DISCRETE TOMOGRAPHIC RECONSTRUCTION METHODS FROM THE THEORIES OF OPTIMIZATION AND INVERSE PROBLEMS: APPLICATION IN VLSI MICROCHIP PRODUCTION

OSMAN ÖZGÜR

JANUARY 2006

DISCRETE TOMOGRAPHIC RECONSTRUCTION METHODS FROM THE THEORIES OF OPTIMIZATION AND INVERSE PROBLEMS: APPLICATION IN VLSI MICROCHIP PRODUCTION

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF APPLIED MATHEMATICS

OF

THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

OSMAN ÖZGÜR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF SCIENTIFIC COMPUTING

JANUARY 2006

Approval of the Graduate School of Applied Mathematics

Prof. Dr. Ersan AKYILDIZ
Director

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

Prof. Dr. Bülent KARASÖZEN
Head of Department

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

Prof. Dr. Gerhard Wilhelm WEBER
Supervisor

Examining Committee Members

Prof. Dr. Bülent Karasözen

Prof. Dr. Gerhard Wilhelm Weber

Prof. Dr. Nevzat Güneri Gençer

Assoc. Prof. Dr. Tanıl Ergenç

Assist. Prof. Yeşim Serinağaoğlu Doğrusöz

.

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

Name, Last name: Osman Özgür

Signature:

Abstract

DISCRETE TOMOGRAPHIC RECONSTRUCTION METHODS FROM THE THEORIES OF OPTIMIZATION AND INVERSE PROBLEMS: APPLICATION IN VLSI MICROCHIP PRODUCTION

Osman Özgür
M.Sc., Department of Scientific Computing
Supervisor: Prof. Dr. Gerhard Wilhelm Weber

January 2006, 127 pages

Optimization theory is a key technology for inverse problems of reconstruction in science, engineering and economy. Discrete tomography is a modern research field dealing with the reconstruction of finite objects in, e.g., VLSI chip design, where this thesis will focus on. In this work, a framework with its supplementary algorithms and a new problem reformulation are introduced to approximately resolve this NP-hard problem. The framework is modular, so that other reconstruction methods, optimization techniques, optimal experimental design methods can be incorporated within. The problem is being revisited with a new optimization formulation, and interpretations of known methods in accordance with the framework are also given. Supplementary algorithms are combined or incorporated to improve the solution or to reduce the cost in terms of time and space from the computational point of view.

Keywords: Discrete Tomography, Reconstruction, Inverse Problems, VLSI Microchip Design, Constrained Optimization, Derivative Free Optimization.

Öz

OPTİMİZASYON VE TERS PROBLEMLER TEORİSİNDEN AYRIK TOMOGRAFİ YENİDEN OLUŞTURMA METODLARI: VLSI MİKROÇİP ÜRETİMİNDE UYGULAMASI

Osman Özgür
Yüksek Lisans, Bilimsel Hesaplama Bölümü
Tez Yöneticisi: Prof. Dr. Gerhard Wilhelm Weber

Ocak 2006, 127 sayfa

Optimizasyon teorisi bilim, teknoloji ve ekonomideki ters problemler için anahtar bir teknolojidir. Ayrık Tomografi, sonlu objelerin yeniden oluşturulması ile ilgilenen yeni bir araştırma alanıdır. Bu tezde, Ayrık Tomografinin VLSI çip dizayn ve üretiminde uygulanmasına yer verilecektir. Bu tezde, farklı bir optimizasyon formülasyonu, bu formülasyonun kullanıldığı yeni bir çerçeve, ve bu çerçevede kullanılmak üzere bazı yardımcı algoritmalar sunarak NP-hard olan bu probleme (Ayrık Tomografi) bir çözüm buluyoruz. Yeni yaklaşımımız, diğer yeniden oluşturma metodlarının, optimizasyon tekniklerinin, optimum deneysel dizayn metodlarının da kullanılabileceği modüler bir yapıya sahiptir. Problem yeni bir optimizasyon formülasyonu ile incelenmiş ve ayrıca bilinen bazı diğer çözüm metodlarının da bu yeni çerçevede kullanımı değerlendirilmiştir.

Anahtar Kelimeler: Ayrık Tomografi, Ters Problemler, VLSI Mikroçip Dizaynı, Koşullu, Türevsiz Optimizasyon.

Acknowledgment

I would like to express my gratitude to all those who supported me in any means to complete this thesis. I am grateful to my supervisor Professor Dr. Gerhard Wilhelm Weber for giving me the possibility to do the necessary research work, for guiding me through my graduate study and his valuable suggestions and encouragement in all the time of writing this thesis. I thank him for his careful examination of this thesis and for many valuable contributions.

Special thanks to Professor Dr. Bülent Karasözen for supporting me with his suggestions and helping me in developing my knowledge throughout my studies in the Scientific Computing area. The greatest share in the preparation of my knowledge was met by his valuable lectures.

I would like to thank to the members of my committee, who have given encouragement and suggestions in the progress of this thesis, Assist. Prof. Yeşim Serinağaoğlu Doğrusöz , Assoc. Prof. Dr. Tanıl Ergenç, and Prof. Dr. Nevzat Güneri Gençer.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The word *tomography* is derived from the combination of the Greek words *tomos* (meaning **slice**), and *graphia* (meaning **describing**). So, *tomography*, as a word can be considered as 'reconstruction from projections'. In other words, reconstructing the inner structure of the object just using cross sections without scattering or damaging the object. The word *"object"* here, may be a function, or a mathematical model of a real object. So, *tomography* can be thought of as a reconstruction of a function from its line or hyperplane integrals, which is an *inverse problem* [45].

The process of obtaining the *density distribution* within the an object from multiple *projections* (i.e., images from different angles) is called *reconstruction*. Because, only the *projections* (images obtained from different angles) are known, and we want to obtain the *density distribution*. A known example from brain tomography consists of obtaining some cross section images and trying to reconstruct the actual structure as in Figure 1.1:

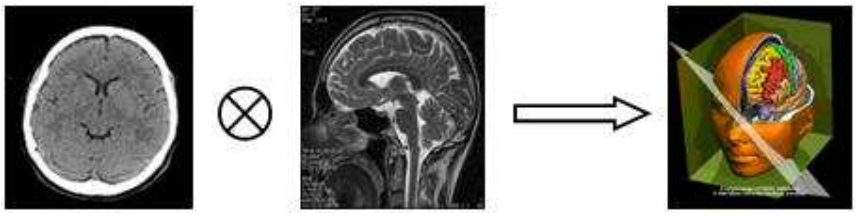

Figure 1.1: 2D cross section images are used to reconstruct 3D structure [33].

1.1 Technology Used in Tomography

The problem of tomography has been studied by many researchers. Mathematical model of the problem has been developed and many algorithms have been introduced together with their theoretical aspects. Since the process of this reconstruction is hard to solve, in practice, computer help is needed. Therefore, with the computer usage, the problem sometimes makes use of *computerized(or computed) tomography (CT)* or *computer-assisted tomography (CAT)*.

The internal property of the object to be reconstructed, such as density, space dependent attenuation coefficient, molecular or atomic distribution, etc., is generally referred to as *internal distribution*.

The fundamental idea of the tomography depends on the fact that, objects consist of different types of atoms or molecules. So, any object to be investigated has different physical properties, such as *electrical resistivity*, *magnetic flux response*, *conductivity*. Using this fact, instruments have been developed to measure these differences and values of the object. With these instruments, the values such as *electrical resistivity*, *magnetic flux response*, or *conductivity* (according to the instrument used) of the object are measured. Then using the measured values, the object's *internal distribution* is tried to be reconstructed.

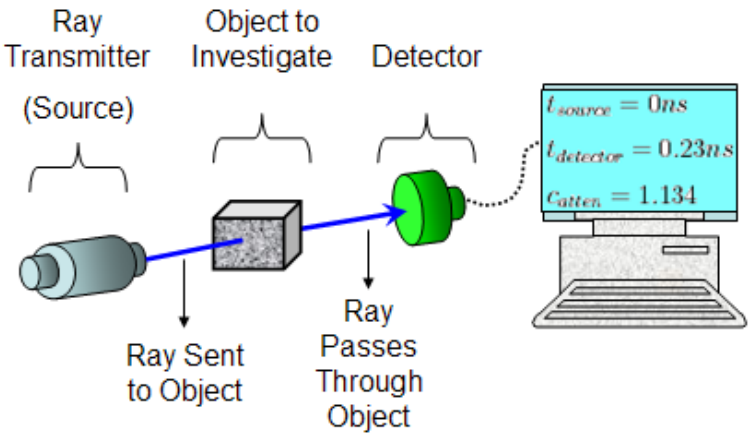

Figure 1.2: Illustration of a simple X-ray measurement.

The physical instruments to obtain the cross sections may differ from X-rays, gamma rays, visible light, electrons or neutrons to ultrasound waves or nuclear magnetic resonance signals. Figure 1.2 illustrates a simple X-ray measurement.

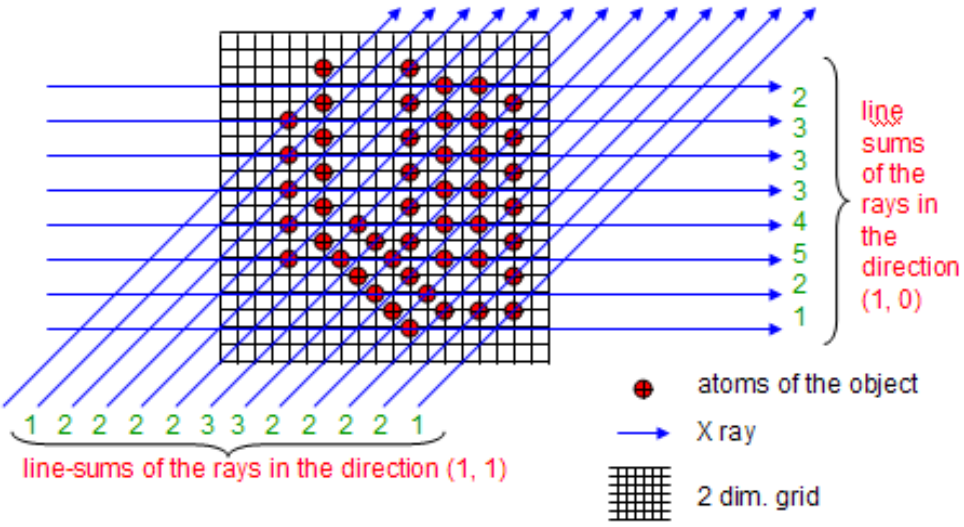

Figure 1.3: Illustration of line sums of a 2-dimensional object.

The instruments take measurements from different angles, distances, and powers. The instruments make use of the values such as *latency of the ray from source to detector*, *attenuation values* and the physical properties of the object to output cross sectional values. Using these information, *the atomic or molecular density of the object along a ray (line) can be obtained*. Here, the *density* may be the inverse of the *ray's latency*, *the number of atoms each ray touched*, etc. The density along a line is called ***line integral*** or ***line sum***. In Figure 1.3, the line sums of a 2-dimensional object is illustrated. In the figure, the red points represent the particles of the object, and the *line sums correspond to the "number of atoms each ray touched or hit"*. Using electron microscopy, with nowadays technology, it is possible to count the number of atoms in each atomic column of a crystal, in the case there is only one type of atom ([30, 48]).

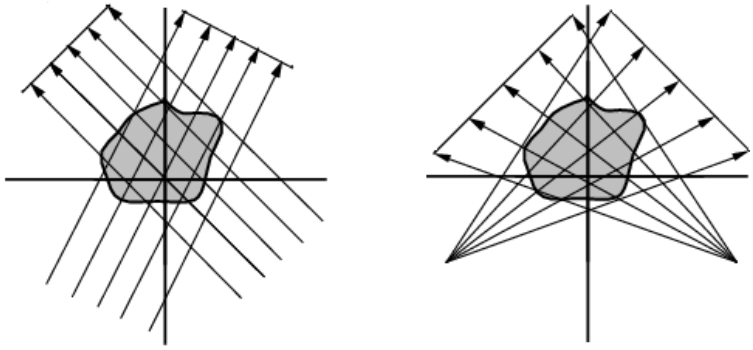

Figure 1.4: Types of beam geometries: *Left*: parallel beam, *Right*: fan beam [45].

Here, line integrals or sums can be obtained in two different beam geometries: *Parallel beam* or *fan beam* geometries are used to obtain the line integrals or sums, as illustrated in Figure 1.4.

The tomographic reconstruction problem is concerned with the reconstruction of an object from a set of line integrals or line sums through the object. After obtaining the projections (line sums or integrals) from different angles, these are used to reconstruct the original object's density distribution or atomic (or molecular) structure of it.

Mathematically, *the object corresponds to a real-valued function* defined over a subset of n-dimensional space, and the so-called *reconstruction* problem, is to reconstruct the function from its integrals or sums over subsets of its domain. The mathematical model and its correspondence to this context will be given in Section 2.1.

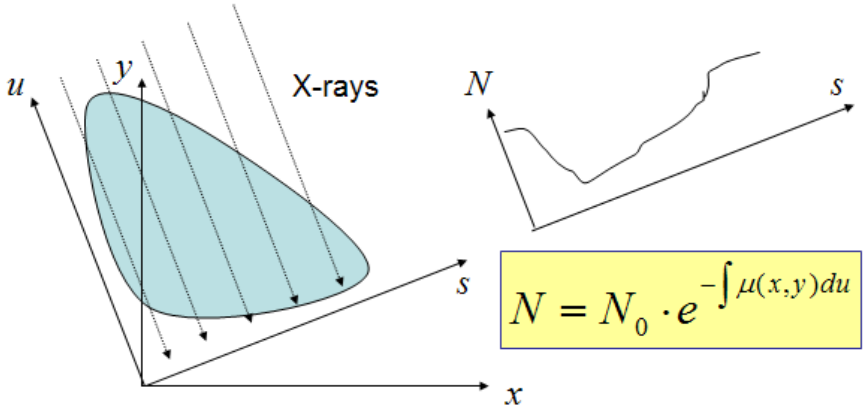

Figure 1.5: Line sums of a 2-dimensional object [33].

According to the probe of the region, there are mainly 2 types of tomography. If the probe is *outside* the object, then this type of tomography is called *transmission computerized tomography (TCT)*. If the probe is inside the object to be reconstructed, then this type of tomography is called *emission computerized tomography (ECT)*. The *ECT* has also 2 variants: *SPECT (single particle ECT)* where the radiation along a half line is detected, and *PET (positron emission tomography)* in which radiation emitted along opposite direction is detected. There are also other types of tomography [45], such as *reflection tomography* [28], *electric impedance tomography* [52], *biomagnetic imaging* [26], and diffraction tomography [22, 23].

1.2 Applications of Computerized Tomography

In general terms, computerized tomography is used in geology, geodesics, medical imaging, diagnostic medicine, industrial manufacturing, electron microscopy, crystallography, combinatorics, computer vision, and many other areas. For examples of applications of computerized tomography, the reader is referred to [23], and applications of discrete tomography can be found in [25].

1.3 Difficulties of Computerized Tomography

Although today's technology offers more accurate imaging and high computation power, there are still difficulties to be solved in most practical applications of computerized tomography.

Remark 1.1. Here, by *"computational complexity of the problem"*, it is meant that, *"computational complexity of the algorithms that solves the problem"*. Computational complexity of an algorithm can be considered roughly as ***the time it takes the algorithm to solve the problem***. This computation time is usually measured by *a function of the algorithm's input length.* Mostly used functions as a computational complexity are *polynomial, exponential, logarithmic* functions. If an algorithm finds the solution of the problem in $n^2 + n$ steps where n is the input length, then it is said that its computational complexity is *polynomial of order 2.* As input length (data) given to an algorithm increases, the computation of a solution will increase according to the computational complexity function of the algorithm. For a detailed description of *computational complexity* and *NP-completeness*, you can refer to the excellent source [15].

One of the important difficulties is that, in most real life tomography problems (such as reconstruction), the problem is *NP-complete* or *NP-hard* [25] (Remark 1.1). Roughly speaking, *NP-complete* or *NP-hard* problems are the class of problems whose exact solutions require too much time. Therefore, finding an exact solution becomes infeasible in practice because of the high computational complexity of the algorithms. This infeasibility is finding a solution in a reasonable time with less storage requirements of computers.

Increasing the number of projections to be taken can yield better solutions, but the trade-off in this point is that, the intensive radiation of the rays may damage the object which is being investigated, or it would cost time. Especially in non-destructive testing, reverse engineering of fragile materials, and more importantly, investigation of human body with harmful rays or instruments,

this becomes an issue.

As it will be explained in Section 2.2, even the problem of reconstruction can be expressed as a linear system of equations, the problem is an inverse problem. Moreover, like most of the inverse problems suffer from being ill-conditioned, reconstruction with the linear system of equations also suffers from this aspect. For inverse problems, some alternative solutions of them, and solutions for the ill-conditioned problems can be found in [1].

1.4 Discrete Tomography: Intuitive Definition

When the object to be reconstructed is considered to be a real-valued function defined over a subset of n-dimensional space, the domain of the function is said to be *continuous* if it contains all points in a region. Similarly, the range of the function is *continuous* if it can have any value in an interval.

The tomographic inversion problem may be continuous or discrete. In *continuous tomography*, both the domain and the range is continuous, however, in *discrete tomography*, the domain of the function can be either discrete or continuous, but the range is a finite subset of real numbers.

Assume, an unknown function f which has a domain either discrete or continuous, and has a discrete range whose values are known. The main problems of *discrete tomography* are about determining the function f from weighted integrals or line sums. So, the problem is *inverse*, where only the values at some points are known and even the function f is not known, we try to construct either f (if it is possible), or an approximate one.

1.5 Brief History of Discrete Tomography

The name *discrete tomography (DT)* is due to Larry Shepp in 1994. DT has its own mathematical theory based mostly on discrete mathematics and has

connections with combinatorics and geometry.

Many problems of DT were first discussed as *combinatorial problems* between 1950 and 1960. In 1957, Ryser [46] published a necessary and sufficient consistency condition for a pair of integral vectors of a $(0-1)$-matrix, which is a special case of Lorentz's result [34]. First reconstruction algorithm is due to Ryser [46, 47], with a reconstructive algorithm. The 2-dimensional case on $(0-1)$-matrices has been studied by him and the concepts related to variances and invariants, namely, *interchange operators*, *interchange operations*, and *structure matrix* have been introduced by Ryser [47].

Some theoretical studies have been performed for the three main problems of DT, namely, consistency, uniqueness, reconstruction [4, 7, 6, 13, 14, 17, 25, 57]. These are studied with their algebraic solutions and computational complexities.

Theoretical results have been found for the reconstruction problem subject to some constraints such as (non-)orthogonality of projections [7, 6], limited number of projections [27, 31, 32], horizontal/vertical convexities [3], polyominoes [2, 8, 54], connectedness, different geometries of beam scans. These are studied with their algebraic and approximate solutions and computational complexities.

Some studies are on applications of DT results to medical and industrial fields. These applications include medical imaging (e.g., angiography of heart chamber [42]), computer vision, pattern recognition, electron microscopy and many other applications [53, 55, 56].

1.6 Objective and Scope of the Thesis

This thesis focuses on the application of Discrete Tomography in **VLSI (Very Large Scale Integrated Circuit) microchip design**. In VLSI technology, the microchips are printed on a silicone material. The silicone material should be very homogenous and must not contain any holes or impurities. Therefore, the density distribution and the atomic structure of the silicone material should be investigated *before* production of VLSI microchip. Moreover, *after* the pro-

duction, the quality of the product may also be required. For these two cases, the advances in Discrete Tomography can be used. In this thesis, **reconstruction** of the *density distribution (atomic structure)* of the silicone material will be investigated, treating as any object. Here, *X-rays* with *parallel beam* geometry will be used and the line sums are considered to be *number of particles* along a ray. However, there are constraints on this problem. Some of these are: the number of projections should be minimal, to prevent damaging the object from the intensive X-ray beams, the solution must be found in a reasonable time and space limit, and the solution must reveal as much information as possible of the material investigated.

1.7 Outline of the Thesis

The organization of this thesis will be as follows:

In Chapter 2, the mathematical model and the main definitions will be introduced. These are presented with the notation presented in G. T. Herman and A. Kuba's book ([25]). In Section 2.2, the main problems of Discrete Tomography will be given with its mathematical definition.

In Chapter 3, some of the solutions of reconstruction problems have been given. In each section, a different solution from the literature is introduced. In Section 3.9, I comment on the studies presented throughout the chapter.

In Chapter 4, some information on VLSI microchip production will be given. Section 4.3, introduces the application areas of Discrete Tomography in VLSI microchip production.

Finally, in Chapter 5, I present my solution proposal and the experimental results of it. In Section 5.3, a small comparison with my proposed method and some of the existing methods, in terms of reconstruction results will be given.

Chapter 2

Mathematical Model and Foundations

2.1 Definitions

Let $\mathbb{Z}$ stand for the set of integers, and $\mathbb{N}_0$ denotes the set of natural numbers including 0. The following definitions are generally used in literature with different notation [25].

Remark 2.1. In this chapter and in Chapter 5, the notation presented here will be used. However, in Chapter 3, the works of the researchers will be presented with their notation.

Definition 2.2. Discrete sets F, $F \subseteq \mathbb{Z}^d$ which are *finite* subsets of integer vectors are called ***lattice sets***.

Definition 2.3. The embedding set $\mathbb{Z}^d$, called a ***lattice***, or a rectangular subset of it containing a given lattice set F can be considered as a regular grid of points or positions. The latter ones are also called ***cells***.

Remark 2.4. In DT, we work on d-dimensional Euclidean space. In Euclidean space, a **lattice** is defined as the set of all linear combinations with integer coefficients of a fixed set of d linearly independent vectors. Since any such lattice is isomorphic to the integer lattice $\mathbb{Z}^d$ under a nonsingular transformation, in DT it is enough to study the case of the integer lattice set $\mathbb{Z}^d$ ([25]).

Definition 2.5. ***Lattice directions*** are nonzero vectors v in the lattice $\mathbb{Z}^d$, but over the field $\mathbb{Q}$ of rational numbers, which implies $v \in \mathbb{Q}^d$. A finite sequence

of distinct lattice directions will be denoted by D, hence, for some $q \in \mathbb{N}$, $q \geq 2$, these are $v_j \in \mathbb{Q}^d$, $(j = 1, ..., q)$ such that

$$D = (v_1, v_2, \ldots, v_q). \tag{2.1.1}$$

Definition 2.6. A ***lattice line*** l is parallel to a vector v_k where v_k is the k^{th} component of a sequence of D of lattice directions and, furthermore, it has a nonempty intersection with the lattice: $l \cap \mathbb{Z}^d \neq \emptyset$.

Remark 2.7. In this context, "$v_k \in D$" will be used instead of "v_k is the k^{th} component of a sequence of D of lattice directions" for simplification.

For a visualization of lattice lines and vectors see Figure 2.1.

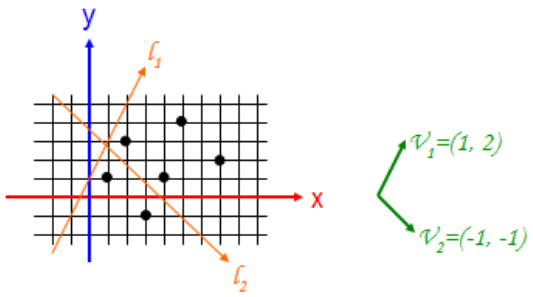

Figure 2.1: Lattice lines l_1 and l_2 in the lattice directions v_1 and v_2.

The set of all lattice lines which are parallel to $v_k \in D$ (Remark 2.7) is denoted by L^k, and $\mathcal{E}$ will be a class of finite sets in $\mathbb{Z}^d$.

The collection of a set of lattice lines determined by D is represented by

$$L = (L^1, L^2, \ldots, L^q). \tag{2.1.2}$$

Definition 2.8. The ***projection of a lattice set in direction*** v_k is $p_F^{(k)} : L^{(k)} \to \mathbb{N}_0$ such that

$$p_F^{(k)}(l) = |F \cap l| = \sum_{x \in l} f(x), \tag{2.1.3}$$

where f is the characteristic function of F, i.e., $f(x) = 1$ if $x \in F$, and $f(x) = 0$ if $x \notin F$.

Definition 2.9. Two lattice sets F and F' are said to be ***tomographically equivalent*** with respect to the directions D, if the following equality is satisfied:

$$p_F^{(k)} = p_{F'}^{(k)} \quad (k = 1, \ldots, q). \tag{2.1.4}$$

Here, the relation between the mathematical model (definitions above) and tomography in real life (Section 1.1) is as follows:

- $\mathbb{Z}^3$ is the lattice corresponding to space,
- F is the object to be reconstructed ($F \subseteq \mathbb{Z}^3$ in real life),
- *Cells* or *points* of F are the particles of the object,
- *Lattice directions*, $D = (v_1, ..., v_q)$ corresponds to the directions from which the projections will be obtained,
- *Lattice line* l, corresponds to the X-rays,
- $p_F^{(k)}$ is the projection (line sum/integral) at direction v_k.

2.2 Main Problems of Discrete Tomography

Now, the three main problems which discrete tomography is concerned with can be defined: consistency, uniqueness, and reconstruction.

Consistency $(\mathcal{E}, L)$

Given: Functions $p^{(k)} : L^{(k)} \to \mathbb{N}_0$ $(k = 1, \ldots, q)$, with finite support (i.e., for every k: $p^{(k)}(l) \neq 0$ for finitely many $l \in L^{(k)}$ only).

The consistency problem is being investigated by Gardner and Gritzmann [14]. A fundamental result there is that*Consistency* $(\mathcal{E}, L)$ is NP-complete for $q \geq 3$. In the case of $q = 2$, the problem can be solved in polynomial time (Table 3.1).

Question: Does there exist an $F \in \mathcal{E}$ such that $p_F^{(k)} = p^{(k)}$ for $k = 1, \ldots, q$?

Uniqueness $(\mathcal{E}, L)$

Given: An $F \in \mathcal{E}$.

Question: Does there exist a different $F' \in \mathcal{E}$ such that F and F' are disjoint and tomographically equivalent with respect to the directions of D?

If a set is nonunique, then it cannot be distinguished by its projections from some other set in $\mathbb{Z}^d$.

The problems of *Uniqueness* $(\mathcal{E}, L)$ and *Consistency* $(\mathcal{E}, L)$ with their relation are discussed in [14]. Kong and Herman [57] proved that in case of $q \geq 3$ the uniqueness of a discrete set can not be decided by simply finding certain patterns of 0's and 1's unlike in the case of $q = 2$. From Ryser [47], it was known that for the case of $q = 2$, existence of certain type of (2×2)-submatrices is equivalent to non-uniqueness.

Reconstruction $(\mathcal{E}, L)$

Given: Functions $p^{(k)} : L^{(k)} \to \mathbb{N}_0$ $(k = 1, \ldots, q)$, with finite support.

Task: Construct a finite set $F \in \mathcal{E}$ such that $p_F^{(k)} = p^{(k)}$ for $k = 1, 2, \ldots, q$.

In this thesis, the emphasis will be on reconstruction. Since from projections, a unique solution can not be reconstructed always, a priori information may be needed in the field of VLSI microchip production. The atomic density of the silicone, approximate number of particles in the material, an estimate of the ingredients of the material, an approximate value for the purity of the material, dimensions and the geometry of the material, etc. can be considered as a priori information in this context.

Suppose that there are given functions $p^{(k)} : L^{(k)} \to \mathbb{N}_0$ $(k = 1, \ldots, q)$ having finite support with cardinalities m_k $(k = 1, ..., q)$. Let $M = m_1 + ... + m_q$. It is clear that if F is a finite set having projections $p^{(1)}, ..., p^{(q)}$ in the directions $v^{(1)}, ..., v^{(q)}$, respectively, then $F \subseteq G$, where G consists of lattice directions $z \in \mathbb{Z}^d$ for which $p^{(k)}(l) > 0$, l being the lattice line passing through z in direction

$v^{(k)}$ $(k = 1, ..., q)$. Because the functions $p^{(k)}$ have finite support, G is also finite, $|G| = N$. Then, the discrete reconstruction problem can be reformulated as the following linear feasibility problem (***if the object is known to be composed of $\bar{d}$ different types of particles, atoms , molecules, pixels, etc.***):

$$\text{find} \quad Px = b, \quad \text{such that} \quad x \in \{0, ..., \bar{d}\}^N, \tag{2.2.5}$$

where $P \in \{0, 1\}^{M \times N}$ and $b \in \mathbb{N}_0^M$. Namely, if the smallest rectangular box containing the finite set to be reconstructed has dimension $n_1 \times n_2$, then, $M = n_1 + n_2$ and $N = n_1 \times n_2$. Hence, M is the number of lattice lines, in this case being parallel to the directions $(1, 0)$ and $(0, 1)$ on which there is at least one element from our discrete set, and N is the total number of points or positions considered in our reconstruction problem.

According to the projections taken we define the matrix P, sometimes referred to as a *view matrix* [19]. The matrix P describes the geometric relation between the points of G and the lattice line l; that is, it specifies which points of G are on a line l. Each equation in (2.2.5) corresponds to a line sum on a lattice line. The vector x represents the set G. The non-negative integral vector b consists of the raywise recorded experimental data, that is, the values of the functions $p^{(k)}$, $(k = 1, ..., q)$. Considering that the surface or tissue to become reconstructed is discretized by cells, this matrix will consist of rows whose components are 1 for any cell where the ray (represented by a row of P) goes through, and 0 if the ray does not meet that cell. More generally, coming from an underlying and discretized continuous problem, P may also have non binary values of distance or values of further physical, biological or chemical dimensions [1].

Example 2.10. (From [53, 55, 56]). Consider the lattice set given in Figure 2.2, which consists of only $\bar{d} = 1$ type of atom or molecule. It is contained in a (3×2)-rectangle, hence, $M = 5$ and $N = 6$. If we choose $D = (v_1, v_2)$ where $v_1 = (1, 0), v_2 = (0, 1)$, then $L^1 = (l_1, l_2, l_3)$, $L^2 = (l_4, l_5)$ and $L = (L^1, L^2)$. Here, we know the projections along 3 horizontal and 2 vertical directions.

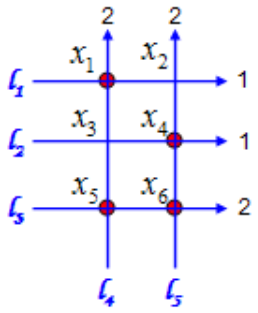

Figure 2.2: Illustration of the projections on two lattice lines in the directions $(1, 0)$ and $(0, 1)$.

