1996

Digital signal processing and micromorphological visualization of crystal lattices of thin films.

Indu. Cidambi
University of Windsor

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Digital Signal Processing and Micromorphological Visualization of Crystal Lattices of thin films

by

Indu Cidambi

A Thesis
Submitted to the Faculty of Graduate Studies and Research
through the School of Computer Science in Partial
Fulfillment of the Requirements for the Degree of
Master of Science at the
University of Windsor
Windsor, Ontario, Canada
1996

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

0-612-30894-4

Canada

Abstract

A graphically-based software has been developed for acquiring, processing, displaying and manipulating scattering intensity data. The program generates the Radial Distribution Function from the scattering intensity data, thereby supporting the interpretation of the intensity data to visualize crystal lattice structures. It allows the user to handle a wide variety of data sets and provides for direct user interaction through an easy-to-use graphical interface. The software can handle complex mathematical procedures such as Fast Fourier Transforms, digital filtering, apodization and peak searching. Although developed specifically for the interpretation of diffraction pattern data, the software is generic and can be used for the analysis and interpretation of other kinds of scientific data.

Dedicated to my mother

Acknowledgements

I express my sincere gratitude to my supervisor Dr. R. D. Kent of the School of Computer Science, who has always been ready to assist me. He has been giving me an incredible amount of guidance and has been very supportive of my work all through. I convey my heartfelt gratitude to my co-supervisor Dr. M. Schlesinger of the Physics Department, who has been an eternal source of inspiration and has given me all the guidance, encouragement and support required during the course of this research. I am extremely grateful to my Internal Reader Dr. R. A. Frost of the School of Computer Science, who has always been ready to guide me and to provide invaluable suggestions. The Physics Department's Lab Technicians Mr. Bernard Masse and Mr. John S. Robinson, have been very helpful and I am especially grateful to Mr. Patrick Seguin for all his help. I would also like to express my sincere thanks to Mr. Walid Mnaymneh of the School of Computer Science, who has been very helpful. I express my sincere gratitude to my mother and sister who have given me all the moral support and encouragement required.

TABLE OF CONTENTS

List of Figures

Chapter 1 INTRODUCTION

1.1 Overview

This research was conducted in order to devise an easier method to process scattering intensity data produced due to diffraction and to investigate the techniques possible to visualize crystal lattice structures from the Radial Distribution Function from which the properties of thin film structures can be deduced. Also the enhancements required in the imaging of microscopes are discussed.

Electron diffraction patterns produced by thin films are used to study crystallite lattice structures, crystallite sizes and lattice distortions in thin films. Henceforth, any reference to specimens is intended to refer to thin film structures. The scattering intensity data generated from diffraction patterns have to be refined and Fourier transformed to obtain the Radial Distribution Function. The various stages in the processing of digital signals (raw data) are filtering of noise, windowing and Fourier Transformation. The Radial Distribution Function is used to compute interatomic distances. The crystal lattice structure of the thin film is deduced and the lattice parameter is evaluated. These are eventually used to visualize crystal lattice structures.

In a nutshell, the graphical user interface for the acquisition of scattering intensity data and the processing of the digital signals was achieved by the use of LabVIEW software. The processed scattering intensity data are read from text

files and the Radial Distribution Function obtained from its Fourier Transform is displayed with the help of Java. Java has been used to provide a web interface for the application and it was ultimately used to display the unit cell model of a crystal lattice since Java is platform-independent and has graphics capabilities.

This thesis is divided into five chapters. The second chapter will review literature related to the present work. This chapter will give an overview of electron microscopes. Transmission Electron Microscopes in particular. and will introduce the significance of Radial Distribution Function. Chapter 3 will describe the experimental setup. experimental details and will discuss the algorithms used for digital signal processing. data acquisition. analysis and implementation of the graphical user interface for the above using LabVIEW. The fourth chapter discusses the Java-based implementation of a graphical interface for the analysis and interpretation of the Radial Distribution Function and the visualization of crystal lattice structure. Chapter 5 gives the conclusions drawn during the course of this research.

1.2 Statement of the Problem

The objective was to implement a graphical interface for the acquisition and processing of scattering intensity data and for the display and interpretation of the Radial Distribution Function. Since computers have become an essential part of modeling and research for both the acquisition and analysis of data. a wide

variety of commercial packages are available today which support the process of data acquisition and processing of laboratory data. While they may be quite powerful, commercial programs often fail to meet the special needs of researchers.

It has been proven that the application of Radial Distribution Function techniques is a powerful means to study the micromorphology of crystalline materials. [8]. Thus there was a need to develop a suitable intensity data analyzer for the interpretation of diffraction patterns from the Transmission Electron Microscope. As part of this development, there was a need for comprehensive, user-friendly software that could acquire, process, display, and manipulate the scattering intensity data generated by the equipment and graphical user interfaces have been implemented for this purpose as they are easy to use and are visually appealing : As the old saying goes, "A picture is worth a thousand words."

The software can handle analysis and visualization of the scientific data with a convenient graphical interface. It was used to render the crystal lattice model. It can handle both small and large data sets. The main goal of this thesis is to give a meaningful interpretation to the scattering intensity data and the Radial Distribution Function obtained from its Fourier Transformation.

Chapter 2 REVIEW OF LITERATURE

2.1 Introduction to Transmission Electron Microscopes (TEM)

Microscopes are used to view objects that are invisible to the naked eye. Microscopes have a long history. The first microscopes were the optical microscopes which only have an objective and an eyepiece. Rays from the object fall on the objective which then creates a real and magnified image of the object, which is eventually converted into a virtual, inverted and enlarged image by the eyepiece. This is the image that is observed by the user of the microscope. But there is a limit on the resolution possible with such microscopes. The resolution can be improved by either :

- decreasing the Wavelength of light or
- increasing the Numerical Aperture.

An increase in the Numerical Aperture means an improvement in the Refractive Index of the medium between the objective and object and for this purpose, an oil immersion objective is used.

The Transmission Electron Microscope (TEM) was the first electron microscope. The working principle of an electron microscope is that electrons which are accelerated by high voltages can be deflected by magnetic and electrostatic fields. An electron gun is used for emitting a cylindrical beam of electrons and

these are then passed through magnetic and electrostatic lenses which are placed perpendicular to the direction of motion of electrons. The cylindrical beam is converted into a cone of electrons which is directed towards the object. Due to the fact that thick specimens totally absorb the beam, thin films are currently being employed for the study. [14].

2.1.1 Resolving power of Transmission Electron Microscopes

The TEM is capable of producing a transmitted image of a thin specimen magnified from 10^2 to approximately $5*10^5$ times with a resolving power of approximately 0.2 nm. An electron beam penetrates a sample to produce a standard bright-field transmitted image. The quality of the image in a TEM depends not only on the expertise of the microscopist, but also on the quality of the sample preparation. [1].

The TEM is used for a variety of purposes : to examine interfaces in composite materials, dislocations in metals, the fine structure of polymers, thin metal films, the crystal lattice structure of catalysts, layered clays, and other specimens and to localize elements within specimens. The fundamental principles of optics are used as the basis of the TEM imaging process. The resolving power of the TEM is related to the wavelength of the energy source used to form the images. De Broglie defined the wave nature of electrons and the variables that affect wavelength: his equation states that $\lambda = h / mv$, where λ is the wavelength, h is

the Planck's constant, m is the mass of the particle, and v is the velocity of the particle. [11]. From this equation, it is seen that the wavelength of a particle or a beam of particles can be decreased by increasing the velocity of the particles.

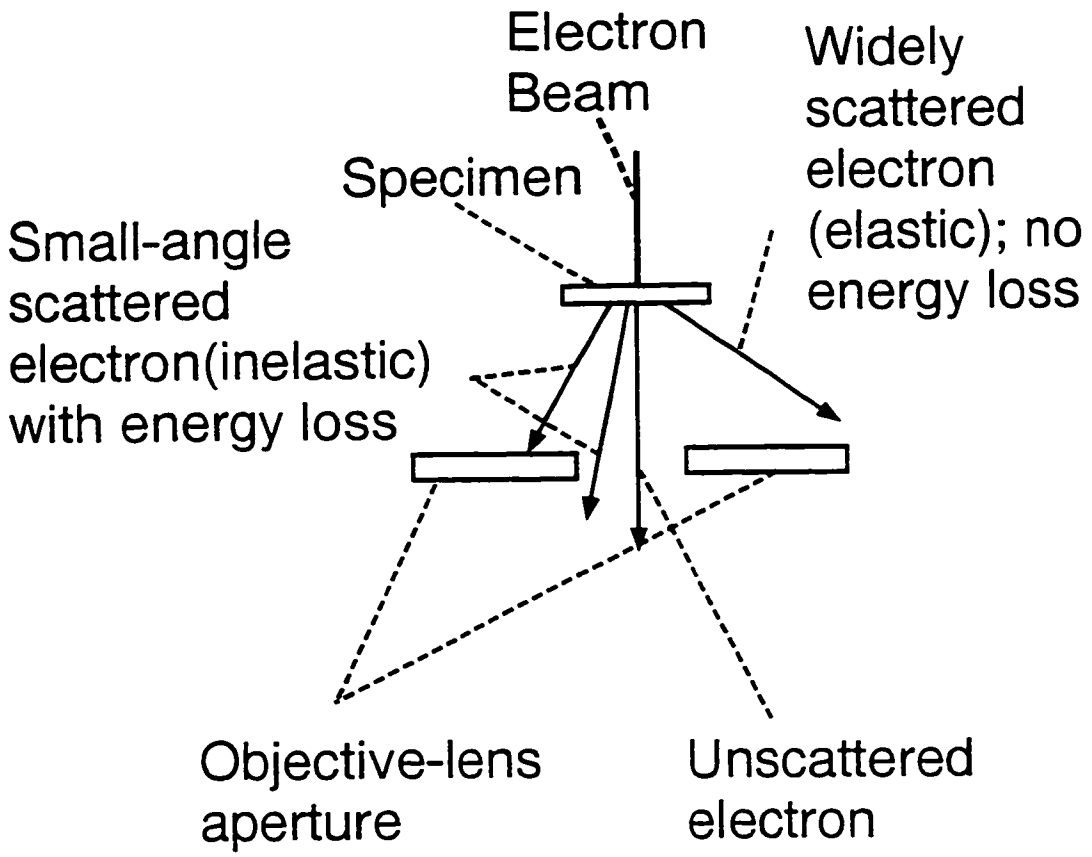


Figure 2.1 The specimen-beam interaction and the aperture of the Transmission Electron Microscope which imparts image contrast.

Abbe's equation states that the resolving power = $0.61\ \lambda\ /\ NA$, where the Numerical Aperture NA is defined as $n\ sin\ \alpha$, where α is the aperture angle of the lens (approximately 0.3° for a TEM), and n is the refractive index of the

medium (approximately 1 for vacuum in a TEM). Numerical Aperture defines the maximum cone of light that can be taken up by the lens from a point on the specimen. Angular aperture is the angle within the lens in which the most divergent rays can pass and form an image. In general, a large angular aperture improves the resolution. This benefit, however, has practical limitations because of the increased spherical abberation that occurs with large angular apertures. [1].

Relating Abbe's and De Broglie's equations demonstrates that aside from lens abberations, resolving power is limited by the wavelength of the energy source used to construct magnified images and that wavelength is related to both the velocity and the mass of the energy source. The wavelength of light is much less when compared to the wavelength of an electron at a high accelerating voltage. Increasing the velocity of electrons results in a shorter wavelength and consequently, increased resolving power. In actuality, because of lens abberation constraints especially spherical abberation, the electron microscope provides an excellent improvement in resolving power over the light microscope. [1].

Specifically, the spherical abberation varies with the cube of the numerical aperture and the chromatic abberation increases linearly with numerical aperture. Both spherical and chromatic abberation thus contribute to the practical limitation of resolution in a TEM. The extent to which chromatic abberation limits resolution is strongly dependent on specimen thickness, which in turn is strongly dependent on the quality of the specimen preparation procedure. In a thick specimen,

multiple inelastic collisions result in electrons of many different energy-loss levels that contribute to the loss of resolution. The ultimate resolving power depends not only on the instrument itself, but also on the specimen preparation, especially the effect of specimen thickness on chromatic abberation. For amorphous samples, for example, "the rule of thumb" states that resolution can be no more than one-tenth the specimen thickness.

To summarize, the resolution of the TEM is dependent on :

- Numerical and Angular Apertures of the lenses in the instrument.
- The wavelength of energy source, which is dependent on the velocity of the electrons.

The practical limitations of resolution of the TEM are :

- Spherical abberation, which in turn depends on the Numerical Aperture.
- Chromatic abberation, which is dependent on specimen thickness, which in turn depends on the specimen preparation procedure.

2.1.2 Diffraction and Diffraction Patterns

Diffraction occurs due to the interference between waves in the area beyond an obstruction. As a secondary wavefront is formed when a primary wavefront interacts with an obstacle, due to the differing path lengths between primary and secondary fronts, wave interference is created. (Figure 2.2 below). The extent of diffraction depends on the wavelength of the energy source. The longer

the wavelength, the greater the spreading of the secondary waves. Significant diffraction into the region behind the obstacle occurs only if the size of the obstacle is smaller than the wavelength of the energy source. For these reasons, diffraction is a major limiting factor for resolution in the light microscope. [2].

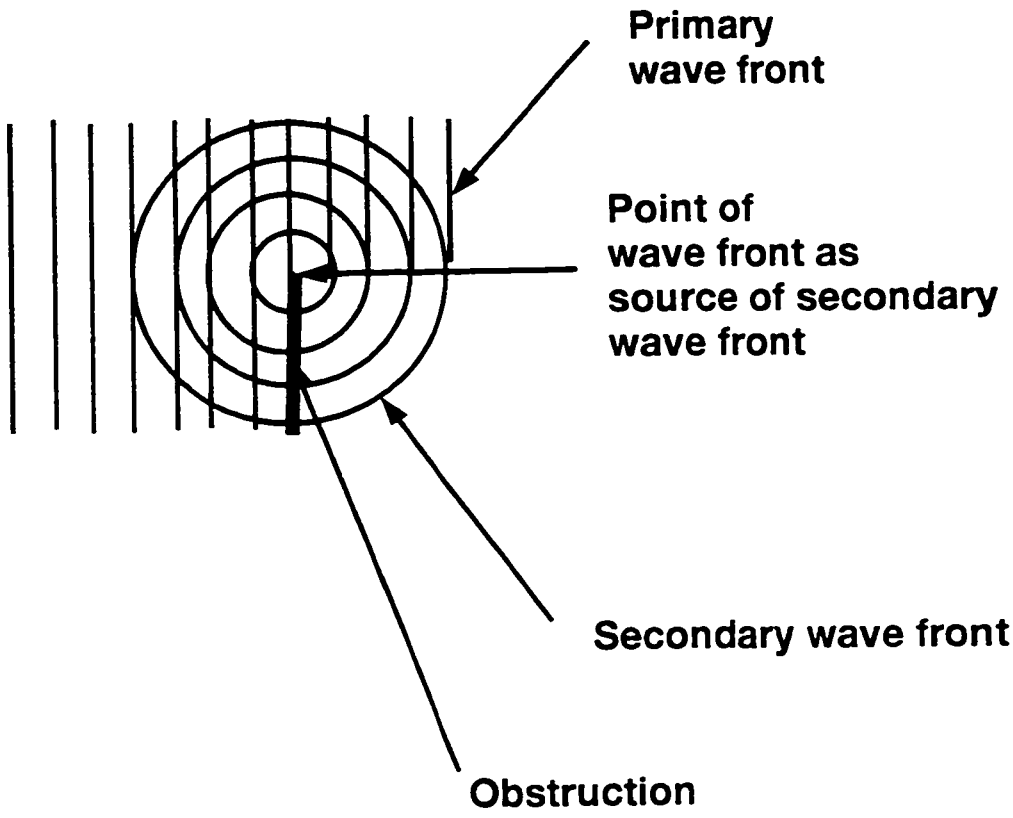


Figure 2.2 Diffraction - Interference of primary and secondary wavefronts.

Diffraction is a useful phenomenon in the study of objects using the TEM. Consider the case where a beam of electrons is passed through an aperture. As the electron beam strikes an edge, diffraction is observed as a result of the interaction

between the unscattered beam (primary wavefront) and the beam scattered by the edge (secondary wavefront). This interaction results in a series of bright lines, the brightest of which is apparent at the edge of the aperture, inside the aperture, or outside the aperture.

Diffraction is useful because it contributes to contrast in images of crystalline specimens and hence it is used to determine crystal lattice structures. Elastically scattered electrons are those produced when electrons from the beam interact with the nuclei in atoms from the specimen (Figure 2.3 below). The electrons undergo a large deviation in their path but little or no energy loss. Elastically scattered electrons contribute to diffraction contrast in an image. [16].

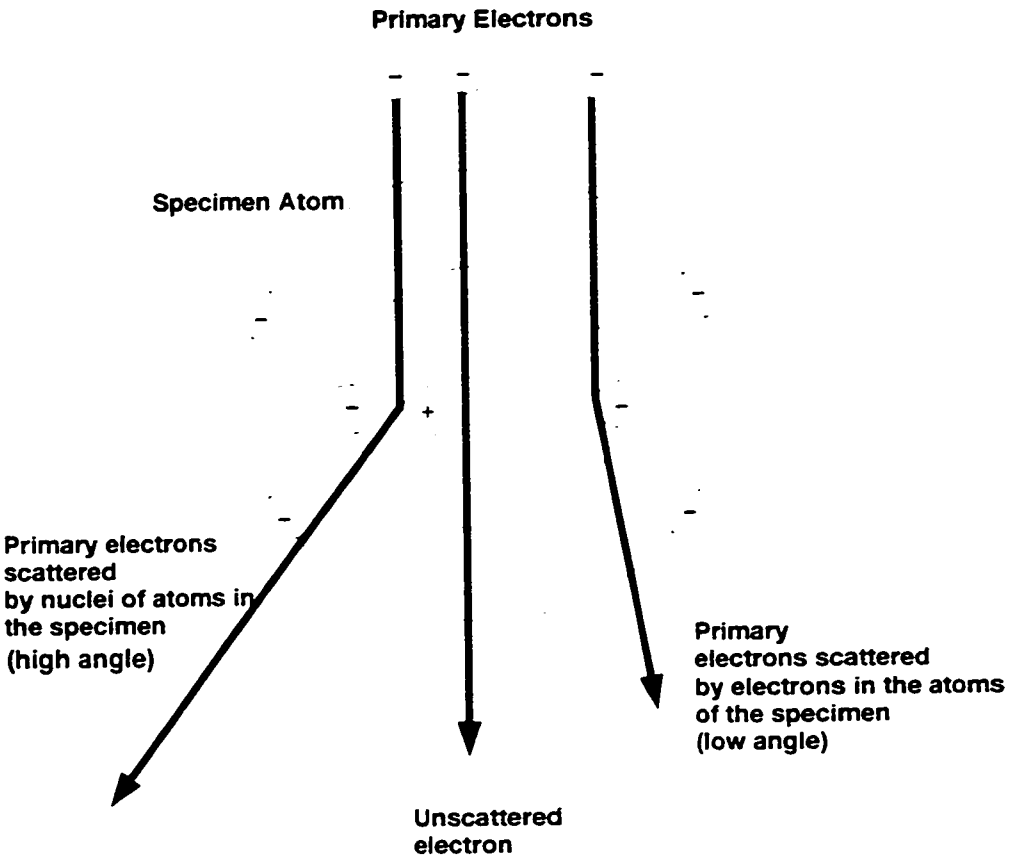


Figure 2.3 Interaction of primary electrons with the nuclei and electrons of the specimen atom.

Figure 2.3 shows the various interactions that occur between the primary-beam electrons and the atoms of the specimen that contribute to the formation of the TEM image. Unscattered electrons pass through the specimen unchanged. Elastically scattered electrons are those that interact with the nuclei of the atoms in the sample. These electrons are scattered through wide angles and suffer no energy loss. Inelastically scattered electrons are those produced when electrons from the beam interact with electrons in atoms from the specimen. These electrons

are characterized by a loss of energy and only a slight deviation in their path. Inelastically scattered electrons are particularly important for imaging samples of low atomic number. They can contribute to chromatic abberation because of energy loss and phase contrast. [12].

