Array-Based GPR SAR Simulation and Image Reconstruction

Amresh S. Desai

A dissertation submitted to the Department of Electrical Engineering, University of Cape Town, in partial fulfilment of the requirements for the degree of Master of Science in Engineering

Cape Town, April 2002

Declaration

I declare that this dissertation is my own, unaided work. It is being submitted for the degree of Master of Science in Engineering at the University of Cape Town. It has not been submitted before for any degree or examination in any other university.

Signature of Author .....

Cape Town
22 April 2002

Acknowledgements

I wish to acknowledge my supervisor, Dr Andrew J. Wilkinson, for his immense help in clarifying theoretical concepts entailed in the project. I appreciate his efforts in motivating me and guiding me throughout the project. Prof Michael. R. Inggs was also of great assistance, giving valuable input to my project.

Dr Richard Lord was of great help, in providing useful feedback about the project. I also appreciate the contibutions of my family, friends, and the members of the Radar Remote Sensing Group at the University of Cape Town.

Finally I would like to thank the University of Cape Town for awarding me the "International Postgraduate Scholarship" for my studies.

Abstract

Subsurface object detection has mainly been carried out using conventional ground penetrating radar (GPR) techniques, which use a single receiving antenna from which a number of range profiles (known as "A Scope" images) are assembled to form a two-dimensional data field (known as a "B Scope" image). These GPR systems have difficulties with high clutter level, surface reflections, limited ground penetration and the required fine resolution. The resolution in the across track and along track directions is limited by the physical aperture in these directions. This project aims at developing a SAR imaging technique, which uses a single transmitting/receiving antenna to synthesize a two-dimensional planar aperture. Thus a three-dimensional reflectivity image of a scene is generated. The resolution in the across track and along track directions is achieved via a SAR aperture synthesis technique. The depth/range resolution is achieved via the transmission of narrowband *Stepped Frequency Continuous Wave* (SFCW) signals.

Contents

List of Figures

List of Tables

List of Symbols

B	—	RF bandwidth
c	—	Speed of light
f_i	—	Centre frequency
F_s	—	Sampling rate
V	—	Backscattered wavefield
ε_r	—	Effective dielectric constant
δx	—	Along track resolution
δy	—	Across track resolution
δr	—	Range resolution
Δx	—	Along track sampling spacing
Δy	—	Across track sampling spacing
λ	—	Wavelength
ν	—	Speed of propagation in medium
θ	—	Beamwidth
$\sigma(x, y, z)$	—	Reflectivity of subsurface object

Nomenclature

Beamwidth—The angular width of a slice through the mainlobe of the radiation pattern of an antenna in the horizontal, vertical or other plane.

GPR—Ground Penetrating Radar.

PRF—Pulse Repetition Frequency.

SFCW—Stepped Frequency Continuous Wave.

Synthetic Aperture Radar (SAR)—A signal-processing technique for improving the azimuth resolution beyond the beamwidth of the physical antenna actually used in the radar system. This is done by synthesizing the equivalent of a very long sidelooking array antenna.

Chapter 1

Introduction

1.1 Background

Synthetic Aperture Radar (SAR) is an electromagnetic imaging technique that is used for terrain mapping [1]. It can produce high resolution images of the planetary terrains from either airborne or spaceborne platforms and is unaffected by weather conditions such as rain and clouds. One important feature of SAR is its ability to operate during the day and night, as it provides its own illumination [8][10].

SAR is basically a signal-processing technique for improving the azimuth (along track) resolution beyond the beamwidth of the physical antenna actually used in the radar system [7]. This is done by synthesizing the equivalent of a very long sidelooking array antenna.

Airborne and spaceborne SAR imaging is referred to as *far field* radar imaging, because the distance between the antennas and the scene is relatively large. However, this SAR technique is currently also being used for *near field* radar imaging [9]. A common example is in *Ground Penetrating Radar* (GPR) studies, where the antennas and the scenes are close to each other. A synthesized array of antennas is moved across a field and the reflectivity image of the scene is reconstructed with the aim of identifying subsurface objects.

1.2 Subsurface Object Detection

Subsurface object detection has mainly been carried out using conventional GPR techniques, which use a single receiving antenna from which a number of range profiles (known as "A scope" images) are assembled to form a two-dimensional data field (known as "B scope" images). However, these GPR systems have difficulties with high clutter level, surface reflections, limited ground penetration and the required fine resolutions. The resolutions in the across and along track directions are limited by the physical aperture size in these directions. To improve these resolutions, SAR imaging techniques can be used.