For this instance we have the following system of equations

$$\begin{aligned} x_1 + x_2 &= 1 \\ x_3 + x_4 &= 1 \\ x_5 + x_6 &= 2 \\ x_1 + x_3 + x_5 &= 2 \\ x_2 + x_4 + x_6 &= 2. \end{aligned}$$

Here, P, x and b are

$$P = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}, \; x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix}, \; b = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{pmatrix}. \quad \blacksquare$$

Since the object consists of only $\bar{d} = 1$ type of atom or molecule, a solution to the system will be $x = (1, 0, 0, 1, 1, 1) \in \{0, 1\}^6$. This solution tells that, there are atoms or molecules at the locations x_1, x_4, x_5, x_6.

Example 2.11. Consider the lattice set given in Figure 2.3, which consists of

$\bar{d} = 2$ type of atom or molecule. Moreover, assume that the atom shown with big circle contributes 2 to the line sums. This object is also contained in a (3×2)-rectangle, hence, $M = 5$ and $N = 6$. If we choose $D = (v_1, v_2)$ where $v_1 = (1, 0), v_2 = (0, 1)$, then $L^1 = (l_1, l_2, l_3)$, $L^2 = (l_4, l_5)$ and $L = (L^1, L^2)$. Here, we know the projections along 3 horizontal and 2 vertical directions.

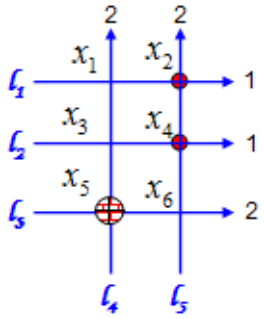

Figure 2.3: Illustration of a 2 valued lattice and its 2 orthogonal projections.

For this instance, the system of equations is the same with the system of equations of the previous example (Example 2.10). However, note that, since the object consists of 2 different types of atoms, the solution x will be a member of $\{0, 1, 2\}^6$. A possible solution therefore will be $x = (0, 1, 0, 1, 2, 0)$. This solution tells us that at locations x_1, x_3, x_6 no particles, at locations x_2, x_4 a particle of type 1, and at location x_5 a particle of type 2 exist.

2.3 Main Theorems

In this section, main results for the 2-dimensional binary case will be given. Here, the problem is restricted to 2-dimensional space and two direction vectors, i.e., the projection function gives the values of the lines in two directions, namely, in x-axis direction and in y-axis direction. In this case, the problem can be modeled by linear algebra. So, we are actually trying to find a binary $(m \times n)$-matrix F, which corresponds to our discrete set, whenever a projection function and two vectors R and S representing the value of row sums and column sums,

are given. The 1's in the binary matrix denote the existence of a tissue (in the case of VLSI microchip design *tissue* will be the atoms or molecules of the silicone material) in that entry and 0's denote nonexistence.

In this section, the fundamentals will be presented for binary case, in $\mathbb{Z}^2$ only. That is, the subjects will be explained for objects, which lie in $\mathbb{Z}^2$ and the objects are assumed to contain only 2 values, i.e. it contains only 1 type of particle ($\bar{d} = 1$).

2.3.1 Fundamentals on Consistency

The consistency problem deals with the problem, given the projection function, p, whether we can find a discrete set F which satisfies p. The solution depends on the number of direction vectors and the dimension of our space. Here, I am going to explain and illustrate the results given in [25].

Definition 2.12. Let $R = (r_1, ..., r_m)$ and $S = (s_1, ..., s_n)$ be nonnegative integral vectors. The class of all binary matrices $A = (a_{ij})$ satisfying the equations

$$\sum_{j=1}^{n} a_{ij} = r_i \quad (i = 1, ..., m), \tag{2.3.6}$$

$$\sum_{i=1}^{m} a_{ij} = s_j \quad (j = 1, ..., n). \tag{2.3.7}$$

is denoted by $\mathcal{U}(R, S)$. The vectors R and S are called the ***row*** and ***column sum vectors*** of any matrix $A \in \mathcal{U}(R, S)$.

Remark 2.13. Here, A is the binary model of the object to be reconstructed, that is, it represents F. On the other hand, P, in Equation 2.2.5 represents the relation between the points of G and the lattice line l. ***Therefore, the view matrix*** P ***presented in the previous section, and the binary matrices*** A ***and*** $\bar{A}$ ***which will be presented in this section should not be confused***.

Definition 2.14. A pair (R, S) of vectors is said to be ***compatible*** if there exist positive integers m and n such that

(i) $R \in \mathbb{N}_0^n$;

(ii) $r_i \leq n$ $(i = 1, ..., m)$, $s_j \leq m$ $(j = 1, ..., n)$;

(iii) $\sum_{i=1}^{m} r_i = \sum_{j=1}^{n} s_j$.

Clearly, if $\mathcal{U}(R, S)$ is not empty, then (R, S) is compatible. Ryser and Gale [46], gave a necessary and sufficient condition under which the class $\mathcal{U}(R, S)$ is nonempty.

Definition 2.15. Consider the matrix $\bar{A}$ in which, for $i = 1, ..., m$, row i consists of r_i 1's followed by $n - r_i$ 0's. A matrix having this property is called ***maximal***.

A maximal binary matrix $\bar{A}$ is uniquely determined by its row sum vector. Let its column sum vector be denoted by $\bar{S}$. Furthermore, let us denote the non-increasing permutations of the elements of R and S, by R' and S', respectively, that is, $r'_1 \geq r'_2 \geq ... \geq r'_m$ and $s'_1 \geq s'_2 \geq ... \geq s'_n$.

Theorem 2.16. *Let $R = (r_1, ..., r_m)$ and $S = (s_1, ..., s_n)$ be a pair of* **compatible vectors**. *The class $\mathcal{U}(R, S)$ is non-empty* if and only if

$$\sum_{j=l}^{n} s'_j \geq \sum_{j=l}^{n} \bar{s}_j \quad (l = 2, ..., n). \tag{2.3.8}$$

Proof. Suppose that $\mathcal{U}(R, S)$ contains binary matrix A. Then, the class $\mathcal{U}(R, S')$ contains a binary matrix A' constructed from A by a suitable permutations of the columns. Now, $\bar{A}$ can be obtained from A' by shifting 1's to the left in the rows of A'. So, it follows that we have Equation 2.3.8. □

Example 2.17. Let (R, S) be given as $R = (2, 3, 4, 1)$ and $S = (4, 2, 1, 2, 1)$, then, $\bar{A}$ would be

$$\begin{matrix} 2 \\ 3 \\ 4 \\ 1 \end{matrix} \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} = \bar{A}.$$

The binary matrix $\bar{A}$ is a requested maximal matrix, since in each row all 1's are shifted to the very left of the matrix. Hence, $\bar{S} = (4, 3, 2, 1, 0)$ and $S' = (4, 2, 2, 1, 1)$.

Example 2.18. Assume (R, S) is given as in Example 2.17. The following observation can be deduced:

$$\begin{gathered} s'_5 \geq \bar{s}_5 \\ s'_4 + s'_5 \geq \bar{s}_4 + \bar{s}_5 \\ s'_3 + s'_4 + s'_5 \geq \bar{s}_3 + \bar{s}_4 + \bar{s}_5 \\ s'_2 + s'_3 + s'_4 + s'_5 \geq \bar{s}_2 + \bar{s}_3 + \bar{s}_4 + \bar{s}_5. \end{gathered}$$

Hence, by Theorem 2.16, $\mathcal{U}(R, S)$ is non-empty.

Now, with the knowledge of compatible vectors, a reconstruction algorithm ([25]) can be given. This algorithm assumes a pair of compatible vectors (R, S) satisfying (2.3.8) is given, and outputs a binary matrix A, solving the reconstruction problem.

Algorithm 2.19.

Input : Compatible pair of vectors (R, S) satisfying (2.3.8)

Step 1. Construct S' from S by permutation π

Step 2. Let $B = \bar{A}$ and $k = n$

Step 3. *while*$(k > 1)$

{

while$(s'_k > \sum_{i=1}^{m} b_{ik})$

{

$j_0 \leftarrow \max_{1 \leq i \leq m} \{\ j < k | b_{ij} = 1, b_{i,j+1} = ... = b_{ik} = 0\ \}$

$i_0 \leftarrow index(j_0)$

$b_{i_0 j_0} \leftarrow 0$ and $b_{i_0 k} \leftarrow 1$ *(i.e., shift the 1 to right)*

}

$k \leftarrow k - 1$

}

Step 4. Construct the matrix A from B by permutation π^{-1} of the columns [25].

Output : Matrix A

Example 2.20. Let the compatible pair of row and column sum vectors be given as (R, S), with $R = (2, 3, 3, 1)$ and $S = (4, 2, 1, 1, 1)$. The binary matrix A, reconstructed by Algorithm 2.19 will be:

2	1	0	1	0	0
3	1	1	0	0	1
3	1	1	0	1	0
1	1	1	0	0	0
	4	2	1	1	1

Example 2.21. Assume the compatible pair of row and column sum vectors be given as (R, S), with $R = (2, 4, 3, 4, 1)$ and $S = (3, 4, 3, 2, 1, 1)$ are given. Using Algorithm 2.19, a stepwise reconstruction will be as shown in the Figures 2.4, 2.5, 2.7, 2.7:

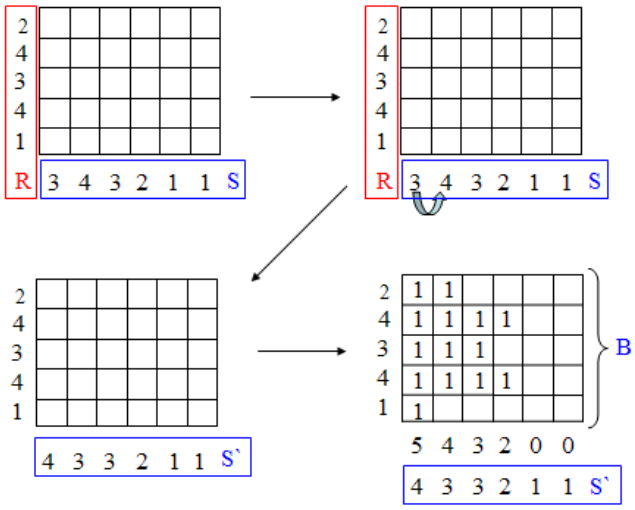

Figure 2.4: Illustration of the reconstruction progress with Algorithm 2.19 [33]; (Part 1 of 4 of Example 2.20).

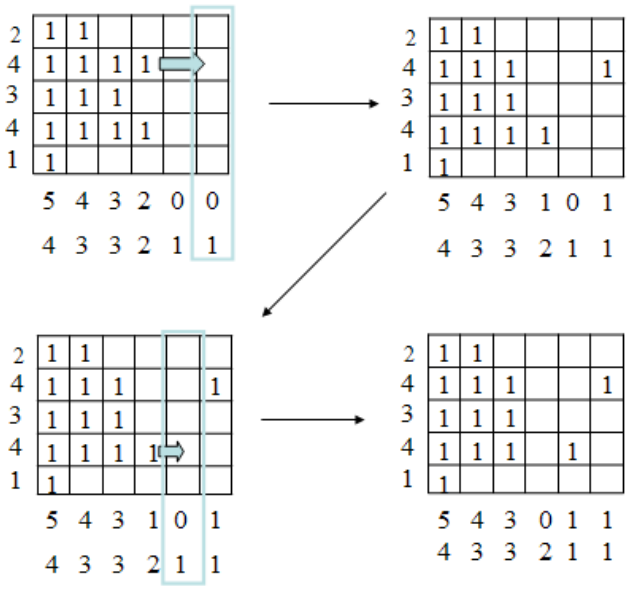

Figure 2.5: Illustration of the reconstruction progress with Algorithm 2.19 [33]; (Part 2 of 4 of Example 2.20).

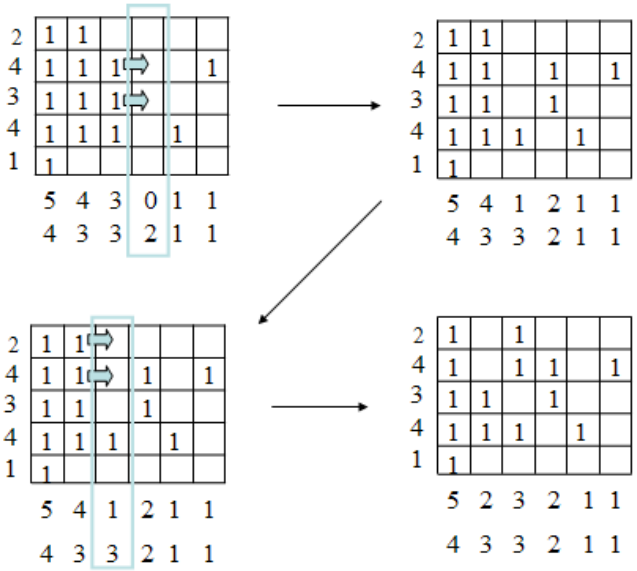

Figure 2.6: Illustration of the reconstruction progress with Algorithm 2.19 [33]; (Part 3 of 4 of Example 2.20).

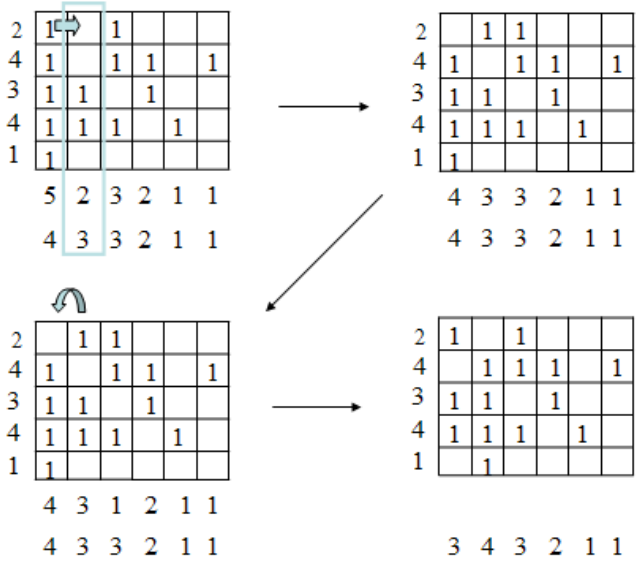

Figure 2.7: Illustration of the reconstruction progress with Algorithm 2.19 [33]; (Part 4 of 4 of Example 2.20).

Remark 2.22. Theorem 2.16 provides an answer to the consistency problem for

binary matrices with special projections, and the proof of it contains Algorithm 2.19. By this property, it gives a solution to the reconstruction problem.

1. The computational complexity of Algorithm 2.19 is $\mathcal{O}(n(m+\log n))$ [25].
2. There are several alternatives of Algorithm 2.19 including Ryser's method [46, 47], which differ in selection of the row i_0.
3. The determination of the precise number of matrices in $\mathcal{U}(R,S)$, is an open problem.

2.3.2 Fundamentals on Uniqueness

Definition 2.23. A binary matrix A is ***non-unique*** (with respect to its row and column sums), if there is a binary matrix $B \neq A$ having the same row and column sums as A. Otherwise, A is ***unique***.

At this point it would be useful to give further insights of Ryser [46, 47].

Definition 2.24. The sub-matrices of a binary matrix A to be reconstructed, S_1 and S_2 defined by Equation (2.3.9) are called ***switching operators*** (due to Ryser [46]).

$$S_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \; S_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{2.3.9}$$

Definition 2.25. Replacing the sub-matrix S_1 to S_2, or S_2 to S_1 in a binary matrix A, is called ***switching***.

A switching is a transformation of the elements of A that interchanges the submatrices S_1 and S_2, which preserves the row and column sums.

Theorem 2.26. *(**Ryser's Theorem**): If A and B are two binary matrices with the same* standard projections *(orthogonal projections in the directions $(1,0)$ and $(0,1)$), then A can be transformed into B by a* finite *number of* switchings.

Theorem 2.27. *A binary matrix is non-unique (with respect to its row and column sums)* if and only if *it has a switching component.*

Now, consider the special case of Theorem 2.16, in which the Equation (2.3.8) is replaced by

$$\sum_{j=l}^{n} s'_j = \sum_{j=l}^{n} \bar{s}_j, \qquad (l = 2, 3, ..., n). \tag{2.3.10}$$

Using the constructive method of Algorithm 2.19, it can be shown that, if (R, S) satisfies the conditions of Theorem 2.16 with (2.3.8) replaced by (2.3.10), then there is a ***unique*** matrix with respect to R and S.

Conversely, if the binary matrix A is unique, then it has no switching component. Consider two columns of A, j_1 and j_2. Suppose that $s_{j_1} \leq s_{j_2}$. If $a_{ij_1} = 1$ for some i, then $a_{ij_2} = 1$. Otherwise, if $a_{ij_1} = 1$ and $a_{ij_2} = 0$, there is at least one row i' such that $a_{i'j_1} = 0$ and $a_{i'j_2} = 1$, which contradicts to the assumption that A has no switching component. In other words, the 1's in column j_1 are in the rows in which there is also a 1 in column j_2. This means that if the columns of A are permuted non-increasingly, then we get just the maximal matrix $\bar{A}$. Therefore, $S' = \bar{S}$ and so (2.3.10) is true.

This leads to the following result: If A is unique binary matrix, then it can be recovered from its row sum R and column sum S using Algorithm 2.19 without Step 3. That is, we can construct the maximal matrix $\bar{A}$ from R and then recover A by a permutation π^{-1} at the columns of $\bar{A}$ (where π is the permutation that produce S' from S). Again, this leads us the following results.

Lemma 2.28. *If A is a maximal binary matrix, then*

$$a_{ij} = 1 \quad \Leftrightarrow \quad s_j \geq |\{k | r_k \geq r_i\}|. \tag{2.3.11}$$

From this, for a unique binary matrix A in $\mathcal{U}(R, S)$, the maximal matrix $\bar{A}$ is the unique element of $\mathcal{U}(R, \bar{S}) = \mathcal{U}(R, S')$. Since A is obtained from $\bar{A}$ by the permutation π^{-1} of the columns, it follows that for a unique binary matrix A

in $\mathcal{U}(R,S)$, Equation (2.3.11) holds (even if A is not maximal). This leads to the following results on uniqueness.

Definition 2.29. An $(m\times n)$ binary matrix $A=(a_{ij})$ $(i=1,...,m)$ $(j=1,...,n)$ is additive if there are vectors $X=(x_1,...,x_m)\in\mathbb{R}^m$ and $Y=(y_1,...,y_n)\in\mathbb{R}^n$ such that, for $i=1,...,m$ and $j=1,...,n$, $a_{ij}=1$ *if and only if* $x_i+y_j\geq 0$.

Theorem 2.30. *A binary matrix is unique* if and only if *it is additive.*

Proof. Let A be an $(m\times n)$ binary matrix with row and column sum vectors R and S. If A is unique, then it satisfies (2.3.11). This implies that A is additive with respect to the vectors X and Y, where $x_i=-|\{k|r_k\geq r_i\}|$ $(i=1,...,m)$ and $y_j=s_j$ $(j=1,...,n)$.

Suppose that A is additive with respect to the vectors $X\in\mathbb{R}^m$, $Y\in\mathbb{R}^n$. Let $B\in\mathcal{U}(R,S)$. Consider the function

$$\mathcal{K}(A,B)=\sum_{i=1}^{m}\sum_{j=1}^{n}(x_i+y_j)(a_{ij}-b_{ij}). \tag{2.3.12}$$

From Definition 2.29, each term in the sum on the right-hand side of Equation (2.3.12) is non-negative. Furthermore,

$$\begin{aligned}\mathcal{K}(A,B) &= \sum_{i=1}^{m}x_i\sum_{j=1}^{n}(a_{ij}-b_{ij})+\sum_{j=1}^{n}y_j\sum_{i=1}^{m}(a_{ij}-b_{ij})\\ &= \sum_{i=1}^{m}x_i(r_i-r_i)+\sum_{j=1}^{n}y_j(s_j-s_j)=0.\end{aligned}$$

This implies that each term in the sum on the right-hand side of Equation (2.3.12) is in fact zero. From Definition 2.29 this implies that if $a_{ij}=0$, then $b_{ij}=0$. However, A and B have the same number of 0's, therefore $A=B$, i.e., A is unique. □

All the findings about uniqueness of a non-empty class $\mathcal{U}(R,S)$ can be summarized in the following theorem.

Theorem 2.31. *Let $R = (r_1, ..., r_m)$ and $S = (s_1, ..., s_n)$ be vectors of non-negative integers such that there is a binary unique matrix $A \in \mathcal{U}(R, S)$. The following conditions are equivalent:*

(1.) A is unique with respect to R and S

(2.) A has no switching component

(3.) Equation (2.3.10) is satisfied

(4.) A is additive.

If it is known that there is a unique binary matrix A in $\mathcal{U}(R, S)$, then the Equation (2.3.11) must be satisfied. Using this and the theorems stated about uniqueness, this special case can be solved by the following algorithm.

Algorithm 2.32.

Input : Compatible pair of vectors (R, S) satisfying (2.3.10)

Step 1. $A = 0$ (zero matrix)

Step 2. Find $i_1, i_2, ..., i_m$ such that $r_{i_1} \geq r_{i_2} \geq ... \geq r_{i_m}$

Step 3. *for*$(j = 1$ *to* $n)$

{

for$(k = 1$ *to* $s_j)$

{

$a_{i_k j} \leftarrow 1$

}

}

Output : Matrix A

Chapter 3

State of the Art

From the arise up to now, some important results have been found in the three basic problem of DT theoretically, namely, for consistency, uniqueness and reconstruction. For 2-dimensional objects (discrete sets), there are algorithms with polynomial-time computational complexity. However, for 3-dimensional objects (discrete sets), usually the problems are NP-complete or NP-hard, or even not solved yet. There are only a few polynomial-time algorithms under some assumptions or special cases. In the following paragraphs, I will present some of the examples of these results.

In 1957, Ryser [46] published a necessary and sufficient consistency condition for a pair of integral vectors being the row and column sum vectors of a binary matrix. By giving a constructive proof of his theorem, he provided the first reconstruction method (Algorithm 2.26). He also recognized the switching operators and operations Definition 2.25, so that any two binary matrices can be transformed into each other if they have the same row and column sums.

Then this problem arose: *When is a planar convex body uniquely determined from its projections?* For projections along parallel lines, an answer is given as follows: For any planar convex body, there exist 3 projections which uniquely determine it. Moreover, it has been proved that it is possible to find 4 projections that will uniquely determine all planar convex bodies. Although these results of convex geometry, more exactly, geometric tomography, are about the reconstruction of *convex* bodies, there are corresponding results for discrete sets.

Obviously, a finite number of projections are not generally sufficient to recon-

struct discrete sets uniquely. However, it has been proved that, if *the number of points in the discrete set to be reconstructed is known*, it is possible to find finitely many projections that will guarantee uniqueness.

In the following, we come to *complexity* results for the reconstruction problems by lattice lines. Table 3.1, below, complexity results are given. Here, **PTA** stands for the existence of a corresponding *polynomial-time algorithm.*

	Consistency	Uniqueness	Reconstruction
$q = 2, d = 2$	PTA	PTA	PTA
$q \geq 3$	NP Complete	NP Complete	NP Hard

Table 3.1: Computational complexities of consistency, uniqueness, reconstruction [25].

The complexity of the problem is not in general developed yet for r-dimensional X-rays when $r > 1$. Only a few results are known. For instance, when $r = 2$, $d = 3$, and L consists of the 3 coordinate planes, the problems still remain open.

If we have $q = 2$ lattice directions only, then, due to *Ryser's Theorem* 2.26 (cf. [46]), by checking whether a known sub-matrix is contained in a binary matrix or not, we can say whether it is unique or not. On the other hand, it is known that one cannot check uniqueness with the same procedure if there are $q \geq 2$ directions.

Another studied case is the computational complexity of the problem of the reconstructing a set from its horizontal and vertical projections with respect to some classes of sets on which some connectivity constraints are imposed. In particular, *horizontally convex (or h-convex)*, *vertically convex (or v-convex)*, and *4-connectivity (or polyomino)*. Below, Table 3.2 shows the computational complexities of these cases for reconstruction. PTA stands for existsnce of a polynomial-time algorithm, while NP-C stands for NP-complete [15])

	hv-convex	h-conv	v-conv	No restriction
Polyomino	PTA	NP-C	NP-C	NP-C
No restriction	NP-C	NP-C	NP-C	PTA

Table 3.2: Computational complexities of hv-, h-, v-convex, with respect to connectivity [25].

Since the problem of reconstruction in higher dimensions becomes NP-complete or NP-hard, most of the research focus on some *approximate* solutions. Unless $\mathcal{P} = \mathcal{NP}$, the search for approximate solutions is suitable for real-world problems, instead of trying all the possible solutions, or running exponential time algorithms. Some of such works are mentioned shortly in the following sections.

3.1 Simple Slice Reconstruct-Merge Approach

The objects being inspected are 3-dimensional if the object being inspected is a real-world object. If the object is 2-dimensional there are algorithms (Algorithm 2.19, Algorithm 2.32) as presented in Section 2.3.2.

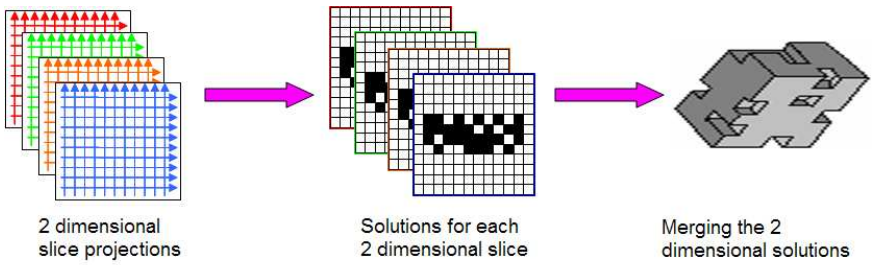

Figure 3.1: Illustration of a 2D slice reconstruction and 3D merge.

Ryser [46] introduced a reconstruction algorithm which gives a solution to the inverse problem of DT. There are also Kaczmarz's algorithm, ART (cf. [17]) and SIRT algorithms for solving the inverse problem of $Px = b$. A good reference for these algorithms is the book [1]. Since in DT, the problem is of a 3-dimensional

nature, the most common method is to use 2-dimensional reconstructions (using Ryser's algorithm, Kaczmarz's algorithm, ART, or SIRT) and to merge them into a 3 dimensional solution, see the figure, Figure 3.1. However, merging the 2 dimensional solutions as in the traditional method would give non-matching results.

There are difficulties of this approach in practice. Since the object to be reconstructed from the projections is not known, the actual dimensions of it are also unknown. Therefore, the reconstructed 2D slices can be merged inaccurately. The planes of 2D slices should be rotated, scaled, or shifted to place them into a correct space. Otherwise, the 3D solution might be far from representing the object to be reconstructed.

3.2 Probabilistic Modeling of Discrete Images

Probabilistic methods are studied widely by Gabor T. Herman, Avi Vardi, Michael T. Chan, and Emanuel Levitan [24]. In their study, the objects are images. Their work is for reconstruction of a particular family of images. That is, reconstruction with probabilistic modeling is suitable if an image to be reconstructed is suited for some distributions. If an image to be reconstructed is not suitable for the distribution used, then a well-suited distribution should be found.

In probabilistic methods, Gibbs distributions (Equation 3.2.1) are used as prior image models. The Gibbs priors describe the local behavior of a *binary* image. The outline is as follows: The images to be reconstructed are assumed to consist of piecewise-homogenous regions, which are again assumed to resemble some images which have Gibbs distributions. Based on this assumption, by projections such piecewise-homogenous regions are tried to be found. That is, sub-regions of the image to be reconstructed are approximated by predefined images whose distributions match with Gibbs distributions:

$$\Pi(w) = \frac{1}{Z} e^{\beta \sum_{h=1}^{H} I_h(w)}. \tag{3.2.1}$$

Here,

- $\Pi(w)$ is the probability of occurrence of image w,
- Z is a normalizing factor so that the sum of $\Pi(w)$ over all possible images is 1,
- β is a parameter defining the peakedness of the distribution,
- $I_h(w)$ is the local energy function for the pixel indexed by h.

For the local energy function $I_h(w)$, two possible approaches are used. The first one is a lookup table derived from some previously studied images, or an image model which is an expectation of the image to be reconstructed is used. The other one is an image model of free parameters.

When an image model of free parameters is used, the function $I_h(w)$ can take these values:

- k_1 for convex values of 0's,
- k_2 for edges,
- k_3 for uniform regions of 1's,
- k_4 for uniform regions of 0's,
- k_5 otherwise.

Now, here is the algorithm for reconstruction of an image:

Algorithm 3.1.
Reconstruction of a binary image:
$w_1 \leftarrow$ A Random Image

do

$w_2 \leftarrow w_1$

invert a randomly chosen sub-region h_1 in w_2

$p \leftarrow \Pi(w_2)/\Pi(w_1)$

if $(p \geq 1)$ **then**

$w_1 \leftarrow w_2$

end if

until a desired image is obtained [25]

Here, to *penalize* the images (images produced in each iteration) that are inconsistent with the projections, p calculated as:

$$p := e^{\beta \sum_{h \in N(h_1)} (I_h(w_2) - I_h(w_1)) - \alpha (F_{h_1}(w_2) - F_{h_1}(w_1))}, \tag{3.2.2}$$

where $F_{h_1}(w)$ denotes the difference between the projection and the image w for the lines passing through the sub-region h_1. $N(h_1)$ represents 6 neighbors of h_1 on a hexagonal grid.

By applying penalization, i.e., checking its consistency with the projections and discarding the inconsistent ones, the initially chosen random image converges to an image whose sub-regions have Gibbs distributions, and also becomes more consistent to the projections in each step. In Figure 3.2, an example result obtained for a 64×64 image is shown.

Figure 3.2: Illustration of a reconstruction of a 64×64 image using Algorithm 3.1. *Left*: The original image, *Right*: Reconstruction result [25].

3.3 Bayesian Methods for Discrete Tomography

In Bayesian approach, the focus is on the discrete-valued reconstruction from noisy projections using statistical methods. Statistical methods model the random nature of the physical data collection process, then they seek the solution which best matches the probabilistic features of the data.

The statistical method which is used is Bayesian *maximum a posteriori (MAP)* estimation. Bayesian MAP estimation reconstructs the image as a *tradeoff* between matching the projection data and regularizing the distribution by a prior probability distribution. MAP estimation formulates the reconstruction by treating the original image as a random field X, whose prior distribution is given by $p(x)$.

