustrated in Figure 51.

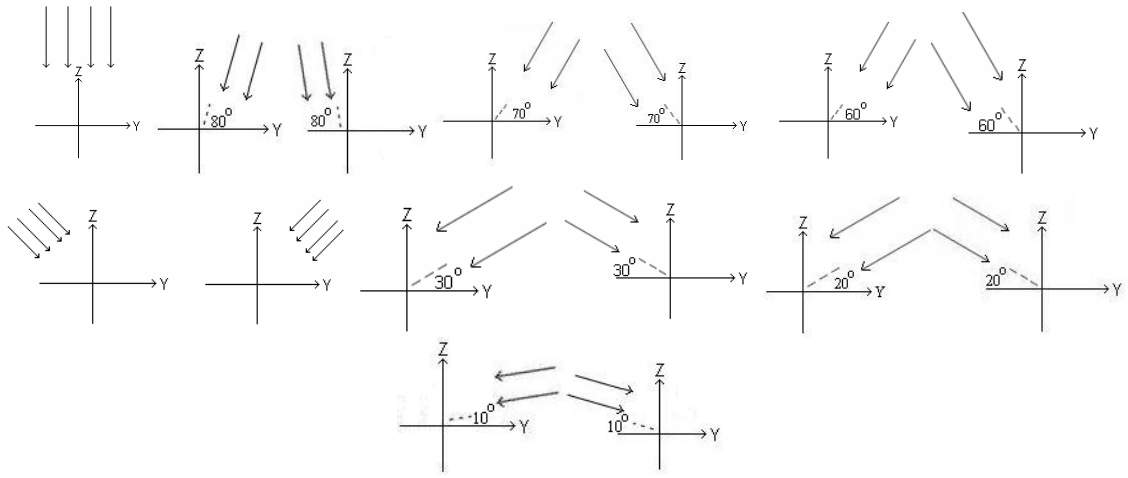

Figure 51 Ray directions used in generating projections. Each ray direction corresponds to a projection plane whose normal lies in the opposite direction of the corresponding ray direction.