As the electron beam interacts with the specimen, a number of signals useful in the formation of the TEM image are generated. Diffraction is one of the most important interactions that contributes to the formation of the TEM image. Electrons may however be absorbed into thick or heavily stained portions of the specimen or into areas of atoms with high atomic numbers, although very few electrons are prevented from passing through the specimen. Electrons are also absorbed into dirt or areas of contamination on the surface of the specimen. If too many electrons are absorbed into a small area of the specimen, such as a puddle of stain or particle of dirt, they may cause a buildup of heat in that area and result in distortion or even destruction of the sample. In this way, such absorption does not contribute to image formation but rather to artifacts in the image. In general, absorption by the specimen does not contribute significantly to image contrast in the TEM, as opposed to the light microscope, which relies on differential absorption as the major contributor to image contrast. [11].

The extent to which the various interactions between electrons in the beam and atoms in the specimen result in scatter, and therefore image formation, depends on the mass-thickness of the specimen. The thickness of the specimen

and the variation in thickness from one portion of the specimen to another, as well as the atomic number of the various atoms making up the specimen (the mass), have an effect on the amount of scattering that takes place. Greater thickness induces proportionally more scattering, although a specimen that is too thick causes increased chromatic abberation, which decreases resolution. Greater atomic number (mass) also results in increased scattering events. This differential scattering between the transmitted and scattered electrons from individual areas of the specimen results in the contrast necessary to form an image on the viewing screen.

2.1.3 Applications of Diffraction Pattern Analysis

In crystalline specimens in preferred orientations, scattering of the incident electrons can occur in specific directions. These directions are defined by the Bragg equation: $n \lambda = 2 d \sin \theta$, where n is the order of the spectrum, which is an integer greater than zero, λ is wavelength of the energy source used, d is the distance between the planes of the crystal also called crystal lattice spacing, and θ is the angle of incidence (or reflection). Figure 2.4 below demonstrates the interaction of the beam with the crystalline specimen. [26].

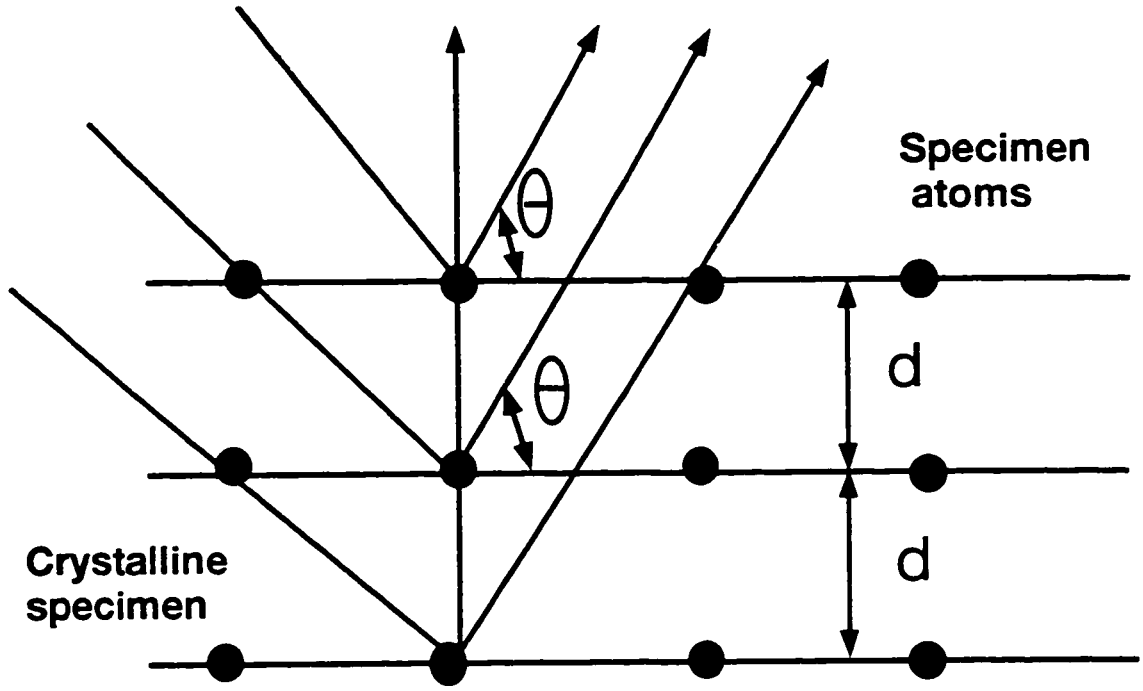


Figure 2.4 Bragg diagram of beam interaction with crystalline specimen.

Depending on the orientation of the crystal, the scattered electrons may pass through the objective lens aperture and produce a bright spot, or more commonly electrons may be absorbed by the aperture, resulting in a dark area. Diffraction contrast can be used to produce dark-field images, and when Bragg's equation is satisfied, crystal lattice structures are determined.

Along with the usual TEM image of a specimen, a *diffraction pattern* is formed at the focal point of the back focal plane of each imaging lens. By altering the current in the diffraction lens, one can view this image in the back focal plane of the objective lens instead of the usual bright-field image. The

diffraction pattern is formed by the elastic scattering of the electron beam by the atoms in the specimen. Such a pattern follows the demonstration by Bragg that crystal structure can be determined by X rays according to a specific mathematical formula.

Briefly, if an area of a specimen is crystalline or amorphous in nature, scattering may occur in a regular manner or in an irregular manner, as the case may be and this gives rise to a pattern of sharp spots or diffuse rings as the case may be. The selected area diffraction may be used to determine areas of ordered or crystalline nature in the specimen or sampie. The pattern of spots or rings contains information about the distances between atoms and thus can be used to determine the arrangement of the atoms. The specific orientation of a crystalline area in the specimen, with respect to the electron beam, determines the pattern and therefore the ease of interpreting the pattern. Specific areas of a specimen may be chosen for examination by selecting a particular diffraction aperture size to define a given area of the specimen. Hence the name, selected area diffraction. [24].

In a crystal, the sites of atoms are determined simply by repeating some sub-unit of the crystal at regular intervals to fill the whole space. Mathematically, we describe a crystal in terms of a regularly arranged set of points whose distribution throughout space looks identical from any point in the set (the lattice), and a prescription that tells us how many atoms of each type to associate with each point and where they should go in relation to that point (the basis). Thus a very

small portion of the crystal sample can be used to study the behaviour of an entire crystal because of the repetitive nature of its structure although this might not necessarily be the case with crystalline thin films. [6, 5].

2.2 Scattering intensity and its relation to interatomic distances — Theoretical Treatment

2.2.1 Introduction to Atomic Elastic Scattering

The derivation of Radial Distribution Functions from interference patterns of X-rays or electrons for materials containing a single species was first achieved by Zernicke and Prins. [35]. The first application of this technique was carried out by Debye and Menke [13] in studying the structure of liquid mercury. Since then, considerable additional research has been done in this area. [16, 15, 33, 14, 32].

The interference pattern obtained by electron diffraction is determined by the manner in which the atoms are distributed in the scattering lattice of the observed sample. Of the various interactions that arise with the passage of an electron beam through the sample, Coulombic interactions are by far the most significant and thus are the only ones that are considered in the first instance. The duration of these interactions are short even with respect to atomic magnitudes and by using accelerating potentials in excess of 30 kV, the range can be reduced to a level that permits the use of approximations.

During the collision process there exists the possibility that either of two different types of events may occur. In one, the incoming electron deflects off the target without energy exchange, and in the other there is an energy exchange. These two processes are referred as elastic and inelastic scattering respectively. These have been dealt with in Section 2.1.2. The elastically scattered electrons retain their original wavelength while inelastically scattered electrons after losing some of their energy obtain a longer wavelength. The difference of wavelengths can be large enough to cause interference effects between the elastic and inelastic parts of the scattered beam, thus to reduce unwanted interference effects, inelastic scattering is always minimized. [8].

2.2.2 Relationship between the amplitude of the scattering intensities and interatomic distances — Significance of the Radial Distribution Function

The scattering of the electron beam by a group of atoms is considered and periodicity is defined by placing other groups of atoms periodically separated from the first. The function representing the amplitude and phase of the beam of electrons scattered by a group of atoms in a particular direction is the Fourier Transform of the group. Thus the Fourier transform of the lattice can be evaluated, and so relative atomic positions in the lattice are determined.

Crystals are composed of groups of atoms repeated at regular intervals, with the same orientation in three dimensions. Each group of atoms can be replaced by

a representative point, and the collection of points so formed is called the space lattice or lattice of the crystal. Thus the crystal is one in which atoms of finite size are located with their mean positions at lattice points. They can be represented only by placing within each unit cell of the lattice a certain arrangement of atoms. It is possible to regard any one set of corresponding atoms in the different unit cells as lying upon a lattice, and thus a crystal with N atoms in the unit cell can be regarded as based upon N identical interpenetrating lattices. [6].

The scattered amplitude at an angle 2θ from the direction of incidence, when an electron wave is incident on a small crystal can be given by

$$A(s) = \sum_{p=1}^{N} f_p(\mathbf{s}) e^{2\pi i \mathbf{s}\mathbf{r}_p} \tag{2.1}$$

where $\mathbf{r}_1$, $\mathbf{r}_2$, ... , $\mathbf{r}_N$ are the positions of N atoms relative to an origin and $f_1(\mathbf{s})$, $f_2(\mathbf{s})$... $f_N(\mathbf{s})$ are the scattering factors and $\mathbf{s}$ is the scattering vector. [31]. Here the term $e^{2\pi i \mathbf{s}\mathbf{r}_p}$ is called the phase factor.

The square of the wave amplitude gives the intensity of the scattered wave and hence from Equation (2.1), we have,

$$I(s) = \sum_{p=1}^{N} f_p(s) e^{2\pi i \mathbf{s}\mathbf{r}_p} \sum_{q=1}^{N} f_q(s) e^{-2\pi i \mathbf{s}\mathbf{r}_q} \tag{2.2}$$

where we have introduced a second set of vectors $\mathbf{r}_q$ which run over the same atomic positions as $\mathbf{r}_p$. [31].

The angles of diffraction produced by a crystal depend upon the dimensions of the lattice. In order to invest the lattice with the power to diffract electrons of the electron beam, it is necessary to give it some material existence. Thus, each lattice point can be assumed to be the site of an atom. Then the positions of the atoms can be specified by the ends of vector r such that $\mathbf{r} = u\mathbf{a} + v\mathbf{b} + w\mathbf{c}$, where **a**, **b** and **c** are the primitive translations of the lattice (the lattice constants or lattice parameters) and u, v and w are integers. [6]. Consider an electron beam of wavelength λ falling on a lattice in a direction defined by the scattering vector **s**. Then to satisfy certain conditions, $\mathbf{r} \cdot \mathbf{s}$ must be an integral number.

Thus $(u\mathbf{a} + v\mathbf{b} + w\mathbf{c}) \cdot \mathbf{s}$ = integer. Since this equation must be true when u, v and w change by integral values, it follows that each of the products separately must be integral; that is u, v and w are already integral.

$$\begin{aligned} \mathbf{s} \cdot \mathbf{a} &= h \\ \mathbf{s} \cdot \mathbf{b} &= k \\ \mathbf{s} \cdot \mathbf{c} &= l \end{aligned} \tag{2.3}$$

where h, k and l are integers, referred to as Miller indices. [5]. These equations (Equations 2.3) are known as Laue's equations. When Laue's equations are

simultaneously satisfied, a diffracted beam of maximum intensity will be produced. The numbers h, k and l specify the order of diffraction.

The mathematical form of Laue's equations made them unsuitable at first for the interpretation of experimental results, and it was not until Bragg placed them on a physical basis that it was possible to make use of them to determine the structure of crystals. Essentially, Bragg's contribution was to identify the integers h, k and l with the Miller Indices of the lattice planes. [5]. The connection between Bragg's law and Laue's equation is brought out by rewriting the latter (Equations 2.3) in the following form:

$$\begin{aligned} \frac{\mathbf{a}}{h} \cdot \mathbf{s} &= 1. \\ \frac{\mathbf{b}}{k} \cdot \mathbf{s} &= 1. \\ \frac{\mathbf{c}}{l} \cdot \mathbf{s} &= 1. \end{aligned} \tag{2.4}$$

Subtraction of the first two equations of Equations 2.4 gives:

$$\left(\frac{\mathbf{a}}{h} - \frac{\mathbf{b}}{k}\right) \cdot \mathbf{s} = 0. \tag{2.5}$$

which means that the vector s is perpendicular to the vector $\left(\frac{\mathbf{a}}{h} - \frac{\mathbf{b}}{k}\right)$. From the following figure Figure 2.5, it can be seen that the latter is in the plane of Miller Indices *(hkl)*.

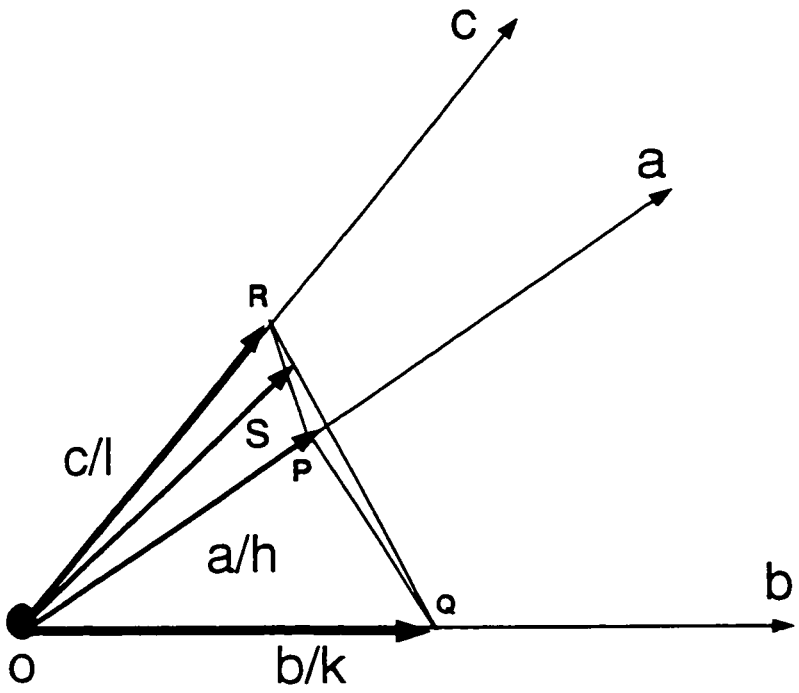


Figure 2.5 Relationship between the scattering vector s and the plane *(hkl)*

where **a**, **b** and **c** are the crystallographic axes. Plane *PQR* is perpendicular to s.

To make the law quantitative, it is necessary to introduce the spacing d_{hkl} of the planes *hkl*: this is the perpendicular distance of the plane PQR from the origin (Figure 2.3) and is the projection of $\mathbf{a}/h$, $\mathbf{b}/k$ and $\mathbf{c}/l$ on the vector s; that is

$$d_{hkl} = \frac{\frac{\mathbf{a}}{h} \cdot \mathbf{s}}{|\mathbf{s}|} \tag{2.6}$$

However, $\frac{\mathbf{a}}{h}.\mathbf{s} = 1$ and $|\mathbf{s}| = \frac{2\sin\theta}{\lambda}$, hence

$$2d_{hkl}\sin\theta = \lambda \tag{2.7}$$

which is Bragg's law.

Circular diffraction patterns are produced by scattered beams that fall on a photographic emulsion. This is because the crystallites in the sample are randomly oriented, and as a result the beam scattered from each crystallite at an angle 2θ from the incident beam will be in a different direction. Therefore the sum of all these scattered beams for a particular angle 2θ produces a cone of radiation. Scanning of the intensity can be performed along any radius of the diffraction rings.

The average scattered intensity for a face-centered cubic space lattice of an N atom crystallite can be evaluated. The primitive translation vectors **a'**, **b'** and **c'** are given by:

$$\begin{aligned} \mathbf{a}' &= \frac{a}{2}(\mathbf{x}+\mathbf{y}) \\ \mathbf{b}' &= \frac{a}{2}(\mathbf{y}+\mathbf{z}) \\ \mathbf{c}' &= \frac{a}{2}(\mathbf{x}+\mathbf{z}) \end{aligned} \tag{2.8}$$

where a is the lattice constant. The primitive basis vectors of the face–centered cubic lattice are shown in Figure 2.6. Using the primitive translation vectors as a basis, the radial distance of any atom p from q in the face-centered cubic crystallite is given by:

$$r_{pq} = \frac{a}{2}\sqrt{(p_1 + p_3 - q_1 - q_3)^2 + (p_1 + p_2 - q_1 - q_2)^2 + (p_2 + p_3 - q_2 - q_3)^2} \tag{2.9}$$

where the p's and q's are arbitrary integers. [5].

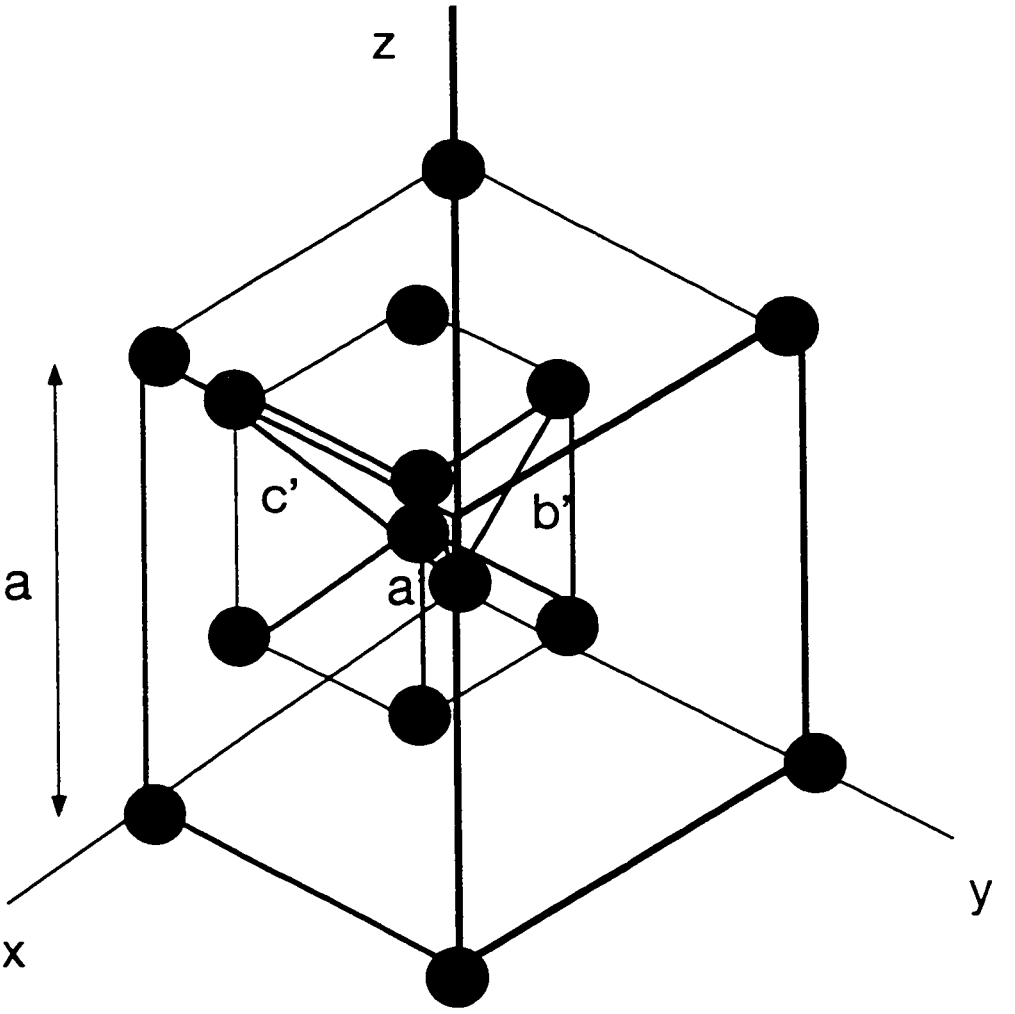


Figure 2.6 The primitive basis vectors of the face-centered cubic lattice.