In order to write the probability density for the measurements, define X as the N-dimensional vector of pixels (original image). Let Y (projections) denote the vector of lines-sums for all M projections. Let P_{ij} correspond to the length of the intersections between the j^{th} pixel and the i^{th} projection. Then, P is

the matrix of elements P_{ij} and P_{i*} denotes the vector formed by its i^{th} row. With this notation, the atom count for Y_i corresponds to projection i and the distribution of of Y_i given x being an entry of X is defined by:

$$\mathcal{P}(Y = y|x) = \prod_{i=1}^{M} \frac{e^{-y_T e^{(-P_{i*}x)}} (y_T e^{(-P_{i*}x)})^{y_i}}{y_i!}. \tag{3.3.3}$$

Since X is not known, for reconstruction an *a posteriori* distribution of X (original image) given the projections Y. It is computed using Bayes' formula. Then, using Equation (3.3.3), the logarithm of the *a posteriori* distribution of X given Y is obtained:

$$\mathcal{P}_p(x|y) = \log \mathcal{P}(X = x|Y = y). \tag{3.3.4}$$

The MAP reconstruction itself is considered and formulated as an optimization problem. Therefore, the maximum *a posteriori* (MAP) estimate $\hat{x}$, which maximizes the *a posteriori* density given the observations y, can be defined as:

$$\hat{x} := \arg\max_x \mathcal{P}_p(x|y), \tag{3.3.5}$$

where the optimization problem can be written as:

$$(\mathcal{P}) \qquad \text{maximize} \quad \mathcal{P}_p(x|y). \tag{3.3.6}$$

There are two commonly used different techniques for solving the optimization problem $(\mathcal{P})$. These are:

- *Expectation-maximization* (EM) (cf. Section 3.4)
- *Iterative coordinate descent* (ICD): a pixel-wise update method.

Because of implementation difficulties and computational simplicity, ICD is preferred rather than EM. The ICD method sequentially updates each pixel of

the image. With each update, the current pixel is chosen to maximize $\mathcal{P}_p(x|y)$.

Algorithm 3.2.

Reconstruction of a binary image using Bayesian approach:

1. Compute an estimate image X using (3.3.5)
2. **do**

    Update the pixels of X using ICD (or EM)

until no pixel update occurs

The computational complexity of Algorithm 3.2 is $NKM_0O(\mathcal{OP})$, where N is the number of pixels in reconstruction, K is the number of discrete values, M_0 is the average number of projections intersecting a pixel, and $O(\mathcal{OP})$ is the complexity of the optimization problem (Equation 3.3.6). In Figure 3.3, the original image and the reconstruction result is shown.

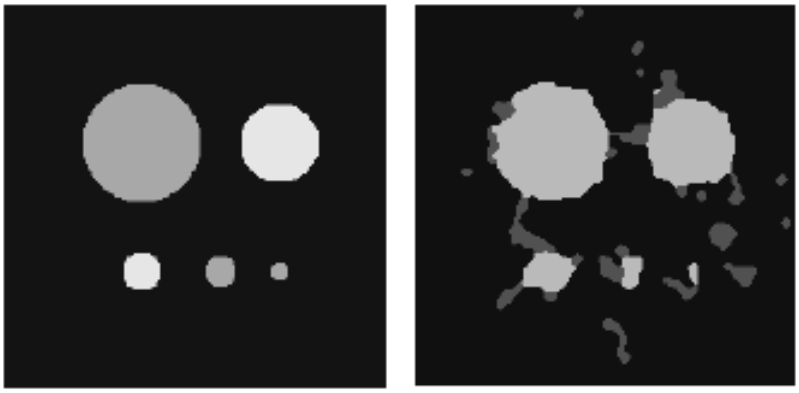

Figure 3.3: Illustration of a reconstruction of a 64×64 non-binary image using Algorithm 3.2. *Left*: The original image, *Right*: Reconstruction result [24, 25].

3.4 Reconstruction of Binary Images via EM Algorithm

The approach is based on *(a)* relaxing the binary constraints $x(i) = 0$ or $x(i) = 1$ to interval constraints $0 \leq x(i) \leq 1$ $(i \in \mathbb{Z})$, and *(b)* applying a minimum distance method (*Kullback-Leibler distance*) [18, 51] to find such an $\hat{x}$ for which the distance between the observed and the theoretical partials is as small as possible.

Let x be a binary function defined on a finite subset of a lattice, $x(i) = 0$ or $x(i) = 1$ $(i \in \mathbb{Z})$. Let $L_1, ..., L_m$ be a collection of subsets of the lattice $\mathbb{Z}$, let the line-sums be

$$y_j = \sum_{i \in L_j} x(i) \qquad (j = 1, ..., m) \tag{3.4.7}$$

and consider the problem of reconstructing the function f from the knowledge of the partial sums $y_1, ..., y_m$.

Here, Kullback-Leibler distance (*KL-distance)* is used as a minimum distance function.

Definition 3.3. (KL-distance):
The KL-distance between two probability vectors $p = (p_1, p_2, ..., p_m)$ and $q = (q_1, q_2, ..., q_m)$ is

$$KL(p, q) = \sum_{j=1}^{m} p_j \log p_j / q_j. \tag{3.4.8}$$

Rewriting (3.4.7) as

$$y_j = \sum_{i=1}^{k} x_i A_{ij} \qquad (j = 1, ..., m), \tag{3.4.9}$$

where $i = 1, ..., m$ is a labeling of the set $\mathbb{Z}$, $x_i := x(i)$ and $A_{ij} = 1_{L_j}$, where $i = 1$ if $i \in L_j$, and $i = 0$ if $i \notin K_j$). Then, the interval constraints are $x_i \in [0, 1]$

$(i = 1, ..., k)$. Before giving the EM algorithm, the following definitions will be necessary:

$$\hat{g}_j := y_j / \sum_{j'=1}^{m} y_{j'}, \tag{3.4.10}$$

$$h_{ij} := A_{ij} / \sum_{j'=1}^{m} A_{ij'}, \tag{3.4.11}$$

$$f_i := x_i (\sum_{j'=1}^{m} A_{ij'}) / \sum_{j'=1}^{m} y_{j'}. \tag{3.4.12}$$

Algorithm 3.4.

Simple EM/ML Algorithm:

1. **Initialize**

   $f^{(0)} = (f_1^{(0)}, ..., f_k^{(0)})$, $a_i \leq f_i^{(0)} \leq b_i$, such that $\sum_{i=1}^{k} f_i^{(0)} = 1$.

2. **Iterate E-Step and M-Step for** $n \geq 0$

   E-Step (Expectation)

   $f^{(n+1/2)} = f_i \sum_{j=1}^{m} h_{ij} \frac{\hat{g}_j}{g_j(f)}$ $(i = 1, ..., k)$

   M-Step (Maximization)

   $\max_p \sum_{i=1}^{k} f_i^{(n+1/2)} \log p_i$

It has been proved that Algorithm 3.4 converges monotonically. You can see the experimental results in [18, 51].

3.5 Iterative Methods for Discrete Tomography

A process, called *binary steering* [5], designed to intervene between the iterative steps of any non-binary algorithm for solving $Px = b$ in a way that would gradually steer the iterates towards a binary solution. This heuristic process is applicable to a plethora of non-binary iterative reconstruction algorithms which solve (asymptotically, depending on the relevant solution concept adopted) the system $Px = b$. Some of these non-binary iterative algorithms perform very well on non-binary image reconstruction problems, efficiently generating acceptable reconstructed images (i.e., approximations to the solution vector x), some of them lend themselves to parallel computations or have other favorable features such as guaranteed convergence even if the system $Px = b$ is inconsistent.

This method is to reconstruct *binary images*, by using non-binary iterative reconstruction algorithms in conjunction with an additional mechanism to steer the non-binary iterates towards an acceptable *binary solution*.

Remark 3.5. The non-binary algorithms, to which binary steering will be applied, should have the following general form:

Algorithm 3.6.

Initialization: $x^0 \in U$, where $U \subseteq \mathbb{R}^n$ is the initialization set dictated by the specific non-binary algorithm.
Iterative Step: Given the k^{th} iterate x^k, and the data of the problem $d \in D$, where D is the data space dictated by the specific non-binary algorithm, calculate:

(1) *Correction calculation*: The k^{th} correction vector c^k is calculated by a formula of the form $c^k = f_k(x^k, d)$, where the functions f_k are dictated by the specific non-binary algorithm.

(2) *Correction application*: The next iterate x^{k+1} is calculated by a formula of the form $x^{k+1} = g_k(x^k, c^k)$, where the functions g_k are dictated by the

specific non-binary algorithm.

The term data ($d \in D$) in this (and the next) algorithm is meant to include not only the measured data (such as b in $Px = b$) but all measured as well as design data, i.e., both P and b in the case of linear equations (Figure 3.4).

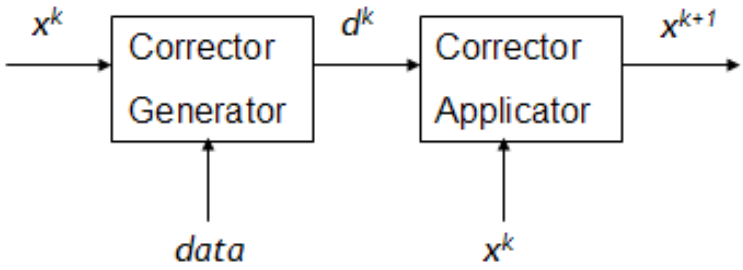

Figure 3.4: Illustration of iterative step of steering mechanism [5].

Definition 3.7. Let $\alpha = (\alpha_k)_{k\geq 0}$, $\beta = (\beta_k)_{k\geq 0}$, and $t = (t_k)_{k\geq 0}$ be three sequential sequences such that $0 \leq \alpha_k \leq t_k \leq \beta_k \leq 1$, and $\beta_{k+1} < \beta_k$ for all $k \geq 0$. Given any sequence $(x^k)_{k\geq 0}$ of vectors $x^k = (x_j^k)_{j=1}^n \in \mathbb{R}^n$, the sequence $(\tilde{x}^k)_{k\geq 0}$ defined for all $k \geq 0$ and $j = 1, 2, ..., n$, by

$$\tilde{x}_j^k = \begin{cases} 0 & \text{if} \quad x_j^k \leq \alpha_k, \\ 1 & \text{if} \quad x_j^k \geq \beta_k, \\ x_j^k & \quad \text{otherwise}, \end{cases} \tag{3.5.13}$$

is called a *sequential binarization of* $(x^k)_{k\geq 0}$ *with respect to the triplet of sequences* (α, β, t).

Definition 3.8. Let $\alpha = (\alpha_k)_{k\geq 0}$, $\beta = (\beta_k)_{k\geq 0}$, and $t = (t_k)_{k\geq 0}$ be three sequential sequences as in Definition 3.7, and let ϵ be an arbitrarily small but *fixed* real number with $0 < \epsilon < 1/10$. Given any two vector sequences $(x^k)_{k\geq 0}$ and $(y^k)_{k\geq 0}$, the sequence $(z^k)_{k\geq 0}$, defined for all $k \geq 0$ and $j = 1, 2, ..., n$, by

$$\tilde{z}_j^k = \begin{cases} t_k - \epsilon & \text{if} \quad x_j^k \leq \alpha_k, y_j^k \geq t_k, \\ t_k + \epsilon & \text{if} \quad x_j^k \geq \beta_k, y_j^k \leq t_k, \\ y_j^k & \text{otherwise,} \end{cases} \tag{3.5.14}$$

is said to *settle sequentially the conflict between* $(x^k)_{k\geq 0}$ and $(y^k)_{k\geq 0}$ *with respect to triplet of sequences* (α, β, t) and ϵ.

Algorithm 3.9.

Initialization: $x^0 \in U$ where $U \subseteq \mathbb{R}^n$ is the initialization set dictated by the specific non-binary algorithm in use.
Iterative Step: Given the k^{th} (current) iterate x^k, do the following:

(1) *Sequential binarization*: Use the sequences (α, β, t) of Definition 3.7 to perform a sequential binarization on x^k to obtain $\tilde{x}^k$.

(2) *Non-binary algorithmic step*: Use the k^{th} sequentially binarized iterate $\tilde{x}^k$ and the data of the problem $d \in D$, where D is the data space dictated by the specific non-binary algorithm in use, to calculate:

(2.a) *Correction calculation*: The k^{th} correction vector c^k is calculated by a formula of the form $c^k = f_k(\tilde{x}^k, d)$, where the functions f_k are dictated by the specific non-binary algorithm in use.

(2.b) *Correction application*: The output iterate y^k of the non-binary algorithmic step is calculated by a formula of the form $y^k = g_k(\tilde{x}^k, c^k)$, where the functions g_k are dictated by the specific non-binary algorithm in use.

(3) *Conflict resolution*: Use the sequences (α, β, t) and the parameter ϵ of Definition 3.8 to calculate the next iterate x^{k+1} of the binary steering process by settling the conflict between y^k and x^k, if any, according to Definition 3.8.

3.6 Reconstruction by Shape Estimation from Projections

An interesting and promising work due to Ali Mohammad-Djafari and Charles Soussen [12, 11] will be presented in this section.

In their work, it is assumed that the object to be reconstructed is a compact homogeneous object and the object lies in a homogeneous background. A reconstruction method is presented for reconstructing the compact homogeneous object from its X-ray projections. The method presented can reconstruct 3-dimensional objects.

The researchers categorize the methods into 3 main groups:

1. *Methods which discretize the object into cells and use an appropriate model for the distribution*: This approach has the advantage of the assumption that the relation between data and the unknowns are linear. As pointed out in [12], one of the major drawbacks of this approach is, the number of cells to be reconstructed increases as the object size increases.

2. *Methods which make the simplification that the object is approximated by a level set of continuous functions*: This is done by estimating the closed contour of the object. The difficulty of this approach is, for 3-dimensional objects finding an estimate of a closed contour has too much computation cost.

3. *Methods which estimate the geometrical shape of the object from its projections*: This approach reduces the number of unknown parameters but the relation between the data and the unknown parameters cannot be assumed linear. In [12], this type of methods can be categorized into 2 subclasses, namely those

   3.a. which use *simple shapes* as a model in the estimation, and those

   3.b. which use *deformable shapes* as a model in the estimation.

In [12], it is shown that exact reconstruction via methods of class 3 above is impractical. The proposed method is an approximation method which uses polyhedral shapes to estimate the shape of the object directly from its projections. Consider the parallel-beam projections model. Let P be the 2-dimensional object and f be the density function of P. Then, $\mathcal{R}_{r,\phi}$, ***X-ray transform of f***, can be defined as:

$$\mathcal{R}_{r,\phi} := \int\int_{\mathbb{R}^2} f(x,y)\delta(r - x\cos\phi - y\sin\phi)dxdy, \tag{3.6.15}$$

where $\phi \in [0,\pi)$, $r \in \mathbb{R}$ and δ is the *Dirac delta function* [1].

With the notation of Equation (3.6.15), the formulation of problem is given as:

$$p(r,\phi) = \mathcal{R}_{r,\phi}(f) + n(r,\phi), \tag{3.6.16}$$

where p is the projection data and n represents the errors of modeling and measurement. In Figure 3.5, an example of a projection of a polygonal shape is given.

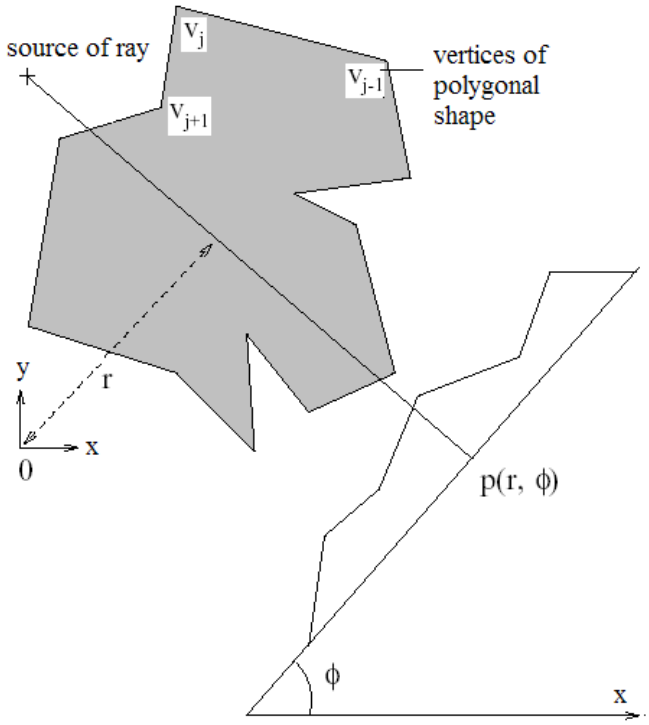

Figure 3.5: Illustration of a projection of a polygonal shape [12].

If the domain is assumed to be discrete, then simply taking the density function f as a discrete valued function, the problem will be a DT problem. An example of a binary density function of a 2-dimensional object P might be:

$$f(x,y) = \begin{cases} 1 & \text{if} \quad (x,y) \in P, \\ 0 & \quad \text{otherwise.} \end{cases} \tag{3.6.17}$$

They proved that, theoretically, it is possible to estimate exactly a polygonal shape from the moments [12] of its projections. However, in practice to fulfill the sufficient conditions to do an estimation will be satisfied if at least $2n_v - 1$ (where n_v is the number of vertices of the polygonal shape) projections are needed. If the situation is a *non-destructive testing* (NDT), then this is almost impossible.

Because of these restrictions in practice, an exact 3-dimensional reconstruction

becomes very hard. In [49], Mohammad Djafari proposed methods to approximately solve this problem based on the regularization theory or its Bayesian estimation interpretation. From these results, the proposed method for a 3-dimensional reconstruction will be explained here.

Assumption 3.6.1. *Assume that the polyhedron has the following properties:*

1. *the polyhedron is composed of only triangular facets,*
2. *the neighborhood relations between the vertices themselves and between the vertices and the facets of the polyhedron are already defined.*

With Assumption 3.6.1, the polyhedron can be represented by only its vertices. Let n_v denote the number of vertices and $v_j = (x_j, y_j, z_j)$ their coordinates $(j = 1, ..., n_v)$. Denote the set of vertices by $v = \{v_j | j = 1, ..., n_v\}$. Then simply with Assumption 3.6.1, the coordinates of the vertices are estimated by the solution of minimization problem:

$$\text{minimize} \quad J(v) = \|p - h(v)\|^2 + \lambda\Omega(v), \tag{3.6.18}$$

where h is a function which computes the projections for any given set v. The chosen *regularization function* is

$$\Omega(v) := \sum_{j=1}^{n_v} \|v_j - \frac{1}{K_j} \sum_{i \in \nu_j} v_i\|^2, \tag{3.6.19}$$

where ν_j stands for the neighborhood of v_j and K_j is the number of vertices that belong to this neighborhood. Here, $v_j - \frac{1}{K_j} \sum_{i \in \nu_j} v_i$ represents the geometric center of all the neighbors of v_j. Computing the optimal solution corresponding to its global minimum is solved by one of the following 2 alternatives:

1. **Simulated Annealing (SA)**: This technique is iterative and involves a parameter T_n called temperature at the iteration n. The sequence of temperatures (T_n) decreases, and converges to 0 as n goes to infinity. For

a fixed value of n, the vertices v_j is modified, according to a random procedure. Furthermore, v_j is generally sampled from a uniform, a Gaussian or using its prior probability distribution. Let v^j denote the vector v, in which v_j has been replaced by its new value. A decision rule indicating which of the configurations v and v^j has to be kept is the following:

- if $J(v^j) < J(v)$, then accept the modification of v_j. The new value is $v := v^j$;
- if $J(v) < J(v^j)$, then accept the configuration v^j with a probability proportional to

$$\exp(-\frac{J(v^j) - J(v)}{T_n}). \tag{3.6.20}$$

There exist sufficient conditions on the sequence (T_n), which insure that this optimization algorithm converges to one of the global minima of J, whatever the initial condition is. In practice, this technique gives satisfactory results [12], but requires a large number of iterations and choosing initial temperature T_0 and the cooling schedule is not obvious.

2. Use of a local optimization technique with an initial solution is used as another alternative. Here, a *good initial solution* becomes important.

   - The initial solution is estimated by calculating its moments from projections. Then, a polygonal or polyhedral shape whose vertices are on an ellipsoid is reconstructed as the initial solution.
   - The SA technique with a slight modification is used. Here, the difference from SA is that the new configuration after modification of v to v^j is accepted *if and only if* $J(v^j) < J(v)$. This technique is called ***Iterated Conditional Modes*** (***ICM***).

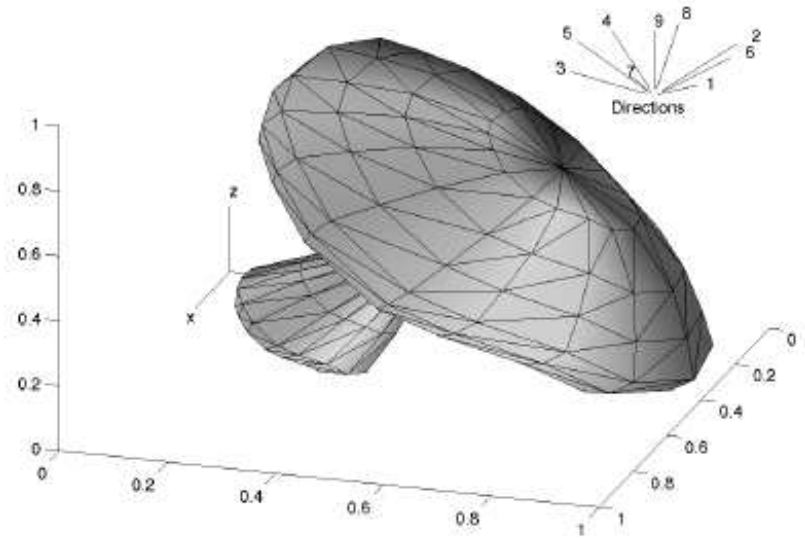

Figure 3.6: Experimented 3-dimensional object and the 9 projections used [12].

A 3-dimensional object is studied and the results are given in [12]. The object to be reconstructed is a compact homogenous object with 40 vertices and the projections are parallel beam geometry with 9 directions as shown in Figure 3.6.

The object in Figure 3.6 has been reconstructed by the proposed methods of Djafari and Soussen [12, 11, 49]. The results are shown in Figure 3.7.

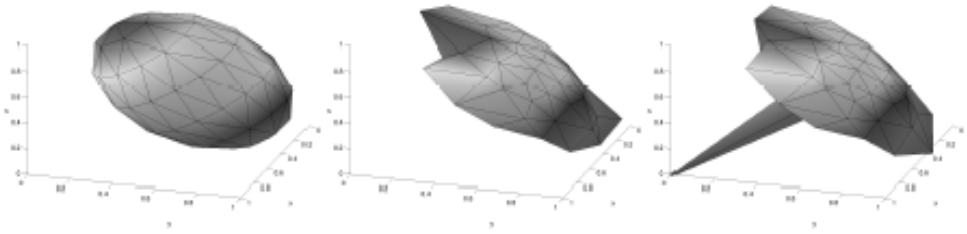

Figure 3.7: Reconstructions of the object after 0, 40 and 100 iteratioins [12].

3.7 Reconstruction by Optimization, Equivariance Analysis and Statistical Learning

It is studied by Gerhard Wilhelm Weber and Öznur Yaşar [53, 55, 56]. In this work, new approaches are brought to DT, such as using *statistical learning, coding theory and optimal experimental design* in an optimization framework.

Here, one problem of reconstruction is considered as an optimization problem by imposing the maximization of $f(x)$, defined as the sum of the components x_i $(i = 1, ..., N)$ with respect to Equation (2.2.5), i.e.,

$$(\mathcal{P}_1) \qquad \text{maximize} \quad f(x) := \sum_{j=1}^{N} x_j, \quad \text{subject to } Px = b \text{ and } x \in \{0,1\}^N.$$

Further optimization problems are coming from the use of coding theory or optimal experimental design.

One of the optimization approaches presented is being developed on equivariant analysis. In the following paragraphs, the outline of this approach will be presented.

The possible cluster is embedded into a little bigger discrete set. Using a series of X-ray measurements, the equivariants (or symmetrical) properties are discovered. This can be done *globally*, but also *locally* or *partially* for a finite number of subsets. Such discrete subsets are called ***windows***.

Then, finally, these equivariances, if verified by the X-ray measurements, help to simplify the representation of the atom (sub-)cluster or of its estimate (iterate) and, herewith, globally, partially or locally to dimensionally reduce the complexity of the reconstruction problem. Later on, when the reconstruction problem is approximately resolved, obits or discrete trajectories are followed back, and the problem result from the points on the section to all the other points on the corresponding orbit are assigned back. This *backward assignment* will orbitwise be done by the same values, or up to a regular change of coordi-

nates of the measurement vector, e.g., by a permutation of the components or another transformation related with the group of orbits.

Now, at this point coding theory is incorporated to reduce the dimension of the problem and for efficient equivariance analysis. Consider a sufficiently large rectangular subset of the lattice with the unknown atom cluster included (to be studied by rays, mainly, X-rays), as a *word* being close to one or another element of a finite linear space $\mathcal{C}$. This word can be easily found by aligning the columns (or rows) of the binary matrix given by the rectangular set and the lattice directions. This initial linear code incorporates any pre-information about the possible location of the atom cluster. At first, there is only a vague initial guess about the atom distribution. The codeword which is closest to the initial guess is tried to be found by *decoding* (or *reconstructing*).

A *training set* and a *test set* of measurements are generated. By training, a code which is supposed to be an element in or approximated is refined and suitably built up. Afterwards, but *iteratively* coupled with training, by testing the validation and improvement of the code and, herewith, the approximation of the real atom cluster is done successively. Within of this learning process, step by step the dimension or complexity of the code becomes ***reduced***. This implies an alternating sequence of training and test errors being analyzed.

Since a linear code is defined as a vector space over some finite field $\mathbb{Z}_p$, where p is some prime number, coding theory is used to find geometrical properties of the atom cluster in a step by step process of measurement and improvement. For the case of discrete tomography in VLSI chip design, the code is over the field $\mathbb{Z}_2$, hence $p = 2$. Here, 1 means existence and 0 means nonexistence of an atom at a lattice point. So, for measuring the difference between the approximative iterate and the closest codewords, *Hamming distance* is used.

For the equivariance detection and processing *cyclicity* (or in terms of coding theory, *cyclic codes*) can be used,

(i) for making simplifications in decoding,

(ii) for finding error-correcting code associated,

(iii) for calculating its minimum distance easily.

Since usually the set to be reconstructed is *non*symmetric, i.e., not ideally equivariant, the following treatment is used: Include those lattice points which are neighboring but not necessarily expected to be in the atom cluster, by giving them zero weights. (In fact, orbits g, e.g., rotations, perturbations or sign changes can still be applied.) Namely, some small neighboring parts of the lattice to get more symmetries can be added. This means a *preprocessing* which may, in addition, also be made at the beginning of every iteration step. At the end of the algorithm, these further auxiliary points are deleted. Moreover, a ***backward assignment*** is performed by means of all the equivariance conditions used, including a successive increase of the dimensions. This concludes the algorithmic concept.

Optimal experimental design is the other ides presented in their work from optimization point of view. Here, statistical learning concepts have been emphasized in the logic of optimal experimental design.

3.8 General Inverse Problems Algorithms

Since a reconstruction problem is an inverse problem, direct application of inverse problem solution methods are also applicable. In this section, some iterative methods which are used for solving general inverse problems will be presented based on [1].

Kaczmarz's algorithm, and the variations of it, namely, ART and SIRT will be presented. These algorithms were originally developed for tomographic applications and are particularly effective for such problems. On the other hand, conjugate gradient least squares (CGLS), which is a general inverse problem solving method by regularization, will be presented.

3.8.1 Kaczmarz's Algorithm

Kaczmarz's algorithm is an easy to implement algorithm for solving a linear system of equations $Px = b$ (P is assumed to be $(M \times N)$ matrix as given in 2.2.5). Let Kaczmarz's algorithm starts with an initial solution $x^{(0)}$, and then moves to a solution $x^{(1)}$ by projecting the initial solution onto the line sum defined by the first row in P. Next, $x^{(1)}$ is similarly projected onto the line sum defined by the second row in P, and so forth. The process is repeated until the solution has been projected onto all M line sums defined by the system of equations. At that point, a new cycle of projections begins. These cycles are repeated until the solution has converged sufficiently.

Let $P_{i,.}$ be the i^{th} row of P, and b_i be the i^{th} row of b. Consider the hyperplane defined by $P_{i+1,.}x = b_{i+1}$. Because the vector $P^T_{i+1,.}$ is perpendicular to this hyperplane, the update to x^i from the constraint due to row $i+1$ of P will be proportional to $P^T_{i+1,.}$. Denote this proportion by β. Then, the update formula of the Kaczmarz's algorithm will be:

$$x^{(i+1)} = x^{(i)} + \beta P^T_{i+1,.}. \tag{3.8.21}$$

Using $P_{i+1,.}x^{(i+1)} = b_{i+1}$, and Equation (3.8.21), the solution for β will be:

$$\beta = -\frac{P_{i+1,.}x^i - b_{i+1}}{P_{i+1,.}P^T_{i+1,.}}. \tag{3.8.22}$$

Then, substituting β into the Equation (3.8.21), the update formula will be

$$x^{(i+1)} = x^{(i)} - \frac{P_{i+1,.}x^{(i)} - b_{i+1}}{P_{i+1,.}P^T_{i+1,.}}P^T_{i+1,.}. \tag{3.8.23}$$

Algorithm 3.10.

Input : P ($M \times N$ matrix) and b ($N \times 1$ matrix) (2.2.5)

Step 1. Let $x^{(0)} = 0$ be the initial solution

Step 2. **for** $i = 0, ..., M$

{

$$x^{(i+1)} \leftarrow x^{(i)} - \frac{P_{i+1,.}x^{(i)} - b_{i+1}}{P_{i+1,.}P_{i+1,.}^T} P_{i+1,.}^T$$

}

Step 3. **If** the solution has not yet converged, **go to** Step 2.

Output : x, solution of $Px = b$

It can be shown that if the system of equations $Px = b$ has a unique solution, then Kaczmarz's algorithm (Algorithm 3.10) will converge to this solution. If the system of equations has many solutions, then the algorithm will converge to the solution that is closest to the point $x^{(0)}$. In particular, if $x^{(0)} = 0$ is used, then a minimum length solution will be obtained. If there is no exact solution to the system of equations, then the algorithm will fail to converge, but will typically bounce around near an approximate solution. Since Kaczmarz's algorithm makes use of projections of previous step's solution onto hyperplanes (line sums) of P, the convergence becomes slow if the hyperplanes (line sums) described by the system of equations are nearly parallel. Indeed, if the rows of P are nearly parallel, then the system of equations tend to be rank deficient as most of the inverse problems.