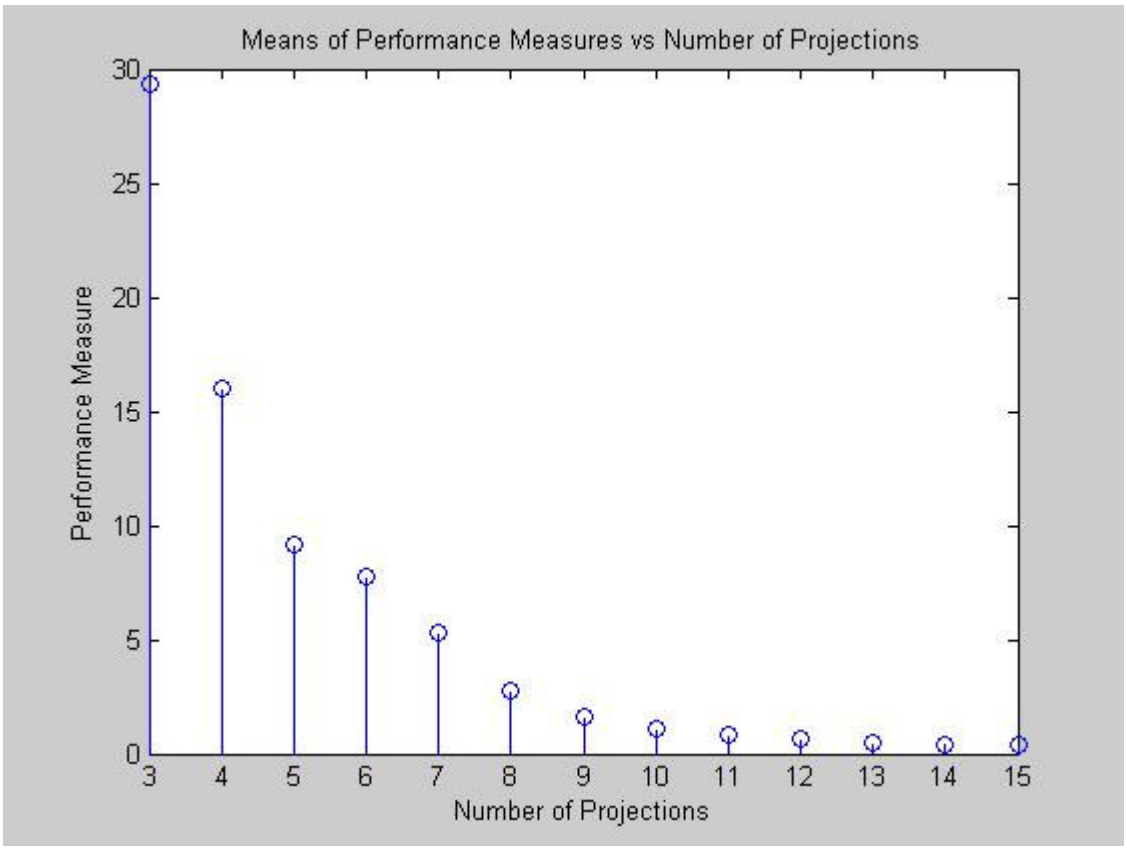

Figure 52 Means of performance measures obtained from 50 simulations each involving 200 points.

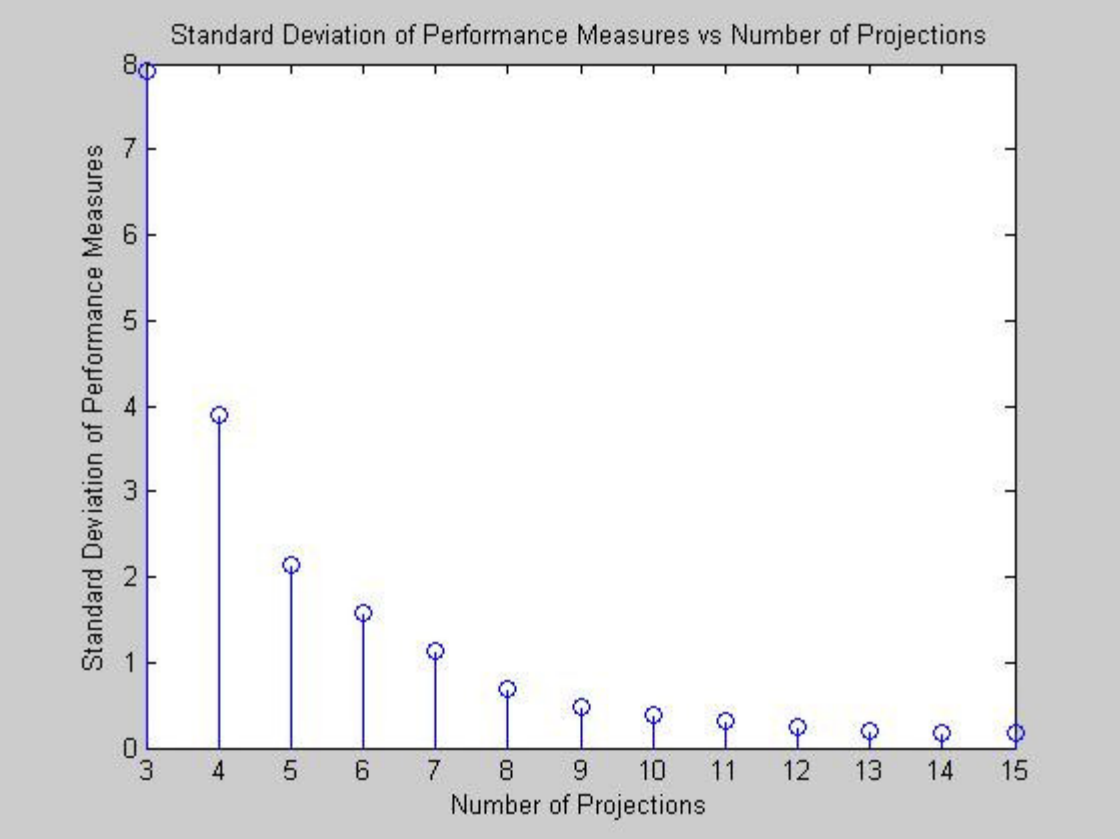

Figure 53 Average standard deviations of performance measures obtained from 50 simulations each involving 200 points.

Figure 52 illustrates means of performance measures obtained from 50 simulations each involving 200 points and the same projections. The number of projections is increased up to 15 projections to investigate the effect of number of projections towards performance measure. The number 3 in the x-axis of Figure 52 corresponds to three projections whose ray directions are the first 3 ray directions in Figure 51. Projections using the first four ray directions in Figure 51 are equivalent to four projections that are formed by adding the fourth projection in Figure 51. Likewise, nine projections are taken using the first nine ray directions in Figure 51 in generating Figure 52. The same is true for Figure 53 except that it illustrates average of standard deviations of performance measures over 50 experiments. There is a considerable decrease in average performance measure in Figure 52 until there are nine projections. Upon increasing number of projections after nine projections, average performance measure decreases at a much slower rate. Therefore, nine projections is the breakpoint of slope of Figure 52 and Figure

53. It can be concluded that reconstructions do not improve greatly after nine projections whose ray directions are as shown in Figure 51.