The factors affecting the resolution of the Transmission Electron Microscope have been listed in Section 2.1.1. These were studied mainly to be able to determine the factors that have to be considered while building a Virtual-Reality Interface for the microscope to further enhance the visualization of their images. Determination of lattice parameter and other crystal lattice variables that are required for the determination and visualization of crystal lattice structure, have been discussed in Section 2.2.2.

In a similar fashion, the lattice parameters for other types of crystal lattice structures can be evaluated. This research deals only with crystalline thin film samples which mostly have face–centered cubic structures.

Chapter 3 DATA ACQUISITION AND DIGITAL SIGNAL PROCESSING ALGORITHMS

The present study uses an experimental apparatus consisting of a TEM to generate diffraction patterns and a data acquistion system to study diffraction patterns of thin films from a TEM . The thin film structures were grown using electroless deposition techniques. A variety of previously grown samples were used to produce diffraction patterns which were recorded on photographic films using a TEM. The thin film samples used for this research were prepared by special methods devised by Dr. M. Schlesinger of the Department of Physics. [10]. The photographic films with diffraction patterns were scanned by a photodensitometer which produced voltage signals corresponding to the intensity contrasts in the diffraction patterns. The signals were measured using the photovoltmeter and the weak signals were amplified and acquired by the digital oscilloscope to be converted into digital signals which were then acquired by the PC.

The digital signals so acquired were further processed for refinement of the scattering intensity data, i. e. they were filtered for noise, windowed and fourier tranformed using LabVIEW, which is a software that uses the Graphical programming language G and which is convenient for data acquisition, creation of good graphical interfaces and for display of graphs before ultimate visualization using Java.

3.1 Experimental Setup

The photographic films with diffraction patterns were scanned by a photodensitometer. The photodensitometer has a photomultiplier which sends a small signal to the photovoltmeter which in turn generates a voltage in the range of 0–200 mV. The voltage of the signal corresponds to the brightness (intensity contrasts) of the diffraction pattern scanned by the photodensitometer. The range of the photovoltmeter was set so that the upper limit corresponds to the central dark spot (maximum density position) of the pattern and the lower limit corresponds to the final last ring (minimum density position) of the diffraction pattern. The sensitivity of the photovoltmeter was adjusted so as to secure maximum precision of measurements. The photovoltmeter was initially connected to a strip chart recorder to obtain the scattering intensity curve corresponding to the diffraction pattern on the photographic film. This was done to check if the connections have been properly made, to fix the appropriate ranges for the measuring instruments and to verify if the scattering intensity data are being acquired in a proper manner. The weak incoming signal from the photovoltmeter was amplified using a voltage amplifier.

Figure 3.1 shows the present experimental setup.

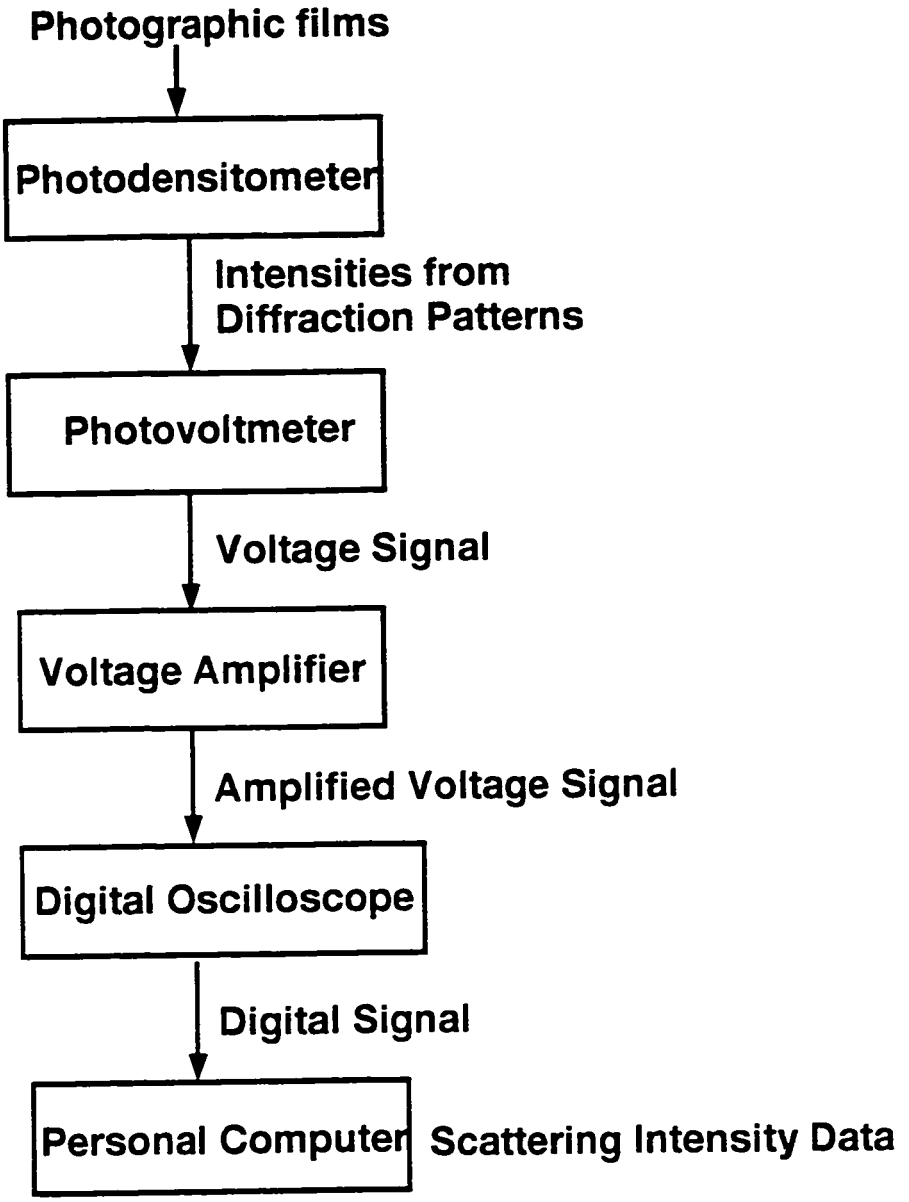


Figure 3.1 Experimental Setup

The voltage amplifier was in turn connected to a Nicolet 2000 digital oscilloscope to acquire data. The power requirement for the oscilloscope is 115V.

The oscilloscope can store eight waveforms. It has a maximum sweep length of 4096 points. The digitizing rate was adjusted to correspond to the speed of the photodensitometer. The photodensitometer eyepiece is moved to the desired starting point and the scan motor is switched on to conduct a left-to-right scan or a right-to-left scan. The time per data point was set to 200 mS so that the photodensitometer could make a complete left-to-right scan of the photographic film at the rate of 5mm/minute. The range of the oscilloscope was set to ± 400 mV. A single channel was employed and a manual trigger was provided.

The oscilloscope has a disk recorder which is used to record waveforms on a floppy. The waveforms are composed of small dots called datapoints. There are 4096 datapoints. Five characters are needed for each data point. The principal function of the oscilloscope was to convert the incoming analog signals into digital signals. In other words, it was used as an Analog-to-Digital converter.

The oscilloscope has storage control buttons that provide three modes of signal acquisition. The Live button is for new signals that are acquired each time a valid trigger is received. The Hold Last button is for the signal that followed the last valid trigger (before the Hold Last button was pressed) and the signal is stored in memory. A signal that is acquired when the button is pressed is held in memory. The Hold Next button is used to store the signal following the next valid trigger in memory and to display it on the screen.

The oscilloscope has an IEEE-488 General Purpose Interface Bus interface which can transfer ASCII code. It was necessary to interface this setup with a Pentium PC for real-time data processing and analysis using the General Purpose Interface Bus Board.

3.2 Data Acquisition Hardware

Data Acquisition is the process of bringing a real-world signal such as voltage, into the computer for processing, analysis, storage and other data manipulation. Hewlett Packard's General Purpose Interface Bus (GPIB) was used to facilitate the communication between the oscilloscope and the computer which in this case is a Pentium PC running Windows 95. The data from the oscilloscope was acquired, using its GPIB interface.

3.2.1 General Purpose Interface Bus (GPIB)

A bus is simply a means by which computers and instruments transfer data. The GPIB is a digital 24–connector parallel bus. It uses an 8–bit parallel, byte-serial asynchronous data transfer scheme. In other words, whole bytes are sequentially hand-shook across the bus at a speed determined by the slowest participant in the transfer. [34].

The computer controls the bus. The controller addresses one talker and one listener to transfer instrument commands and data on the bus. The data bits are

then hand shook from the talker to the listener. The controller manages the flow of information on the GPIB by sending commands to all devices.

Each device that is connected to the GPIB must be assigned an unique address (a decimal number) so that it can be distinguished from other devices on the GPIB. Each device may have a "talk" address for outputting data and a "listen" address for inputting data. The oscilloscope was used as the talker and the computer acted as the Controller-in-Charge. The GPIB address of the oscilloscope is set via a DIP switch in the back cover of the scope. This GPIB address should not be used by any other device on the GPIB. The oscilloscope was assigned a single specific GPIB address.

The GPIB is like an ordinary computer bus, except that the computer has its circuit cards interconnected via a backplane, while the GPIB has stand-alone devices interconnected via a cable. The GPIB interface system consists of 16 signal lines and 8 ground lines. The sixteen signal lines are grouped into data lines, handshake lines and interface management lines. [34].

Commands can be given to the oscilloscope for reading data from the oscilloscope and sending data to the oscilloscope. The commands used for ASCII data transfer are D0 and D1. D0 commands the oscilloscope to input or output data in ASCII code. D1 commands the oscilloscope to input or output data in ASCII code for addresses specified.

Driver software is the lowest level of GPIB software and it interfaces directly with the GPIB hardware. Driver software automatically handles the GPIB protocol and bus-management details. The NI-488.2M driver software is the de facto industry standard for GPIB applications. NI-488.2M software is compatible across a number of different computer platforms. The NI-488.2M runs under multitasking operating systems and can perform multiple tasks and access different devices through the same interface board.

Why was it not possible to perform data acquisition using low-level programming languages? The data acquisition using LabVIEW is straightforward unlike in low-level programming languages which entails the writing of device drivers, connections to appropriate I/O ports (serial or parallel), verification of the status of the read operation and eventually, data acquisition for subsequent processing. If the GPIB board was absent, a cable like the RS-232 C should have been employed which uses the built-in serial port in the computer. The data is sent one bit at a time to the receiver and they have lower data transfer rates and hence are slower and less reliable than the GPIB but do not require the use of a board; they also do not require special hardware. Communication is possible with only one device.

3.3 LabVIEWand the Graphical Programming Paradigm

The Student Edition of LabVIEW (Laboratory Virtual Instrument Engineering Workbench) software was used to acquire data with the help of the GPIB and to

convert the data into a form suitable for analysis. It was used to filter the noise and to perform real-time Fourier Transformation on the waveforms. It was also used to display the acquired signals and the processed waveforms. [34]. I/O components of LabVIEW communicate directly with the plug-in boards and with external physical instruments through the GPIB and this made LabVIEW software ideal for data acquisition. LabVIEW has been used to provide a neat graphical user interface for the data retrieval and refinement and it also has been used to display graphs and store the refined data so they can be visualized later in Java which is platform–independent and which was used to provide a good web interface.

The application programs created in LabVIEW are called Virtual Instruments (VIs). A Virtual Instrument is a software construction that has the characteristics of an actual instrument. It has controls which simulate input devices such as knobs and switches, and indicators which simulate output devices both of which are displayed on the computer screen. VIs are used because they:

- make it possible to develop test programs that exactly emulate the physical functionality and even the appearance of the instruments, yet retain programmability of the instruments in the system.

- provide the necessary high-level programmatic interface to an instrument without sacrificing performance or flexibility.

- leave the definition of the instrument in the hands of the user.

- are configurable and reusable for other applications.

As each VI can be called from another, like a subroutine, the modularity and hierarchy associated with traditional programming languages is preserved.

The VI has a front panel which acts as an interactive user interface and is displayed on the computer screen and operated via the keyboard and mouse. The next component is the block diagram which performs the virtual instrument's function in lieu of the assembly of electronic components. The block diagram is the VI's source code. There are nodes in the block diagram which are analogous to statements, operators, functions and subroutines in a standard programming language.

LabVIEW has been written in a Graphical Programming language called G. The G language consists of graphical objects called icons, each of which represents a specific operation. LabVIEW uses an extensive set of these icons, that are in principle, the building blocks of the program. The LabVIEW program is written by arranging the appropriate icons. The arranged icons are connected together on the screen with graphical wires. This method of programming is intuitive because the user sees the data flow of the program.

One significant difference between G and ordinary character-based languages is that special care must be taken to ensure proper flow of execution. In G, when all the input values of a node are available, the node can execute at any time. If

multiple nodes have valid inputs, they can execute in any order. Execution order can be controlled with sequence structures and by artificial data dependency.

The graphical programming language G was convenient to use because the data flow was visible which means that they are analogous to flow charts. It is possible to run a VI using a single-step program execution mode (node-by-node execution of the block diagram) and this facilitates debugging. Also the Arithmetic functions like add, multiply and divide are polymorphic, i. e. the inputs to these functions may be of different data types.

Disadvantages of the LabVIEW environment include the fact that it is not a standardized program. LabVIEW's code cannot execute "line by line". The principle that governs the program execution is called dataflow. When data arrives at all its input terminals, the node executes. The node supplies data to all its output terminals when it finishes executing. Dataflow contrasts strikingly with the control flow method of executing a text-based program like in C, in which instructions are executed in the sequence in which they are written. Traditional execution flow is instruction driven, while dataflow execution is data driven or data dependent. Since visualization of the crystal lattice structure using LabVIEW was not possible because of the nature of the software, Java had to be used to visualize crystal lattice structures.

What were the advantages is using the Graphical programming paradigm so

special? LabVIEW facilitated the building of a Graphical User Interface and the appendices reflect this. The quality of the graphs obtained are definitely better than those that would have been obtained if low-level programming langages were used. Since it is iconic in construction, it is easier to build VIs. Since it is possible to utilize the current VI that has been implemented and build a new VI extending the current VI, it is going to aid in the faster implementation of another VI that might have new interface elements and input/output components added to it. The new VI can extend the old VI for other applications and in the process add new functionalities to the existing VI. For example, one can use the implemented VI shown in Appendix A to perform new functionalities like varying the filter order or other parameters of the filter used and in the process, provide flexibility to the user. At the same time, one can extend it to implement a data acquisition and processing system which essentially performs a whole new set of processing on the data read. Also, different instruments can be used each with unique GPIB addresses and one can thus reuse the software for data acquisition. It is also possible to simulate new instruments and in the process, add new components. The old VI is generally used as a sub VI in the block diagram of the new VI. Calling of sub VIs in the main VI is analogous to calling subroutines in C. In this implementation, the main VI calls sub VIs such as the Fast Fourier Transform (FFT) VI to perform the Fast Fourier Transformation of the scattering intensity data to obtain the RDF. Programming is modular since it makes it possible to

build a VI to accomplish subtasks of an application and to combine VIs on a block diagram to accomplish a certain task. It can be seen that a VI can have many sub VIs and the sub VIs can have sub VIs and so VIs are hierarchical in nature. This modular approach makes the applications easy to debug, understand and modify.

3.4 Stages in the Processing of Digital Signals

The digital signal, on acquisition by the GPIB, was processed prior to presentation using the data acquisition software. Digital Signal Processing is the theory and application of filtering, coding, transmitting, estimating, analyzing, recognizing, synthesizing, recording and reproducing signals from digital or analog devices using goal-specific techniques.

Digital Signal Processing involves a series of mathematical procedures ranging from sampling, filtering, windowing to Fourier Transformation. The procedures that are employed have been described in the following sections. The different stages in the refinement of scattering intensity data are demonstrated in Figure 3.2.

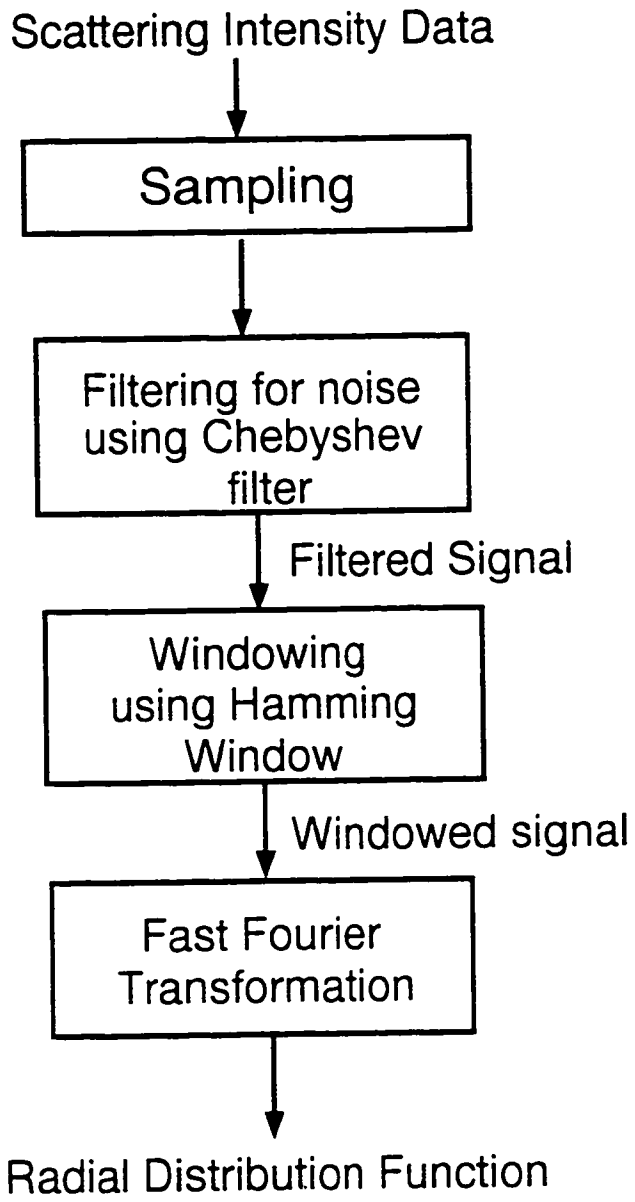


Figure 3.2 Digital Signal Processing stages

3.5 Sampling

When using digital signal processing techniques, it is necessary to first convert analog signals into digital ones. Consider an analog signal $x(t)$ and let the sampling interval be Δt. The signal $x(t)$ can be represented by the discrete sequence of samples as :

$$x(t) = \{x(0), x(\Delta t), x(2\Delta t), x(3\Delta t), \ldots, x(k\Delta t), \ldots\}. \tag{3.10}$$

Because Δt establishes only the sampling rate and has no bearing on the actual sampled (digitized) value, the sample at $t = i\ \Delta t$, for $i=0, 1, 2, \ldots$, corresponds to the ith element in the sequence. Thus,

$$x_i = x(i\Delta t) \tag{3.11}$$

and $x(t)$ can be represented by the sequence X whose values are

$$X = \{x_0, x_1, x_2, x_3, \ldots, x_k, \ldots\}. \tag{3.12}$$

If n samples are obtained from the signal $x(t)$, then the sequence

$$X = \{x_0, x_1, x_2, x_3, \ldots, x_{n-1}\} \tag{3.13}$$

is the digital representation or the sampled version of $x(t)$.