3.8.2 ART

The *Algebraic Reconstruction Technique* (*ART*) is a version of Kaczmarz's algorithm that has been modified especially for the tomographic reconstruction problem [17]. A rough approximation to the Kaczmarz update, used in ART, is to replace all of the nonzero elements in row $i + 1$ of P with 1s.

Let σ_{i+1} be the approximation to the travel time along ray path $i + 1$. Define σ_{i+1} as:

$$\sigma_{i+1} := \sum_{\text{cell } j \text{ in ray path } i+1} x_j c. \tag{3.8.24}$$

The difference between σ_{i+1} and b_{i+1} is roughly the residual for the predicted travel time of ray $i+1$. ART takes the total error in the travel time for ray $i+1$ and divides it by the number of cells in ray path $i+1$, Σ_{i+1}, and by the cell dimension, c. This correction factor is then multiplied by a vector that has ones in cells along the ray path $i+1$.

Taking into account that the ray path lengths actually will vary from cell to cell, and denoting L_{i+1} as the length of ray path $i+1$, the corresponding update formula for the tomography problem can be written as:

$$x_j^{(i+1)} = \begin{cases} x_j^{(i)} + \frac{b_{i+1}}{L_{i+1}} - \frac{\sigma_{i+1}}{c\Sigma_{i+1}} & \text{cell } j \text{ in ray path } i+1, \\ x_j^{(i)} & \text{cell } j \textbf{ not} \text{ in ray path } i+1. \end{cases} \tag{3.8.25}$$

Algorithm 3.11.

Input : P ($M \times N$ matrix) and b ($N \times 1$ matrix) (2.2.5)

Step 1. Let $x^{(0)} = 0$ be the initial solution

Step 2. **for** $i = 0, ..., M-1$, $j = 1, ..., N$

{

Calculate $x^{(i+1)}$ using Equation (3.8.25)

}

Step 3. **If** the solution has not yet converged

{

let $x^{(0)} \leftarrow x^{(M)}$, **go to** Step 2

}

Else

{

return the solution $x = x^{(M)}$

}

Output : x, solution of $Px = b$

3.8.3 SIRT

The *Simultaneous Iterative Reconstruction Technique* (*SIRT*) is a variation on ART which gives slightly better images in practice, at the expense of a slightly slower algorithm [1]. In the SIRT algorithm, all (up to M nonzero) updates are computed for each cell j of the model, for each ray that passes through cell j. The set of updates for cell j are then averaged before updating the appropriate model element (solution element) x_j.

Let K_j $(j = 0, ..., N)$ be the number of ray paths that pass through cell j and

$$\Delta x_j = \begin{cases} \Delta x_j + \frac{b_{i+1}}{L_{i+1}} - \frac{\sigma_{i+1}}{c\Sigma_{i+1}} & \text{cell} \quad j \quad \text{in ray path} \quad i+1, \\ \Delta x_j & \text{cell} \quad j \quad \textbf{not} \text{ in ray path} \quad i+1. \end{cases} \tag{3.8.26}$$

Algorithm 3.12.

Input: P ($M \times N$ matrix) and b ($N \times 1$ matrix) (2.2.5)

Step 1. Let $x = 0$ be the initial solution

Step 2. Let $\Delta x = 0$

Step 3. **for** $i = 0, ..., M-1$, $j = 1, ..., N$

{

Calculate Δx_j using Equation 3.8.26

}

Step 4. **for** $j = 1, ..., N$

{

$x_j \leftarrow x_j + (\Delta x_j)/K_j$

}

Step 5. **If** the solution has not yet converged, **go to** Step 2.

Output: x, solution of $Px = b$

3.8.4 CGLS Method

The *Conjugate Gradient* (*CG*) *method* by itself can only be applied to positive definite systems of equations, and is thus not directly applicable to general least squares problems. In the *Conjugate Gradient Least Squares* (*CGLS*) *method*, the least squares problem:

$$(\mathcal{P}) \qquad \text{minimize} \quad \|Px - b\|_2, \tag{3.8.27}$$

by normal equations:

$$\text{minimize} \quad \|P^T Px - P^T b\|_2. \tag{3.8.28}$$

Then, by iterating the equation $P^T Px - P^T b$, the residual is tried to be minimized. To avoid round off errors, in the iterations, at iteration step k, $s^{(k)} := Px^{(k)} - b$ is used. The details can be found in [1] (Section 6.3).

3.9 Observations about Existing Works

In this subsection, I will give observations and comments on the existing works from the literature. Here, I will present some challenges which I worked out and will make some constructive proposals.

Early methods for discrete-valued reconstruction aimed at reconstructions of binary arrays by horizontal and vertical projections only. The deterministic projections were treated as a system of linear equations. Attention was paid particularly on ambiguity of the reconstruction. Algorithms for unambiguous reconstruction have been developed under some assumptions, such as connectedness in 2D, or convexity in 3D. However, such assumptions may not hold in real world problems. So, a lot of *prior knowledge* is needed (h-convexity, v-convexity, hv-convexity, etc.).

Probabilistic methods try to satisfy some probability distribution under the projection constraints (Section 3.2, Section 3.3). Therefore, the resulting reconstruction is a mixture between an image that has the probability distribution used in the method, and the projection information.

Moreover, they do not initially start with a reconstruction using the projections. In the iteration, they eliminate (*penalize* as in Section 3.2) or modify (*using ICD/EM* as in Section 3.3) the images that do not match projections. Although they give results that match projection information, the result may not reflect the real density distribution. Because, the main aim is to produce images that converge to an image which has the desired probability distribution. If the probability distribution used is not convenient for the object, then a probability distribution which gives good approximation to the real object must be found. To get such a distribution itself is also another hard problem to solve.

There is a known disadvantage of the probabilistic approach mentioned in Section 3.2, namely, the algorithm may get stuck in local minima. To resolve this problem, additional prior information, such as the typical number and sizes of objects may be needed (cf. [24]).

For the work discussed in Section 3.7, it can be said that, there are new ideas from different areas, and the works introduces pioneering approaches to DT. If some comments needs to be stated, at first glance, I find the *(a)* preprocessing *(adding and removing additional pixels)*, *(b)* finding symmetries, *(c)* transformation from lattice set to the linear code-words (encoding) and the other-way (decoding), and *(d)* error analysis, may introduce expensive computation cost. On the other hand, the authors use statistical learning methods which aim at a saving of unnecessary costs also.

Chapter 4

Semi-conductors and VLSI Microchip Design

4.1 Introduction

The birth of the transistor in 1947 represents the start of the semiconductor industry. Since then, semiconductor manufacturing and fabrication techniques have advanced significantly (Figure 4.1). Many individual transistors can now be fabricated and interconnected to form complex "integrated circuits". Semiconductors called *Very Large Scale Integration circuits* (*VLSI*), often containing millions of transistors, are presently being manufactured. Image on the top left of Figure 4.1 is a *Triode* due to Lee De Forest in 1906. The top right image in that figure if the first *point contact transistor(germanium)* developed by John Bardeen and Walter Brattain in Bell Laboratories in 1947. The image on the lower left is the first *integrated circuit(germanium)* developed by Jack S. Kilby at Texas Instruments in 1958. The lower right image shows one of the today's chips manufactured by Intel Corporation.

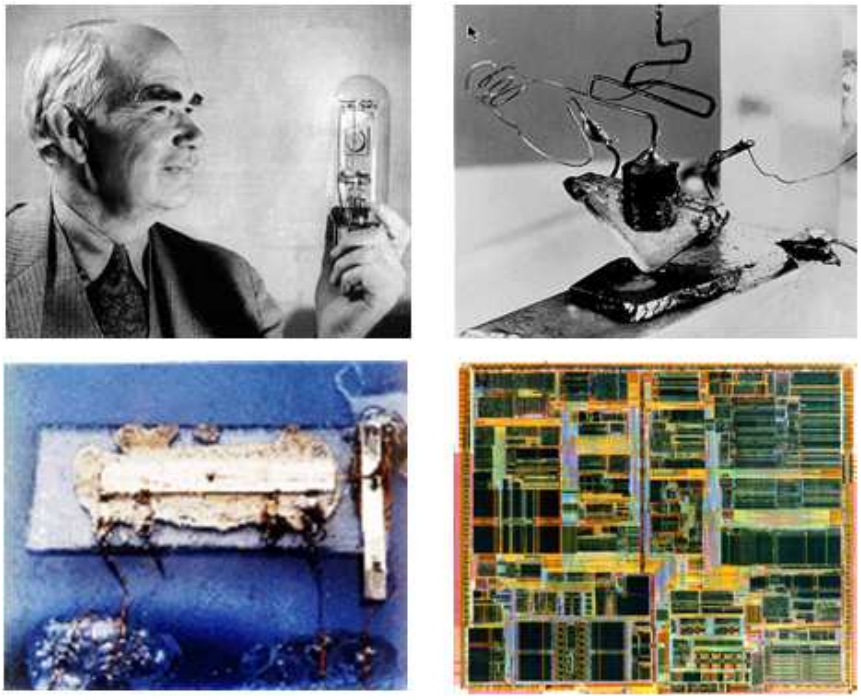

Figure 4.1: Illustration of development of transistor technology up to semiconductor chips era [39].

The electronics industry has achieved a phenomenal growth over the last two decades, mainly due to the rapid advances in integration technologies, large-scale systems design, in short, due to the advent of VLSI. The number of applications of integrated circuits in high-performance computing, telecommunications, and consumer electronics has been rising steadily, and at a very fast pace. Typically, the required computational power of these applications is the driving force for the fast development of this field.

As more and more complex functions are required in various data processing and telecommunications devices, the need to integrate these functions in a small system/package is also increasing. The level of integration as measured by the number of logic gates in a monolithic chip has been steadily rising for almost three decades, mainly due to the rapid progress in processing technology and interconnect technology. Table 4.1 shows the evolution of logic complexity in integrated circuits over the last three decades, and marks the milestones of

each era. Here, the numbers for circuit complexity should be interpreted only as representative examples to show the order-of-magnitude. A logic block can contain anywhere from 10 to 100 transistors, depending on the function.

ERA	DATE	COMPLEXITY (logic blocks per chip)
Single transistor	1959	less than 1
Unit Logic(1 gate)	1960	1
Multi-function	1962	2-4
Complex function	1964	5 - 20
Medium Scale Integration	1967	20 - 200 (MSI)
Large Scale Integration	1972	200 - 2000 (LSI)
Very Large Scale Integration	1978	2000 - 20000 (VLSI)
Ultra Large Scale Integration	1989	20000 - ? (ULSI)

Table 4.1: Evolution of logic complexity in integrated circuits [35].

The design flow starts from the algorithm that describes the behavior of the target chip. The corresponding architecture of the processor is first defined. It is mapped onto the chip surface by floor planning. The next design evolution in the behavioral domain defines *finite state machines* which are structurally implemented with functional modules such as registers and *arithmetic logic units* (ALUs). These modules are then geometrically placed onto the chip surface using CAD tools for automatic module placement followed by routing, with a goal of minimizing the area and signal delays. The third evolution starts with a behavioral module description. Individual modules are then implemented with leaf cells. At this stage, the chip is described in terms of logic gates, which can be placed and interconnected by using a cell placement and routing program. The last evolution involves a detailed Boolean description of leaf cells followed by a transistor level implementation of leaf cells and mask generation. In standard-cell based design, leaf cells are already pre-designed and stored in a library for logic design use.

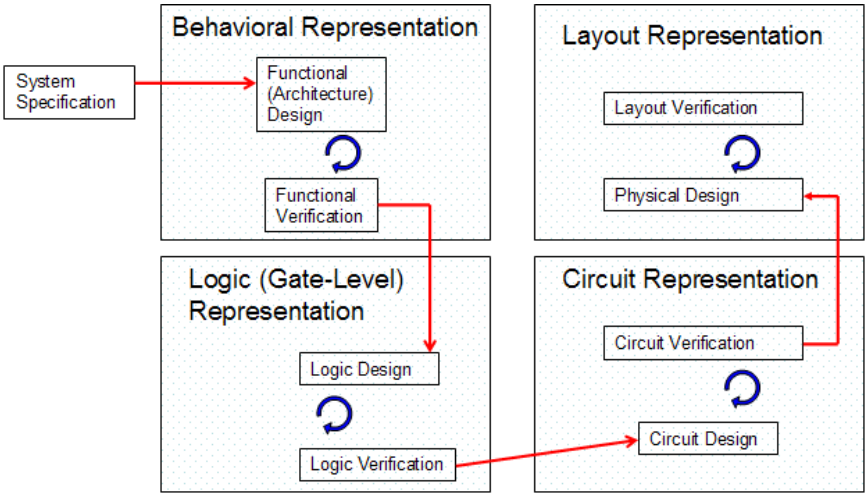

Figure 4.2: Simplified linear VLSI design flow [35].

This thesis is concerned with the production of VLSI chips from the point of view of material testing and after production testing, which is in the production and packaging phase.

4.2 Fabrication: From Silicone to VLSI

Semiconductor circuits are initially manufactured in what is called *wafer* form. A *wafer* is a circular slice of silicon used as a foundation upon which many individual circuits are built. An individual circuit within a wafer is called a *die*, with *dice* being the plural form of the word. Each die is isolated from, and completely independent of, all other dice contained within the wafer.

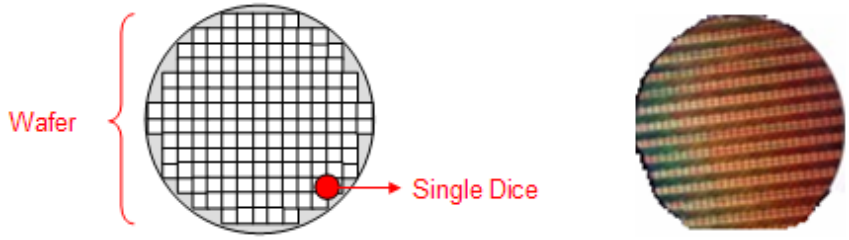

Figure 4.3: Illustration of a wafer and dice on it. Right : A 300mm diameter wafer (Intel Corp.) [37].

Figure 4.4: Illustration of a silicone ingot and cut wafers [37].

Wafers are obtained by cutting circular slices from a *silicone ingot* (Figure 4.4). A wafer has a flat spot or notch which is used to insure proper orientation during the fabrication and testing process. Wafers often have process monitor dice which are the same on all wafers regardless of the product (Figure 4.5). Since these process monitor dice are the same on all wafers, their electrical characteristics are known and checked at specific points during the fabrication process to verify that the process is being performed correctly. Ink dots mark bad dice.

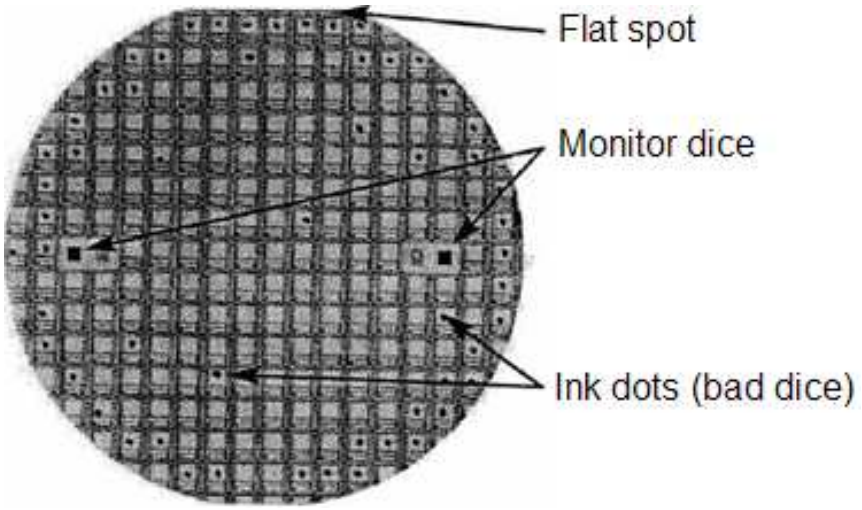

Figure 4.5: Illustration of a wafer with flat spot, monitor dice and ink dots (bad dice) [41].

When the manufacturing process is complete, each die must be thoroughly tested. Testing a wafer is called *wafer probing* or *die sort*. During this process, each die is tested to insure that it properly meets its device performance specification. This involves verifying voltages, currents, timings and functionality. When a die does not meet its specification, it is marked to indicate that it has failed the test process. Failures are typically indicated by placing an ink dot on defective dice.

After all dice on the wafer are probed, the wafer is cut to separate the dice. This is known as sawing the wafer. Any defective dice shown by an ink dot are thrown away. The production may be summarized as shown in Figure 4.6.

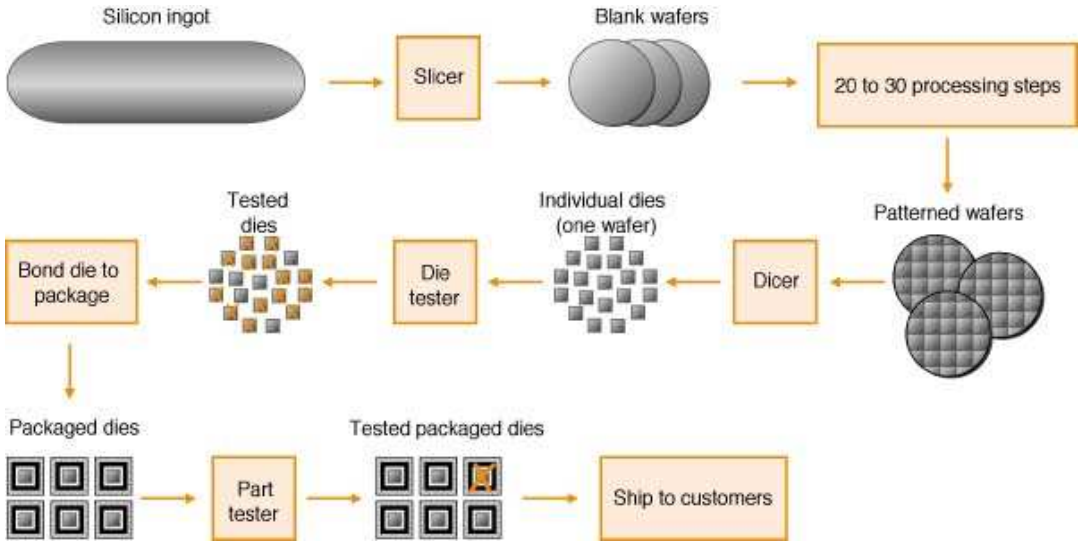

Figure 4.6: Illustration of a fabrication from silicone to VLSI chips [40].

The key material in VLSI production are ***Polysilicon, Silicon Dioxide*** (SiO_2), ***Aluminium***, ***Copper***. Typical impurities that can be found in semiconductor is the presence of ***Arsenic*** and ***Boron***. SiO_2 is used to insulate transistor gates and to insulate layers of wires. The insulators can be grown on Silicon or chemically deposited. Polysilicon (polycrystalline silicon) is the key material for transistor gates and also it is used for short wires by adding chemical deposition. Aluminum or Copper metals are used for wiring between transistors.

Transistors and wiring are made from many layers (usually between 10 and 15) built on top of one another. The first half-dozen or so layers define transistors, and the second define the metal wires between transistors. ***Lambda*** (λ) is the smallest resolvable feature size imprinted on the *integrated circuit* (IC). Roughly λ is half the length of the smallest transistor, approximately $0.2\mu m$ in current technology.

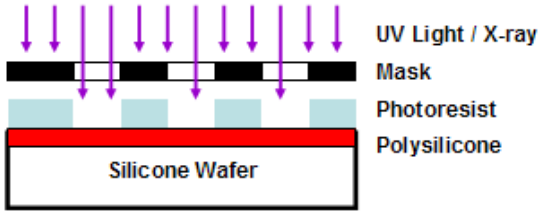

Figure 4.7: Illustration of a silicone wafer patterning via UV or X-rays [37].

In processing wafers, the wafers are patterned. The approach is roughly as follows (in order):

1. Adding materials (such as metal wires, polysilicon gates, oxide) on to silicone wafer surface,
2. Coating the materials with a kind of filter called ***photoresist*** (***PR***),
3. Coating the PR with a mask which filters the materials from light source (e.g., UV, X-rays),
4. Exposing light (e.g., UV, X-rays) to the surface of the filter (Figure 4.7),
5. Removing exposed PR and material.

So, another application area of DT can be detecting the material exposed in the patterning of a wafer.

4.3 Applications of Discrete Tomography in VLSI Microchip Production

After computer usage has started to increase and the computer prices dropped to manageable prices for personal use, the production has also increased. However, while this need was increasing, although the prices of the chips seemed to drop, the producers started to investigate the ways of lowering the prices.

In the process of VLSI production, silicone material plays an important role. To be able to print a circuit on the silicone material, the wafers should be homogenous and must not contain any holes, cracks, or impurities. Otherwise, most of the effort will be wasted. Table 4.2 shows the importance of the cost of the production per die.

Chip	Wafer ($)	Defect per cm^2	Area (mm^2)	Dies/Wafer	Yield	Die ($)
386DX	$900	1.0	43	360	71%	$4
486DX2	$1200	1.0	81	181	54%	$12
PowerPC 601	$1700	1.3	121	115	28%	$53
HP PA 7100	$1300	1.0	196	66	27%	$73
DEC Alpha	$1500	1.2	234	53	19%	$149
SuperSPARC	$1700	1.5	256	48	13%	$272
Pentium	$1500	1.5	296	40	9%	$417

Table 4.2: Die cost of different manufacturers measured in 1993 [20].

A formulation which can be found in [20], is as follows, which underlines the importance of *die yield*.

$$\text{DieCost} = \frac{\text{WaferCost}}{\text{DiesPerWafer} \times \text{DieYield}}, \tag{4.3.1}$$

where *die yield* is,

$$\text{DieYield} = \text{WaferYield} \times \left(1 + \frac{\text{DefectsPerUnitArea} \times \text{DieArea}}{\alpha}\right)^{-\alpha}, \tag{4.3.2}$$

with $\alpha \approx 4$. So, from Equation (4.3.1), die cost goes up with the order of 4 of *defects per die area*. To lower the costs, one possible way is to detect the defects on the wafer, before it goes to production, or after the production for the physical material testing as shown in Figures 4.5 and 4.6. To detect the defects

X-ray tomography can be used [50]. This is where **discrete tomography** comes in to play. Another application area of X-ray tomography, specific to VLSI production, is the wafer patterning as described in Section 4.2 (Figure 4.7).

The wafers being inspected have to be analyzed with special instruments (Figure 4.8). These systems take projections (line sums) of the material, and by using these projections the inner structure is tried to be reconstructed.

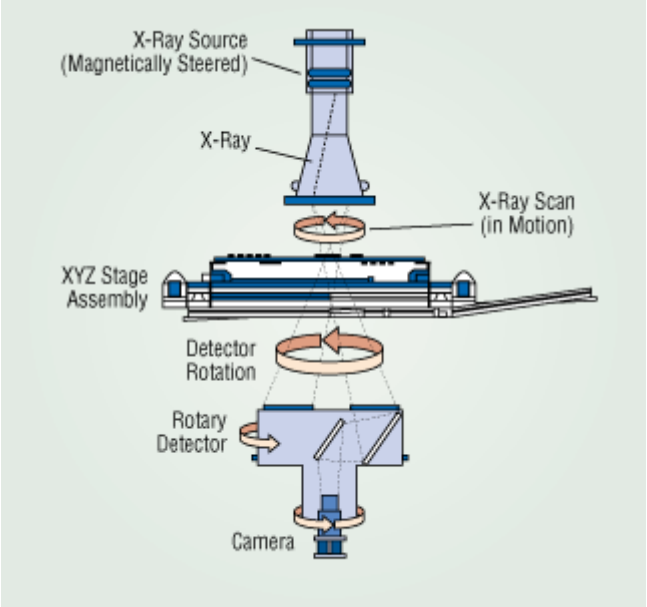

Figure 4.8: Illustration of an X-ray system used in tomography of silicone wafers [50].

Some approximate silicone surface reconstructions will reveal the necessary information of the quality of the material. The applicability of the *X-ray* tomography together with discrete tomography can also be used after the production (Figure 4.9). Then, although the tests are not limited to surface and material testing, this would save time and minimize the costs.

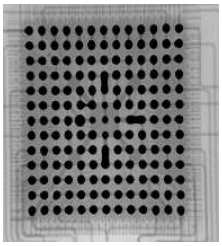

Figure 4.9: Illustration of defected integrated circuit (IC) [50].

Besides the applications of quality testing, with the use of X-rays in wafer patterning on exposing onto photoresist (PR), Discrete Tomography can be used to selectively remove materials in the production also (Section 4.2, Figure 4.7).

Chapter 5

Proposed Solution

5.1 Motivation

In the following sections I will present my ideas. Before entering into details, I want to make clear the objective, scope of the thesis and the connection between DT and VLSI microchip design. In this way, a small summary will also be given.

In VLSI microchip production, one of the main materials is the silicone material on which the VLSI chips are printed (Section 4.1, Figure 4.6). The silicone material should not contain too much defects (e.g., impurities, holes) in order to produce a quality product and to increase the yield. If a low quality silicone material is used, the yield from a silicone wafer decreases and because of this, the cost of a single chip increases as given in Table 4.2. Therefore, the silicone should be investigated, and *any defects should be revealed before and after the production.*

To investigate the silicone material, a good technique that can be used is *X-ray tomography.* The process is simply:

1. Obtaining the projections from different angles,
2. Using the projections, reconstructing an approximation to the original material is performed using a reconstruction algorithm.

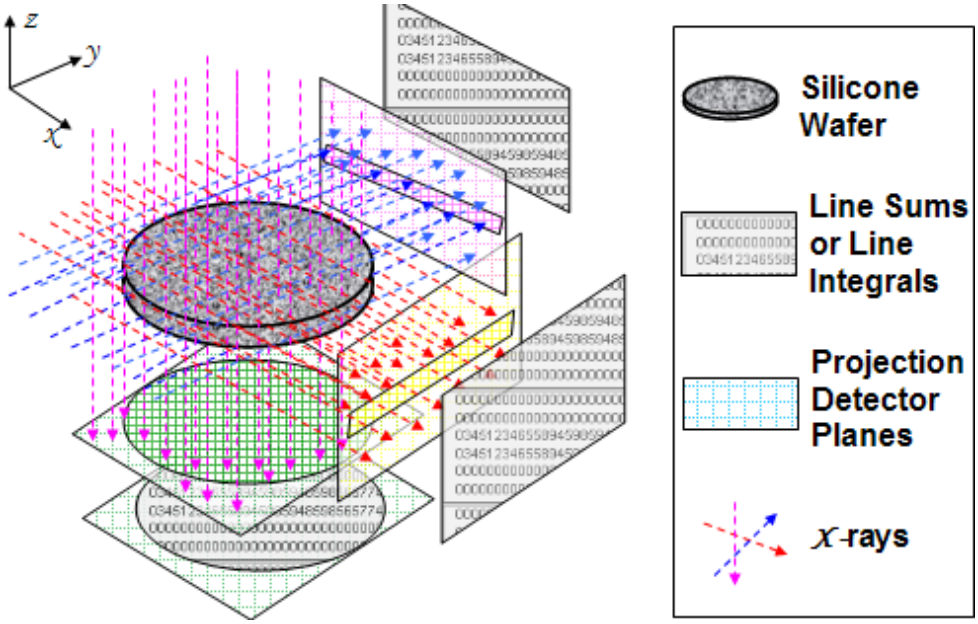

Figure 5.1: Illustration of obtaining 3 orthogonal projections from a silicone wafer.

Since the silicone ingot has been processed to purify it before cutting the silicone wafers, the silicone material contains smaller number of different atoms or molecules. So, this makes modeling the object, as a finite set over the integer lattice set, possible. Using the prior knowledge that the object has smaller number of possible values, the results of DT can be used.

In Section 2.2, the problem has been expressed as a linear system of equations. Equation (2.2.5) ($Px = b$) is this system of equations. Here, P is an $(M \times N)$ matrix whose rows represent the line sums for each ray. So, M is the number of rays sent, and N is the number of cells of a rectangular grid which contains the object. Example 2.10 and Example 2.11 are illustrating this for 2-dimensional object. For the 3-dimensional case, the situation is similar. Since ***reconstruction is an inverse problem*** [19, 1] if the system of equations $Px = b$ has less number of rows than the number of columns, this system usually tends to be ***rank-deficient*** and ***ill-conditioned***. This is usually the case, because available number of projections is usually small. To avoid this, more projections from different angles can be obtained. However, obtaining a lot of

projections has a drawback: *Excessive radiation from UV light or X-rays may be harmful to the object being inspected.* Therefore, the reconstruction methods should be able to *solve the problem using* ***less number of projections***.

Another difficulty, to mention, is the *computational complexity of reconstruction* (Table 3.2) of 3-dimensional objects which have more than 1 different types of atoms or molecules. Since the reconstruction of 3-dimensional objects which have more than 1 different types of atoms or molecules, is ***NP-complete*** [14, 25, 15], finding an exact solution in a *feasible time and computer storage* becomes almost impossible. Therefore, instead of trying to find an exact solution, an *approximate solution* is tried to be found via ***optimization techniques***. In this thesis, I will reformulate the reconstruction problem as a constrained optimization problem. Techniques for solving inverse problems (with examples dedicated to tomography) via optimization methods and regularization can be found in [1]. Optimization for more general problems can be found in [21, 38].

Since the domain is discrete and the reconstruction problem of DT is an inverse problem, *optimization techniques which make use of* ***derivatives*** *are not applicable here.* Because there is no derivative to be used. This is another important point to consider. Therefore, optimization techniques which do not depend on the derivatives would be more convenient. Such methods, i.e., methods which do not make use of derivatives directly (or use only a rough approximation of derivatives), are called ***derivative free optimization*** (***DFO***) ***methods***. Some of these DFO methods are ***trust-region method, simplex method, simulated annealing, genetic algorithms***. These methods are very promising in practice. In this thesis, I will reformulate the reconstruction problem as a constrained optimization problem and solve the problem with a genetic algorithms approach. For more and detailed information on trust-region methods, there is a very good book of A.R. Conn, N.I.M. Gould and P.L. Toint [10].

5.2 Proposed Solution

In this section, I will present my ideas, the implementation of them and the results of the experiments.