There are several SAR processing algorithms. Those suited for near-field imaging are the *Chirp Scaling Algorithm* (CSA), the *Range Migration Algorithm* (RMA), the *Polar Format Algorithm* (PFA) [14], and a purely *time domain processor*. These applicable to the acquisition of backscattered data using planar apertures are the CSA, the RMA, and the time domain processor.

The Chirp Scaling Algorithm works with motion compensation to a line and rectifies more or less the range curvature before compressing the data. The Chirp Scaling Algorithm does not require any sort of interpolation [6][19]. This technique is generally used to focus 2-D airborne and spaceborne SAR data.

The Range Migration Algorithm originates from seismic engineering and geophysics [14]. It is an extension of the $\omega - \kappa$ algorithm that is used to focus 2-D data [3]. The RMA basically works with motion compensation to a line (similar to the CSA), needs a 1-D interpolation (Stolt interpolation [20]), and then fully compensates for the range curvature of the wavefront. However, the Stolt interpolation needs vital computation time and results in the deterioration of image quality, especially when the interpolation kernel is small [19].

The time domain algorithms are generally considerate accurate, but slow [3]. This option is viable for small data set.

This dissertation describes an imaging technique which uses a single antenna to synthesize a two-dimensional planar aperture. Hence a three-dimensional reflectivity image of a scene is generated. This SAR algorithm resembles the Range Migration Algorithm, with the Stolt interpolation replaced by a block processing concept. The resolutions in the across track and the along track directions are achieved via SAR aperture synthesis techniques, while the depth/range resolution is achieved via the transmission of narrowband *Stepped Frequency Continuous Wave* (SFCW) signals [15][26].

1.3 Project Overview

This section provides a general outlook of the project. This project can be split into two main parts: a simulator to generate 3-D backscattered raw data and a focusing algorithm.

Figure 1.1 shows the geometry of the system, where an antenna positioned at (x_a, y_a, z_a) is synthesized to produce a 2-D planar aperture. A single antenna, that transmits narrowband Stepped Frequency Continous Wave (SFCW) signals and that has a bandwidth of 1600 MHz, is used to conform to the specifications of the radar, that was developed by the University of Cape Town for subsurface object detection.

In Figure 1.1, the x, y, and z directions represent the along track, the across track and the range/depth directions respectively.

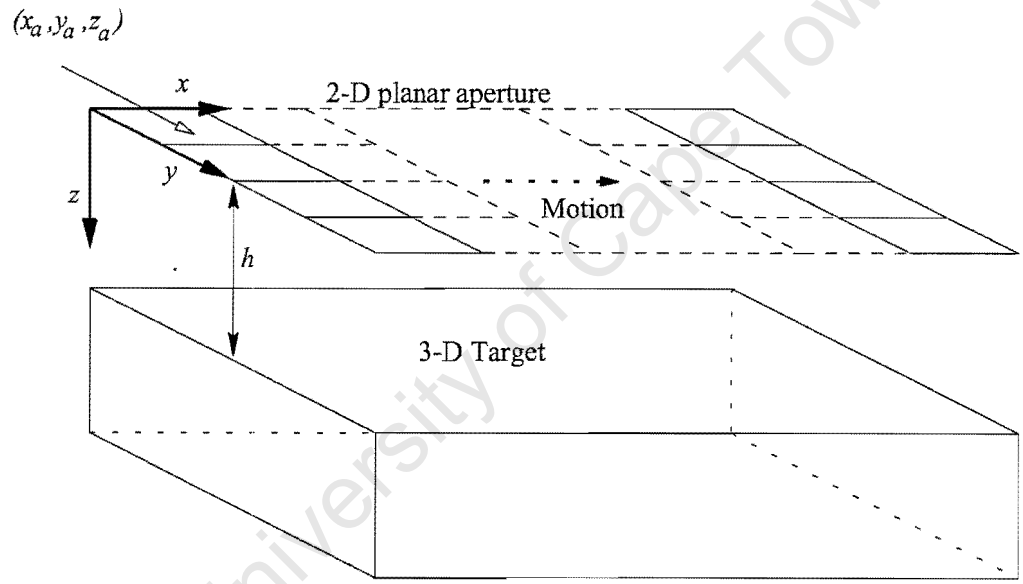

Figure 1.1: Diagram representing the imaging geometry.

The antenna is moved in a stepwise manner in the y direction and a synthesized array of antennas is generated. Thus the across track resolution is determined by the SAR aperture synthesis technique. A physical array of antennas in the across track direction would have been preferred, because the single antenna has to be moved in both the across track and the along track directions. In practice, this is time consuming and the system is more likely to have phase errors.