When performing analysis of any kind from point-sampled data, it is assumed that the data was sampled finely enough to catch all the important details. An examination of the sampling rate required to capture all features of importance is most easily done in the frequency domain. Sampling in the frequency domain implies that the time series is periodic in the time domain. In order to capture all features in a signal, the sampling rate must capture information about all of the signals' components in the frequency domain (Fourier Transform) representation. In particular, it must capture information about the highest frequency component of the signal. The sampling theorem states that a signal can be correctly (and uniquely) reconstructed from a set of point samples whose period of sampling is less than half the period of the highest-frequency component present in the sampled signal. Thus the frequency of sampling must be greater than twice the frequency of any component making up the signal (this frequency is known as the Nyquist frequency).

It is possible to sample the time signal of interest at Δt equally spaced intervals without losing information, and the parameter Δt is known as the sampling interval. The sampling frequency f_s is given by $f_s = 1 / \Delta t$, where Δt is the sampling interval. The Nyquist frequency that the digital system can process is given by :

$$f_{Nyq} = f_s / 2$$

The sampling interval for the experiment conducted was obtained as follows. Since the time constant (in milli seconds) of the oscilloscope was set at 200 mS, the sampling interval Δt = 0.2 seconds (200 mS). This determines the sampling frequency.

So f_s = 1 / 0.2 = 5 Hz and the Nyquist frequency was calculated to be f_{Nyq} = f_s / 2 = 2.5 Hz.

If the signal is sampled with a sampling frequency that is below the Nyquist rate, the signal will be distorted by the sampling. In particular, it will be impossible to correctly reconstruct the high-frequency components of the sample that lie above half the sampling rate. The primary effect of this will be loss of detail in the reconstructed image: sharp edges will become rounded and small features will be overlooked by the sampling.

If loss of detail were the only result of under-sampling, this would be tolerable: the larger features are probably of more interest in large scans anyway. There are quite a few problems due to under-sampling.

Oversampling the number of points in the waveform increases the sampling frequency making it easy to overcome aliasing problems.

Thus the sampling frequency has been chosen appropriately such that the scattering intensity data is sampled without aliasing.

3.6 Filtering of noise

The output signals from most measurement devices are corrupted and imperfect. So also are the outputs from the photodensitometer and photovoltmeter. These signals contain random noise. The random background noise and noise from the detection apparatus have to be filtered before further processing of these signals. Also, the voltage amplifier is a major source of high-frequency noise. The incoming signal is highly distorted and it is virtually impossible to analyse such a signal and hence noise filtering is mandatory for the processing of digital signals.

Filters are generally frequency-selective devices. Signals with certain frequencies are passed whereas signals with certain other frequencies are blocked or attenuated. The frequencies that are allowed to pass through the filter are called the passbands and the frequencies that are blocked are called the stopbands. The filter is specially designed to suit the needs of the application. Analog filters are not preferrable because digital filters are more flexible and software programmable. They are also stable and predictable. They have a superior performance-to-cost ratio.

Savitzky-Golay least square smoothing filters [28] were used in the past. Chebyshev Infinite Impulse Response filters are being employed to perform the smoothening of noise in the scattering intensity data.

The sampling frequency plays a very important part in the design of a filter.

An important parameter in the design of a low–pass digital filter is the cutoff frequency. The cutoff frequency must meet the condition $0 \leq f_c \leq f_s / 2$. Thus the sampling frequency determines the cutoff frequency and is chosen so as to prevent aliasing.

Higher order filters can achieve very sharp cutoff specifications at the price of filter complexity, or the filter order.

A further important property inherent in all filters is that noise can never be suppressed without also destroying some significant information. Therefore, good filters must be able to separate those parts where the noise is stronger.

3.6.1 Infinite Impulse Response (IIR) Filters

Infinite impulse response filters (IIR) filters are digital filters whose impulse response can theoretically be infinite in duration. The general block diagram for IIR filters is shown in Figure 3.3:

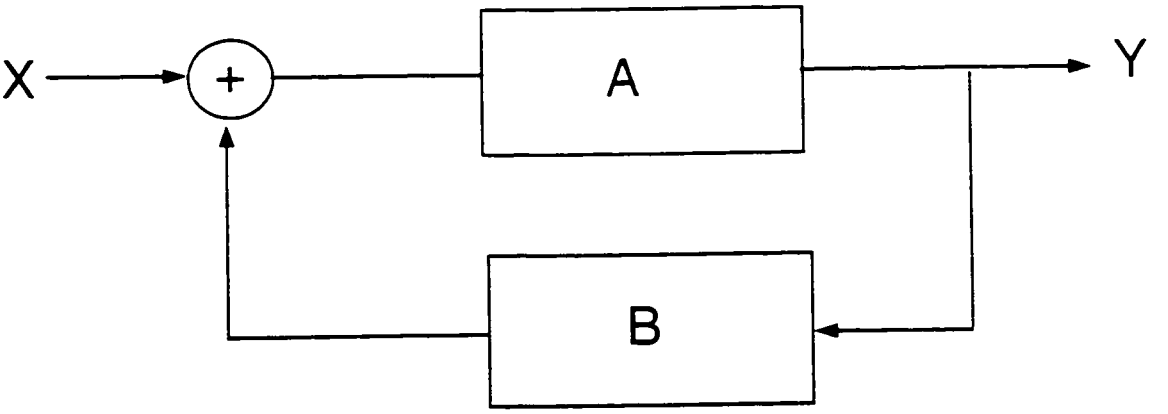


Figure 3.3 General Block Diagram for Infinite Impulse Response Filter

The digital characterization between the input sequence X and the output sequence Y is

$$y_i = \frac{1}{b_0}\left(\sum_{j=0}^{n} a_j x_{i-j} - \sum_{k=1}^{m} b_k y_{i-k}\right) \tag{3.14}$$

where n is the polynomial order of the forward branch labeled A, and m is the polynomial order of the feedback branch labeled B. Essentially, n is the number of forward coefficients (represented by a_j), and m is the number of reverse coefficients (represented by b_k). [34].

The delays associated with the feedback branch of the filter configuration cause the infinite duration of the impulse response. Even when no more external samples are fed into the system, the feedback portion continues to send samples to the filter structure. In practice, the input sequence is finite and the output sequence is truncated to a finite number of samples which is why windowing had to be employed which is explained in the following section. The advantage of using digital IIR filters over finite impulse response (FIR) filters is that IIR filters require fewer coefficients to perform the same filtering operation.

A lowpass filter suits the purpose because it allows low frequency signals to pass and rejects the high-frequency noise signals. There are different types of low pass filters which can be used for different purposes. These are :

- Butterworth Filters which are based on Butterworth Polynomials.

- Chebyshev Filters which are based on Chebyshev Polynomials.
- Bessel Filters which are based on Bessel Polynomials.

Butterworth filters have a smooth, monotonically decreasing frequency response. Butterworth filters have a slow rolloff between the passband, which is the portion of interest in the spectrum, and the stopband, which is the unwanted portion of the spectrum. Chebyshev filters have an equi-ripple magnitude response in the passband, monotonically decreasing magnitude response in the stopband, and a sharper rolloff than Butterworth filters. The Chebyshev filter is prefered for the reasons stated above.

For the above reasons, the raw scattering intensity data was smoothened using the Chebyshev Infinite Impulse Response filter to remove noise. LabVIEW's Chebyshev filter Virtual Instrument was used for this purpose.

3.7 Windowing and Apodization

It is generally necessary that a finite set of samples be considered because in practice, it is impossible to process (or Fourier transform) an infinite set of samples coming from the Chebyshev filter. Hence the time signal is usually multiplied by another signal to terminate the time signal. The time series is generally multiplied by a square wave truncation function (rectangular window) and this act of multiplication to give the termination effect is referred to as windowing. The peaks do get broader but the discontinuities in the time series at the edges

give rise to spurious peaks or ripples and this is referred to as ringing and the ripples are referred to as sidelobes. The ringing (also called leakage) is due to the wideband response to these discontinuities because they determine the amount of frequency leakage and are undesirable artifacts of the windowing process. They must be suppressed as much as possible because they may be confused with the mainlobes of the weaker sinusoids that may be present or in other words, they may be mistaken for real information.

The technique for suppressing the sidelobes is to use a non-rectangular window – a window that cuts off to zero less sharply and more gradually than the rectangular one. This significantly reduces the ripple effect because its Fourier Transformation contains oscillations of smaller amplitude than those of the square wave truncation function. The window that is used is the Hamming Window which tapers off gradually at its end points. It provides a suppression of the sidelobes by atleast 40 dB. The Hamming Window function and its Fourier Transform are shown in Figure 3.4 below.

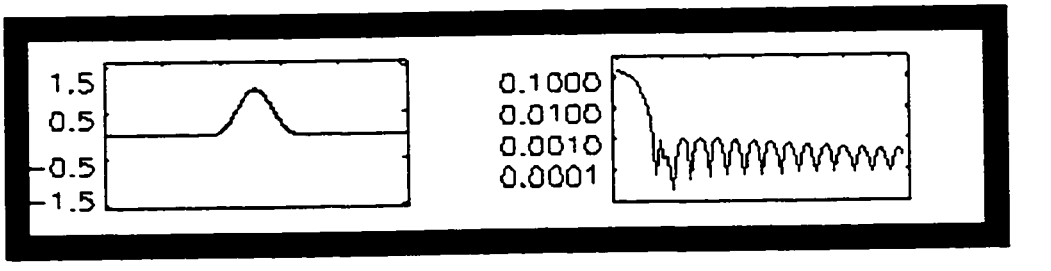


Figure 3.4 Hamming Window Function on the left and its Fourier Transformation on the right.

The Hamming window is a raised cosine type of window and is defined as follows:

$$w(n) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi n}{L-1}\right), if 0 \le n \le L-1 \\ 0, otherwise. \end{cases} \tag{3.15}$$

At its center, $n = (L - 1) / 2$, the value of $w(n)$ is 0.54 + 0.46 = 1, and at its endpoints, $n = 0$ and $n = L - 1$, its value is 0.54 – 0.46 = 0.08. Because of the gradual transition to zero, the high frequencies that are introduced by the windowing process are deemphasized. The sidelobes are still present but are barely visible because they are suppressed relative to the mainlobe. The main tradeoff in using this particular non-rectangular window is that its mainlobe becomes wider and shorter, thus reducing the frequency resolution capability of the windowed spectrum.

Given a finite record of L samples, $x(n)$, $n = 0 .. L - 1$, the windowed signal is defined by

$$x_L(n) = w(n) * x(n) = \left[0.54 - 0.46\cos\left(\frac{2\pi n}{L-1}\right)\right] x(n) \tag{3.16}$$

for $n = 0 .. L - 1$.

Apodization is the process of convolving the spectrum of the time series with the apodizing function in accordance with the convolution theorem. An apodizing function is the Fourier Transform of the window function. The convolution

theorem states that multiplication of two signals in the time domain is equivalent to convolving their spectra in the frequency domain. In general, the multiplication in one domain is equivalent to convolution in the other domain. [4]. In other words, the convolution theorem states that given two time signals *g(t)* and *h(t)*, their product is equivalent to the convolution of the frequency spectra *G(s)* and *H(s)* where *G(s)* is the frequency spectrum of *g(t)* and *H(s)* is the frequency spectrum of *h(t)*. This can be written as follows:

$$f(t) = g(t) * h(t) <==> F(s) = G(s) * H(s) \tag{3.17}$$

The * in the case of *g(t)* * *h(t)* denotes multiplication and the * in the case of *G(s)* * *H(s)* denotes convolution.

The Filtered scattering intensity signal is multiplied by the Hamming Window function to give the termination effect. For this purpose, the time signal was terminated using the Hamming Window Virtual Instrument provided by LabVIEW.

3.8 Fast Fourier Transformation (FFT)

The Scattering Intensity Data have to be Fourier tranformed in order to obtain the Radial Distribution Function.

The mathematical procedure to calculate the frequency components of periodic functions, needed to reproduce the signal is called Fourier Transformation. [4].

Fourier Tansformations can be discrete and continuous. To compute the spectrum of the analog signal digitally, a finite duration of the signal should be sampled and the resulting samples are transformed to the frequency domain by the FFT algorithm.

3.8.1 Continuous Fourier Transformation

For a continuous function of one variable *f(t)*, the Fourier Transform *F(f)* is defined as :

$$F(f) = \int_{-\infty}^{+\infty} f(t) e^{-j2\pi ft} dt \tag{3.18}$$

and the inverse Fourier Transform is given by

$$f(t) = \int_{-\infty}^{+\infty} F(f) e^{j2\pi ft} df \tag{3.19}$$

where j is $\sqrt{-1}$ and e denotes the natural exponent.

$$e^{j\phi} = \cos\phi + j\sin\phi. \tag{3.20}$$

3.8.2 Discrete Fourier Transformation (DFT)

Consider the complex series *x(k)* with N consecutive samples of the form x_0, x_1, x_2,... x_k,... x_{N-1} where x is a complex number of the form

$$x_i = x_{real} + j x_{imag}. \tag{3.21}$$

The Discrete Fourier Transform of this series is denoted by *x(k)* and it will also have *N* samples. The forward transform is defined as:

$$X(n) = \frac{1}{N} \sum_{k=0}^{N-1} x(k) e^{-jk2\pi n/N} \tag{3.22}$$

for $n = 0 \,..\, N - 1$.

The inverse transform is defined as

$$x(n) = \sum_{k=0}^{N-1} X(k) e^{jk2\pi n/N} \tag{3.23}$$

for $n = 0 \,..\, N - 1$.

Although the functions here are complex series, real valued series are represented by setting the imaginary part to 0. In general, the transform into the frequency domain will be a complex valued function with magnitude and phase given by:

magnitude = $|X(n)| = (x_{real} * x_{real} + x_{imag} * x_{imag})^{0.5}$

and phase = $\tan^{-1}\left(\frac{x_{imag}}{x_{real}}\right)$..

3.8.3 Fast Fourier Transformation Algorithm

The Fast Fourier Transform in Digital Signal Processing is used for the numerical computation of the frequency spectrum of a signal. The Discrete Fourier Transform is a mathematical tool to describe the relationship between the time domain and frequency domain representation of discrete signals and hence,

$$Time(Signal) =^{FFT}=> Frequency(Spectrum). \tag{3.24}$$

In other words,

$$x(t) =^{FFT}=> X(f). \tag{3.25}$$

where $x(t)$ represents the time signal and $X(f)$ represents the frequency spectrum.

Different algorithms have been suggested in the past of which the Cooley-Tuckey [7] was very popular. The algorithm that has been used to implement the Fast Fourier Transform has been explained below. The algorithm splits the Discrete Fourier Transform of length N into two DFTs of length $N / 2$ and splits the two DFTs of length $N / 2$ into four DFTs of length $N / 4$ and so on. These are then merged together. The output values are then rearranged into bit-reversed order because the DFT samples are produced in bit-reversed order. It uses the divide and conquer approach.

In all further references, the $x(n)$ is represented as x_n, $x(2n)$ is represented as x_{2n} and $x(2n+1)$ is represented as x_{2n+1} using the subscript notation.

Given a sequence $x(n)$, $n = 0 .. N-1$, of length N, its N-point discrete fourier transform $X(k) = X(\omega_k)$ can be written in the component form of

$$X(k) = \sum_{n=0}^{N-1} W_N^{kn} x_n \tag{3.26}$$

where $k = 0, 1, ..., N-1$ and $W_N^{kn} = \frac{1}{N} e^{jk2\pi n/N}$.

It is possible to split the N-point input sequence x_n into two $(N/2)$-point sequences x_{2n} and x_{2n+1} corresponding respectively, to the even and odd samples of $x(n)$. Since the summation index n ranges over both odd and even values in the range $0 \leq n \leq N-1$, by grouping the even and odd-indexed terms, it is possible to rewrite $X(k)$ as :

$$X(k) = \sum_n W_N^{k2n} x_{2n} + \sum_n W_N^{k(2n+1)} x_{2n+1}. \tag{3.27}$$

To determine the proper range of summations over n, the two terms are considered separately. For the even-indexed terms, the index $2n$ must be within the range $0 \leq 2n \leq N-1$. But because N is even (a power of 2), the upper limit $N-1$ will be odd. Therefore, the highest even index will be $(N-2)$. This gives the range :

$0 \le 2n \le N - 2 \Rightarrow 0 \le n \le N/2 - 1.$

Similarly, for the odd-indexed terms, we must have $0 \le 2n + 1 \le N - 1$. Now, the upper limit can be realized, but the lower limit cannot; the smallest odd index is unity. Thus we have :

$1 \le 2n + 1 \le N - 1 \Rightarrow 0 \le 2n \le N - 2 \Rightarrow 0 \le n \le N/2 - 1.$

Therefore the summation limits are the same for both terms :

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} W_N^{-k(2n)} x_{2n} + \sum_{n=0}^{\frac{N}{2}-1} W_N^{-k(2n+1)} x_{2n+1}. \tag{3.28}$$

The expression leads us to define the two length $N/2$ sequences:

$$\begin{aligned} g(n) &= x_{2n} \\ h(n) &= x_{2n+1} \end{aligned} \tag{3.29}$$

where $n = 0 \,..\, N/2 - 1$ and their $(N/2)$-point DFTs are :

$$G(k) = \sum_{n=0}^{\frac{N}{2}-1} W_{\frac{N}{2}}^{-kn} g(n) \tag{3.30}$$

$$H(k) = \sum_{n=0}^{\frac{N}{2}-1} W_{\frac{N}{2}}^{kn} h(n) \tag{3.31}$$

where $k=0, 1, ..., N/2 - 1$.

Then the two terms of Equation 3.28 can be expressed in terms of *G(k)* and *H(k)*. The twiddle factors W_N and $W_{N/2}$ of orders N and $N/2$ are related as follows :

$$W_{\frac{N}{2}} = e^{-2\pi \frac{j}{N/2}} = e^{-4\pi \frac{j}{N}} = W_N^2. \tag{3.32}$$

Therefore this may be written as:

$$W_N^{-k(2n)} = \left(W_{\frac{N}{2}}^{-2}\right)^{kn} = W_{\frac{N}{2}}^{-kn} : W_N^{-k(2n+1)} = W_N^{-k} W_N^{-k2n} = W_N^{-k} W_{\frac{N}{2}}^{-kn}. \tag{3.33}$$

Using Equations 3.29 and Equations 3.33, Equation 3.28 can be written as

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} W_{\frac{N}{2}}^{-kn} g(n) + W_N^{-k} \sum_{n=0}^{\frac{N}{2}-1} W_{\frac{N}{2}}^{-kn} h(n) \tag{3.34}$$

and using Equations 3.30 and 3.31, the above equation can be written as,

$$X(k) = G(k) + W_N^{-k} H(k) \tag{3.35}$$

where $k = 0 \ldots N - 1$.

This is the basic merging result. It states that *X(k)* can be rebuilt out of the two *(N/2)*-point DFTs *G(k)* and *H(k)*. There are N additional multiplications. Using the periodicity of *G(k)* and *H(k)*, the additional multiplications may be reduced by half to $N / 2$. To see this, it is necessary to split the full index range $0 \le k \le N - 1$ into two half-ranges parametrized by the two indices k and $k+N/2$.

$0 \le k \le N/2 - 1 \Rightarrow N/2 \le k + N/2 \le N - 1.$

Therefore we may write the N equations as two groups of $N/2$ equations:

$$X(k) = G(k) + W_N^k H(k) \tag{3.36}$$

and

$$X\left(k + \frac{N}{2}\right) = G\left(k + \frac{N}{2}\right) + W_N^{(k+\frac{N}{2})} H\left(k + \frac{N}{2}\right). \tag{3.37}$$

Using the periodicity property that any DFT is periodic in k with period equal to its length, we have $G(k+N/2) = G(k)$ and $H(k+N/2) = H(k)$. We have the twiddle factor property:

$$W_N^{\frac{N}{2}} = \left(e^{-2\pi\frac{j}{N}}\right)^{\frac{N}{2}} = e^{-j\pi} = -1. \tag{3.38}$$

Then, the DFT merging equations become:

$$X(k) = G(k) + W_N^k H(k) \tag{3.39}$$

$$X\left(k + \frac{N}{2}\right) = G(k) - W_N^k H(k) \tag{3.40}$$

where $k = 0, 1, \ldots, N/2 - 1$. They are known as butterfly merging equations.