I will map the real-world problem into the mathematical model given in Section 2.1 with the notation presented there. Before mapping the real world to a mathematical model, I want to state a few basic assumptions which hold in this section and the following sections. (Indeed, these assumptions are made in real world problems also):

1. *Parallel X-ray beams* are assumed as the projection instrument.
2. It is assumed that the dimensions of the object being inspected is approximately known (also dimensions of its minimum bounding box).
3. It is assumed that the *number of different types of atoms or molecules* is approximately known.

Recalling the mathematical model and the definitions given in Chapter 2, the mapping from real world to the mathematical model is done as follows:

- $\mathbb{Z}^3$ is the lattice corresponding to space,
- F is the 3-dimensional grid where the object to be investigated (e.g., silicone wafer) is contained in,
- *Lattice directions*, $D = (v_1, ..., v_q)$ $(q \geq 2)$, correspond to the directions from which the projections will be obtained,
- *Lattice line* l, corresponds to one X-ray,
- $p_F^{(k)}$ is the projection (line sum) at direction v_k $(k = 1, ..., q)$.

To be familiar with the notation and the mathematical model mapping given above, I will give a 3-dimensional example which is similar to Example 2.10

given in Section 2.2. In this example (Example 5.1), reconstruction of the lattice set, F, will be transformed into a linear system of equations of the form 2.2.5 ($Px = b$) given in Section 2.2.

Example 5.1. Consider the *grid* F (lattice set) given in Figure 5.2. F contains an imaginary object whose atoms are represented by circles. The dimensions of F are: width $= 4$ $(x - axis)$, height $= 3$ $(y - axis)$, depth $= 3$ $(z - axis)$. So, F has $4 \times 3 \times 3 = 36$ points. For each point c of F, define the mapping $c \leftrightarrow x_j$, with $j \leftarrow ((c_z - 1) * width * height + (c_y - 1) * width + (c_x - 1) + 1$, where (c_x, c_y, c_z) are the Cartesian coordinates of points c.

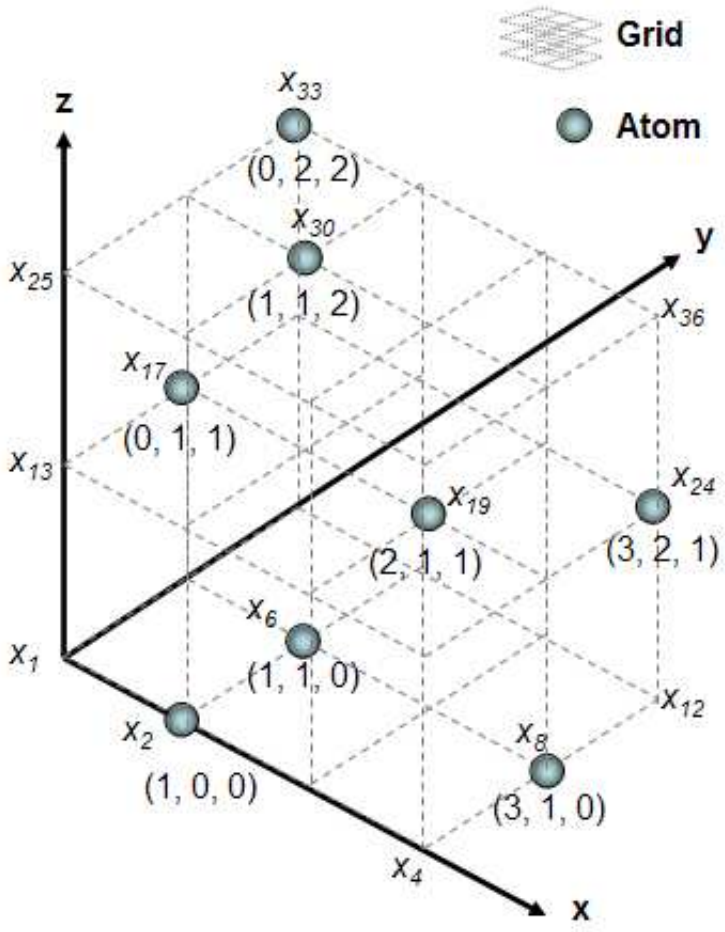

Figure 5.2: Illustration of a 3-dimensional grid and the atoms of an imaginary object.

Here, the vector $x = (x_1, ..., x_{36})^T$ is the unknown vector of the reconstruction problem $Px = b$ (Equation 2.2.5). The atoms of the object reside at the Cartesian coordinates: $(1, 0, 0)$, $(1, 1, 0)$, $(2, 1, 0)$, $(0, 1, 1)$, $(2, 1, 1)$, $(3, 2, 1)$, $(0, 2, 2)$, $(1, 1, 2)$.

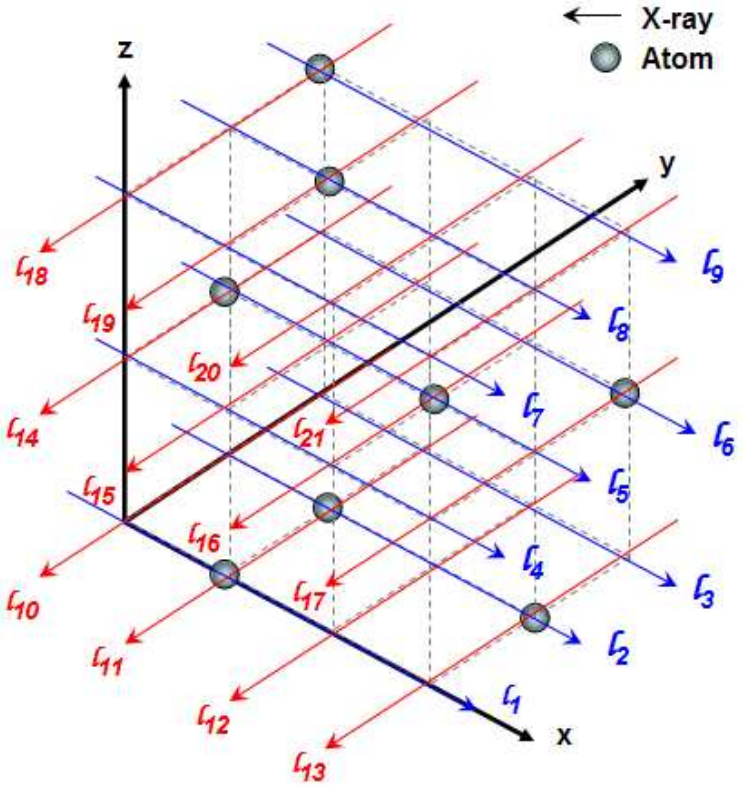

Figure 5.3: Illustration of a 3-dimensional grid and 2 orthogonal X-ray beams.

If the vector $x = (x_1, ..., x_{36})^T$ can be found, then by using the inverse map, it is possible to recover F and therefore the object residing in it. To do this, let us take 2 orthogonal projections with parallel-beam geometry. Choosing the set of projection directions $D = (v_1, v_2)$ (set of lattice directions) with $v_1 = (1, 0, 0)$ and $v_1 = (0, -1, 0)$ will accomplish these 2 orthogonal projections. Then, the collection of the set of lattice lines determined by D will be $L = (L^1, L^2)$, where L^1 is the set of lattice lines parallel to v_1 and L^2 is the set of lattice lines parallel to v_2. In Figure 5.3, $L^1 = l_1, ..., l_9$, and $L^2 = l_{10}, ..., l_{21}$.

Let $P_{i,.}$ be the i^{th} row of P. As explained in Section 2.2, each row of P, i.e., $P_{i,.}$ is the information; *which points of F does ray i passes through.* To be precise, if N is the number of points of F, $P_{i,.} = (P_{i,1}, ..., P_{i,N})$, where $P_{i,j} = 1$ if l_i *passes through* x_j, and $P_{i,j} = 0$ if l_i *does not pass* through x_j $(j = 1, ..., 36)$. For example, in Figure 5.3, l_{13} passes through the point x_8, so, $P_{13,8} = 1$. With the lattice directions determined, there will be 21 rays. So, P will be a (21×36)-matrix of 0's and 1's.

After taking lines sums, the data will be obtained. This is the projection data required. Here, *the number of atoms each ray touched* will be used, i.e., the projection functions will be $p_F^{(k)} : L^{(k)} \rightarrow \mathbb{N}_0$, where k is the index of lattice direction used ($|D| = 2$ $k = 1, 2$ in this example). Using the equation given in Section 2.1 (Equation 2.1.3), the projections are calculated. That is, for each lattice direction v_1 and v_2, the projections $p_F^{(1)}$ and $p_F^{(2)}$ are found. In this example, for direction v_1, the lattice lines $L^1 = (l_1, ..., l_9)$ are used: $p_F^{(1)}(l_{i1}) = |F \cap l_{i1}|$, $(i1 = 1, ..., 9)$. For direction v_2, the lattice lines $L^2 = (l_{10}, ..., l_{21})$ are used: $p_F^{(2)}(l_{i2}) = |F \cap l_{i2}|$, $(i2 = 10, ..., 21)$. Inspecting Figure 5.3, $p_F^{(2)}(l_{11}) = |F \cap l_{11}| = 2$, because the ray l_{11} touches 2 atoms of the object. Furthermore, $p_F^{(1)}(l_3) = |F \cap l_3| = 0$, because the ray, l_3, does not touch any atoms of the object.

After obtaining the projection data $p_F^{(k)}$, the right hand side of the linear system of equations $Px = b$, namely b, can be formed. Since b consists of raywise projection data, it is a (21×1)-vector in this example. Each row of b, b_i, corresponds to the projection value of ray (lattice line) l_i. So, $b_i = p_F(l_i)$ $(i = 1, ..., 21)$. For this example (Figure 5.3), $p_F^{(2)}(l_{11}) = 2$, so, $b_{11} = p_F(l_{11}) = 2$. Finally, $p_F^{(1)}(l_3) = 0$, so, $b_3 = p_F(l_3) = 0$.

My solution proposal consists of:

1. **Reformulation of the reconstruction problem** by a **constrained optimization** formulation,

2. **A new framework** which is based on the reformulation (i.e., designed

for the reformulation),

3. **A new algorithm** to be used in the framework to obtain an initial solution which uses geometric information.

5.2.1 Problem Reformulation

In this section, I will present my problem reformulation. Before presenting my reformulation, I will mention the variations of the feasibility problem given in Section 2.2 by (2.2.5). In literature, for the linear feasibility problem given in Section 2.2 by (2.2.5), different interpretations have been given in terms of reformulations of *linear programming* (LP). That is, the linear feasibility problem (2.2.5) is transformed into the LP problem. The problem of reconstruction is considered as a LP problem as follows:

$$\text{maximize} \quad \sum_{j=1}^{N} x_j, \quad \text{subject to} \quad Px = b \quad \text{where} \quad x \in \{0, \bar{d}\}^N. \tag{5.2.1}$$

Here, N is the number of points of F, and $\bar{d}$ is the number of different types of atoms or molecules of the object. By *relaxing* the constraint $x \in \{0, \bar{d}\}^N$, to $x \in [0, \bar{d}]^N$ by a suitable mapping, the LP problem Equation (5.2.1), the problem turns to be:

$$\text{maximize} \quad \sum_{j=1}^{N} x_j, \quad \text{subject to} \quad Px = b \quad \text{and} \quad x \in [0, \bar{d}]^N. \tag{5.2.2}$$

Now, I will present my reformulation of the reconstruction problem. My reformulation is a ***constrained optimization*** problem formulation.

Before presenting my reformulation, I will define **minimum bounding object (MBO)** of the object which will be reconstructed. Minimum bounding object

(MBO) will be used in my reformulation and it functions as a boundary condition in the reformulation. Moreover, it is used as an initial solution in the framework. In Section 5.2.3, an algorithm is given to construct the MBO.

Definition 5.2. Let F be a lattice set, and O be the object contained in the lattice set F. The ***minimum bounding object*** (***MBO***) of O, is the object when it is embedded into F, it gives projections values *greater or equal to* the projection values of O. Let $p^{(k)}_{F_O}(l)$ be the projection of F, with the original object O embedded, and let $p^{(k)}_{F_{MBO}}(l)$ be the projection of F, with the *minimum bounding object* MBO embedded. Then, ***minimum bounding object***, MBO, of O, is a wrapping object which satisfies the Inequality (5.2.3), when embedded into F instead of O.

$$p^{(k)}_{F_{MBO}}(l) \geq p^{(k)}_{F_O}(l). \tag{5.2.3}$$

To have more idea about minimum bounding object, Figure 5.4 will be useful.

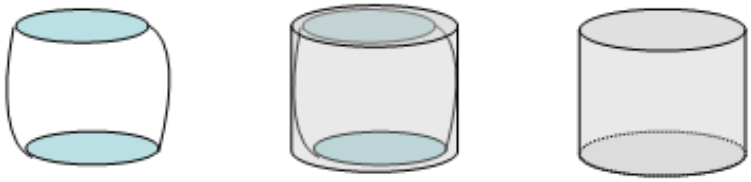

Figure 5.4: Illustration of a minimum bounding box (MBO) of an object O. *Left*: Original object O, *Middle*: MBO covering O, *Right*: MBO.

Now, my reformulation will be presented. Let the solution be x^*, and let the lattice set and the object corresponding to x^* be denoted as F^*, and O^*. My reformulation depends on the following:

1. ***Minimizing*** $\|Px = b\|_1$ as the objective function,
2. The solution, x^*, ***cannot have any points outside of the MBO*** (Constraint 1).

3. The solution, x^*, ***must have non-negative integer values, and the maximum entry of*** x^****, cannot be greater than the number of different types of atoms found in original object*** O (Constraint 2).

4. The solution, x^****, should not contain less cells than the sums of line sums*** (Constraint 3).

To be precise, my reformulation of the reconstruction problem of DT, is:

$$(\mathcal{P}) \begin{cases} \text{minimize} & \|Px - b\|_1 \\ \text{subject to} & p^{(k)}_{F_{MBO}}(l) \geq p^{(k)}_{F^*_{\bar{O}^*}}(l), \\ & x \in \{0, 1, ..., \bar{d}\}^N, \\ & \sum_{j=1}^{N} \chi_j \geq \sum_{i=1}^{M} b_i, \quad \chi_j = \begin{cases} 0, & \text{if} \quad x_j = 0, \\ 1, & \text{otherwise.} \end{cases} \end{cases} \tag{5.2.4}$$

5.2.2 A New Framework

My reformulation is a constrained optimization interpretation of the reconstruction problem of DT. There are known methods to solve constrained optimization problems. However, some methods can be applied to unconstrained problems more suitably. To make use of those methods, constrained optimization problems can be transformed into unconstrained optimization problems. Although some restrictions may apply for some problems, there are effective ways of this transformation. For the details of such transformations, [21, 38] can be referred.

Here, before introducing my proposed framework, I want to give a transformation to transform $(\mathcal{P})$ (constrained optimization problem) into a unconstrained optimization problem. Although, generic methods can be used [21, 38], it will be more suitable to do the transformation according to the domain of this thesis.

Definition 5.3. Denote the algorithm chosen to solve $(\mathcal{P})$, by $\mathcal{A}$. Let $x_{current}$ be the solution at the *current* iteration of the algorithm $\mathcal{A}$. Then, the func-

tion, $fitness_{MBO}(x_{current}) : \mathbb{R}^N \to \mathbb{N}_0$, is ***fitness of the solution to the constraint 1 of*** $(\mathcal{P})$, and is defined by:

$$fitness_{MBO}(x_{current}) := \|p^{(k)}_{F_{MBO}}(l) - Px_{current}\|_1. \tag{5.2.5}$$

Definition 5.4. Denote the algorithm chosen to solve $(\mathcal{P})$ by $\mathcal{A}$. Let $x_{current}$ be the solution at the *current* iteration of the algorithm $\mathcal{A}$. Then, the function, $fitness_{Discrete}(x_{current}) : \mathbb{R}^N \to \mathbb{N}_0$, called the ***fitness of the solution to the constraint 2 of*** $(\mathcal{P})$, is defined by:

$$fitness_{Discrete}(x_{current}) := \begin{cases} 0, & \text{if} \quad x_j \in \{0, 1, ..., \bar{d}\} \\ 1, & \text{otherwise.} \end{cases} \tag{5.2.6}$$

Definition 5.5. Denote the algorithm chosen to solve $(\mathcal{P})$ by $\mathcal{A}$. Let $x_{current}$ be the solution at the *current* iteration of the algorithm $\mathcal{A}$. Then the function, $fitness_{CellCount}(x_{current}) : \mathbb{R}^N \to \mathbb{N}_0$, called the ***fitness of the solution to the constraint 3 of*** $(\mathcal{P})$, is defined by:

$$fitness_{CellCount}(x_{current}) := (\sum_{j=1}^{N} \chi_j - \sum_{i=1}^{M} b_i), \quad \chi_j = \begin{cases} 0, & \text{if} \quad x_j = 0 \\ 1, & \text{otherwise.} \end{cases} \tag{5.2.7}$$

Definition 5.6. Denote the algorithm chosen to solve $(\mathcal{P})$ by $\mathcal{A}$. Let $x_{current}$ be the solution at the *current* iteration of the algorithm $\mathcal{A}$. Let $\omega, \alpha, \beta, \gamma \in \mathbb{R}$ be real-valued constants. Then, the function, $fitness_P(x_{current}) : \mathbb{R}^N \to \mathbb{N}_0$, called the ***fitness of the solution of*** $(\mathcal{P})$, is defined by:

$$\begin{aligned} fitness_P(x_{current}) := \quad & \omega \ \|Px_{current} - b\|_1 + \\ & \alpha \ fitness_{MBO}(x_{current}) + \\ & \beta \ fitness_{Discrete}(x_{current}) + \\ & \gamma \ fitness_{CellCount}(x_{current}). \end{aligned} \tag{5.2.8}$$

Definition 5.7. Choosing $\omega, \alpha, \beta, \gamma \in \mathbb{R}$ suitably according to the algorithm $\mathcal{A}$, the constrained optimization problem can be converted to an unconstrained

form. Denote the ***unconstrained form of*** $(\mathcal{P})$ (Definition 5.2.4) by $(\mathcal{P}_\mathcal{U})$. Then, it will be:

$$(\mathcal{P}_\mathcal{U}): \qquad \text{minimize} \quad fitness_\mathcal{P}(x). \tag{5.2.9}$$

With these definitions, my framework can be presented.

Algorithm (Framework) 5.8.

Step 1. Form the matrices P, b as illustrated by Example 5.1

Step 2. Construct an initial solution $x^{(0)}$, satisfying constraints of $(\mathcal{P})$

Step 3. Choose an algorithm $\mathcal{A}$ to solve $(\mathcal{P})$

Step 4. **if $\mathcal{A}$ can solve constrained optimization problems** (like $(\mathcal{P})$)

{

Solve $(\mathcal{P})$, using $\mathcal{A}$

}

else

{

Transform $(\mathcal{P})$ into $(\mathcal{P}_\mathcal{U})$

Solve $(\mathcal{P}_\mathcal{U})$, using $\mathcal{A}$

}

5.2.3 Supplementary Algorithm

Algorithm 5.9. MBO Construction:

Input : P, b, $\bar{d}$, ($width$, $height$, $depth$) of F, lattice directions D

Step 1. Let $F_{MBO_{L^{(k)}}} \leftarrow [0]_{(width \times height \times depth)}$ $(k = 1, ..., |D|)$

Step 2. **for** $k = 1, ..., |D|$

{

for each $l \in L^{(k)}$

{

if l **passes through** x_j

{

$c_x \leftarrow mod(j-1, width)+1$

$c_y \leftarrow mod((j-c_x)/width, height)+1$

$c_z \leftarrow (j-c_x-(c_y-1)width)/(width \times height)+1$

$F_{MBO_{L^{(k)}}}[c_x, c_y, c_z] \leftarrow \bar{d}$

}

}

}

Step 3. Let $F_{MBO} \leftarrow [0]_{(width \times height \times depth)}$ $(k = 1, ..., |D|)$

Step 4. **for** $j = 1, ..., N$

{

$c_x \leftarrow mod(j-1, width)+1$

$c_y \leftarrow mod((j-c_x)/width, height)+1$

$c_z \leftarrow (j-c_x-(c_y-1)width)/(width \times height)+1$

if for all $k = 1, ..., |D|$, $F_{MBO_{L^{(k)}}}[c_x, c_y, c_z]) = \bar{d}$

{

$F_{MBO}[c_x, c_y, c_z] \leftarrow \bar{d}$

}

}

Output : F_{MBO} grid (lattice set) containing MBO of O.

To make the MBO Construction Algorithm (5.9) clear, an example will be given.

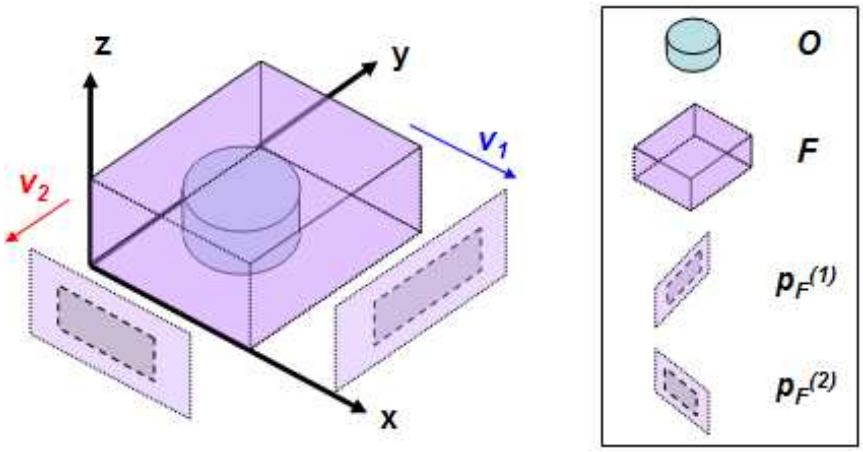

Figure 5.5: Illustration of F, O, v_1, v_2, $p_F^{(1)}$ and $p_F^{(2)}$ used in Example 5.10.

Example 5.10. Let O (cylindrical shape) be the object residing in the grid (lattice set) F as illustrated in Figure 5.5. Assume, two lattice directions $v_1 = (1, 0, 0)$, and $v_2 = (0, -1, 0)$ were chosen and let $p_F^{(1)}$ and $p_F^{(2)}$ be the line sums (projections) in the directions v_1 and v_2.

At *Step 2* of Algorithm 5.9, for each lattice direction v_k, the grid (lattice set) $F_{MBO_{L^{(k)}}}$ is filled with the value $\bar{d}$, along the lattice lines, l, as if the lattice set F is full of that object in the direction v_k. Figure 5.6 shows $F_{MBO_{L^{(1)}}}$ in direction v_1, and Figure 5.7 shows $F_{MBO_{L^{(2)}}}$ in direction v_2.

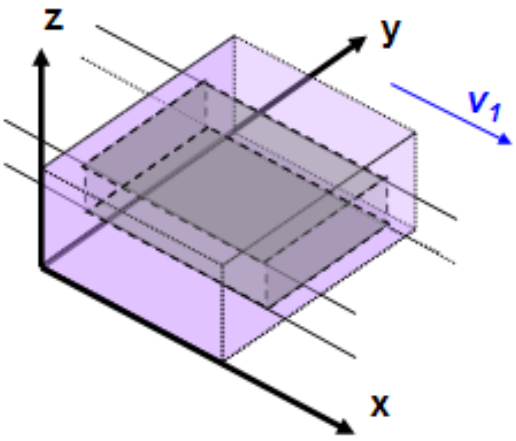

Figure 5.6: Illustration of $F_{MBO_{L^{(1)}}}$ used in Example 5.10, showing Step 2 of Algorithm 5.9.

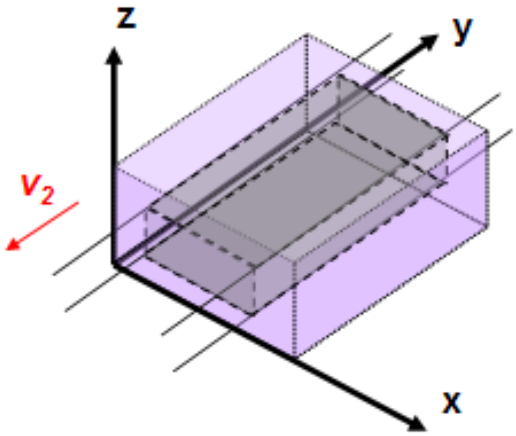

Figure 5.7: Illustration of $F_{MBO_{L^{(2)}}}$ used in Example 5.10, showing Step 2 of Algorithm 5.9.

At *Step 4* of Algorithm 5.9, in fact, intersection $\bigcap_{k=1}^{|D|} F_{MBO_{L^{(k)}}}$ is calculated. The resulting intersection is the desired F_{MBO}. A geometric illustration is given in Figure 5.8.

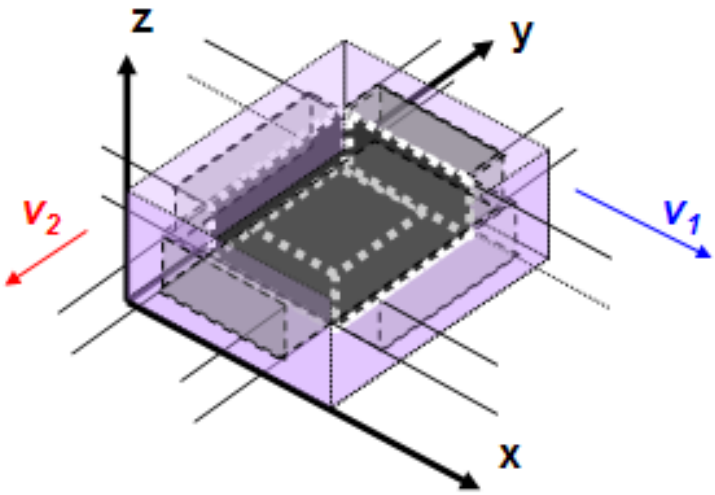

Figure 5.8: Illustration of intersection of $F_{MBO_{L^{(1)}}}$ and $F_{MBO_{L^{(2)}}}$, showing Step 4 of Algorithm 5.9.

The resulting MBO, the lattice set F and the original object O will be as shown in Figure 5.9:

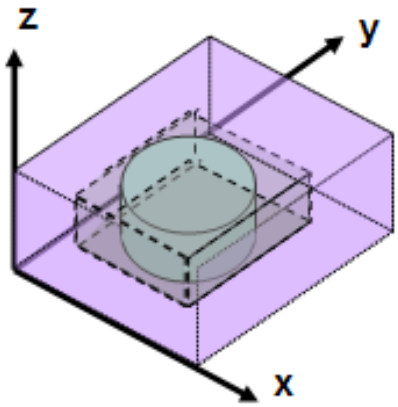

Figure 5.9: Illustration of the resulting MBO, the lattice set F and the original object O for Example 5.10.

5.2.4 Implementation Details

5.2.4.1 Framework

The proposed solution with its supplementary algorithm is implemented in MATLAB.

The algorithm, $\mathcal{A}$, used in the framework (Framework 5.8) is a *Genetic Algorithm* (*GA*). The pool size of GA is chosen as 100. The MBO of the object to be reconstructed is chosen as the initial solution $x^{(0)}$ and the first two entries of the pool is initialized with MBO.

The number of mutations to perform is determined by $\|Px^{(0)} - b\|_1$, where $x^{(0)}$ is MBO of the object to be reconstructed.

The coefficients ω, α, β and γ of the unconstrained optimization problem $(\mathcal{P}_\mathcal{U})$ given in Equation (5.2.8) are determined by experimenting exhaustively in the interval $(0, 1]$. According to these experiments, on the average for $\omega = 0.8$, $\alpha = 0.6$, $\beta = 0.5$ and $\gamma = 0.3$, the solution converges better to the original object. It has also been observed that, when the noise increases, the ω should be decreased and α should be increased to obtain better results.

5.2.4.2 Experiment Data

The objects to be reconstructed in the experiments are hand-made binary 3-dimensional objects. The experiments are done with 5 different types of sample binary 3-dimensional objects with lattice dimensions ($10\times10\times10$), ($50\times50\times50$), ($16 \times 16 \times 3$), and ($16 \times 16 \times 10$).

One of the sample binary (has only $\bar{d} = 1$ type of atom) object is shown in Figure 5.10. It is a cylindrical shape with dimensions ($16 \times 16 \times 3$).

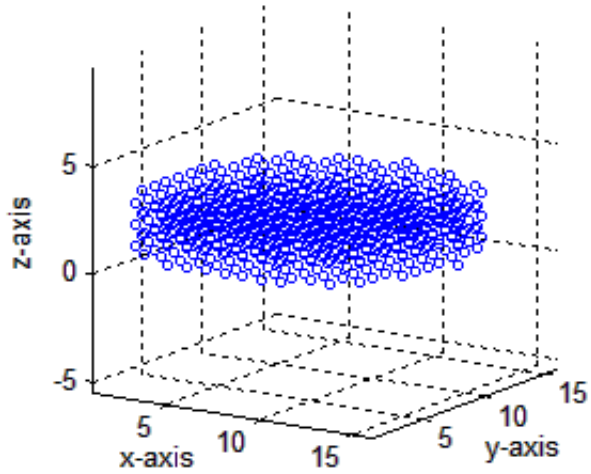

Figure 5.10: Illustration of 3-dimensional binary sample object used in experiments.

5.2.4.3 Projections

The projections are simulated, since no real data could be obtained. The lattice directions $D = (v_1, ..., v_q)$ $(q \geq 2)$ are given to the simulation program as (θ, ϕ), where $0 \geq \theta \geq 90$ is the angle of the direction on $xy - plane$, and $0 \geq \phi \geq 90$ is the angle of the direction on $xz - plane$, and both are in degrees.

Figure 5.11 shows 3 orthogonal projection images of the binary sample object given in Figure 5.10. The gray-scale coloring shows the projections values. The "black" pixels represent non-existence of atoms or particles, while the lighter colors represent higher intensity of atoms on that ray of projection.

Figure 5.11: Projections of the sample object (Figure 5.10), *Left*: $(\theta = 0, \phi = 0)$, *Middle*: $(\theta = 0, \phi = 90)$, *Right*: $(\theta = 90, \phi = 0)$.

The projection resolution is determined by the minimum distance of projected points on the projection plane.

5.2.4.4 Error and Noise

To simulate the measurement errors, *zero-mean Gaussian noise* is added to the projections. That is, the noise is added to b of the system of equations $Px = b$.

Let e be the *zero-mean Gaussian noise* vector. After obtaining the projections b which do not contain any error, the simulated error e is added to b. Then, the experiments are done with the updated b.