The next experiment aims to investigate the effect of total number of points in the discrete 3D lattice to the performance of reconstruction. For this reason, simulations using increasing number of projections have been performed by increasing the number of points by 50 each time. The average of performance measures in logarithmic gray level are displayed as an image in Figure 54.

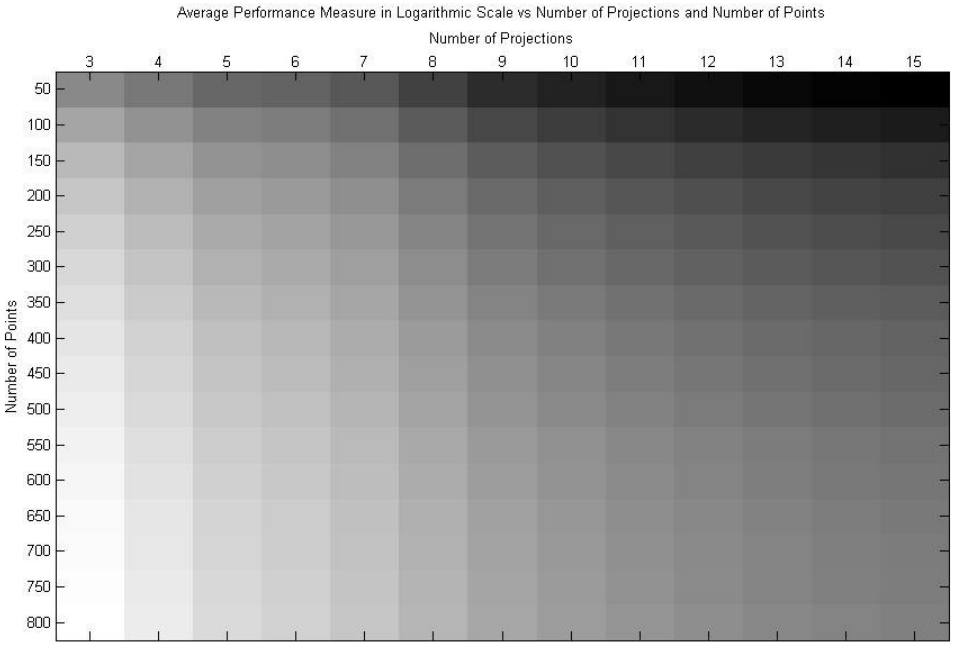

Figure 54 Average performance measure displayed in logarithmic gray level. The x-axis stands for number of projections while y-axis for number of points.

The average is taken over 300 performance measures for each pixel in Figure 54 resulting from 300 simulations. For a fixed number of points, the performance measure decreases until there are nine projections for each number of points. However, the decrease is weak after nine projections for any number of points. The ray directions of the projection planes can be seen in Figure 51. Therefore, it is observed that reconstructions do not improve considerably after there are nine projections.