Thus it is apparent that the computation proceeds in t stages, each stage denoted by i , with $i = 1, 2, \ldots, t$, and that each stage must compute the $N/2$ butterfly operations :

$$x_l^i = x_l^{i-1} + W^d x_{l+\frac{N}{2}}^{i-1}$$
$$x_{l+\frac{N}{2^i}}^i = x_l^{i-1} - W^d x_{l+\frac{N}{2^i}}^{i-1} \tag{3.41}$$

where x_l^{i-1} and x_l^i are respectively, the input and output data samples corresponding to the i^{th} stage, where l is the index. Since the input and output samples in the above equations have the same indices, the computation may be executed in place, by writing the output results over the input data. Thus the FFT may be implemented with only N complex storage locations, plus auxiliary storage registers to support the butterfly computation.

The algorithm produces the DFT output samples in bit-reversed order. Thus, these samples must usually be reordered at the end of the computation by performing a bit-reversal operation on the indices l. The bit-reversal process has been implemented by counting in bit-reversed notation. For an 8–bit DFT, a conventional 3–bit counter yields the successive integers 0, 1, 2, 3, 4, 5, 6, 7. If the counter bit positions are reversed, we have 0, 4, 2, 6, 1, 5, 3, 7, which gives the one-to-one correspondence between the natural order sequence and the bit-reversed order sequence. The coefficients W^d are computed via recursion formula $W^d = WW^{d-1}$ in each stage. The bit-reversal process is shown in Figure 3.5.

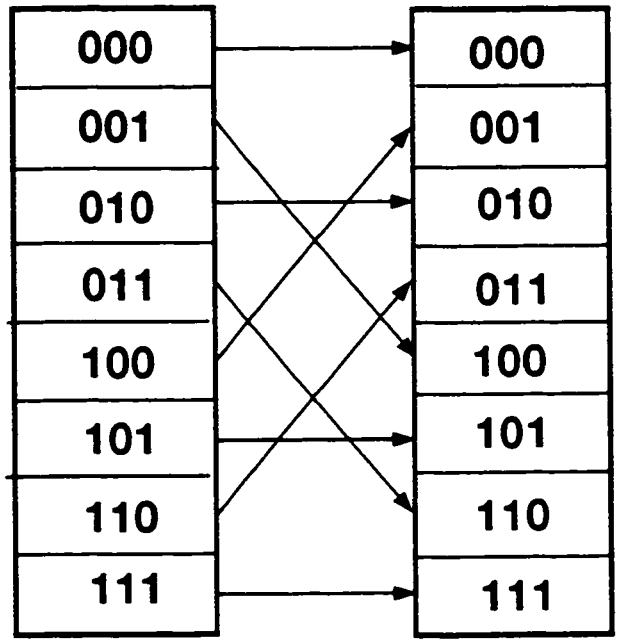


Figure 3.5 The bit-reversal for the Fast Fourier Transformation algorithm.

The algorithm implemented has two basic steps :

1. Performing the butterfly merging operations of the DFTs after splitting the initial DFT further.
2. Bit-reversal.

3.8.4 Significance of the FFT algorithm-Time Complexity = O (n log n)

Thus the Discrete Fourier Transform X_k of an N-point sequence $x(n)$ with $N = 2^t$, can be replaced by two DFTs of length $N/2$ plus N additions and $N/2$ multiplications by W_N^{kn}. The same procedure can be applied again to replace the two DFTs of length $N / 2$ by 4 DFTs of length $N / 4$ at the cost of N additions and

$N / 2$ multiplications. A systematic application of this method computes the DFT of length 2^t in $t = log_2 N$ stages, each stage converting 2^i DFTs of length 2^{t-i} into 2^{i+1} DFTs of length 2^{t-i-1} at the cost of N additions and $N / 2$ multiplications. Consequently, the number of complex multiplications M and complex additions A required to compute a DFT of length N by the FFT algorithm is $M = (N / 2) log_2 N$ and $A = N log_2 N$. The FFT algorithm can be used if the number of data points or data length is an integral power of 2.

Since the FFT algorithm computes a DFT with $N log_2 N$ operations instead of N^2 for the direct approach, practical reduction of the computation load can be very large. In the case of a 1024–point DFT, for instance, where $N = 2^{10}$, the direct computation requires 2^{20} complex multiplications. On the other hand, the FFT algorithm computes the same DFT with only $5 * 2^{10}$ $(N / 2 log_2 N)$ complex multiplications or about 200 times fewer multiplications. Significant additional reduction can be obtained by noting that a number of the multiplications are trivial multiplications by ± 1 or $\pm j$. The space complexity of the algorithm is $O(n)$.

LabVIEW's Fast Fourier Transform Virtual Instrument was used to obtain the Radial Distribution Function from the scattering intensity data.

3.9 LabVIEW Implementation of Data Acquisition and Signal Processing

3.9.1 Data Retrieval and storage

The GPIB Read Virtual Instrument was used to read data from the Digital Oscilloscope. The input to this Virtual Instrument is the number of bytes to be read. The GPIB address of the Digital Oscilloscope is the other input to the VI which is known after initializing the GPIB address corresponding to the oscilloscope's DIP switch specifications. The address string contains both primary and secondary addresses which are decimal values. The primary address is the GPIB board number. The secondary address is the oscilloscope's GPIB address which is set with the help of the oscilloscope's DIP switch. One of the outputs, the status byte is set to true or false, according to whether the read operation is successful or not. The other important output is the set of data strings, which are parsed and stored in an array after conversion to a float value. The float values are stored in a numeric array. Thus the software that has been developed has been used to read data from the instrument. It can also read data from a list of data points stored in ASCII text format, with columns separated by tabs and rows separated by carriage returns.

The waveform collection procedure is as follows. The user specifies the number of bytes required and presses the READ button. Then the required number of bytes are read. There is a status display which allows the user to know if an

error has occured during data acquisition. The software graphs the waveform of the data acquired.

The acquired scattering intensity data from the Digital oscilloscope can also be stored in ASCII text files. The ASCII text files have input data separated by either tabs or commas. The text file of extracted scattering intensity values can be directly imported into an EXCEL spread sheet or into any other kinds of data manipulation software like Spectral Plot for further manipulation. Thus very rapidly, a specific and wide range of multi-line intensity values can be extracted and processed via other Windows application software. This is one of the salient features of the software.

3.9.2 Data Processing

The data was filtered using a Chebyshev Digital Low Pass Filter. The Chebyshev filter is a versatile digital filter based on a Fourier domain approach to digital filtering and is capable of band pass, band reject, and low pass operations. The Chebyshev Low-Pass Filter Virtual Instrument simulates a digital Chebyshev Low Pass filter using sampling frequency, cut-off frequency, ripple in decibels and the filter order. The Virtual Instrument filtered the input using the digital model to obtain the desired filtered data sequence. If the sampling frequency is less than or equal to zero or if the cut-off frequency does not observe the Nyquist criterion or if the ripple and filter order are less than or equal to zero, the Virtual

Instrument sets the filtered data to an empty array and writes all errors to a file.

The Chebyshev filter Virtual Instrument used a low cut-off frequency of 2.0 Hz computed according to the Nyquist criterion and the condition $0 \leq f_{low} \leq f_{high} \leq \frac{f_s}{2}$ where f_{low} is the low cut–off frequency, f_{high} is the high cut-off frequency and f_s is the sampling frequency. The other inputs are the ripple of 0.4 decibels and the filter order of 5. These values can be varied if necessary. The filtered data was displayed on the screen.

The Filtered scattering intensity data was then divided into two sets with the division occuring at the maximum intensity value. One set represents the first half of the scattering intensity curve and the other set represents the other half of the curve, each being the replica of the other. This is done because the diffraction rings exhibit the same pattern on both sides of the central portion. Hence, it was sufficient to analyze one set of data. Also, since the oscilloscope data was inverted, the values had to be reinverted to display the data as required. The acquired/stored input data was then displayed before further processing.

The Filtered data was then windowed in the time domain with the Hamming Window function to terminate it.

The windowed time signal was fed to the Fast Fourier Transform Virtual Instrument which computed the Fourier transform of the data. The data length should be an integral power of two and the program will handle array sizes of

upto 65536 points long. If the number of data points is not an integral power of 2, the software will use the Discrete Fourier Transform routine to perform the Fourier Transform of the data. It is possible to output the real array, the imaginary array and amplitude spectrum although in this case, it was only required to handle real array output. The frequency spectrum was displayed on the computer screen. Provision has been made to convolve the resulting spectrum with the apodizing function if it was not windowed earlier.

The validity of the Fourier transform routine was tested by performing a Fourier Transform of the Radial Distribution Function obtained. The graph that was obtained was a duplicate of the scattering intensity graph that was seen before the Fourier Transformation.

The frequency spectrum (RDF) obtained, was displayed. The graphs that appeared on the screen could be scaled appropriately using the Graphical User Interface.

Appendix A shows the program implemented using LabVIEW software. Appendix B shows the raw scattering intensity data immediately after acquisition; it shows the noisy and noise-filtered real-world data. Appendix C shows the graphs of the real-world scattering intensity data and the Radial Distribution Function. Appendix D shows the Scattering intensity data graph before Fourier transformation and the Fourier transform of the Radial Distribution Function.

This was done to test the Fourier transform routine. Appendix E shows the switch turned on for reading data, the GPIB board number and GPIB address of the oscilloscope, the number of bytes to be read and the status of the read to display error messages during the read operation i.e. inputs and outputs. Also the characters read have been displayed and were stored in the Numeric Array. The length of this array was computed and stored in Length. The filtered array was stored in Filter Array after filtering of data in the Numeric Array. The Filter Array was inverted and the data were stored in New Array. The New Array was split into two new arrays Array3 and Array4 about the maximum intensity value. Array3 was reversed to obtain Array5. Each of these arrays Array4 and Array5 were windowed and Fourier Transformed to obtain the Radial Distribution Function and the RDF graphs (Graph7 and Graph6) were displayed. Graph7 was stored in an ASCII text file. These graphs were again Fourier Transformed to test the Fourier Transform routine to obtain the graphs Graph12 and Graph13 which resembled their respective scattering intensity curves.

Chapter 4 IMPLEMENTATION OF VISUALIZATION OF CRYSTAL LATTICE STRUCTURES IN JAVA

4.1 Java for Data Analysis and Interpretation

Java was used to read data files and perform a Fourier Transform of the processed scattering intensity data to obtain the Radial Distribution Function, to display the Radial Distribution Function Graphs and then to pictorially visualize the structure of the crystal lattice. The peak distances from the Radial Distribution Function were used to determine the interatomic distances.

Java is a CPU-independent, simple, object-oriented, architecture-neutral, portable, multithreaded, secure and distributed language. [21]. Java is a secure environment since access to unauthorized memory locations is not allowed because Java does not allow dynamic memory allocation or pointers.

The principal reason for using Java was to provide a World Wide Web (WWW) interface and, thereby, to make the software platform–independent and portable across different operating systems. Java is an object-oriented language which allows for creation of flexible and modular programs and reusing code. Java is useful for networking, display and creation of user-interface elements.

The Java source files compile to machine-independent "bytecodes". The system includes a bytecode compiler and a virtual machine runtime. This provides architecture independence, enforces the use of verifiable code and implements

security checks. Bytecodes are essentially very simple instructions for a virtual machine. Java allows code to be included with HTML pages and is currently implemented on Solaris, Windows 95 and Windows NT. Java programs can run as stand-alone applications which have "main" methods and they can run as applets which do not contain "main" methods and run in a Java compliant browser. The runtime is typesafe and supports a form of secure loading, so that code from untrusted sources can be added dynamically. Java allows download of libraries from the Internet and allows their execution on the user's machine without security hazards. The language contains mechanisms to verify and execute binary Java programs in a controlled environment, protecting the user's computer from potential viruses and security violations.

To compile a Java program, the command : javac programname.java is used. To run it with the interpreter, the command : java programname is used. Import statements declare class files for the compiler/dynamic class loader.

HOTJAVA, appletviewer (provided as a test vehicle) and Netscape are the major Java compliant browsers which run Java programs. They all implement the Java security model.

Applets are java programs that appear in a Web page much in the same way as images do, but applets are dynamic and interactive. An applet is a software component that typically runs inside some kind of a container (like the web

browser for instance). Applets are used to create animations and figures or for areas that can respond to user inputs. Applets are written in Java and compiled using a Java compiler. The compiled applet can be referred to in the Hyper Text Meta Language (HTML) Web pages. The browser of the person viewing the page with the embedded applet, downloads that applet to the local system and executes it and the reader can view and interact with the applet. The applet is referred to by the <APPLET> tag in the HTML document. The CODE attribute is used to indicate the name of the class that contains the applet. To call an applet, the Java <APPLET> tag is embedded in the HTML page and information is passed with the <PARAM> tag. The browser first finds the applet and downloads it and runs it in the browser.

WIDTH and HEIGHT attributes are used to indicate the size of the applet. Java supports all kinds of variable types and statements and operations present in conventional programming languages. With Java applications, it is posible to pass parameters to the main class through the use of applet parameters in the <APPLET> tag. The <PARAM> tag is used for this purpose. The <PARAM> tag has two attributes which are the NAME and VALUE.

The software developed for viewing the Radial Distribution Function graph, using Java, accepts the frequency of the sine/cosine waves given to the Fourier Transformation function from the keyboard. It also reads scattering intensity data stored in an ASCII text file which has been generated using LabVIEW software

and these can reside in the hard disk of the same PC. The software generates a crystal lattice structure in correspondence with the interatomic distances computed from the Radial Distribution Function, obtained after Fourier transformation of the data that is read from the ASCII text file. Java has excellent graphics capabilities which were employed in the software for the display of lattice structures which actually depends on the text model constructed which depend on the peaks in the RDF graph. The user interface for the software, the graph display and rendering of the model have been achieved using the Abstract Window Toolkit or AWT package.

4.2 Java and the Object-Oriented Programming Paradigm

A graphical interface for the visualization of crystal lattice structures has been implemented using Java. The program has provisions to display sine/cosine waves and their Fourier Transforms and the frequency of the sine/cosine waves can be accepted as input from the keyboard. It is also possible to pass these frequency values as parameters from the HTML document calling the applet employing the <APPLET> and <PARAM> tags.

A class is a template for multiple objects with similar features. Classes embody all the features of a particular set of objects. An instance of a class is another word for the actual object. The classes are the generic representation of an object and an instance is the concrete representation. Instance variables define the

attributes of an object. Methods are functions defined inside classes that operate on instances of those classes. Java includes a set of class libraries that provide basic data types, system input and output capabilities, and other utility functions. It also has classes to support networking, common internet protocols, and user interface toolkit functions.

Java eliminates pointer arithmetic, structs, typedefs, unions, operator overloading and the need to free memory unlike in C++. It provides null pointer checking, array bounds checking, exceptions, automatic garbage collection, a portable and single compilation and is a genuine object language. It does provide single inheritance class hierarchy only unlike C++ which provides multiple inheritance which may cause errors while using overloaded methods and Java does not provide the operator overloading facility. Like any other object-oriented language, it allows software reuse and it provides data and method encapsulation, inheritance and method overloading. Every class contains data variables and methods and every class descends from another (atleast the Object class). The original class is called the super class and the derived class is called the sub class. Packages group java classes in a modular fashion and are imported unlike in C++ where header files are included.

The Applet class methods are : 1) init () which is called once at applet load time. 2) paint (Graphics g) which is called everytime the applet needs refresh. 3) start () which is called when an applet is visited. 4) stop () which is called

when the applet is no longer viewable. 5) destroy () which is called when the applet is no longer alive. 6) resize (int width, int height) which is called when the viewable area of the applet has to be changed. 7) size () which returns an object's Dimension (width and height). 8) repaint () schedules an update (). 9) update (Graphics g) sets background color to default; paints a filled rectangle of width and height; sets the foreground color to default; and calls paint (Graphics g). To access applet parameters in the <PARAM> tag, the getParameter () method is used. The Graphics Object is used for drawing in applets. The methods for basic drawing are drawLine (), fillRect (), drawString (), etc. Other methods are used for image retrieval and display. The context of the Graphics object is owned by the browser and passed to the applet in the paint () method. The Image Methods used are getImage () and drawImage ().

The mouse event methods are mouseEnter (), mouseExit (), mouseMove (), mouseDrag (), keyDown (), keyUp (), gotFocus (), lostFocus () and action (). Applets can create their own Graphics contexts and can pass Graphics context to other methods besides paint (). The play () method provides the easiest access to audio clips.

The software implementation of visualization was implemented as an applet. The applet methods init (), resize (), size (), repaint (), paint () and update () were used when appropriate for the implementation and they had the functionalities stated above. The Graphics Object was employed for drawing and the drawLine

() method was used to display graphs. The action () method was used to perform a certain action on the occurence of some event, say click of a button.

The String class of the java.lang package was used to perform string functionalities. The Math class of the java.lang package was used to implement functions like sine, cosine and power for exponentiation. Classes had been designed such that there was the public class which was the subclass of the Applet class and this has a frame and a panel inside the frame. The java.awt package was used to display common Graphical User Interface (GUI) components like texts, labels and buttons on the panel. The flowlayout manager was used to arrange the GUI components (widgets) on the panel.

When a program runs, it starts executing, runs its initialization code, calls methods or procedures, and continues running and processing until it is complete or until the program is exited. That program uses a single thread — where the thread is a single locus of control for the program. Multithreading, as in Java, enables several different execution threads to run at the same time inside the program, in parallel, without interfering with each other. With the help of threads, it has been possible for the different graphs and the crystal lattice to be displayed because they can all be executed at the same time.

The other classes were subclasses of (or extended) the Thread class and implemented the Runnable interface. Threads were started when a button was

clicked and stopped when the "cancel" button was clicked. Multithreading would have been very useful if the graphs were very complex and if a multi-processing machine was available. Also the class which performed the Fourier Transformation and that which displayed the graphs can be reused for an application which may require only the computation and display of the Radial Distribution Function. The Model class which displayed the model can be reused to display other kinds of models, the text models of which may be different. In all, classes had been designed so that they may be reused in future applications which may require specific types of modeling and various kinds of processing. The software on the whole was made modular to make comprehension and debugging of the program easy. For instance, the computation of the sine/cosine functions and the computation of the Fourier transformation were performed in modules and the display of the graphs was performed using other modules. The bounding box of the model was evaluated in a separate module of the Model class and the model was displayed using another module.

The File class in the java.io package has been used to define files and the FileInputStream class and FileOutputStream class have been employed to read from the data file if it exists and to write to a new model file. Also exceptions have been thrown and caught to display error messages when the file does not exist and to sleep for a certain amount of time while the graph/model is being displayed and to clear the screen so that the next graph/model could be displayed. The Model

class was used to display the model from the atoms created using the Atom class. The atoms are created as balls using different shades of Red/Blue/Green (RGB) colors and displayed using the drawImage () method in the Atom class. The java program essentially will help intensity data sent over the Internet to be fourier transformed, to display the RDF graphs and to display crystal lattice models. The classes can essentially be reused for other applications. Although the Java environment is slow, the computation—intensive fourier transformation of the data did proceed at a reasonably good pace because of the time complexity of the Fast Fourier transform algorithm that was employed. If it is necessary to achieve the same speed as C/C++, it would be required to convert the machine-independent bytecode into machine instructions.

Java can also be used to provide Network client and server socket connections (Transmission Control Protocol/Internet Protocol (TCP/IP) and User Datagram Protocol (UDP)) and Uniform Resource Locator (URL) connections. Another important feature is that native methods can be used with the help of which C functions can be called.

The software can read processed scattering intensity data from ASCII text files. The Java program reads the data from the ASCII text file in which the scattering intensity data are stored and ultimately computes the Radial Distribution Function and displays the Radial Distribution Function graph on the screen and ultimately displays the lattice model. There were five threads which displayed

the graphs and model, one for the sine wave, one for the cosine wave, one for the Fourier Transform of the sine wave, the other for the Fourier Transform of the cosine wave and the last to perform the Fourier transform of the stored data to obtain the Radial Distribution Function graph, display it and to visualize the unit cell of the crystal lattice shown in Figure 4.1.