The error measure used is ***signal-to-noise ratio***, often written ***S/N*** or ***SNR***. It is defined as the measure of signal strength relative to background noise. The ratio is usually measured in *decibels* (dB).

$$SNR := 20 \log_{10}(\frac{\|b\|_2}{\|e\|_2}). \tag{5.2.10}$$

For SNR measurement, Equation (5.2.10) is used. Here, the projection data b is the signal and e is the noise. The experiments are done with 3 SNR values of 20 dB, 15 dB, 10 dB and without noise.

5.2.5 Experiment Results

The experiments are done with 5 different types of sample binary 3-dimensional objects with lattice dimensions ($10 \times 10 \times 10$), ($50 \times 50 \times 50$), ($16 \times 16 \times 3$), and ($16 \times 16 \times 10$). The lattice directions for projection simulations chosen were: 3 orthogonal, and equally spaced 9 and 16 directions. Because of memory issues arising from my inefficient implementations, neither with my proposed method, nor with the methods presented in Chapter 3 were successful. However, smaller sized problems have been solved with satisfactory results. Here, I will present the experiment results of only 2 sample objects with 3 orthogonal projections and 16 projections of equally spaced directions. These objects to

be reconstructed will be denoted as *Sample Object 1* and *Sample Object 2*. The samples are 3-dimensional, and binary (has only $\bar{d} = 1$ type of atom). In the experiments, $\omega = 0.8$, $\alpha = 0.6$, $\beta = 0.5$ and $\gamma = 0.3$ values are used.

5.2.5.1 Experiment Results on Sample Object 1

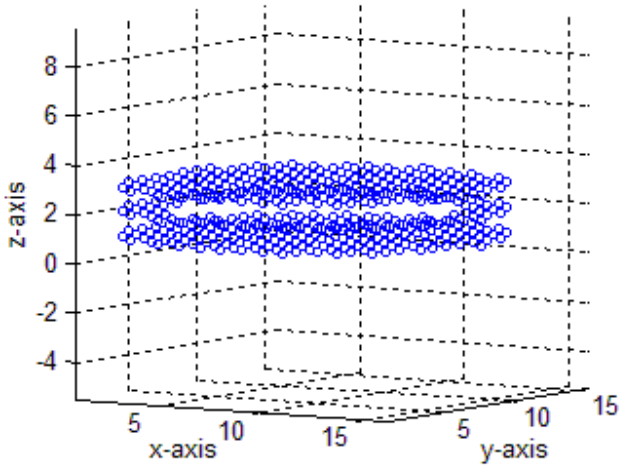

Figure 5.12: Sample object experiment ($16 \times 16 \times 3$).

Consider the sample object, shown in Figure 5.12, used for experimenting. This sample object to be reconstructed has the following properties:

1. Sample object is in a ($16 \times 16 \times 3$) grid F.
2. Sample object has 1 type of atom (i.e. $\bar{d} = 1$, or binary).
3. Sample object is a cylindrical shape with a cylindrical hole in it.

The sample object shown in Figure 5.12, has been experimented with 3 orthogonal and 16 projections using my framework (Framework 5.8) with $\mathcal{A}$ chosen as a genetic algorithms.

Here, I will present the reconstruction of sample object shown in Figure 5.12 using 3 orthogonal projections with lattice directions $D = (v_1 = (\theta = 0, \phi = 0), v_2 = (\theta = 0, \phi = 90), v_3 = (\theta = 90, \phi = 0))$.

Figure 5.13: 3 orthogonal projection images of the sample object shown in Figure 5.12.

The 3 orthogonal projections are simulated and for each lattice direction, (v_1, v_2, v_3), the line sums are calculated. In Figure 5.13, the projection images are shown for each of the lattice directions.

Reconstruction of sample object shown in Figure 5.12 from 3 orthogonal projections defined above is done using my framework (Framework 5.8) with $\mathcal{A}$ chosen as a genetic algorithm. The genetic algorithm $\mathcal{A}$ is run for 10 iterations.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	85	115	5.54
20 dB	80	153	5.48
15 dB	77	176	5.63
10 dB	75	192	5.59

Table 5.1: Reconstruction results of sample object shown in Figure 5.12 using my framework (Framework 5.8) from 3 orthogonal projections, for different noise values.

For each iteration, some measures are taken. Table 5.1 lists these results. The second column of Table 5.1, namely "Correctly Rec. (%)", is the percentage of correctly reconstructed number of atoms over the total number of cells of

the grid (lattice set F). That is, if $x_{reconstructed}$ is the reconstruction result and $x_{original}$ is the original object model, then the correctly recovered percentage is calculated by:

$$100(1 - \frac{\|x_{reconstructed} - x_{original}\|_1}{N}). \tag{5.2.11}$$

Here, N is the number of cells of the lattice set F. The total time for this reconstruction was around 5.6 seconds.

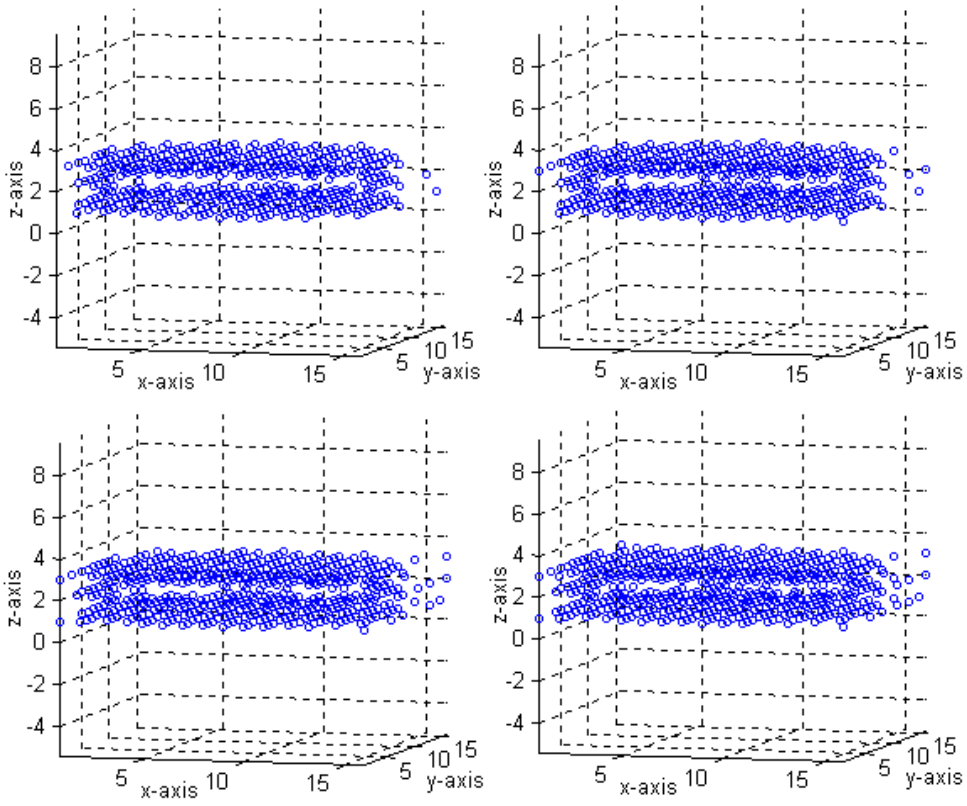

Figure 5.14: Reconstruction of object shown in Figure 5.12 from 3 orthogonal projections using my proposed solution. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Figure 5.14 shows the reconstruction results of *Sample Object 1* (shown in Figure 5.12) from 3 orthogonal projections for different noise values using my proposed solution.

Here, I will present the reconstruction of sample object shown in Figure 5.12 using 16 projections with lattice directions $D = (v_1, ...v_{16})$ where $v_k = \{0, 30, 60, 90\}^2$, $(k = 1, ..., 16)$.

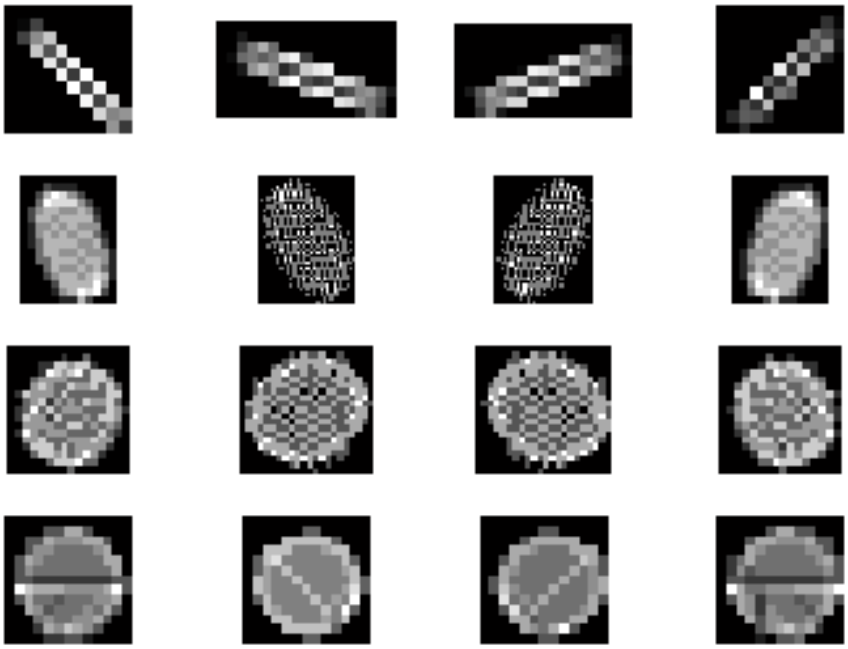

Figure 5.15: 16 projection images of the sample object shown in Figure 5.12.

The 16 projections are simulated and for each lattice direction, $(v_1, ..., v_{16})$, the line sums are calculated. In Figure 5.15, the projection images are shown for each of the lattice directions.

Reconstruction of sample object shown in Figure 5.12 from 16 projections with equal interval lattice directions is done using my framework (Framework 5.8) with $\mathcal{A}$ chosen as a genetic algorithm. The genetic algorithm $\mathcal{A}$ is run for 10 iterations.

For each iteration, some measures are taken. Table 5.2 lists these results. The values of the column "Correctly Rec. (%)" are calculated by using Equation (5.2.11).

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	98	15	66.05
20 dB	93	53	65.67
15 dB	89	84	65.74
10 dB	83	130	66.81

Table 5.2: Reconstruction results of sample object shown in Figure 5.12 using my framework (Framework 5.8) from 16 projections, for different noise values.

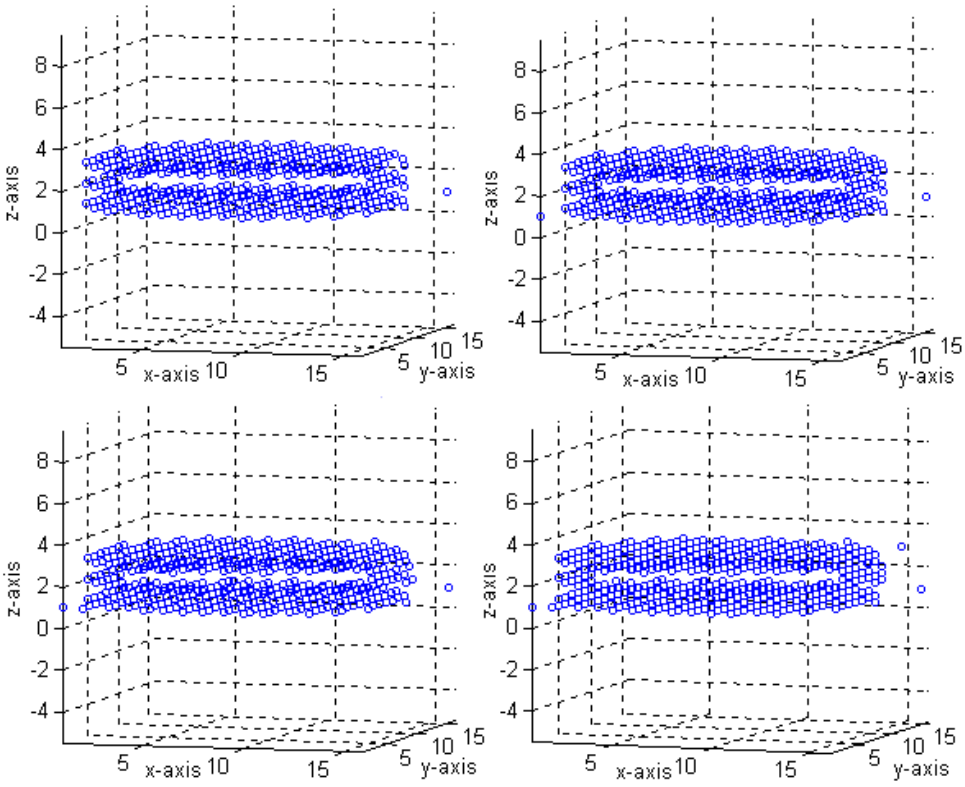

Figure 5.16: Reconstruction of object shown in Figure 5.12 from 16 projections using my proposed solution. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Figure 5.16 shows the reconstruction results of *Sample Object 1* (shown in Figure 5.12) from 16 projections for different noise values using my proposed solution.

5.2.5.2 Experiment Results on Sample Object 2

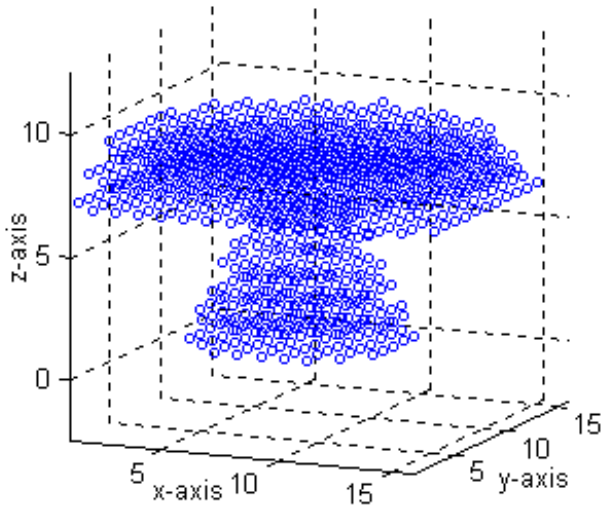

Figure 5.17: Sample object experiment $(16 \times 16 \times 10)$.

Consider the sample object, shown in Figure 5.17, used for experimenting. This sample object has the following properties:

1. Sample object is in a $(16 \times 16 \times 10)$ grid F.
2. Sample object has 1 type of atom (i.e. $\bar{d} = 1$, or binary).
3. Sample object is a cylindrical shape with a torus shaped hollow residing on the outer boundary of the cylindrical shape.

The sample object shown in Figure 5.17, has been experimented with 3 orthogonal and 16 projections using my framework (Framework 5.8) with $\mathcal{A}$ chosen as a genetic algorithms.

Here, I will present the reconstruction of sample object shown in Figure 5.17 using 3 orthogonal projections with lattice directions $D = (v_1 = (\theta = 0, \phi = 0), v_2 = (\theta = 0, \phi = 90), v_3 = (\theta = 90, \phi = 0))$.

Figure 5.18: 3 orthogonal projection images of the sample object shown in Figure 5.17.

The 3 orthogonal projections are simulated and for each lattice direction, (v_1, v_2, v_3), the line sums are calculated. In Figure 5.18, the projection images are shown for each of the lattice directions.

Reconstruction of sample object shown in Figure 5.17 from 3 orthogonal projections defined above is done using my framework (Framework 5.8) with $\mathcal{A}$ chosen as a genetic algorithm. The genetic algorithm $\mathcal{A}$ is run for 10 iterations.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	79	537	38.12
20 dB	75	640	39.71
15 dB	73	698	39.38
10 dB	71	742	38.24

Table 5.3: Reconstruction results of sample object shown in Figure 5.17 using my framework (Framework 5.8) from 3 orthogonal projections.

For each iteration, some measures are taken. Table 5.3 lists these results. The values of the column "Correctly Rec. (%)" are calculated by using Equation (5.2.11). Using only 3 orthogonal projections in the lattice directions $v_1 = (\theta = 0, \phi = 0), v_2 = (\theta = 0, \phi = 90), v_3 = (\theta = 90, \phi = 0)$, at least 71% of the object could be reconstructed for different noise values.

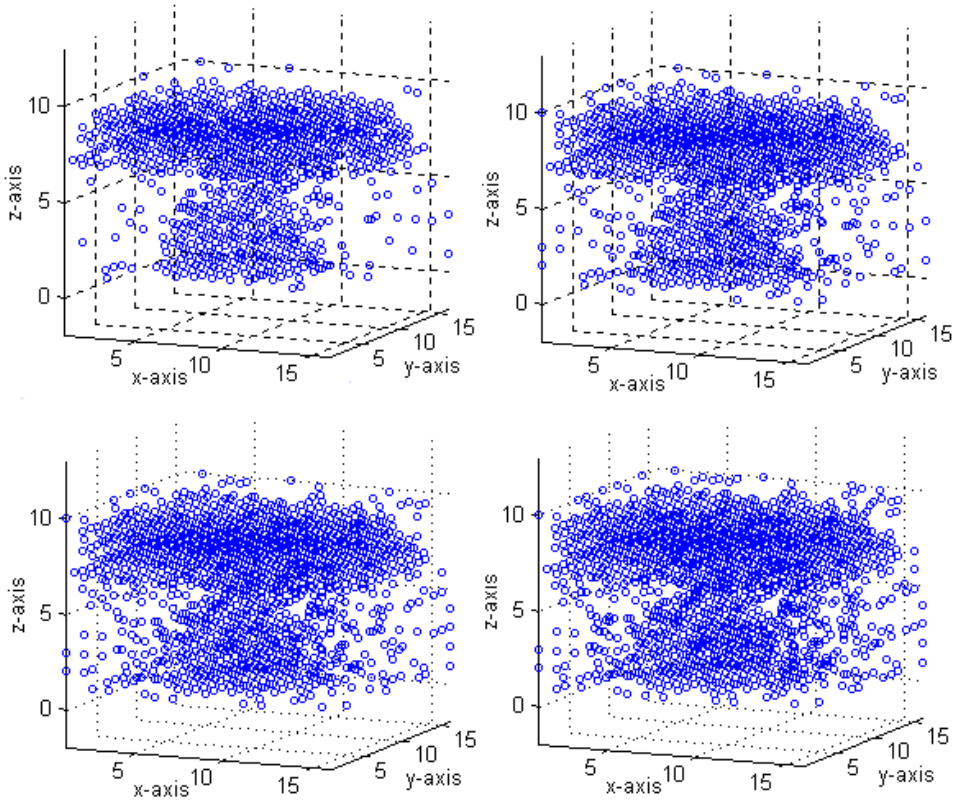

Figure 5.19: Reconstruction of object shown in Figure 5.17 from 3 orthogonal projections using my proposed solution. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Figure 5.19 shows the reconstruction results of *Sample Object 2* (shown in Figure 5.17) from 3 orthogonal projections for different noise values using my proposed solution.

Here, I will present the reconstruction of sample object shown in Figure 5.17 using 16 projections with lattice directions $D = (v_1, ...v_{16})$ where $v_k = \{0, 30, 60, 90\}^2$, $(k = 1, ..., 16)$.

The 16 projections are simulated and for each lattice direction, $(v_1, ..., v_{16})$, the line sums are calculated. In Figure 5.20, the projection images are shown for each of the lattice directions.

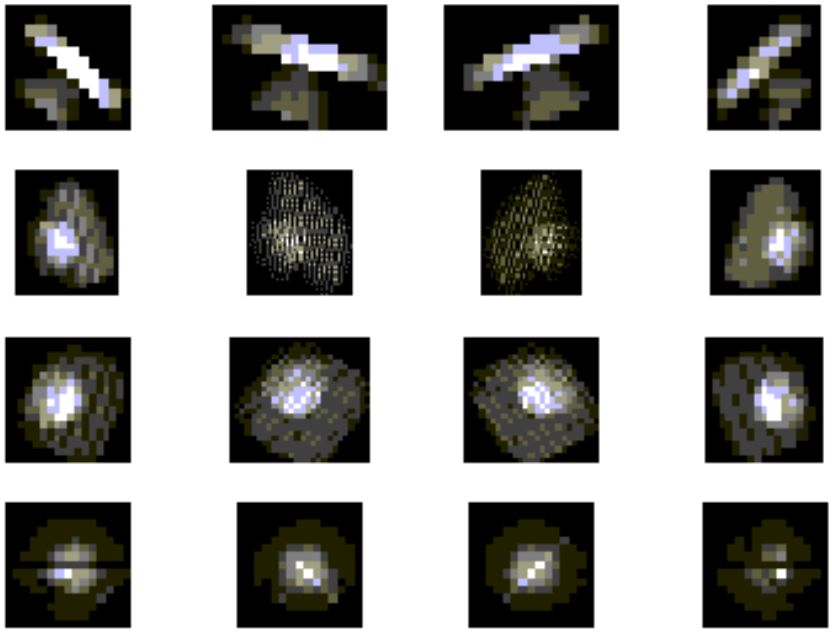

Figure 5.20: 16 projection images of the sample object shown in Figure 5.17.

Reconstruction of sample object shown in Figure 5.17 from 16 projections defined above is done using my framework (Framework 5.8) with $\mathcal{A}$ chosen as a genetic algorithm. The genetic algorithm $\mathcal{A}$ is run for 10 iterations.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	93	179	1015.89
20 dB	86	350	1017.65
15 dB	83	435	1012.37
10 dB	79	537	1016.15

Table 5.4: Reconstruction results of sample object shown in Figure 5.17 using my framework (Framework 5.8) from 16 projections.

For each iteration, some measures are taken. Table 5.4 lists these results.

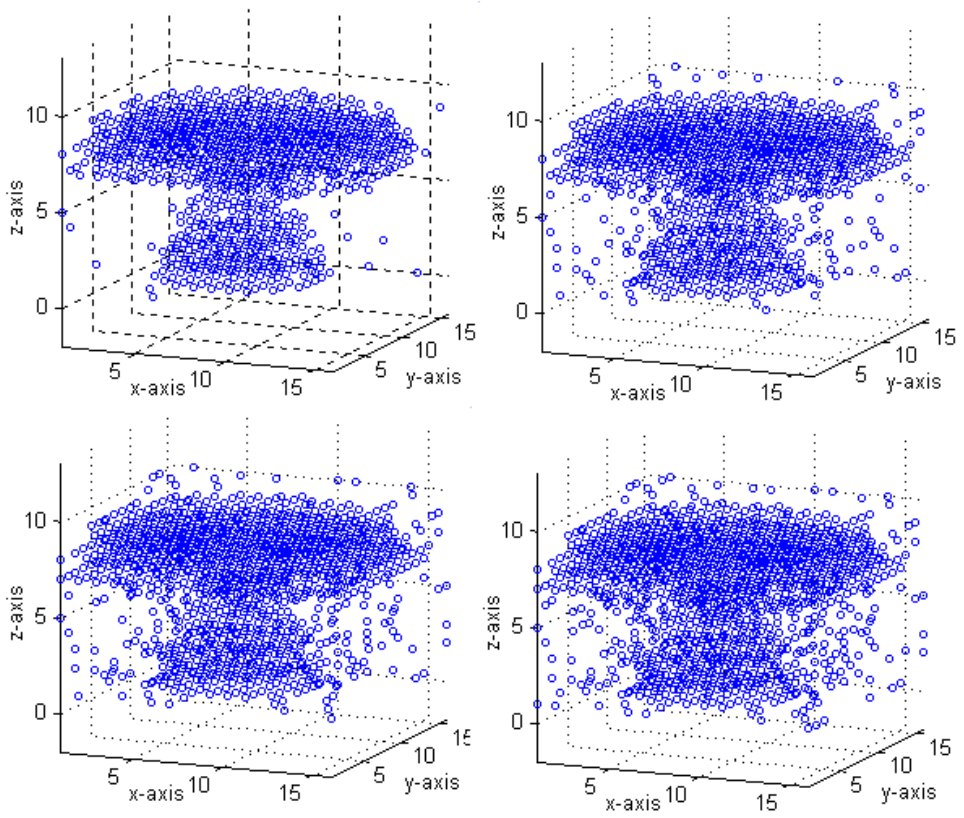

Figure 5.21: Reconstruction of object shown in Figure 5.17 from 16 projections using my proposed solution. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Figure 5.21 shows the reconstruction results of *Sample Object 2* (shown in Figure 5.17) from 16 projections for different noise values using my proposed solution.

5.3 Comparative Study: Proposed Solution vs Existing Works

In this section, I will present comparisons between my ideas and some of the existing works presented in Chapter 3, experimented over some set of sample data.

5.3.1 Polyhedral Shape Estimation

In this section, I will give my reconstruction result which has been applied to sample data shown in Figure 3.6 of Section 3.6. This sample object is in a $(15 \times 15 \times 10)$ grid and has been experimented by Djafari [12, 11, 49] with 9 projections. To simulate the measurement errors zero-mean Gaussian noise is added with SNR equal to 40 dB with the Equation (5.2.10).

The methods introduced by Djafari and Soussen's work, yield the reconstruction results shown in Figure 5.22. In that experiment, 9 projections have been used and a polyhedral shape has been reconstructed.

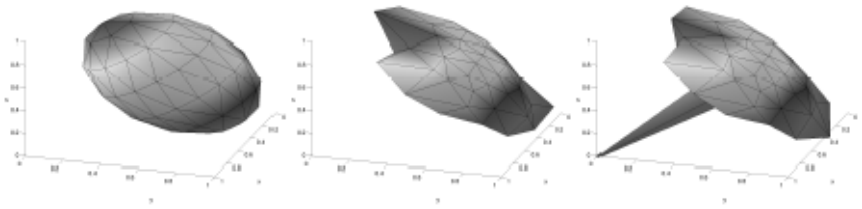

Figure 5.22: Reconstructions of the object from 9 projections after 0, 40 and 100 iterations with SNR 40 dB, using methods presented in [12, 11, 49].

The data shown in Figure 3.6 has been simulated in the geometric sense. I have used $(16 \times 16 \times 10)$ grid and 9 lattice directions $D = (v_1, ...v_9)$ where $v_k = \{0, 45, 90\}^2$, $(k = 1, ..., 9)$. To make an intuitive comparison I used my proposed framework with $\mathcal{A}$ being a genetic algorithms and with the initial reconstruction of MBO Algorithm (5.9).

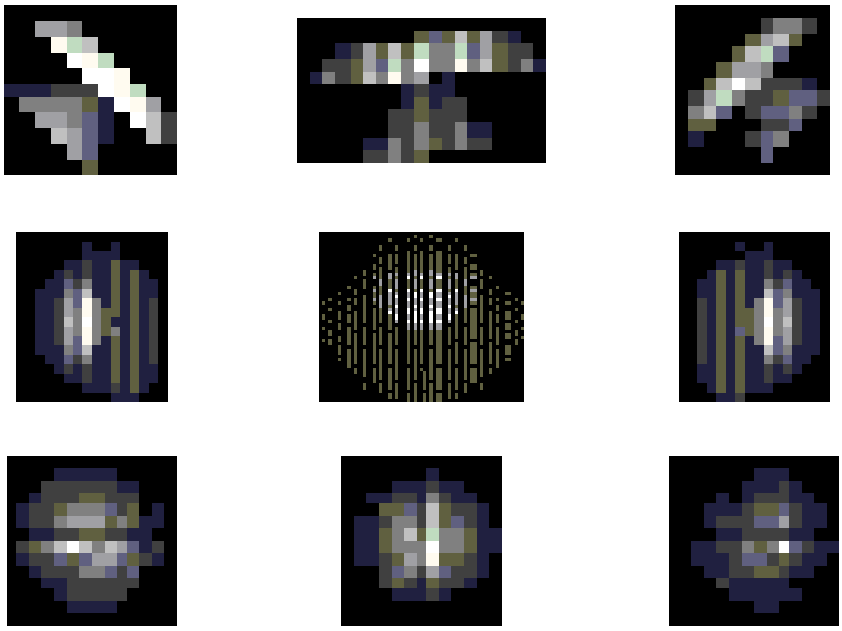

Figure 5.23: 9 projection images of the sample object shown in Figure 3.6.

The 9 projections are simulated and for each lattice direction, $(v_1, ..., v_9)$, the line sums are calculated. In Figure 5.23, the projection images are shown for each of the lattice directions.

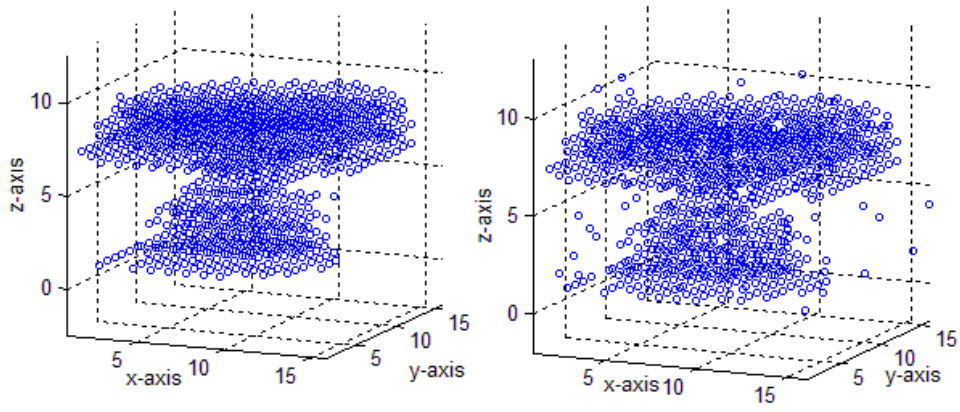

Figure 5.24: *Left*: MBO, *Right*: Reconstruction of the sample object shown in Figure 3.6 from 9 lattice directions using my proposed methods with 40 dB SNR.

Using my MBO construction algorithm (Algorithm 5.9) with 9 projections de-

fined by lattice directions $D = (v_1, ...v_9)$ where $v_k = \{0, 45, 90\}^2$, $(k = 1, ..., 9)$, the MBO is obtained as shown in left of Figure 5.24.

Reconstruction of sample object shown in Figure 3.6 from 9 projections defined above is done using my framework (Framework 5.8) with $\mathcal{A}$ chosen as a genetic algorithm. The genetic algorithm $\mathcal{A}$ is run for 10 iterations. Even with 10 iterations, the result seems very promising as shown in right of Figure 5.24.

5.3.2 Comparison with SIRT

Here, I will show the experiment results of SIRT (Algorithm 3.12) with the *Sample Object 1* and *Sample Object 2*.

The first experiments were on the sample object shown in Figure 5.12 in Section 5.2.5.1. It has been experimented with 3 orthogonal, and 16 projections. The 16 projections are chosen as $D = (v_1, ...v_{16})$ where $v_k = \{0, 30, 60, 90\}^2$, $(k = 1, ..., 16)$ and the projection images are shown in Figure 5.13 and in Figure 5.15.