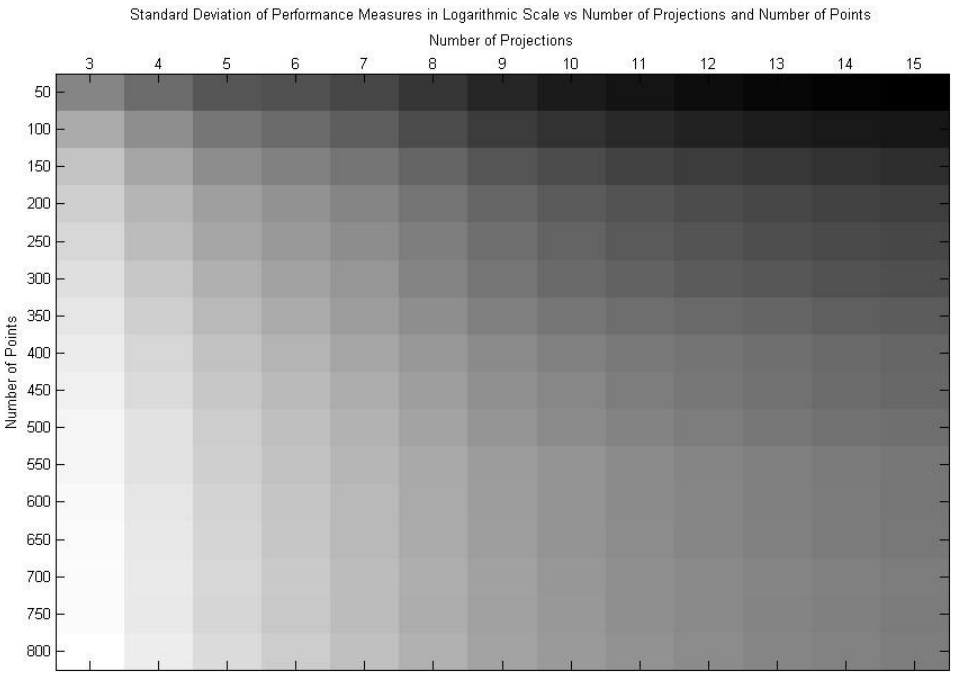

Figure 55 Standard deviation of performance measures is displayed in logarithmic gray level. The x-axis stands for number of projections while the y-axis represents the number of points.

Standard deviation of performance measures is displayed in logarithmic scale in Figure 55. This figure has the same axes as Figure 54. The standard deviation of performance measure indicates the impact of the pattern of point clouds on performance measure and thus reconstruction. The impact is noticeable until there are nine projections for each number of points. After there are 11 projections, the distribution of points has a slight effect on reconstruction.

Chapter 6

Conclusions and Future Work

Inspection of a method to reconstruct point clouds using multi-view orthographic projections is the subject of this thesis. Point cloud generation is performed by using stochastic processes. Generated point clouds mimic microcalcification formation in breast tissue. The point cloud generation includes the Gibbs sampler algorithm. Multi-view orthographic projections of point clouds are generated. Reconstruction from these orthographic projections is performed using volumetric intersection method. It is possible to encounter erroneous reconstructed points. These erroneous points are classified according to their causes and a performance measure based on linear combination is devised. The effect of the number of projections and the number of points to performance of reconstruction are investigated through experiments. Better reconstructions that are more similar to the original point clouds are generated by increasing the number of projections and decreasing the number of points. However, it is reported that reconstructions do not improve considerably as the number of projections are increased after some number. This method of reconstruction is suitable to find locations of original points.

Perfect reconstructions of point clouds are achieved that are exactly the same as the original point clouds in a number of simulations. There are reconstructions that are barely different from the original data along with reconstructions that are noticeably different. However, original point clouds are included in almost all the simulations as expected. The devised performance measure helps to assess reconstructions. Erroneous reconstructed points are classified based on their causes. Three classes of errors are detected; (1) errors due to discrete lines intersecting in more than one voxel as illustrated in Figure 22, (2) errors due to false intersections of discrete lines, (3) errors resulting from unreconstructed points of original data. Each class is given weights according to their degree of importance. Weights are multiplied by number of points in each class and

linear combination of the results gives the performance measure. Therefore, smaller the performance measure, the better the reconstruction. Better reconstructions are carried out as number of projections from different viewpoints is increased. Decreasing total number of points resulted in better reconstructions as well. For this reason, perfect reconstructions occur when number of projections is large enough and total number of points is small enough as expected. In addition, it is also observed that reconstruction does not improve considerably after some number of projections.

The author tried search algorithms to find correct correspondences between orthographic projections of point clouds. Search algorithms failed to find correct correspondences. Therefore, this method based on volumetric intersection is adopted. This method does not require search for correct correspondences neither by machine nor human. Therefore, it is faster than those that require search algorithm and it is less likely to suffer from errors made in finding correct correspondences.

Carr [17] proposes methods to improve accuracy of localizations of breast lesions so that breast biopsy can be done using a needle. His method of localization involves search for lesions performed by a physician. Searching for lesions is followed by triangulization using basic trigonometry. He [17] reported in his paper that physicians cannot differentiate and thus his method fails to reconstruct lesions that are obscured by other lesions. This method reconstructs points even if they are obscured by other points and thus invisible in projection images.

In a future work, the reconstruction of point clouds using real digital mammography images could be considered. Since digital mammography images are not binary images, further processing of these images is required to apply volumetric intersection method. Additional errors are going to be introduced to this method like imperfect parallelization of low-dose X-rays when this method is used with real mammography images. Moreover, noise in these images has to be handled properly.