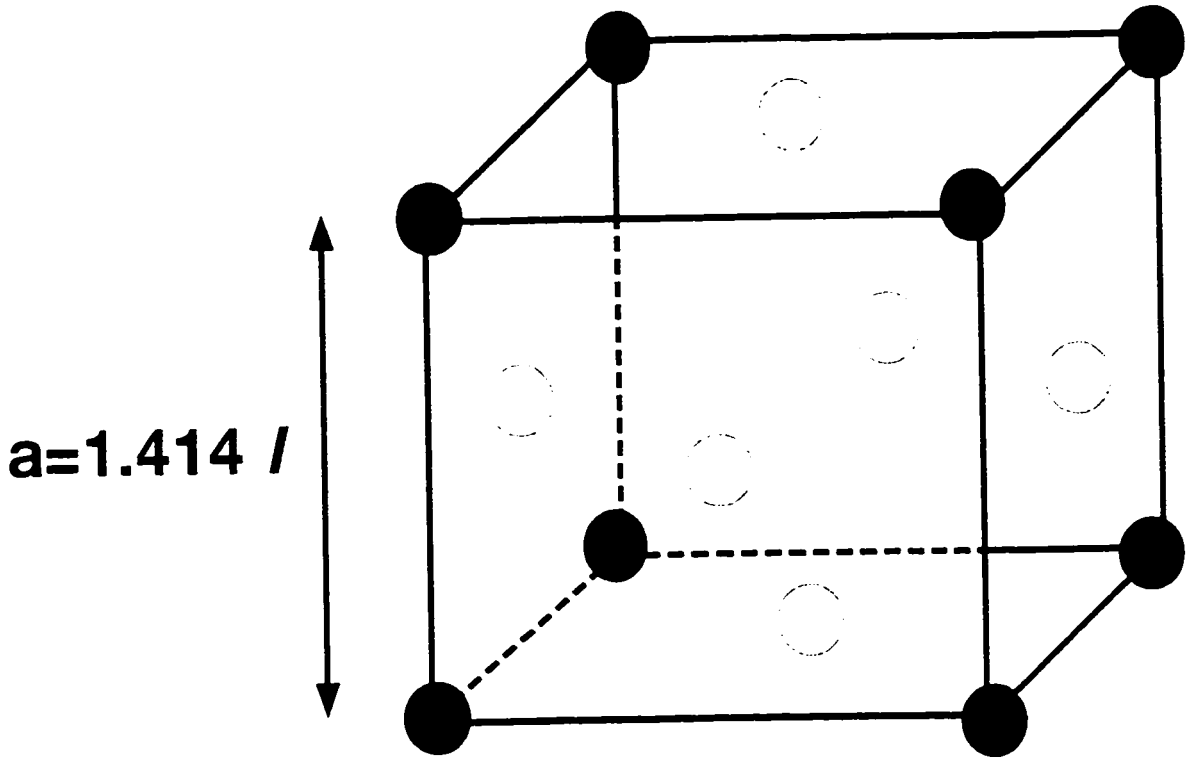


Figure 4.1 Face-centered cubic cell unit

The inputs to the Java program are the scattering intensity data and the Radial Distribution Function graph and the model of the unit cell of the crystal lattice were generated and displayed.

4.3 Visualization of the crystal lattice structure from the Radial Distribution Function graph

Thin Copper films and crystalline metallic samples have been studied in the past to analyze the microstructure of their crystal lattices and to study their images produced by the Transmission Electron Microscope. [17, 30, 22, 3, 23, 29]. In all further references to the Radial Distribution Function peaks, the first peak is not considered because it is an undesirable artifact caused due to the windowing of the time signal. The time signal is terminated at the beginning and the frequencies are zeroed down and hence the artifact. The Radial Distribution Function :

- is the Fourier Spectrum of the scattering intensity data.
- represents the average atomic distribution around a lattice atom as a function of the distance from that atom.
- shows average one dimensional information about the locally varying three dimensional structure of thin films. [10].

The first peak in the radial distribution function represents the average nearest neighbour distance.

The thin film sample is a few hundred Angstroms in thickness. Increase in crystallinity of the sample gives rise to sharper peaks. Hence, as the crystallite size decreases, the peaks tend to broaden. The peaks may broaden due to the presence of stresses and strains in the sample and such broadening occurs in the

direction of the stress and strain.

In summary, the nature of peaks in the RDF depend on:

- crystallinity of sample.
- stress and strain.

If the material of the sample considered is crystalline in nature, the peaks are sharp and well-defined. Once the crystallinity of the sample decreases, the peaks broaden and shift their position. RDF would show a sharp first peak followed by neighbouring peaks at distances relative to that of the first peak. The peaks are identified; the peak distances determine the interatomic distances. The relationship between the interatomic distances determines the lattice structure. The lattice parameter gives the dimensions of the sub-unit. The lattice parameter can be determined from the lattice structure and the interatomic distances. Hence the crystal lattice structure can be visualized. The above method has been illustrated below by considering a Copper sample.

The peaks will be at distances 1, $\sqrt{2}$, and $\sqrt{3}$, relative to the distance of the first peak, if the crystalline sample has a Face-centered Cubic structure. From this it follows that, if the ratios of the interatomic distances are 1, $\sqrt{2}$, and $\sqrt{3}$, relative to the first peak's distance, then the lattice has a Face-centered cubic structure because for a cube of side a, the distance of an atom from the origin atom can be a or $\sqrt{2}a$ (which is the atom diagonally opposite to the origin atom on the

same plane as the origin atom), or $\sqrt{3}a$ for the atom diagonally opposite to the origin atom and on a different plane with respect to the plane of the origin atom. A sample RDF has been displayed below in Figure 4.2.

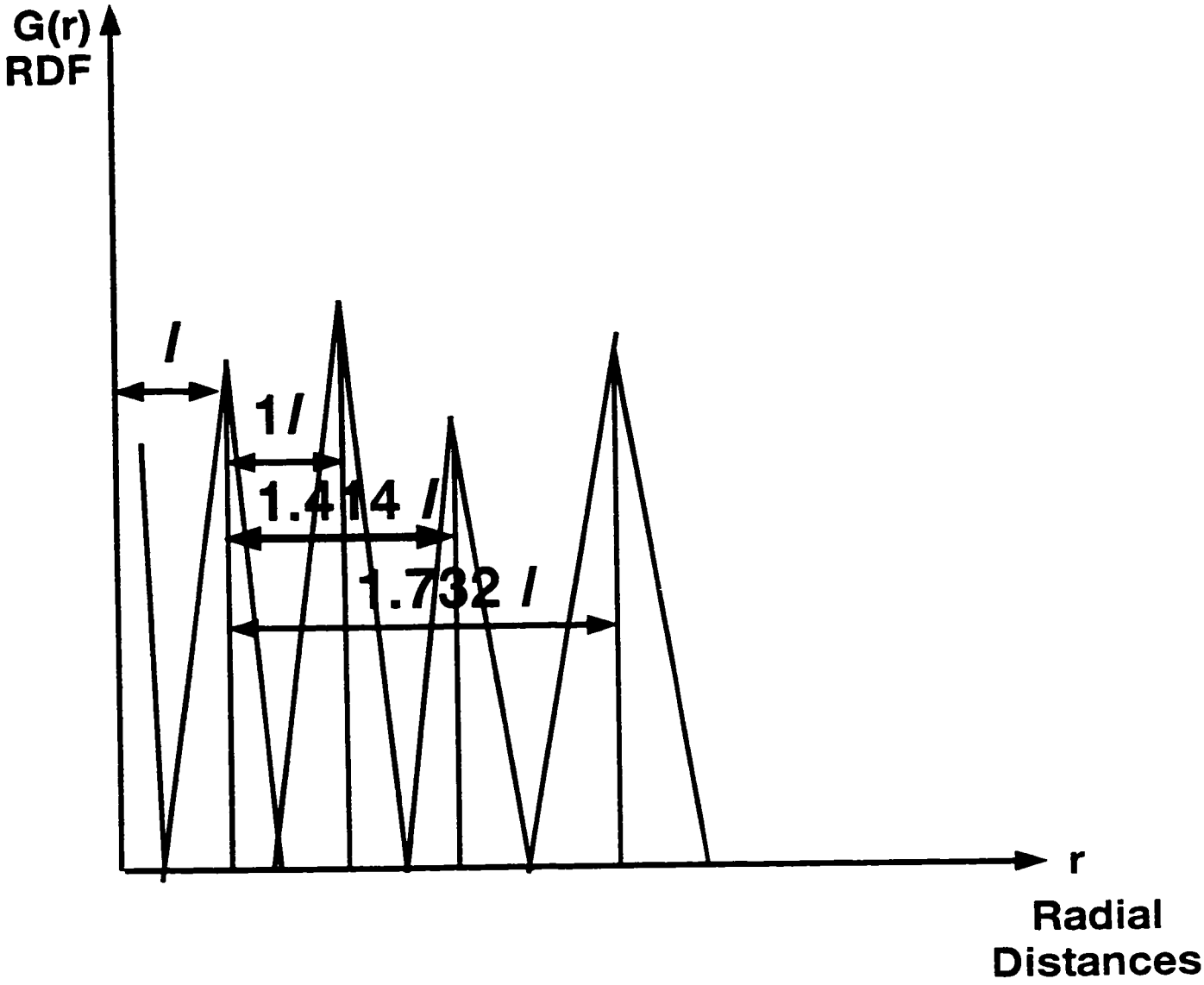


Figure 4.2 RDF graph showing peak distances (interatomic distances) and their relationship. The first peak is usually an artifact and hence is not considered.

So once the ratios have been evaluated, the lattice structure is determined.

Once the lattice structure is determined, the lattice parameter can be evaluated from the interatomic distance. If a is the lattice parameter of the lattice structure, then for a Face-centered cubic lattice, the interatomic distance is $l = \frac{a}{\sqrt{2}}$ which gives the dimension of the unit cell of the face-centered cubic crystal which is the length of one side of the cube. The unit cell is displayed by considering an arbitrary point as the origin atom and displaying the other atoms at distance a, $\sqrt{2}a$, and $\sqrt{3}a$ from the origin atom.

Figure 4.1 shows the unit cell (face-centered cubic cell) with the lattice parameter a evaluated from the value of l evaluated from the RDF graph of Figure 4.2 and here $a = \sqrt{2}l$ or approximately $a = 1.414\ l$.

4.3.1 Analysis of RDF graph

There is a certain region called the wash-out region which is an indication that the Radial Distribution Function peaks subside. The peaks broaden and it is apparently not very desirable to use the region after the wash-out region to interpret the data. This distance approximately gives the radius of the crystallite. An average crystallite radius between 16 Angstroms and 20 Angstroms indicates that the film consists of randomly oriented cubic crystallites with sides of approximately six lattice parameters in length. Hence the size of the crystallites can be obtained from the wash-out distances. The theoretical scattering intensity values can be computed from the crystallite sizes and correlations between the

theoretically calculated RDF curves and experimental ones are used to determine the degrees of stress or strain (or lattice distortions) present. The amplitudes of the peaks are sometimes small if there are voids present within the crystal lattice. [31]. The crystallite radius is half the wash-out distance length. The stress and strain in the lattice are determined from the width of the peaks.

The deformation of a crystal (or rather the variation of the crystal orientation) can affect the orientation of the crystal depending on the direction in which the deformation is applied. Also in certain cases the rotation of the lattice can lead to a change in the orientation of the lattice. It has been proved that changes in the lattice parameter can induce variation in the shear modulus. In the face-centered cubic orientation, changes in the wavelength can cause variations in the shear modulus. But lattice rotation can cause softening of the shear modulus. Hence lattice rotation and changes in the lattice parameter have an effect on the softening of the crystal. [20].

Thin films when grown are placed in metallizing baths with a specific pH. As the pH of the metallizing baths increases, the peaks become better resolved. Also crystallinity of the film increases with increase in pH. The crystallite sizes also depend on pH. It has been shown that the crystallite sizes have doubled for an increase in pH from 8 to 12. [9]. Also the intensities of the peaks vary and this variation can be attributed to the variation in the density of the samples. If there are fewer voids (vacant sites), the density is lower and this gives rise to peaks

of lower amplitude. The voids actually tend to distort the crystallites. Also, the presence of impurity atoms can propagate distortions within crystallites.

4.3.2 Java Implementation of micromorphological Visualization of crystal lattices

The peaks were identified from the Fourier Transformed scattering intensity data. The interatomic distances (distances of neighbouring peaks in the RDF, relative to the first peak) were computed after this. The lattice type was determined from the interatomic distances. In this case, the interatomic distances are compared with those of a face-centered cubic crystal and if they matched, the structure was determined to be face-centered cubic. The lattice parameter was evaluated from the lattice type and the interatomic distance; the dimensions of the unit cell (lattice parameter) that makes up the crystal were computed; the crystal lattice structure was visualized. The face-centered cubic model could be visualized once the lattice parameter was evaluated and the atoms were displayed in a fashion deduced from the crystal lattice structure. The distance between these atoms corresponded to the interatomic distances that were evaluated from the RDF graph. Hence, a 3D model of the unit cell of the lattice was displayed by assuming an arbitrary point as the origin atom and displaying the other atoms at specific distances given by the peak distances, as if the atoms were at the corners, center and faces of a cube. It is immaterial as to which point is chosen to be the origin because the lattice is made up of a number of unit cells and all the unit

cells are arranged in a symmetrical manner in the lattice. The program flow has been illustrated in Figure 4.3.

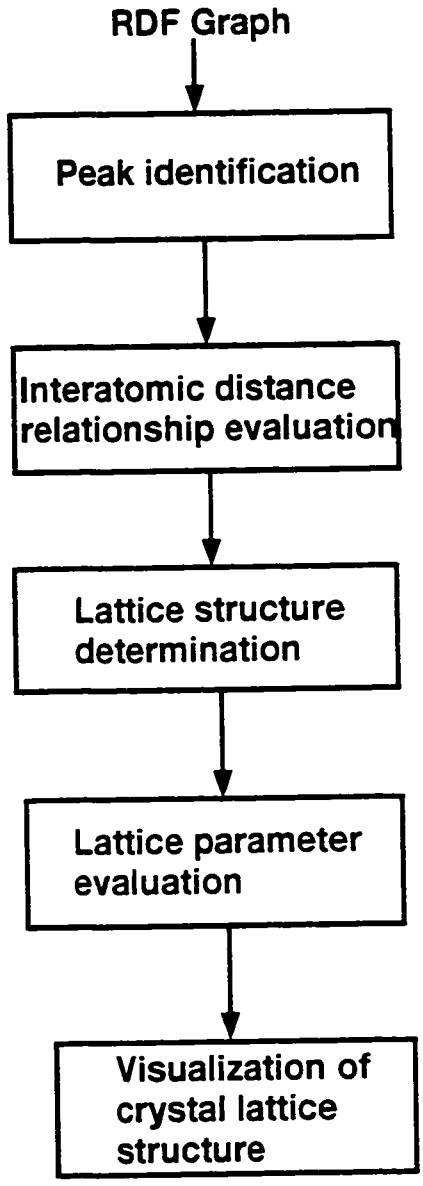


Figure 4.3 Visualization flow diagram

The positions of the atoms with respect to the origin atom were evaluated; these were stored in an ASCII text file. The ASCII text file was read to obtain the coordinates of the atoms. A bounding box was created for the model. A matrix class had been defined to perform 3D transformations such as rotation, scaling and translation; they were performed using Matrix multiplication and addition appropriately to render a neat model. It is also possible to reuse this class in future when one needs to modify the program so as to have new parameters such as scale factor to modify the model size, the rotation angle to rotate the model through a certain angle along the X, Y or Z axes to get a different perspective of the model and to translate it to move the model to a particular position. The transformed 3D unit cell model was then displayed. To summarize, multithreading in the Java program has been illustrated in Figure 4.4.

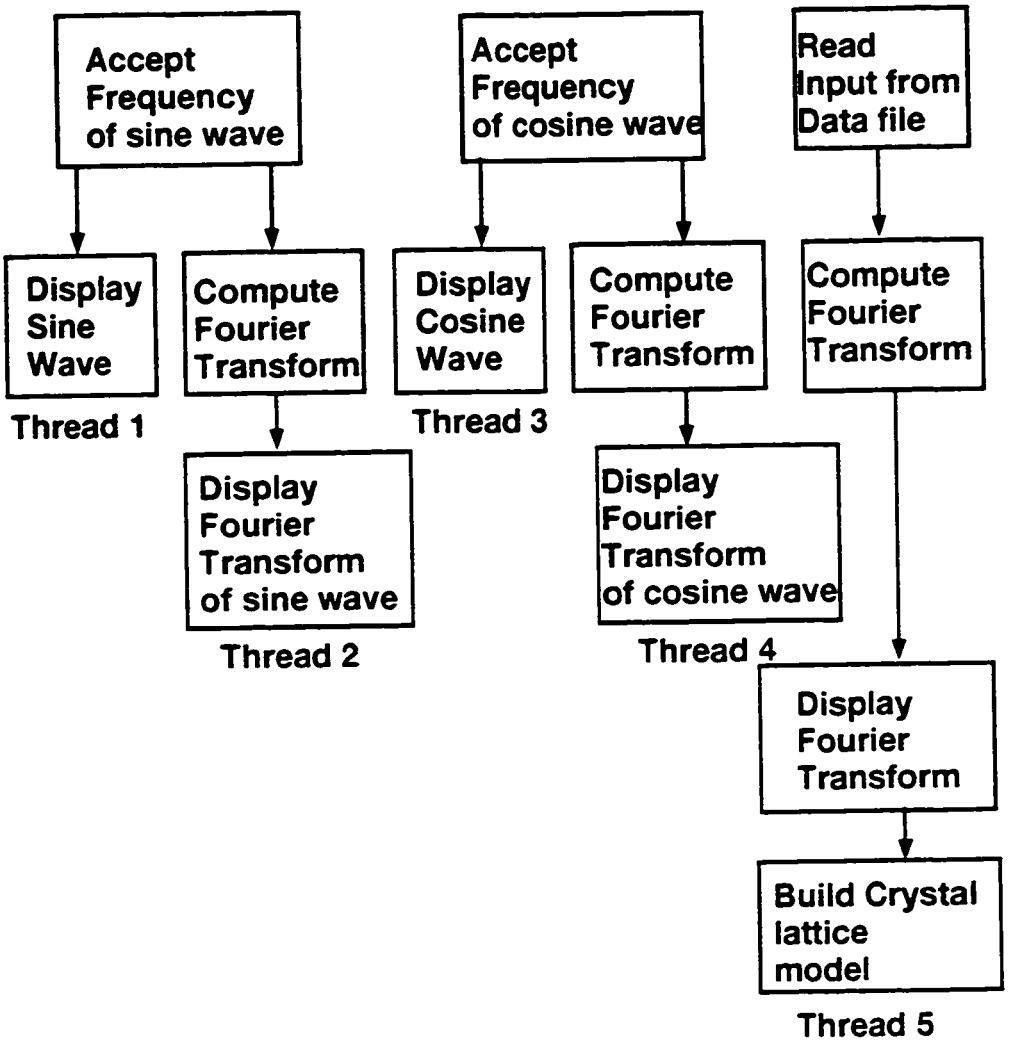


Figure 4.4 The five threads for displaying the various sine, cosine and Fourier graphs and to build the crystal lattice model.

Appendices F and G show snapshots of the Fourier Transformation of the sine wave and the face-centered cubic cell unit model from the Java implementation.

Chapter 5 CONCLUSIONS

A unique feature of the software implemented is that it was written in a graphical environment that allowed a poweful user interface to be constructed and resulted in a program that was easy to interpret and modify.