The theory of using a physical array of antennas is slightly different from that of a single antenna. The transmitter antenna has to be defined as well as the receiver antennas. The resolution in the y direction for a physical array of antennas is determined via the across

track directed array. The theory involved in using a physical array is not discussed in this project.

The synthesized array of antennas is then moved in a stepwise manner in the x direction and hence a 2-D planar aperture is generated as shown in Figure 1.1. The along track resolution is also achieved via the SAR aperture synthesis technique.

The problems that arise due to the multi-layers of the ground are not tackled in this project. A homogeneous medium is assumed and the effective dielectric constant, ε_r, is chosen to be between 4 (dry sand) and 16 (damp soil) [23]. This model can be extended to cater for the heterogeneous nature (different permittivities) of the ground.

1.4 Project Objectives

The objective of this project is to implement a SAR imaging algorithm capable of resolving subsurface objects. In order to implement and verify the algorithm, a SAR simulator has to be developed. The simulator has two main functions:

1. To compute point target responses for the correlation filters, which are part of the focusing algorithm;
2. To generate test data for the focusing algorithm.

A correlation-based focusing algorithm is set up to process the backscattered data from the simulator in order to visualize the targets.

More specifically, this project requires the implementation of a simulator and a SAR focusing algorithm to:

- Simulate numerical data from a 3-D scene using various antenna parameters.
- Focus the data in range, along track and across track directions with appropriate resolutions.
- Display the processed data and hence the positions of the subsurface objects.

The scope of the project does not extend to the experimentation and testing with real data.

1.5 Outline of Thesis

This thesis has the following structure:

- **Chapter 2** describes the theory of Near Field Synthetic Aperture Radar. The signal processing techniques are elaborated in a stepwise manner and the mathematical concepts leading to the processed data are explained. Important factors such as sampling criteria and resolutions in range, along track and across track directions are discussed.

- **Chapter 3** presents the model of the simulator used for generating the test data and for computing the point target responses required by the correlation filters. The program structure of the simulator is elaborated, together with various aspects like the setup parameters, the scene model, the simulated data and the program features. The program is then introduced as a "black box" with input files, output files and file formats.

- **Chapter 4** demonstrates the focusing algorithm and discusses the correlation method used to focus the 3-D data. The Hamming windows and the zero padding techniques, applied to reduce the sidelobes and to increase the number of samples respectively, are then explained. Furthermore, the block processing concept used to improve focusing is described. Various methods to refine the focusing algorithm are briefly examined.

- **Chapter 5** analyses the results of the SAR algorithm for the numerical data. A table of values showing object locations is given, and this table is enhanced with graphs and 2-D visual displays. Finally, the accuracy of the algorithm is discussed.

- **Chapter 6** covers conclusions and recommendations for future work, including possible improvements to the simulator and to the focusing algorithm in order to accommodate subsurface multi-layer models.

Chapter 2

Theory of Near Field Synthetic Aperture Radar

2.1 Introduction

Synthetic Aperture Radar (SAR) is an advanced radar imaging technique that can generate high resolution images [4][13]. Various imaging algorithms like the *Polar Format Algorithm*, the *Range Migration Algorithm* and the *Chirp Scaling Algorithm* use SAR techniques to reconstruct the reflectivity image of a scene [14].

SAR theory is very similar for both near and far field situations. However, one of the major differences is the aperture consideration, which is discussed in Section 2.2. An important characteristic of a SAR image is its resolution. This is the minimum distance at which two closely spaced scatterers of equal strength may be resolved. The resolutions in all three directions for a 3-D data set are dependent on various parameters and these are discussed in Section 2.3. Section 2.4 describes the sampling issues regarding the stepped-frequency waveforms and the antennas.

Finally, a forward and a reverse model to generate and to focus scene data respectively, are discussed in Section 2.5. The forward model (simulator) generates backscattered data from a scene, and the reverse model (SAR focusing algorithm) processes the backscattered wavefield to reconstruct the scene. The focusing algorithm uses signal processing techniques that are similar to those of the Range Migration Algorithm.

2.2 Aperture Considerations for Near and Far Field SAR

The region around any antenna can be split into three sections: reactive near field, radiating near field and far field [2][5]. At the section boundaries, there are no abrupt changes regarding the field composition. However among the different sectors, there are distinct differences in the field composition. Figure 2.1 illustrates the field regions around an antenna where R_1 and R_2 represent the boundaries. Even though all the three regions are discussed, only the distance of demarcation for the radiating near field and far field regions, R_2, is of concern for this project. This is because all the assumptions made are based on either near or far fields.