Firstly, the reconstruction results of SIRT from 3 orthogonal projections will be given. The reconstructed objects for different noise values are shown in Figure 5.25. Table 5.5 shows the results of reconstruction percentage and running times of SIRT from 3 orthogonal projections for different noise values.

Secondly, the reconstruction results of SIRT from 16 projections in the directions of equal interval lattice directions will be given. The reconstructed objects for different noise values are shown in Figure 5.26. Table 5.6 shows the results of reconstruction percentage and running times of SIRT from 16 projections for different noise values.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	85	115	5.08
20 dB	81	145	5.11
15 dB	77	176	5.2
10 dB	74	199	5.16

Table 5.5: Reconstruction results of sample object shown in Figure 5.12 using SIRT from 3 orthogonal projections, for different noise values.

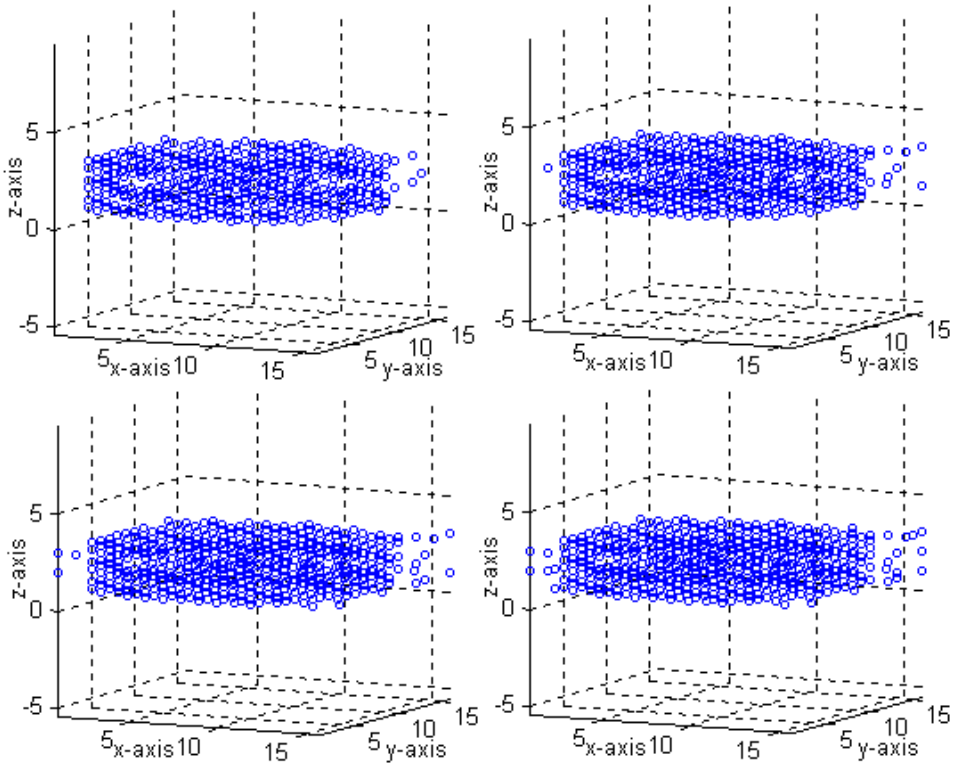

Figure 5.25: SIRT reconstruction of the sample object shown in Figure 5.12 from 3 orthogonal projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Noise	Correctly Rec. (%)	$\|Px-b\|_1$	Running Time (sec.)
No Noise	97	23	62.17
20 dB	91	69	61.56
15 dB	88	92	60.22
10 dB	85	115	61.36

Table 5.6: Reconstruction results of sample object shown in Figure 5.12 using SIRT from 16 projections, for different noise values.

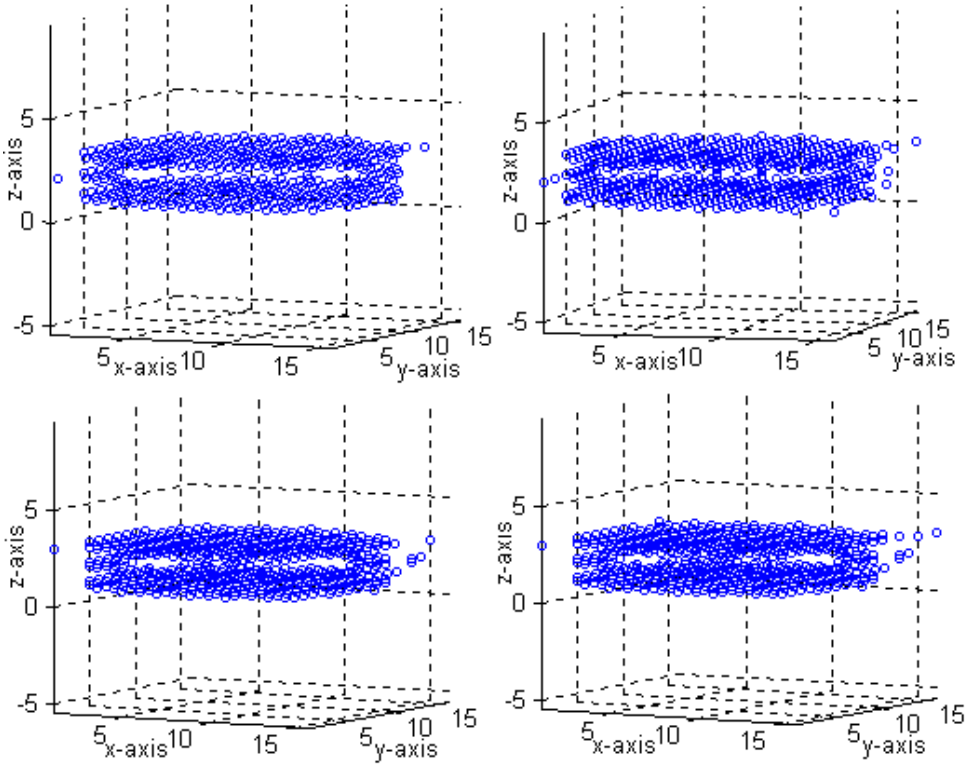

Figure 5.26: SIRT reconstruction of the sample object shown in Figure 5.12 from 16 projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

The next experiments were on the sample object shown in Figure 5.17 in Section 5.2.5.2. It has been experimented with 3 orthogonal, and 16 projections. The 16 projections are chosen as $D = (v_1, ...v_{16})$ where $v_k = \{0, 30, 60, 90\}^2$,

$(k = 1, ..., 16)$ and the projection images are shown in Figure 5.18 and in Figure 5.20.

Firstly, the reconstruction results of SIRT from 3 orthogonal projections will be given. The reconstructed objects for different noise values are shown in Figure 5.27. Table 5.7 shows the results of reconstruction percentage and running times of SIRT from 3 orthogonal projections for different noise values.

Secondly, the reconstruction results of SIRT from 16 projections in the directions of equal interval lattice directions will be given. The reconstructed objects for different noise values are shown in Figure 5.28. Table 5.8 shows the results of reconstruction percentage and running times of SIRT from 16 projections for different noise values.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	78	563	31.95
20 dB	74	665	30.59
15 dB	73	691	30.78
10 dB	70	768	31.46

Table 5.7: Reconstruction results of sample object shown in Figure 5.17 using SIRT from 3 orthogonal projections, for different noise values.

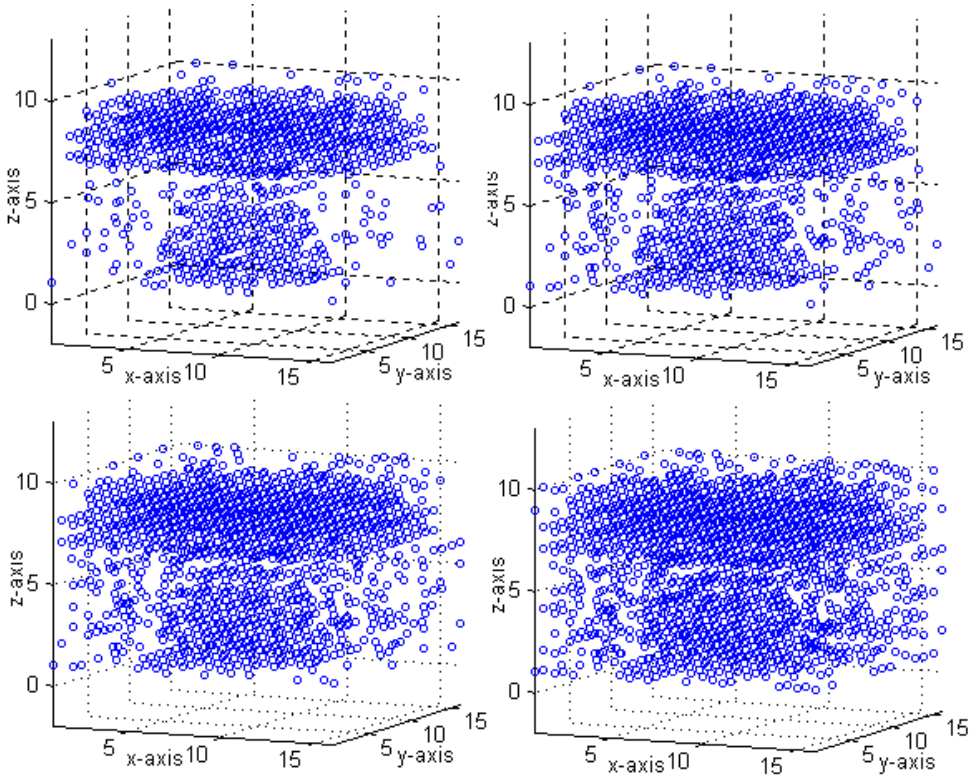

Figure 5.27: SIRT reconstruction of the sample object shown in Figure 5.17 from 3 orthogonal projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	92	204	827.61
20 dB	87	332	830.13
15 dB	82	460	829.01
10 dB	78	563	828.43

Table 5.8: Reconstruction results of sample object shown in Figure 5.17 using SIRT from 16 projections, for different noise values.

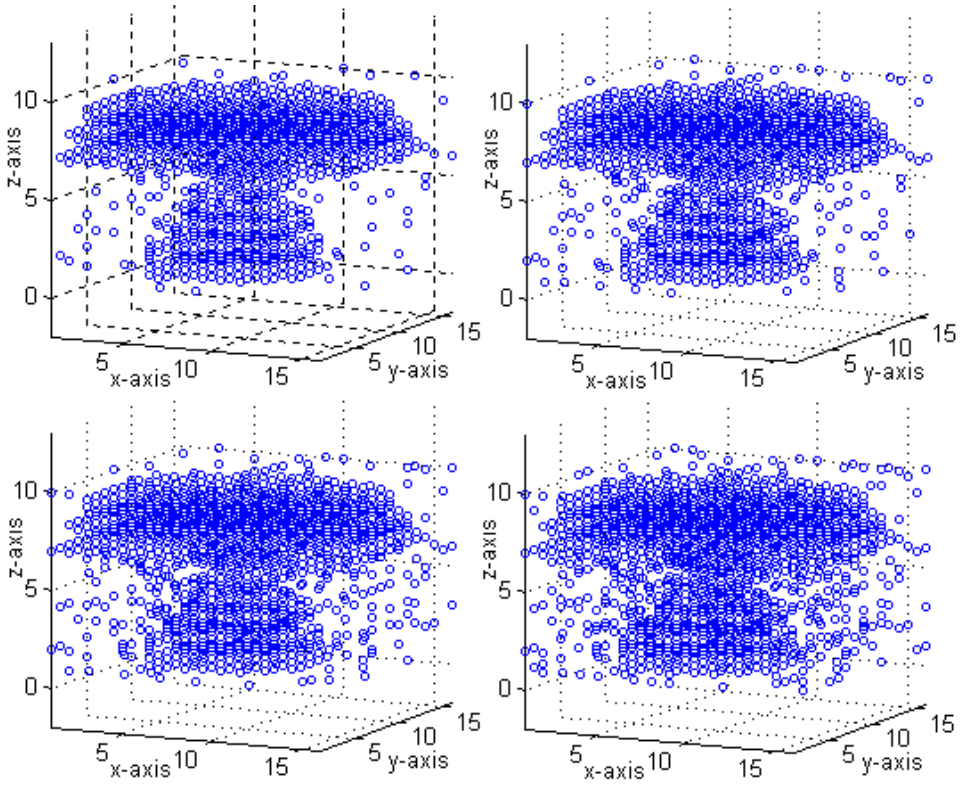

Figure 5.28: SIRT reconstruction of the sample object shown in Figure 5.17 from 16 projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Since SIRT is independent of the initial solution [1], no MBO is supplied as an initial solution.

5.3.3 Comparison with Kaczmarz's Algorithm

Here, I will show the experiment results of Kaczmarz's Algorithm (Algorithm 3.10) with the *Sample Object 1* and *Sample Object 2*.

The first experiments were on the sample object shown in Figure 5.12 in Section 5.2.5.1. It has been experimented with 3 orthogonal, and 16 projections. The 16 projections are chosen as $D = (v_1, ...v_{16})$ where $v_k = \{0, 30, 60, 90\}^2$, $(k = 1, ..., 16)$ and the projection images are shown in Figure 5.13 and in Figure

5.15.

Firstly, the reconstruction results of Kaczmarz's Algorithm from 3 orthogonal projections will be given. The reconstructed objects for different noise values are shown in Figure 5.29. Table 5.9 shows the results of reconstruction percentage and running times of Kaczmarz's Algorithm from 3 orthogonal projections for different noise values.

Secondly, the reconstruction results of Kaczmarz's Algorithm from 16 projections in the directions of equal interval lattice directions will be given. The reconstructed objects for different noise values are shown in Figure 5.30. Table 5.10 shows the results of reconstruction percentage and running times of Kaczmarz's Algorithm from 16 projections for different noise values.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	86	107	4.89
20 dB	73	207	4.82
15 dB	68	245	4.85
10 dB	62	291	4.79

Table 5.9: Reconstruction results of sample object shown in Figure 5.12 using Kaczmarz's Algorithm from 3 orthogonal projections, for different noise values.

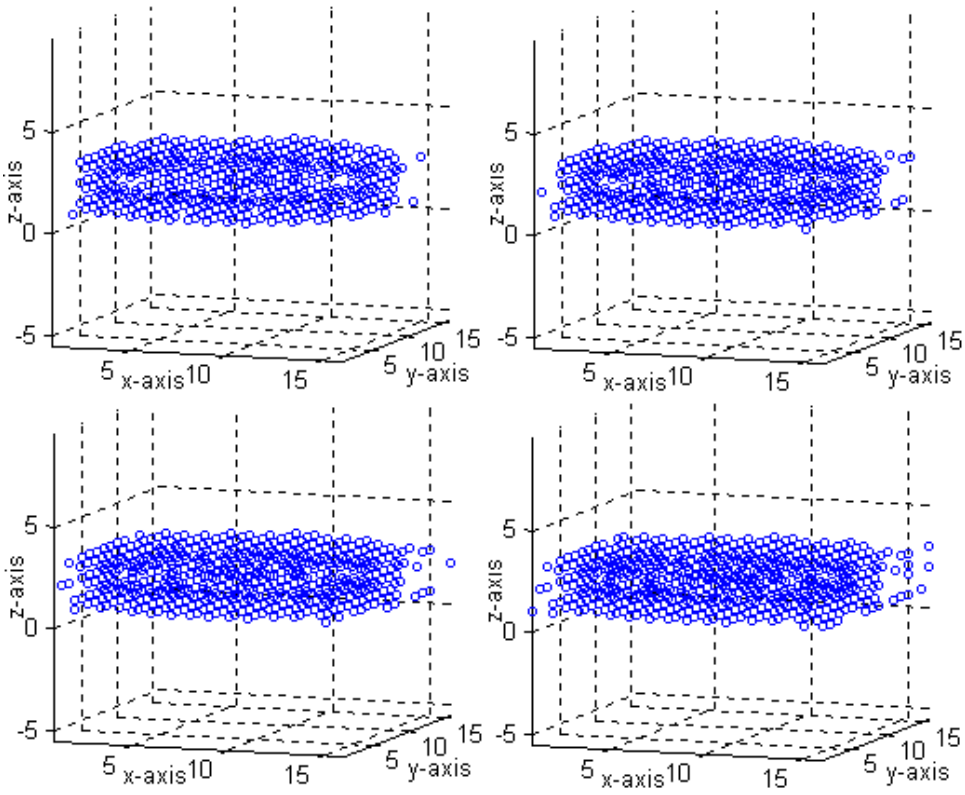

Figure 5.29: Kaczmarz's Algorithm reconstruction of the sample object shown in Figure 5.12 from 3 orthogonal projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	97	23	59.43
20 dB	84	122	58.62
15 dB	80	153	59.33
10 dB	75	192	60.04

Table 5.10: Reconstruction results of sample object shown in Figure 5.12 using Kaczmarz's Algorithm from 16 projections, for different noise values.

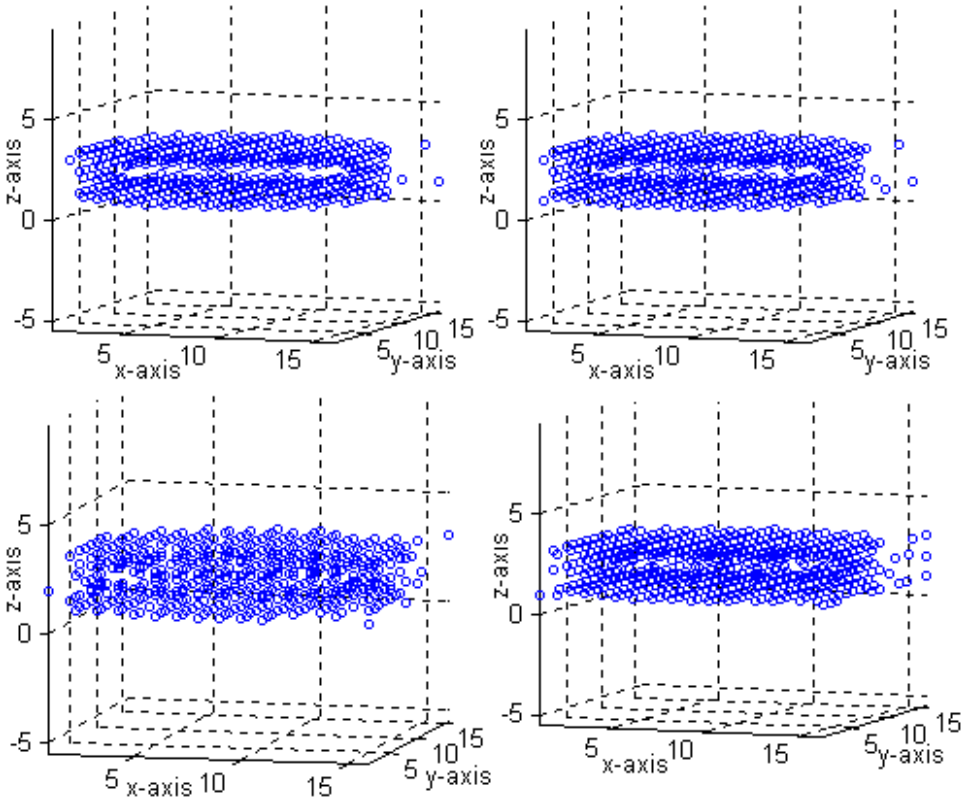

Figure 5.30: Kaczmarz's Algorithm reconstruction of the sample object shown in Figure 5.12 from 16 projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

The next experiments were on the sample object shown in Figure 5.17 in Section 5.2.5.2. It has been experimented with 3 orthogonal, and 16 projections. The 16 projections are chosen as $D = (v_1, ...v_{16})$ where $v_k = \{0, 30, 60, 90\}^2$, $(k = 1, ..., 16)$ and the projection images are shown in Figure 5.18 and in Figure 5.20.

Firstly, the reconstruction results of Kaczmarz's Algorithm from 3 orthogonal projections will be given. The reconstructed objects for different noise values are shown in Figure 5.31. Table 5.11 shows the results of reconstruction percentage and running times of Kaczmarz's Algorithm from 3 orthogonal projections for different noise values.

Secondly, the reconstruction results of Kaczmarz's Algorithm from 16 projections in the directions of equal interval lattice directions will be given. The reconstructed objects for different noise values are shown in Figure 5.32. Table 5.12 shows the results of reconstruction percentage and running times of Kaczmarz's Algorithm from 16 projections for different noise values.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	76	614	29.3
20 dB	67	844	29.17
15 dB	62	972	28.96
10 dB	58	1075	28.82

Table 5.11: Reconstruction results of sample object shown in Figure 5.17 using Kaczmarz's Algorithm from 3 orthogonal projections, for different noise values.

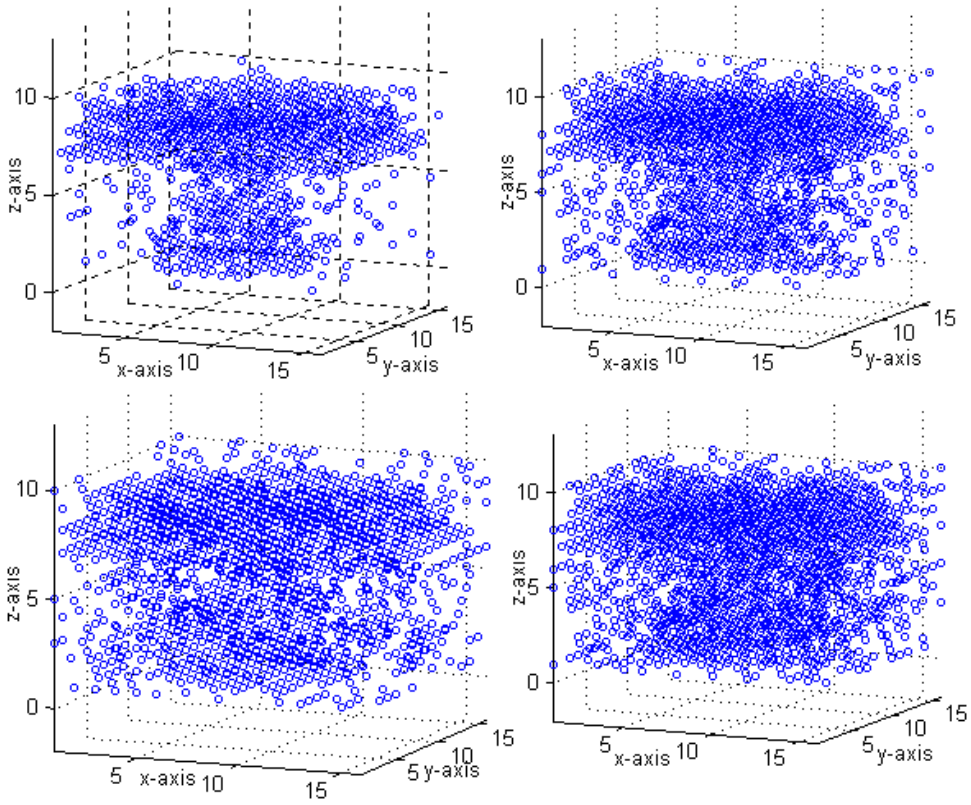

Figure 5.31: Kaczmarz's Algorithm reconstruction of the sample object shown in Figure 5.17 from 3 orthogonal projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	90	256	798.54
20 dB	74	665	790.13
15 dB	70	768	795.42
10 dB	65	896	796.68

Table 5.12: Reconstruction results of sample object shown in Figure 5.17 using Kaczmarz's Algorithm from 16 projections, for different noise values.

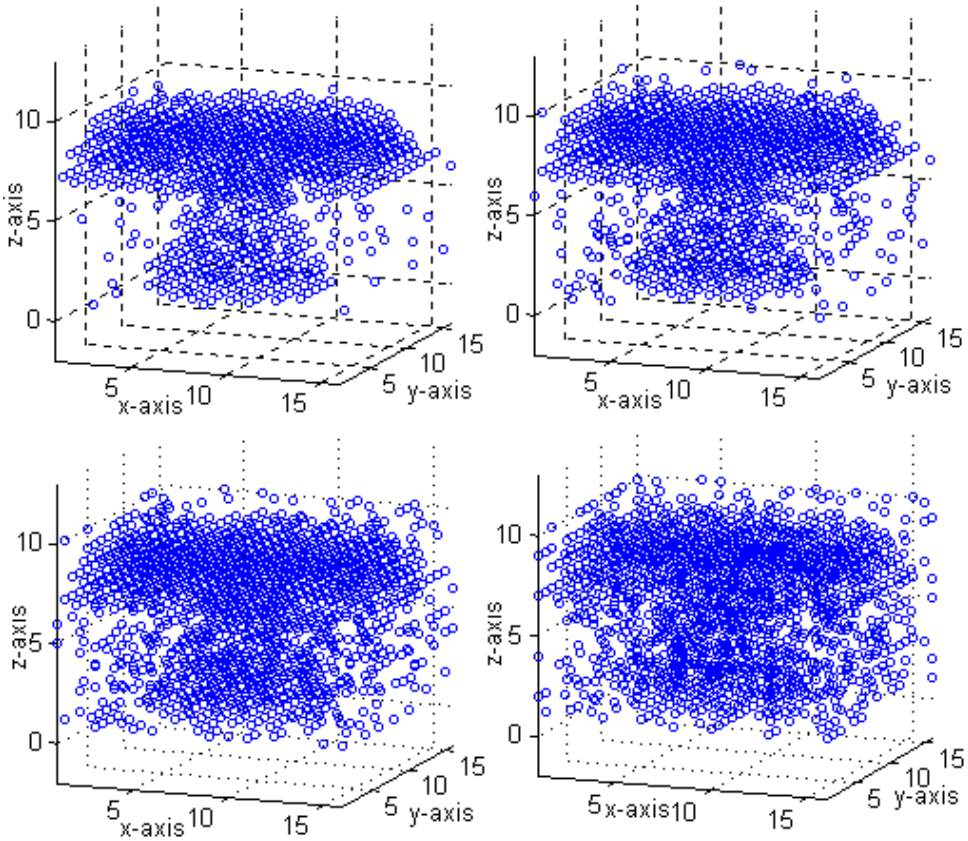

Figure 5.32: Kaczmarz's Algorithm reconstruction of the sample object shown in Figure 5.17 from 16 projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

5.3.4 Comparison with CGLS

Here, I will show the experiment results of CGLS method (Algorithm 3.8.4) with the *Sample Object 1* and *Sample Object 2*.

The first experiments were on the sample object shown in Figure 5.12 in Section 5.2.5.1. It has been experimented with 3 orthogonal, and 16 projections. The 16 projections are chosen as $D = (v_1, ...v_{16})$ where $v_k = \{0, 30, 60, 90\}^2$, $(k = 1, ..., 16)$ and the projection images are shown in Figure 5.13 and in Figure 5.15.

Firstly, the reconstruction results of CGLS method from 3 orthogonal projections will be given. The reconstructed objects for different noise values are shown in Figure 5.33. Table 5.13 shows the results of reconstruction percentage and running times of CGLS method from 3 orthogonal projections for different noise values.

Secondly, the reconstruction results of CGLS method from 16 projections in the directions of equal interval lattice directions will be given. The reconstructed objects for different noise values are shown in Figure 5.34. Table 5.14 shows the results of reconstruction percentage and running times of CGLS method from 16 projections for different noise values.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	84	122	4.76
20 dB	72	215	4.58
15 dB	67	253	4.75
10 dB	63	284	4.49

Table 5.13: Reconstruction results of sample object shown in Figure 5.12 using CGLS method from 3 orthogonal projections, for different noise values.

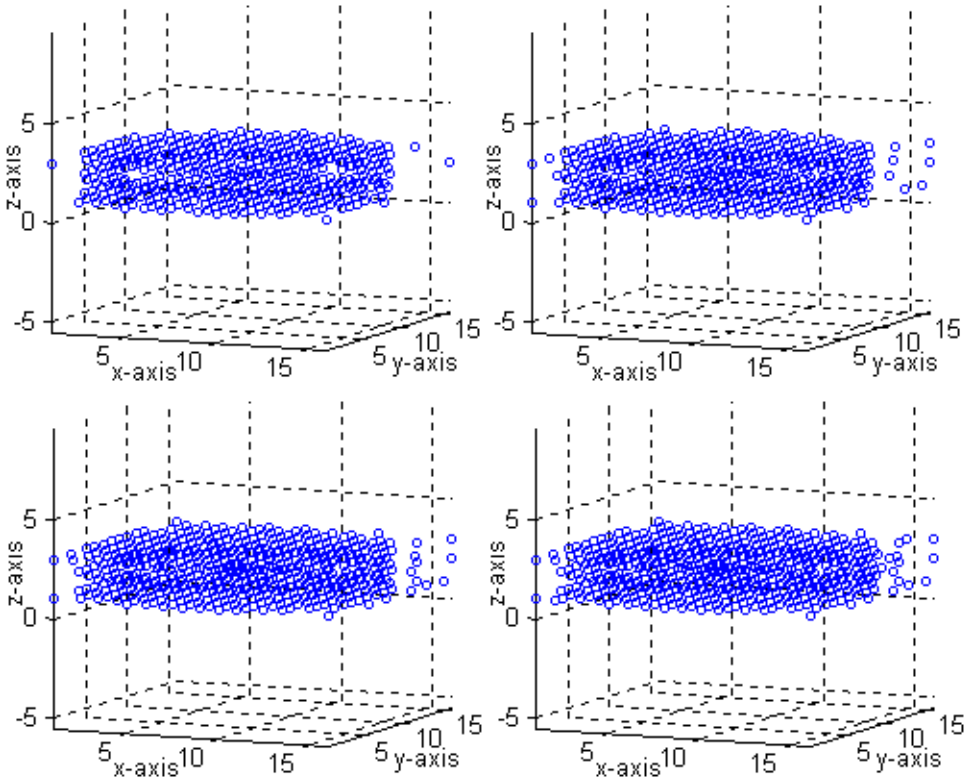

Figure 5.33: CGLS method reconstruction of the sample object shown in Figure 5.12 from 3 orthogonal projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	96	30	56.32
20 dB	85	115	57.52
15 dB	79	161	56.46
10 dB	73	207	56.87

Table 5.14: Reconstruction results of sample object shown in Figure 5.12 using CGLS method from 16 projections, for different noise values.