A graphical analysis program like this can help interpret results easily and effortlessly. It is hoped that more scientists will begin to use this type of computing environment so that useful programs can be more easily created, shared and tailored to specific applications. This software can be combined with existing commercial software programs to produce meaningful interpretations of the data. It is also possible to generate two-dimensional plots of data in real-time. The software has been developed essentially to help in the visualization of scientific data. Several inferences can be drawn based on the research conducted:

• The scattering intensity data can be read, processed, displayed and analyzed in real-time in a convenient way using the graphical interface. User interfaces are a step towards the goal of more accessible, friendlier, more intelligent and distributed analytical instrumentation. They help the researcher to become more productive by allowing data to be presented in a more meaningful manner and by not coming between the researcher and the data. The software developed made the acquisition of data easy. The incoming data can be displayed immediately. The scattering intensity data are processed efficiently and quickly.

• A useful interpretation of Radial Distribution Function Graph has been provided in a more direct and rapid fashion than previous methods. The software facilitates the analysis of the graph's peaks to study the properties of the crystallite sample. The peak identification is accurate and there is little chance of errors occuring at any point in the analysis. This approach is reliable.

• A feasible and practicable means of visualizing crystal lattice microstructure has been developed. The visualization of crystal lattice structures is made possible from a knowledge of the interatomic distances after peak identification in the RDF graph that is obtained.

Thus the Java software read scattering intensity data from files stored in the hard disk of the PC after the data had been acquired and stored in text files using LabVIEW. It then generated the Radial Distribution Function and plotted the graph of the RDF, identified peaks, determined the lattice parameter from the lattice structure and rendered the lattice model. The phases in the software engineering cycle such as requirements analysis, writing of specifications, design, software development, testing and integration of software are the most important aspects of software application development. It is important to realize that the principles of software engineering can be applied universally and virtually to any kind of application development. The advantages that were incurred from the use of Java were that platform–independence, portability and multithreading were achieved. Java supports distributed application development which is one of its

greatest advantages. One major disadvantage of using Java is that compilation of Java programs takes time and the loading of applets is slow.

5.1 Future directions

Software has become the limiting factor in achieving the full potential of computers. Researchers must realize that the software environment is as important as the hardware and must be factored to the design and development of systems.

The instruments mentioned for this application can be manipulated using a Virtual Reality system. It is possible to have a Remote Manipulator which can change the orientation of the sample in accordance with the user's hand movements, to get different kinds of diffraction patterns from the Transmission Electron Microscope which can be visualized to gain a better understanding of the lattice structure. Also, it is possible to have a head mounted display with trackers to track the position of the head, to allow the user to move his/her head over the lattice structure being visualized. That will help the user identify the exact orientation of the lattice and to detect the strains and voids in the lattice. The accelerating potential of the electron beam can be adjusted using another remote manipulator and this can vary the scattering intensities. A Charge Coupled Device can be used for rapid and efficient data acquisition for subsequent processing. It is possible that the whole system be automated so as to visualize lattice structures of samples whose images are produced directly from the TEM without the help of

photographic films. An alternative to the LabVIEW software is the LabWindows software that is similar to LabVIEW but the controlling software for LabWindows is written in C with the aid of code-generation routines. Because C and C++ are popular programming languages in the instrumentation market, LabWindows was designed to make the C programmer as productive as possible. LabWindows is more flexible for this reason. LabVIEW and LabWindows are not compatible with each other.

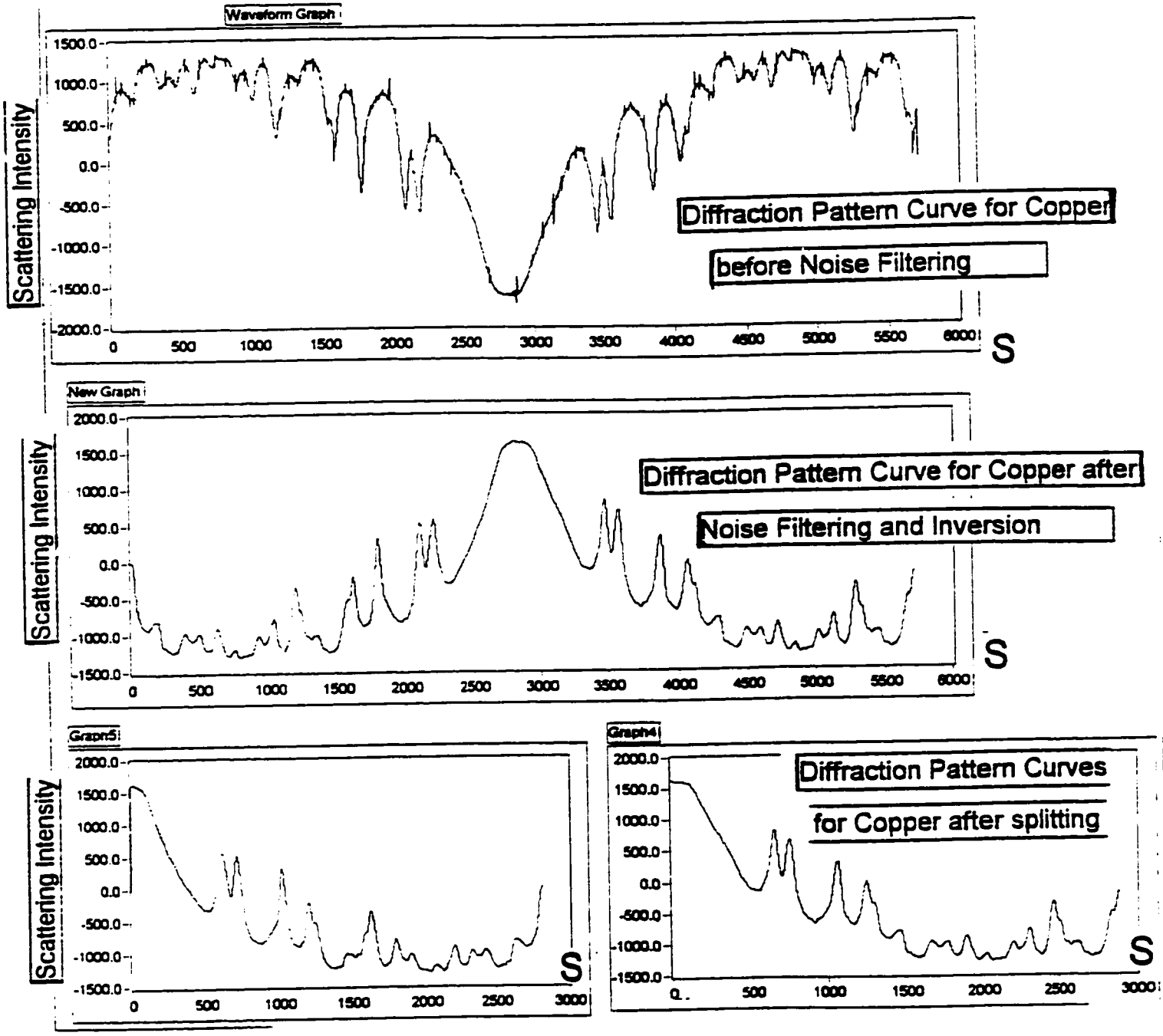
A more recent development was the LabVIEW web server. In general, the job of a web server is to wait for incoming connections from browsers, listen to their requests for information and send them the requested information (HTML documents, graphics etc.). The LabVIEW program or the web server provides the connection to the World Wide Web. When a connection comes in from a browser to see the Radial Distribution Function graph from the scattering intensity data received and to render the lattice structure, the LabVIEW server running on the same computer can use a technique called the server push to capture the graphical image of the LabVIEW VI, send it to the browser, and this can be refreshed continually. The LabVIEW web server technology has been used to provide real-time information updates over the Internet. Hence a scientist can virtually send the samples to be visualized over the Internet and once these samples are received, the lattice structure models can be rendered and this will enable him/her to view them in real time.

APPENDIX A :

LabVIEW program

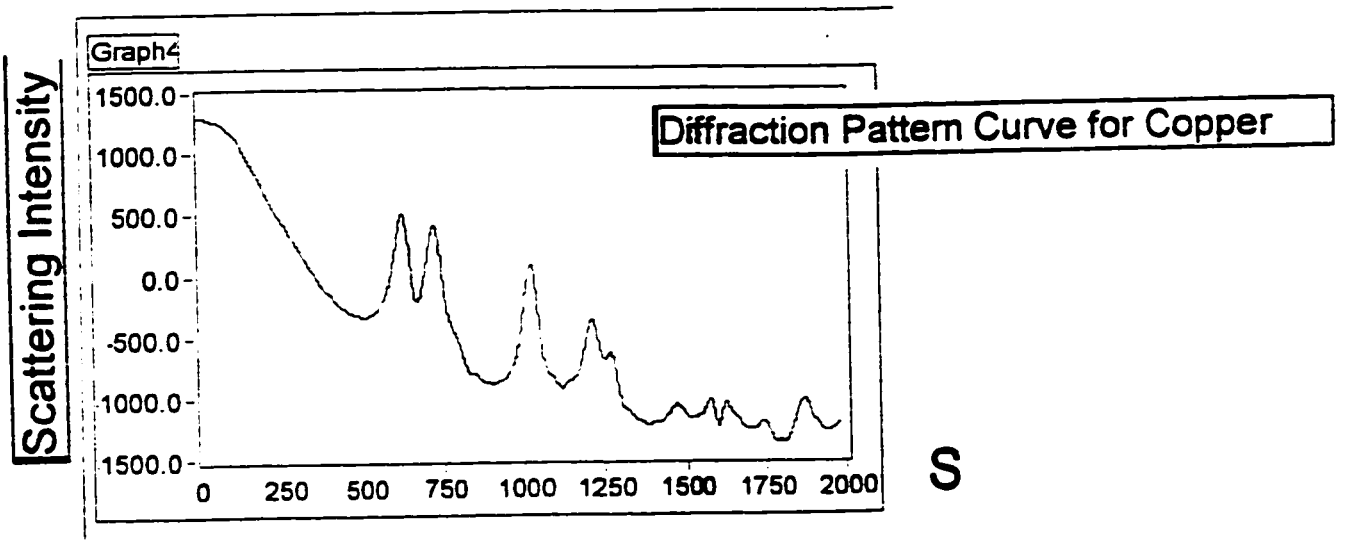
This appendix shows the LabVIEW program's block diagram. The program made it possible to acquire scattering intensity data, filter the data, window the digital signal and display the graph. Eventually it Fourier transformed the data and made it possible to display the Radial Distribution Function.

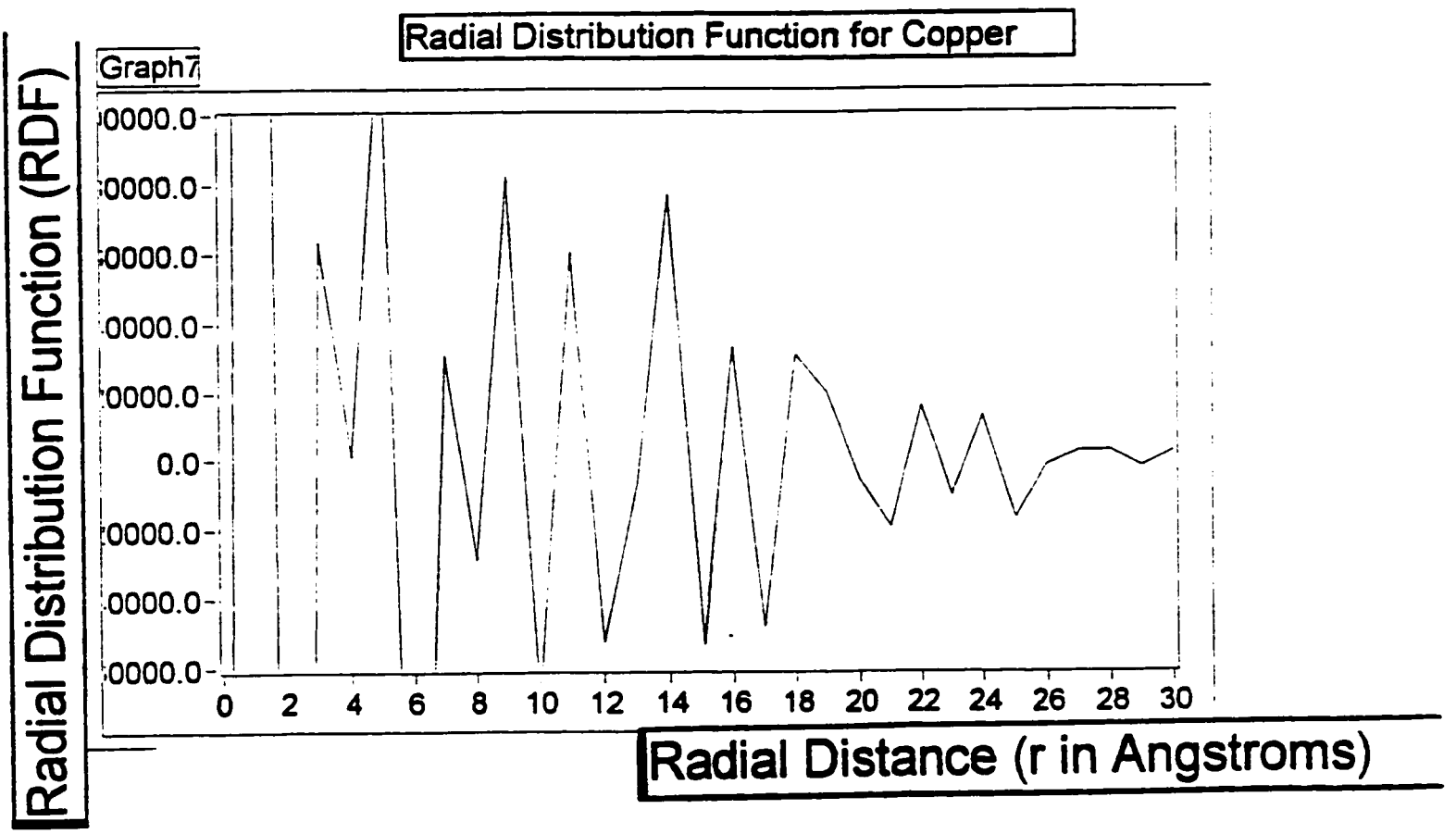
APPENDIX B : Diffraction Pattern Curves and their processing



The diffraction pattern curves before noise-filtering, after noise-filtering and inversion and splitting about the maximum intensity value are displayed with the help of the GUI achieved using LabVIEW.

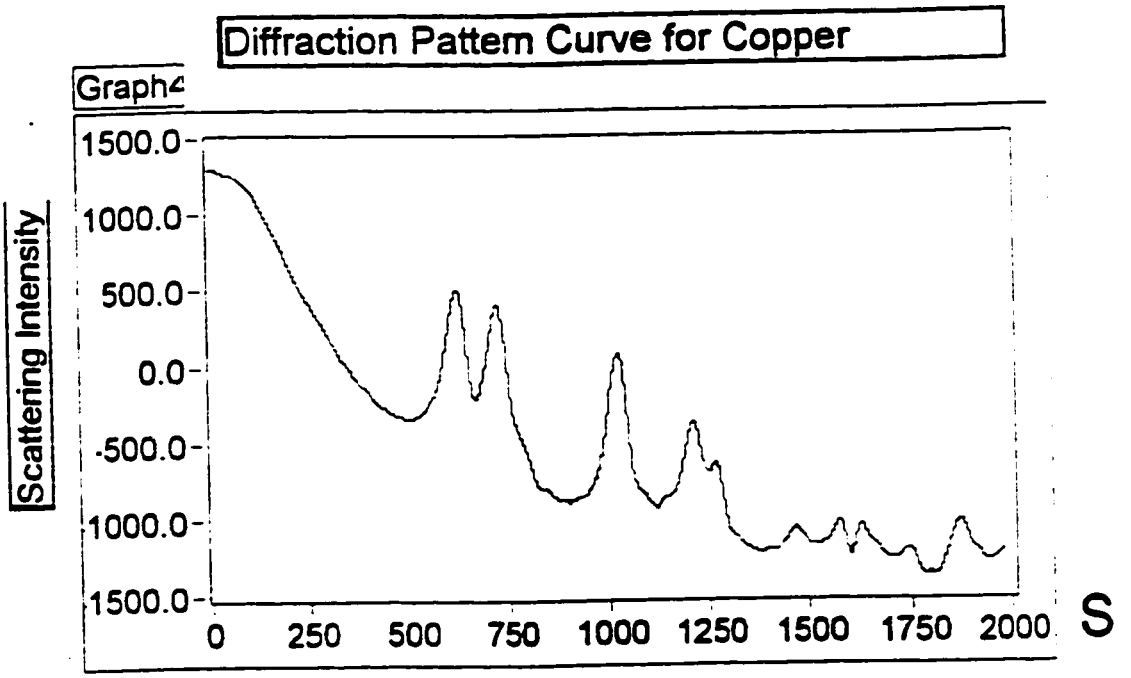
APPENDIX C : Diffraction Pattern Curve and RDF

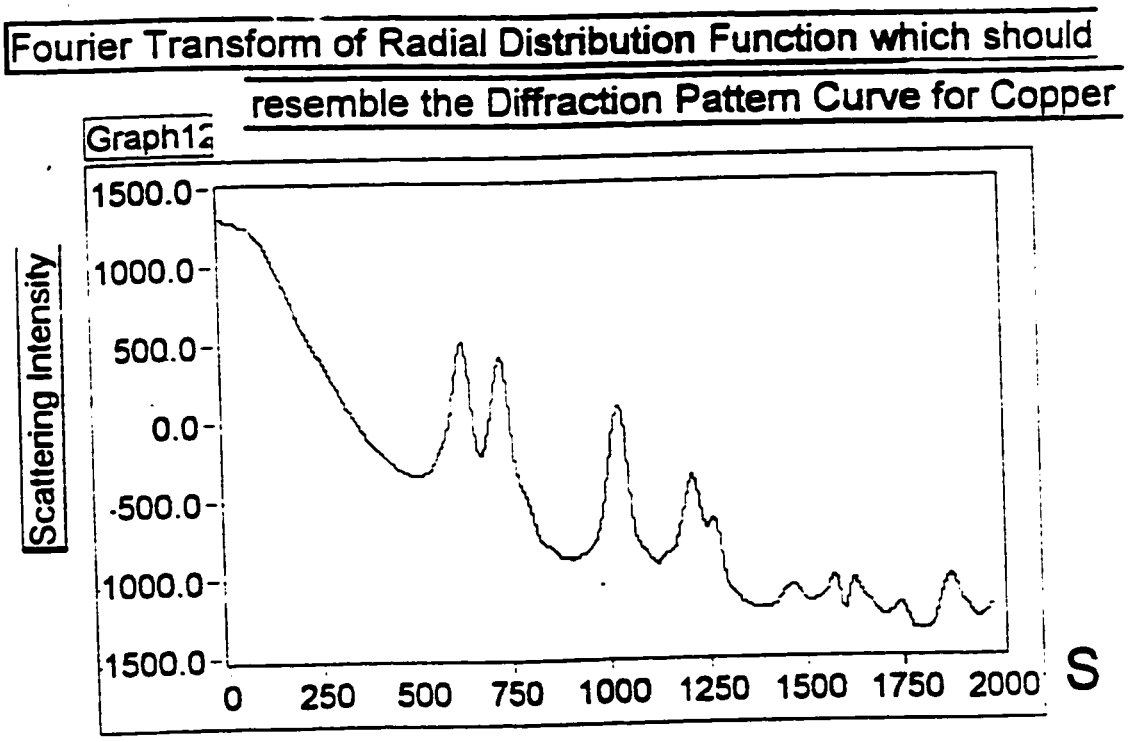




The diffraction pattern curve and the Radial Distribution Function after Fourier Transformation have been displayed with the help of the GUI achieved using LabVIEW.