BIBLIOGRAPHY

1. M. Matsumoto and T. Nishimura, “Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator”, *ACM Trans. on Modeling and Computer Simulations*, 1998.
2. W. H. Press, S. A. Teukolsky, et al., *Numerical Recipes in C++ -- The Art of Scientific Computing*, Cambridge, MA, Cambridge University Press, 2002
3. Pare, E. G., Loving R. O., Hill I. L. *Descriptive Geometry* 3rd Ed. Upper Saddle River, NJ: MacMillan Company, 1965.
4. R. Zhang, P. Tsai, J.E. Cryer, and M. Shah. "Shape from shading: A survey". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 8, pp. 690--706, 1999.
5. A. Blake and C. Marinos, “Shape from texture: estimation, isotropy and moments,” *J . Artificial Intell.,* vol. 45, pp. 323-380. 1990.
6. J. Garding. “Direct Estimation of Shape from texture”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1202–1208, 1993.
7. S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, pp. 721-741, Nov. 1984.
8. F.P.Preparata and M.I.Shamos, *Computational Geometry : An Introduction*, Springer-Verlag, 1985.
9. M. Teillaud, *Towards Dynamic Randomized Algorithms in Computational Geometry*, Springer-Verlag, 1993.
10. J. O'Rourke, *Computational Geometry in C*, Cambridge University Press, 2nd edition, 1998.
11. M. J. Atallah, *Algorithms and Theory of Computation Handbook*, CRC press, 1999.
12. Aldo Laurentini, “The visual hull concept for silhouette-based image understanding”, *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, no. 2, pp. 150-162, February 1994.

13. W. Matusik, C. Bueler, R. Raskar, S. J. Gortler, and L. MacMillan, "Image-based visual hulls", *In SIGGRAPH'00 Proceedings*, pp. 369-374, July 2000.
14. M. Pollefeys, "Tutorial on 3D Modeling from Images", *ECCV 2000*, 2000.
15. T. Jebara, A. Azarbayejani, A. Pentland, "3D Structure from 2D Motion", *IEEE Signal Processing Magazine*, Vol. 16. No. 3, pp. 66-84, May 1999.
16. M. Brady and A. Yuille, "An extremum principle for shape from contour", *IEEE Trans. Patt. Anal. Machine Intell.*, vol.PAMI-6, pp. 288-301, 1984.
17. J. J. Carr, P. F. Hemler, P. W. Halford, R. I. Freimanis, R. H. Choplin, M. Y. M. Chen, "Stereotactic Localization of Breast Lesions: How It Works and Methods to Improve Accuracy", *RadioGraphics*, vol. 21, no. 2, pp. 463-473, 2001.
18. T. Senda and Y. Arimitsu, "Automatic Reconstruction of Solid from a set of the Orthographical Three Views", *IEEE International Conference on Systems Engineering*, pp. 229-233, 1992.
19. Z. Wang and M. Latif, "Reconstruction of a 3D Solid Model from Orthographic Projections", *Proc. Of Intern. Conf. On Geometric Modeling and Graphics*, pp. 75-82, 2003.
20. S. Liu, S. Hu, C. Tai, and J. Sun, "A Matrix-Based Approach to Reconstruction of 3D Objects from Three Orthographic Views", *Proc. The Eighth Pacific Conf. On Comp. Graphics and Applications*, pp.254-261, 2000.
21. N. Ahuja an J. Veenstra, "Generating Octrees from Object Silhouettes in Orthographic Views", *IEEE Trans. Patt. Anal. And Machine Intell.*, vol. 11, no. 2, pp. 137-149, 1989.
22. A. Gelman, J. B. Carlin, H. S. Stern and D. B. Rubin, *Bayesian Data Analysis*, Chapman & Hall, 1995.
23. R. Kasturi, and R. C. Jain, *Computer Vision: Principles*, IEEE Computer Society Press, 1991.
24. H. Stark and J. W. Woods, *Probability, Random Processes, and Estimation Theory for Engineers*, Prentice Hall, 2nd Edition,1994.
25. A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, Addison-Wesley Publishing Company, 2nd Edition, 1994.

26. A. Willis, J. Speicher, D. B. Cooper, “Surface Sculpting with Stochastic Deformable 3D Surfaces”, *Proceedings of ICPR*, vol. 2, 2004.
27. M. K. Bennett, *Affine and Projective Geometry*, Wiley-Interscience, 1995.
28. N. Heredotou, A. N. Venetsanopoulos, and L. Onural, “Image Interpolation Using a Simple Gibbs Random Field Model”, *1995 IEEE International Conference on Image Processing, ICIP'95*, vol I, pp. 494-497, October 1995.
29. P. Srinivasan, P. Liang, S. Hackwood, “Computational Geometric Methods in Volumetric Intersection for 3D Reconstruction”, *Proceedings of International Conference on Robotics and Automation*, vol. 1, pp. 190-195, 1989.
30. D. G. Lowe, “Distinctive Image Features from Local Scale-Invariant Keypoints”, *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.