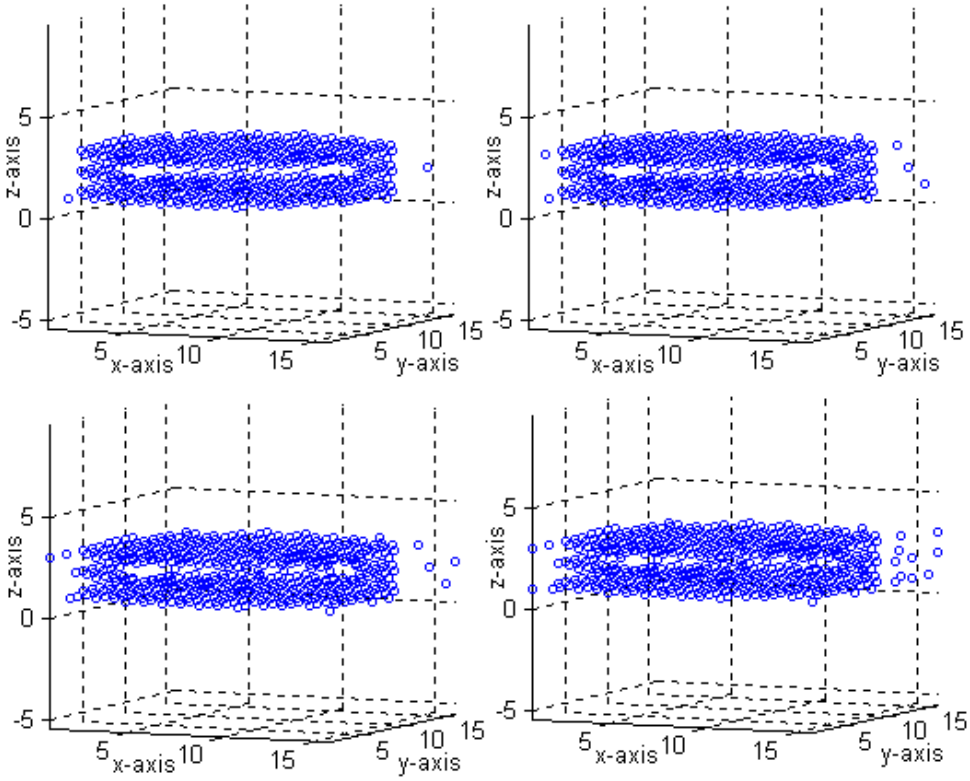

Figure 5.34: CGLS method reconstruction of the sample object shown in Figure 5.12 from 16 projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

The next experiments were on the sample object shown in Figure 5.17 in Section 5.2.5.2. It has been experimented with 3 orthogonal, and 16 projections. The 16 projections are chosen as $D = (v_1, ...v_{16})$ where $v_k = \{0, 30, 60, 90\}^2$, $(k = 1, ..., 16)$ and the projection images are shown in Figure 5.18 and in Figure 5.20.

Firstly, the reconstruction results of CGLS method from 3 orthogonal projections will be given. The reconstructed objects for different noise values are shown in Figure 5.35. Table 5.15 shows the results of reconstruction percentage and running times of CGLS method from 3 orthogonal projections for different noise values.

Secondly, the reconstruction results of CGLS method from 16 projections in the

directions of equal interval lattice directions will be given. The reconstructed objects for different noise values are shown in Figure 5.36. Table 5.16 shows the results of reconstruction percentage and running times of CGLS method from 16 projections for different noise values.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	77	588	27.53
20 dB	65	896	26.79
15 dB	60	1024	27.93
10 dB	56	1126	27.47

Table 5.15: Reconstruction results of sample object shown in Figure 5.17 using CGLS method from 3 orthogonal projections, for different noise values.

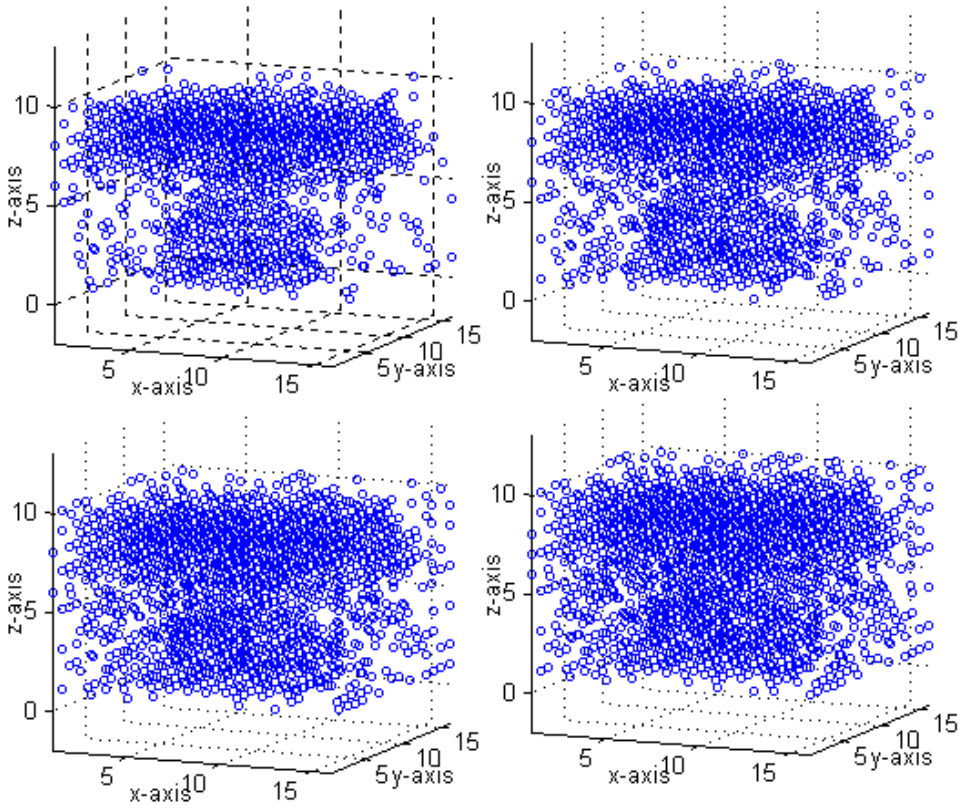

Figure 5.35: CGLS method reconstruction of the sample object shown in Figure 5.17 from 3 orthogonal projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

Noise	Correctly Rec. (%)	$\|Px - b\|_1$	Running Time (sec.)
No Noise	91	230	720.64
20 dB	75	640	726.98
15 dB	68	819	723.84
10 dB	63	947	722.73

Table 5.16: Reconstruction results of sample object shown in Figure 5.17 using CGLS method from 16 projections, for different noise values.

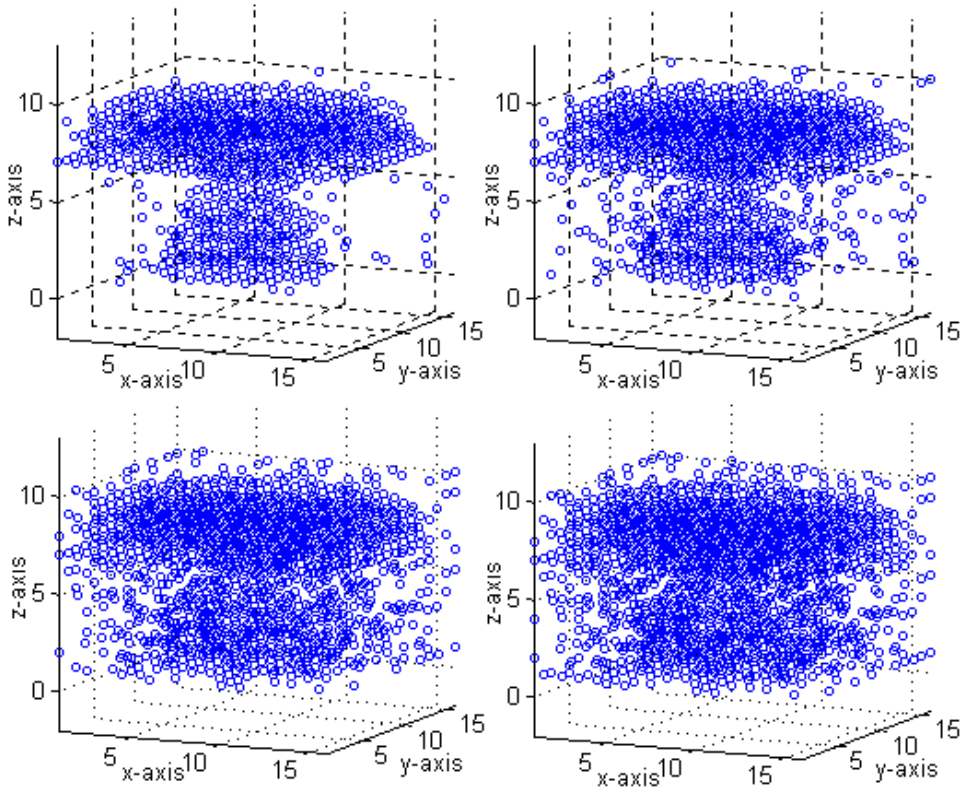

Figure 5.36: CGLS method reconstruction of the sample object shown in Figure 5.17 from 16 projections. *Top Left*: No Noise, *Top Right*: 20 dB SNR, *Bottom Left*: 15 dB SNR, *Bottom Right*: 10 dB SNR.

5.3.5 Comparison: All Methods

In this section the methods are compared according to their reconstruction percentages for different noise values. Here, the results of the methods for *Sample Object 1* and *Sample Object 2* are tabulated and the graphs of these results are presented. The figures 5.37, 5.38, 5.39 and 5.40 shows us that Kaczmarz's Algorithm and CGLS method are affected more than the proposed solution and SIRT when some measurement errors are introduced.

Noise	Proposed Solution	SIRT	Kaczmarz	CGLS
No Noise	85	85	86	84
20 dB	80	81	73	72
15 dB	77	77	68	67
10 dB	75	74	62	63

Table 5.17: Comparison of methods in terms of reconstruction percentages of the sample object shown in Figure 5.12 from 3 orthogonal projections, for different noise values.

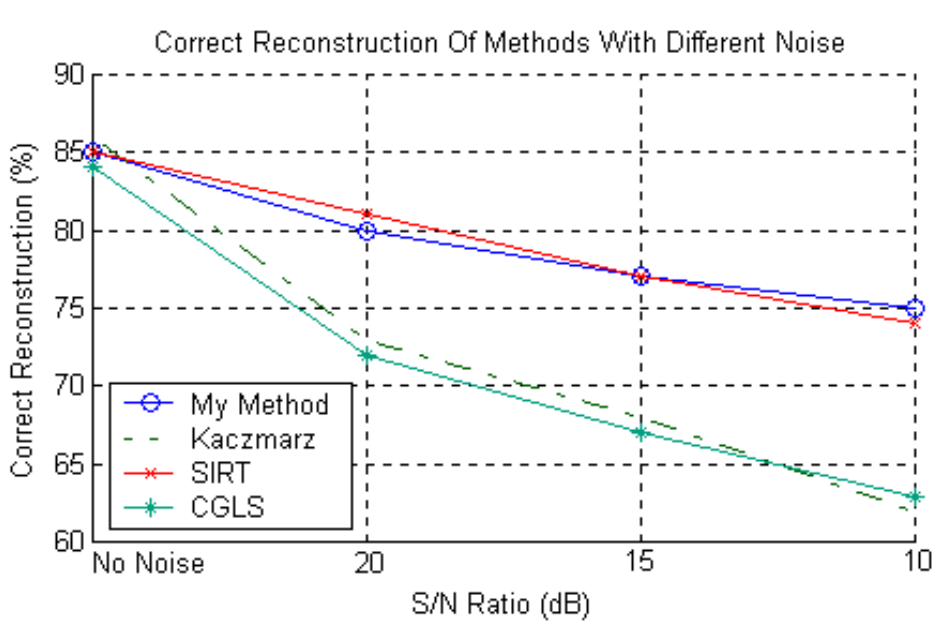

Figure 5.37: Reconstruction of the object shown in Figure 5.12 from 3 orthogonal projections with compared methods in different noise values.

Noise	Proposed Solution	SIRT	Kaczmarz	CGLS
No Noise	98	97	97	96
20 dB	93	91	84	85
15 dB	89	88	80	79
10 dB	83	85	75	73

Table 5.18: Comparison of methods in terms of reconstruction percentages of the sample object shown in Figure 5.12 from 16 projections, for different noise values.

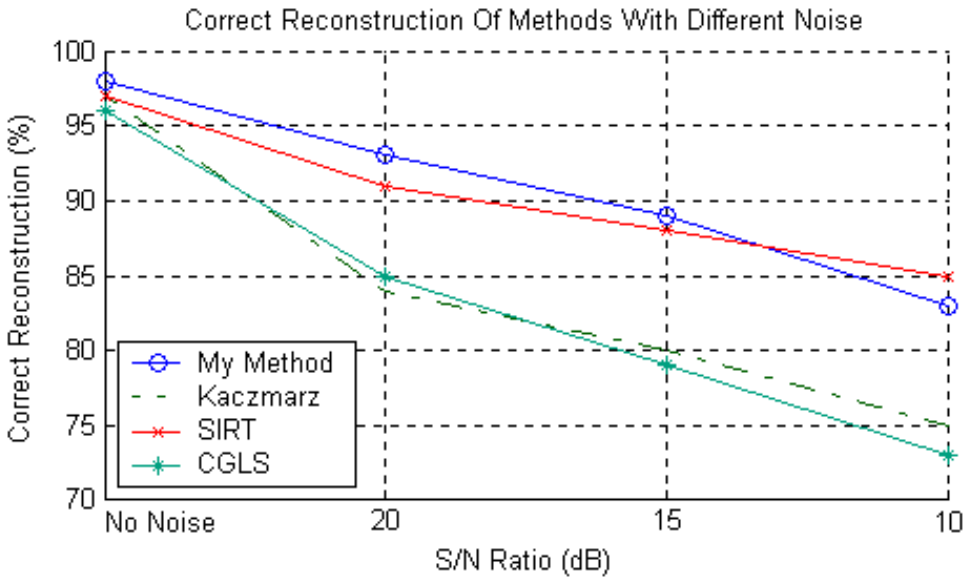

Figure 5.38: Reconstruction of the object shown in Figure 5.12 from 16 projections with compared methods in different noise values.

Noise	Proposed Solution	SIRT	Kaczmarz	CGLS
No Noise	79	78	76	77
20 dB	75	74	67	65
15 dB	73	73	62	60
10 dB	71	70	58	56

Table 5.19: Comparison of methods in terms of reconstruction percentages of the sample object shown in Figure 5.17 from 3 orthogonal projections, for different noise values.

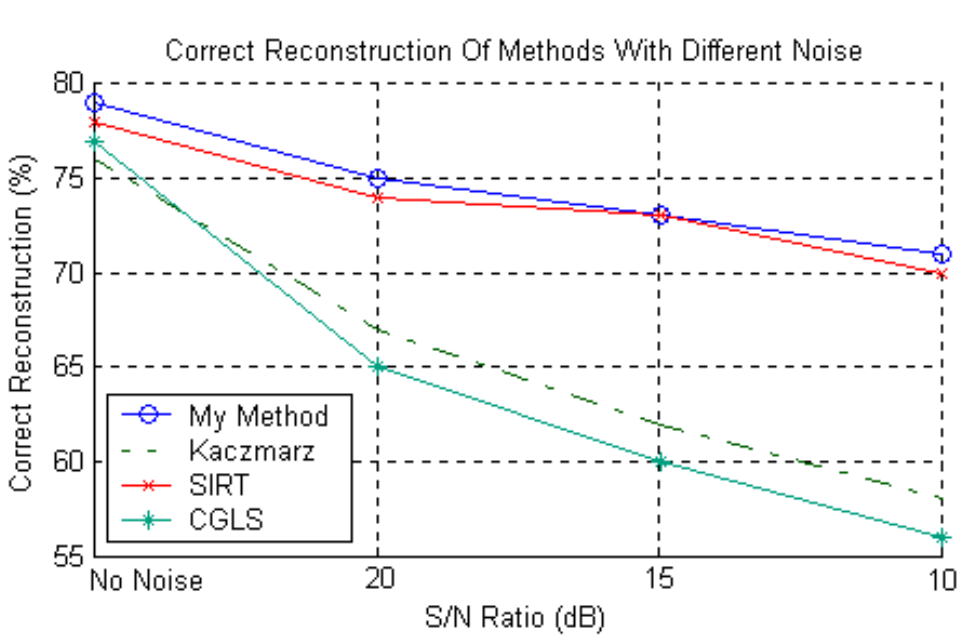

Figure 5.39: Reconstruction of the object shown in Figure 5.17 from 3 orthogonal projections with compared methods in different noise values.

Noise	Proposed Solution	SIRT	Kaczmarz	CGLS
No Noise	93	92	90	91
20 dB	86	87	74	75
15 dB	83	82	70	68
10 dB	79	78	65	63

Table 5.20: Comparison of methods in terms of reconstruction percentages of the sample object shown in Figure 5.17 from 16 projections, for different noise values.

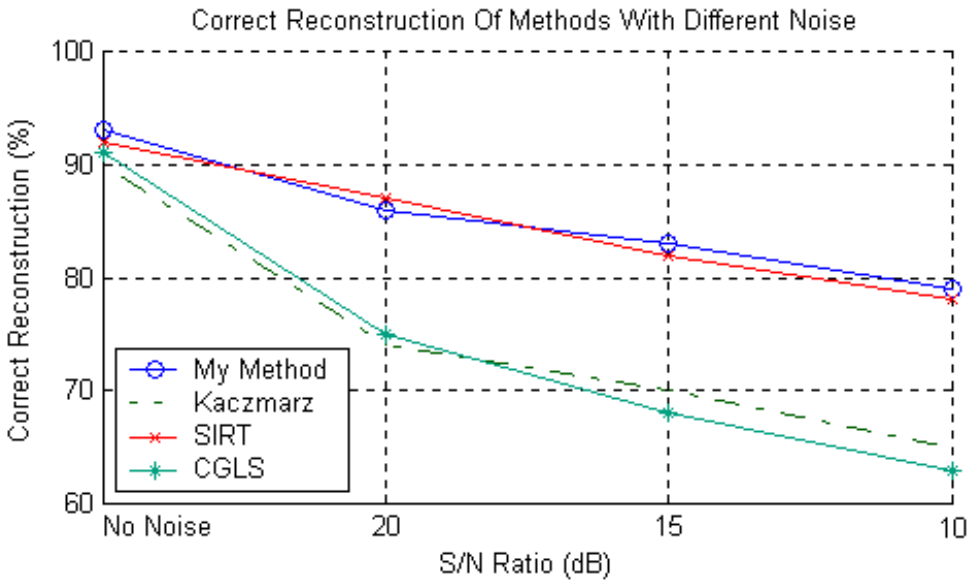

Figure 5.40: Reconstruction of the object shown in Figure 5.17 from 16 projections with compared methods in different noise values.

Chapter 6

Conclusion

In this thesis, Discrete Tomography (DT) has been introduced and one of the main problems of Discrete Tomography, namely *reconstruction* has been pointed out. As an application area of DT, an industrial production area, *VLSI production* has been mentioned. The difficulties of reconstruction problem have been considered and with these motivation, a new approach has been tried to be invented. To reach the main aim of the thesis, some existing methods have been examined.

A new formulation (Section 5.2.1) for an approximate solution of the reconstruction problem based on constrained and unconstrained optimization has been introduced. This reformulation is a formulation of an inverse problem in terms of constrained optimization problem. Moreover, it has been illustrated that how a reconstruction problem can be converted to this type formulation.

Using the methods from optimization theory and inverse problems theory, *a new framework* has been proposed to solve the new reformulation. *A new algorithm* (Algorithm 5.9) is introduced to be used as an initial solution and as a geometric boundary for the formulation.

The experiments show us that the proposed reformulation with the framework can give good approximate solutions even with *less number of projections* and *less iterations*. The comparison results verify this in an experimental way, such that in some cases, the proposed solution yields better results than the existing methods compared in this thesis.

The proposed solution is affected less than Kaczmarz's Algorithm and CGLS method when some measurement errors are introduced. SIRT method per-

formed better in most of the experiments in terms of reconstruction percentage and durability to noise. Although the proposed solution is slower than other methods compared, the reconstruction results are geometrically better than other methods. This might be the effect of geometric constraint imposed by the MBO in the reformulation. Therefore, proposed reformulation solved with proposed framework, can be used as an alternative method instead of existing methods.

However, the implementation should be improved further to use less memory, and with an improved iteration time. Moreover, instead of a genetic algorithm, a faster derivative free optimization method is also worth to investigate.

References

[1] Aster, R.C., Borchers, B. and Thurber, C. , *Parameter Estimation and Inverse Problems*, Academic Press, 2004.

[2] Barcucci, E., Del Lungo, A., Nivat, M. and Pinzani, R., *Medians of polyominoes: A property for the reconstruction*, Int. J. Imaging Systems and Techn. 9, 69-77, 1998.

[3] Barcucci, E., Del Lungo, A., Nivat, M. and Pinzani, R., *Reconstructing convex polyominoes from horizontal and vertical projections*, Theor. Comp. Sci. 155, 321-347, 1996.

[4] Bianchi, G. and Longinetti, M., *Reconstructing plane sets from projections*, Discrete Comp. Geom. 5, 223-242, 1990.

[5] Censor, Y. and Matej, S., *Binary steering of nonbinary iterative algorithms*, in: G.T. Herman, A. Kuba (Eds.), Discrete Tomography: Foundations, Algorithms, and Applications, Birkhuser, Boston, MA, pp. 285296, 1999 .

[6] Chang, S.K. and Chow, C.K., *The reconstruction of three-dimensional objects from two orthogonal projections and its applications to cardiac cineangiography*, IEEE Trnas. Comp. C-22, 18-28, 1973.

[7] Chang, S.K. and Wang, Y.R., *Three-dimensional reconstruction from orthogonal projections*, Pattern Recognition 7, 167-176, 1975.

[8] Chrobak, M. and Dürr, C., *Reconstructing hv-convex polyominoes from orthogonal projections*, Preprint, Dept. of Comp. Sci., Uni. of California, Riverside, CA , 1998.

[9] Conn, A.R., Scheinberg, K. and Toint, P.L., *A derivative free optimization method via support vector machines*, Technical Report TR98/11, Department of Mathematics, The University of Namur, Namur, Belgium, 1998.

[10] Conn, A.R., Gould, N.I.M. and Toint, P.L., *Trust-Region Methods*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2000.

[11] Djafari, M.A., *Image Reconstruction of a Compact Object from a few number projections*, in IASTED, Int. Conf. on Signal and Image Processing(SIP'96), Orlando-Florida-USA, pp.325-329, 1996.

[12] Djafari, M.A. and Soussen, C., *Compact Object Reconstruction*, Kluwer Academic Publications, pp.121-136, 1996.

[13] Gardner, R.J. and Gritzmann, P., *Discrete Tomography: Determination of finite sets by X-rays*, Trans. Amer. Math. Soc. 349, 2271-2295, 1997.

[14] Gardner, R.J., Gritzmann, P. and Prandenberg, D., *On the computational complexity of recontructing lattice sets by X-rays*, Discrete Math. 202, 45-71, 1999.

[15] Garey, M.R., and Johnson, D.S., *Computers and Intractability : A Guide to the Theory of NP-Completeness*, W. H. Freeman and Co. New York, NY, USA, 1990.

[16] Geman, S. and McClure, D., *Statistical methods for tomographic image reconstruction*, In Proceedings of the 46^{th} Session of the ISI, Bulletin of the ISI, 52, 5-21, 1987.

[17] Gordon, R., Bender, R. and Herman, G.T., *Algebraic reconstruction techinques(ART) for three-dimensional electron microscopy and X-ray tomography*, J. Theor. Biol. 42, 1-32, 1970.

[18] Green, P., *Bayesian recsonstructions from emmision tomography data using a modified EM algorithm*, IEEE Trans. on Medical Imaging 9, 84-93, 1990.

[19] Groetsch, C.W., *Inverse Problems in the Mathematical Sciences*, Vieweg, Braunschweig, 1993.

[20] Gwennap, L., *Estimating IC Manufacturing Costs* , Microprocessor Report, p. 15, August 2, 1993.

[21] Heath, M., *Scientific Computing: An Introductory Survey*, McGraw-Hill, Boston, 2002.

[22] Herman, G.T., *Image Reconstruction from Projections, the Fundamentals of Computerized Tomography*, Academic Press, New York, 1980.

[23] Herman, G.T., guest ed., *Special issue on Computerized Tomography*, Proc. IEEE , 71, No. 3, 291-435, 1983.

[24] Herman, G.T., Chan, M., Censor, Y., Levitan, E. and Lewitt, R.M., *Maximum a posteriori image resonstruction from preojections (In Image Models)*, Springer-Verlag, New-York, 53-89, 1984.

[25] Herman, G.T., and Kuba, A., *Discrete Tomography - Foundations, Algorithms and Applications*, Birkhäuser, 1999.

[26] Herman, G.T., Louis, A.K. and Natterer, F., *Mathematical Methods in Tomography* (Proceedings Oberwolfach 1990): Springer, Berlin, 1991.

[27] Huang, L., *The reconstruction of uniquely determined plane sets from two projections in discrete case*, Preprint Series UTMS 95-29, University of Tokyo, 1997.

[28] Kak, A.C. and Slaney, M., *Principals of Computerized Tomographic Imaging*, New York, IEEE Press, 1988.

[29] Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P., *Optimization by simulated annealing*, Science 220, 1983.

[30] Kisielowski, C., Schwander, P., Baumann, F.H., Seibt, M., Kim, Y. and Ourmazd, A., *An approach to quantitative high-resolution transmission electron microscopy of crystalline materials*, Ultramicroscopy 58, pp.131-155, 1995.

[31] Kuba, A., *The reconstruction of two-directionally connected binary patterns from their two orthogonal projections*, Comp. Vision, Graphics, Image Proc. 27, 249-265, 1984.

[32] Kuba, A., *Reconstruction of unique binary matrices with prescribed elements*, Acta. Cybern. 12, 57-70, 1995.

[33] Kuba, A., Yaşar, Ö., Weber, G.W., Özgür, O. and Dağlı, E., *Discrete Tomography: New Approaches and Results by Optimization Theory and Statistical Learning*, ISPRS 2005 SPATIAL DATA MINING WORKSHOP METU, Ankara, Turkey, November, 24-25, 2005

[34] Lorentz, G.G., *A problem of plane measure*, Amer. J. Math., 71, 417-426, 1949.

[35] Mlynek, D. and Leblebici, Y., *Design of VLSI Systems*, lecture notes, EPFL-DE Publication, 1997.

[36] Nash, S.C. and Sofer, A., *Linear and Nonlinear Programming*, McGraw-Hill, 1996.

[37] Nilagupta, P., *Digital Design Automation*, lecture notes, Department of Computer Engineering Kasetsart University, Taiwan, Fall 2003.

[38] Nocedal, J. and Wright, S.J., *Numerical Optimization*, Springer Series in Operations Research, 1999.

[39] Paulo, M., *Introduction to VLSI ASIC Design and Technology*, lecture notes, CERN, Switzerland, Fall 2001.

[40] Patterson, D. and Hennessey, J., *Computer Organization and Design*, Morgan Kafumann, 1996.

[41] Perry, G., *The Fundamentals of Digital Semiconductor Testing*, Soft Test Inc., AK Sharma, 1996.

[42] Prause, G.P.M. and Onnasch, D.G.W., *Binary reconstruction of the heart chambers from biplane angiographic image sequences*, IEEE Trans. Medical Imaging MI-15, 532-546, 1996.

[43] Prince J. and Wilsky, A., *Convex set reconstruction using prior shape information*, CVGIP, vol. 53, no. 5, pp. 413-427, 1991.

[44] Prince J. and Wilsky, A., *Reconstruction convex sets from support line measurements*, IEEE-PAMI, vol. 12, no. 3, pp. 377-389, 1990.

[45] Roerdink, J.B.T.M., *Computerized tomography and its applications: A guided tour.* Nieuw Archief voor Wiskunde 10, 3, 277-308, 1992.

[46] Ryser, H.J., *Combinatorial properties of matrices of zeros and ones*, Canad. J. Math. 9, 371-377, 1957.

[47] Ryser, H.J., *Traces of matrices of zeros and ones*, Canad. J. Math. 12, 463-476, 1960.

[48] Schwander, P., Kisielowski, C., Seibt, M., Baumann, F.H., Kim, Y. and Ourmazd, A., *Mapping projected potential, interfacial roughness, and composition in general crystalline solids by quantitative transmission electron microscopy*, Pyhs. Rev. Let. 71m pp. 4150-4153, 1993.

[49] Soussen C. and Djafari, M.A., *Multiresolution approach to the estimation of the shape of a 3D compact object from its radiographic data*, in Proc. SPIE, Mathematical Modeling, Bayesian Estimation, and Inverse Problems, *F. Prteux, A. Mohammad-Djafari, and E. R. Dougherty, Eds.*, Denver, CO, USA, July pp. 150160, 1999.

[50] Titus, J., *X-Ray Systems Reveal Hidden Defects*, Test and Measurement World, Newton, MA. pp. 2936. February 1998.

[51] Vardi, Y. and Lee, D., *The discrete Radon transform and its approximate inversion via the EM algorithm*, Intern. J. of Imaging Systems and Techn. 9, 110-117, 1998.

[52] Viergever, M.A. and Todd-Pokropek, A., *Mathematics and Computer Science in Medical Imaging*, (NATO ASI Series), vol. F39: Springer, Berlin, 1988.

[53] Weber, G.W. and Yaşar, Ö., *Discrete tomography: A modern inverse problem reconsidered by optimization*, in: Part 1 of common special issue of Journal of Computational Technologies 9 (2004) and The Bulletin

of Kazakh National University 42, 3 (Mathematics, Mechanics and Informatics Issue; 2004) 115-121, at the occasion of International Conference Computational Technologies and Mathematical Models for Research, Engineering and Education, Almaty (October 2004).

[54] Woeginger, G.J., *The reconstruction of polyominoes from their orthogonal projections*, (Techn. Rep. F003, TU-Graz), 1996.

[55] Yaşar, Ö., Diner, Ç., Doğan, A., and Weber, G.W., Özbudak, F., and Tiefenbach, A., *On the applied mathematics of discrete tomography*, Journal of Computational Technologies 9, 5, 14-32, 2004.

[56] Yaşar, Ö. and Weber, G.W., *Discrete tomography: A joint contributions by optimization, equivariance analysis and learning*, Proceedings of CASYS 2005, Liege, Belgium, August 2005.

[57] Yung Kong, T. and Herman, G.T., Tomographic equivalence and switching operations, in: *Discrete Tomography - Foundations, Algorithms and Applications*, Birkhäuser, 59-83, 1999.