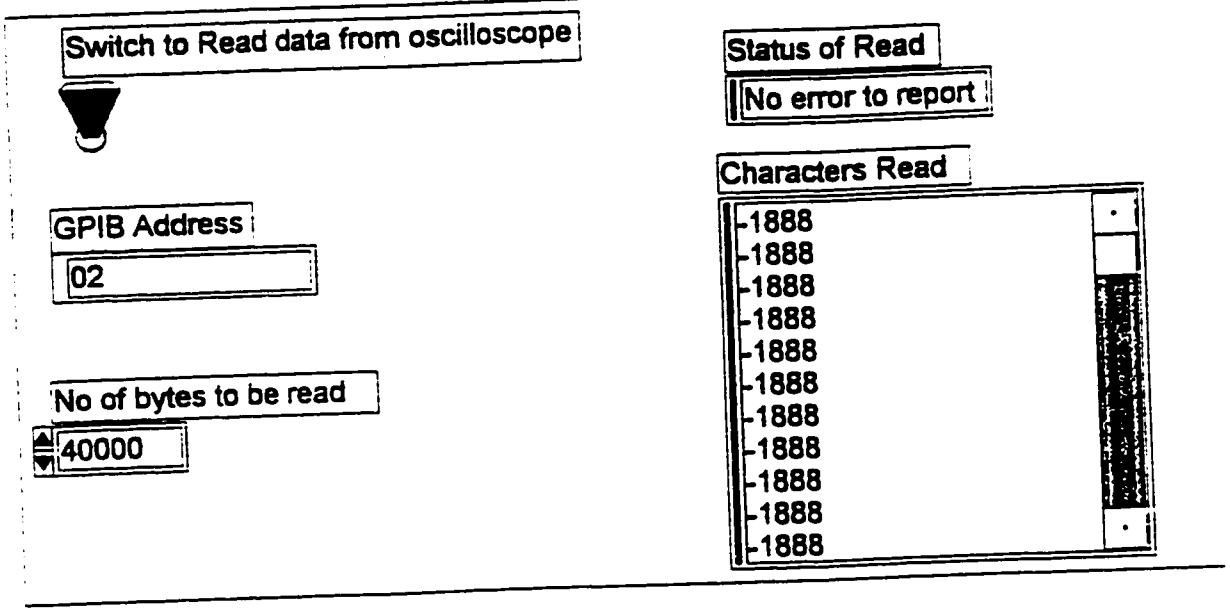
APPENDIX D : Testing of Fourier Transform routine

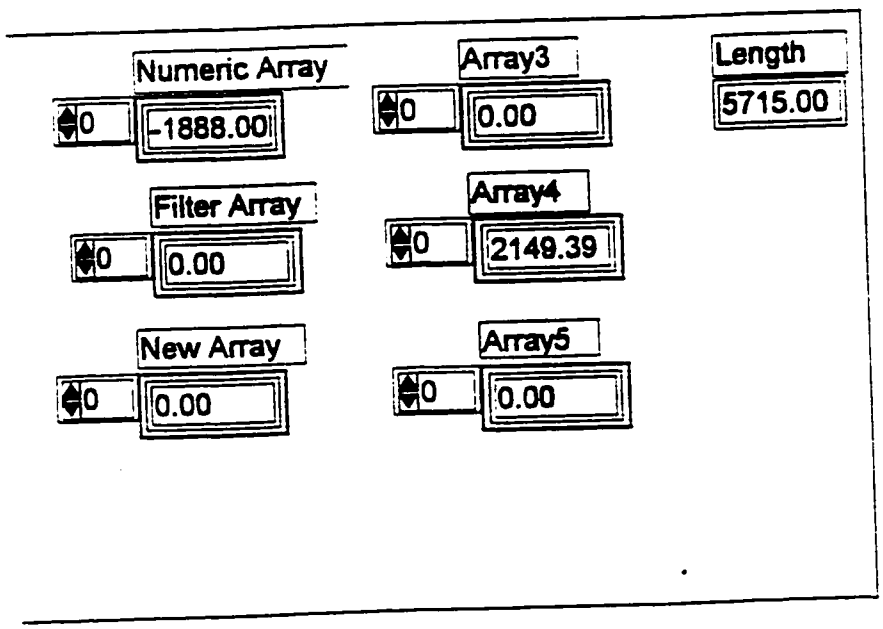




The diffraction pattern curve before Fourier Transformation and the one obtained after Fourier Transformation of the Radial Distribution Function are displayed with the help of the GUI achieved using LabVIEW.

APPENDIX E : Graphical User Interface using LabVIEW (Inputs and Outputs)

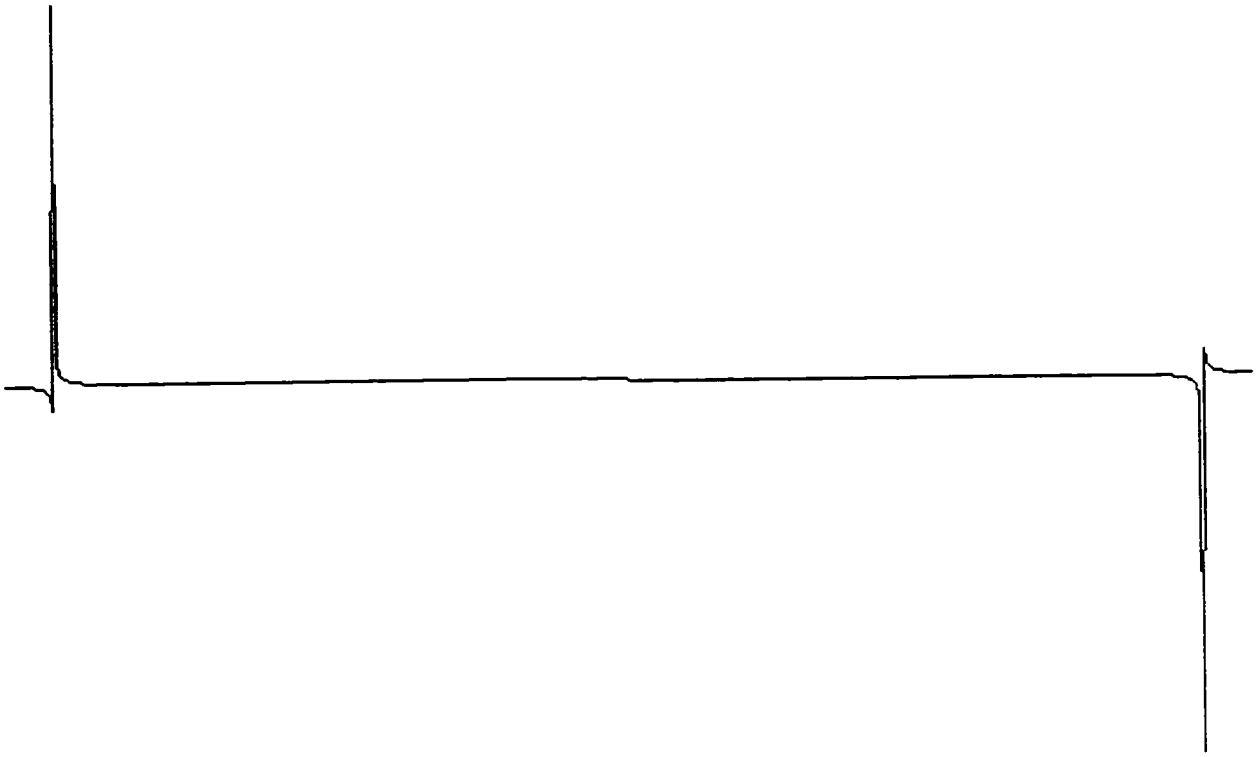




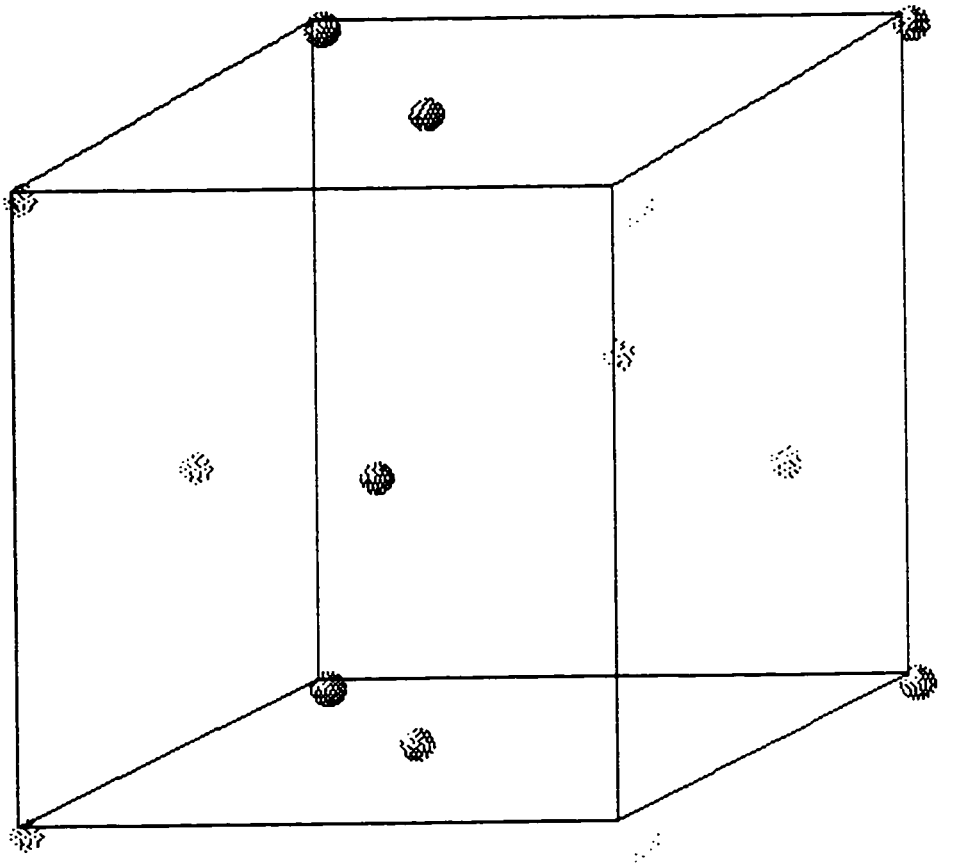
APPENDIX E (contd.) : Graphical User Interface using LabVIEW (Inputs and Outputs) (textual description)

The LabVIEW program's front panel has been displayed. The GUI has many components like switches, arrays, numeric constants and variables, string constants and variables and their like which have been used for input/output. The read switch was used for reading data from the oscilloscope. The GPIB board address and GPIB address of the oscilloscope were given as input. The number of bytes to be read was given as input and the status of the read operation (error message if any) was displayed. The characters read were displayed and were stored in the Numeric Array. The length of this array was computed and stored in Length. The filtered array was stored in Filter Array after filtering of data in the Numeric Array. The Filter Array was inverted and stored in New Array. The New Array is split into two arrays Array3 and Array4 about the maximum scattering intensity value. Array3 was reversed to obtain Array5.

APPENDIX F : Fourier Transform of a sine wave from the Java implementation.



APPENDIX G : Face-centered cubic cell unit from the Java implementation.



BIBLIOGRAPHY

[1] A. W. Agar, R. H. A., and Chescoe, D. *Principles and Practice of Electron Microscope Operation*. North-Holland, Amsterdam, and Elsevier, New York, 1974.

[2] B. E. P. Beeston, R. W. H., and Markham, R. *Electron Diffraction and Optical Diffraction Techniques*. North-Holland, Amsterdam, and Elsevier, New York, 1972.

[3] B. K. W. Baylis, A. Busuttil, N. E. H., and Schlesinger, M. Tin (IV) Chloride Solution as a Sensitizer in Photoselective Metal Deposition. *J. Electrochem. Soc. 123 Pt. 1* (1976), 348.

[4] Brigham, E. O. *The Fast Fourier Transform*. Prentice-Hall: Englewood, New Jersey, 1974.

[5] Brown, P. J., and Forsyth, J. B. *The Crystal Structure of Solids*. Edward Arnold, 1973.

[6] Buerger, M. J. *Crystal-structure analysis*. John Wiley & Sons, Inc., 1960.

[7] Cooley, J. W., and Tuckey, J. W. *Math. Comp. 19* (1965), 297.

[8] Cortijo, R. Structural Studies of Electroless Thin Ni-P Films grown in an Alkaline Environment. Master's thesis, University of Windsor, 1983.

[9] Cortijo, R. O., and Schlesinger, M. *Solid State Comm. 49* (1983), 283.

[10]Cortijo, R. O., and Schlesinger, M. Structural Studies of Electroless Thin Ni-P films grown in an Alkaline Environment. *J. Electrochem. Soc. 130* (1983), 2341.

[11]Cosslett, V. E. *Practical Electron Microscopy*. Academic Press, New York, 1951.

[12]D. C. Joy, A. D. R., and J. I. Goldstein, e. *Principles of Analytical Electron Microscopy*. Plenum, New York, 1986.

[13]Debye, P., and Menke, H. *Physikal. Zeit. 31* (1930), 797.

[14]Dove, D. B. *Physics of Thin films*. Academic Press, Vol. 7, 1, 1973.

[15]Guinier, A. *X-Ray Diffraction*. Freeman : San Francisco, Chapters 2 and 3, 1963.

[16]Hosemann, R., and Bagchi, S. N. *Direct Analysis of Diffraction by Matter.* North Holland Publ. : Amsterdam, 1962.

[17]Kiflawi, I., and Schlesinger, M. Tin (IV)-based photoselective metal deposition of Cobalt and Nickel from Alkaline baths. *Journal Electrochem. Soc. 130* (1983), 872.

[18]Kittel, C. *Introduction to Solid State Physics, 6th edition.* John Wiley and Sons : New York, 1986.

[19]Koehler, J. *Phys. Rev. B 2* (1970), 547.

[20]Koike, J., and Nastasi, M. Microstructural Evolution of Fe grown on a Cu Film and its implication to the Elastic Anomaly in metallic superlattices. *Evolution of Thin-Film and Surface Microstructure — Symposium Proceedings-Materials Research Society 202* (1991), 13.

[21]Lemay, L., and Perkins, C. L. *Java.* Sams, 1996.

[22]M. Schlesinger, J. P. M. *J. Appl. Phys. 40* (1969), 8.

[23]M. Schlesinger, X. M., and Snyder, D. D. Electroless Ni-Zn-P films. *J. Electrochem. Soc. 137 (No. 5–8)* (1990), 1858.

[24]Misell, D. L., and Brown, E. B. *Electron Diffraction : An introduction for Biologists (Selected-area diffraction).* North-Holland, New York, 1987.

[25]Mott, N. F., and Massey, H. S. W. *The theory of Atomic Collisions, 3rd edition.* Oxford University Press : New York, 1965.

[26]Reimer, L. *Transmission Electron Microscopy. Physics of Image Formation and Microanalysis.* Springer-Verlag, Berlin, 1984.

[27]S. L. Chow, N. E. Hedgecock, M. S., and Rezek, J. Electron Microscope study of the nucleation and growth of electroless Cobalt and Nickel. *J. Electrochem. Soc. 119* (1972), 1614.

[28]Savitzky, A., and Golay, M. J. E. *Analytical Chemistry 36*, 8 (1964), 1627.

[29]Schlesinger, M., and Marton, J. P. *J. Phys. Chem. Solids 29* (1967), 188.

[30]Schlesinger, M., and Marton, J. P. The Nucleation, Growth and Structure of Thin Ni-P films. *Journal Electrochem. Soc. 115* (1968), 16.

[31]Simunovich, D. Morphological studies of Quasi Amorphous Systems using Atomic Radial Distribution Functions. Master's thesis, University of Windsor, 1992.

[32]Wagner, C. N. J. *Journal of Non-crystalline Solids 42* (1980), 3.

[33]Warren, B. E. *X-Ray Diffraction*. Addison-Wesley : Reading, Massachussetts, 1969.

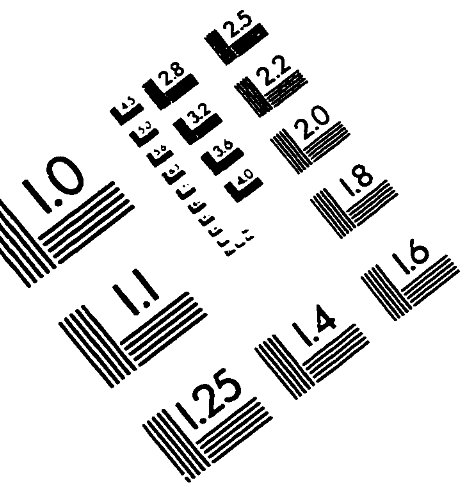
[34]Wells, L. K. *LabVIEW Student Edition*. Prentice Hall, 1994.

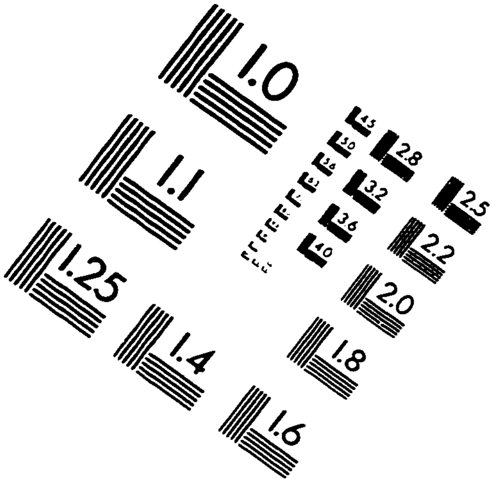
[35]Zernike, F., and Prins, J. A. *Z. Phys 41* (1927), 184.

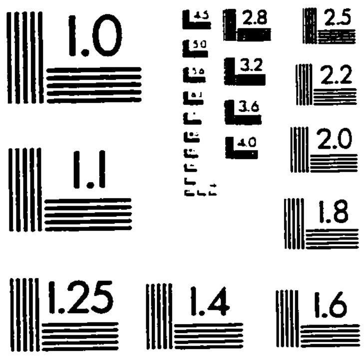
VITA AUCTORIS

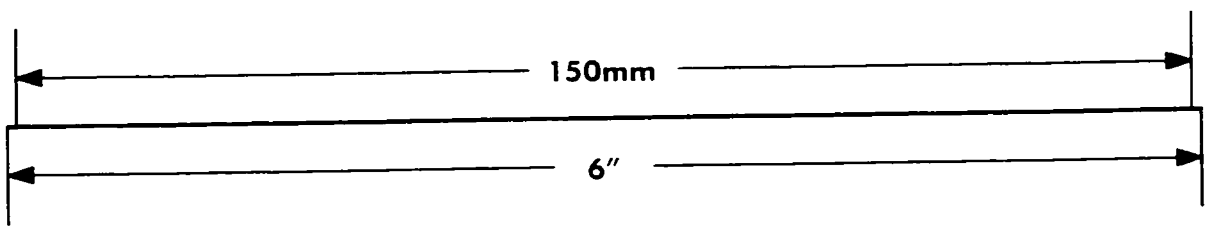
Indu Cidambi was born in 1968 in **Malaysia.** She graduated from **High School** in 1985. From there she went on to the **Post secondary school** where she obtained a B. E. (Bachelor of Engineering) in Computer Science and Engineering in 1992. She is currently a candidate for the Master's degree in Computer Science at the University of Windsor and hopes to graduate in the Fall of 1996.

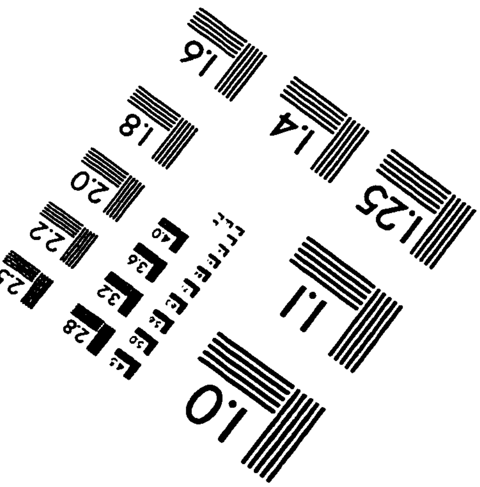
IMAGE EVALUATION TEST TARGET (QA-3)











APPLIED IMAGE . Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989