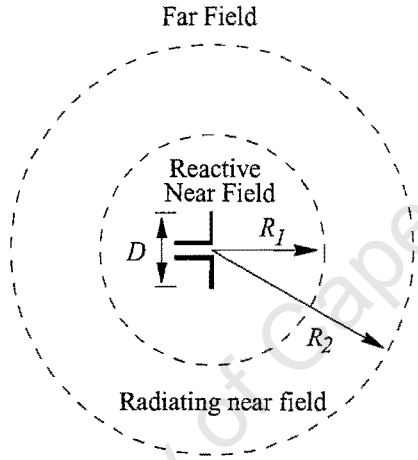

Figure 2.1: Field regions of an antenna.

- The reactive near field region is the section around the antenna where the reactive field predominates. For most antennas, this region is represented by

$$R < 0.62\sqrt{\frac{D^3}{\lambda}} \tag{2.1}$$

where R is the distance betweeen the antenna surface and the boundary, D is the dimension of the antenna, and λ is the wavelength [2].

- The radiating near field is the region between the outer boundary of the reactive near field and the inner boundary of the far field. Here, radiation fields prevail and the angular field distribution is dependant on the distance from the antenna [2].

$$\frac{2D^2}{\lambda} > R \geq 0.62\sqrt{\frac{D^3}{\lambda}} \tag{2.2}$$

- The far field region is where

$$R \geq \frac{2D^2}{\lambda} \tag{2.3}$$

In the far field region, the angular field distribution is independent of the distance from the antenna [2]. Note that D must be large compared to the wavelength for Equations 2.1, 2.2, and 2.3 to be valid.

2.3 Resolution

The radar resolution is defined as the "width" of the point target response, measured between the 3dB points or the half power points. The resolutions in the across track, along track, and range (depth) directions are discussed below.

2.3.1 Range resolution

The range resolution, δr, is achieved via the transmission of narrowband *Step Frequency Continuous Wave* (SFCW) signals. For a radar system, transmitting signals of bandwidth B, and having a speed of propagation, v, in the scene medium, the range resolution is given by [12][22]

$$\delta r = \frac{v}{2B} \tag{2.4}$$

Thus for a bandwidth of 1600 MHz (similar to the bandwidth of the radar available for taking measurements) and a speed of propagation of $7.5 \cdot 10^7$ m/sec (effective dielectric constant of medium is 16), the range resolution is 2.34 cm. The Hamming windows, however, deteriorate this value by about 0.25 cm. By selecting the bandwidth of the SFCW signals, one can specify the range resolution as required [11]. Note that the range resolution is independent of any pulse shape or pulse duration.

2.3.2 Along track resolution

The resolution in the along track direction, δx, is achieved via the SAR aperture synthesis technique. The formulas relating the resolution to various antenna parameters are derived below and the assumptions made are clearly stated. In order to have a better understanding

of the variables used in the equations, Figure 2.2 is given, which illustrates an antenna with a beam angle θ.

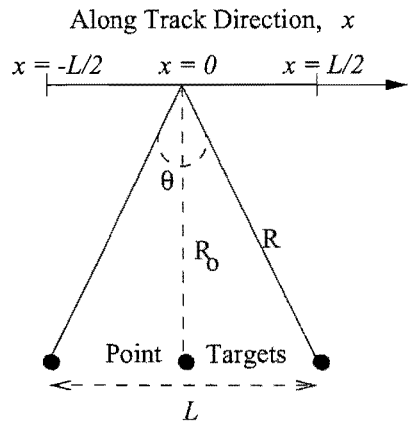

Figure 2.2: Diagram representing an antenna moving in the along track direction.

R_o represents the shortest distance (a constant) between the antenna and the point target, whereas R, the slant range, varies with distance x. L is the maximum distance that the beam pattern can cover. The bandwidth is B_x and the phase of the signal is ψ.

Thus for any field pattern and for any beam angle, the along track resolution without any windowing function is [26]

$$\delta x_{3dB} \approx \frac{0.89}{B_x} \tag{2.5}$$

Where

$$B_x = \frac{1}{2\pi}\left|\frac{d\psi}{dx}\right| \cdot 2 \tag{2.6}$$

For the case of Synthetic Aperture Radar [24][26],

$$\frac{d\psi}{dx} = \frac{d}{dx}\left[\frac{-4\pi R(x)}{\lambda}\right] = \frac{-4\pi}{\lambda}\frac{dR}{dx} \tag{2.7}$$

$$R(x) = \sqrt{R_o^2 + x^2} \approx R_o + \frac{x^2}{2R_o} \tag{2.8}$$

$$\frac{dR}{dx} = \frac{d}{dx}\left[\sqrt{R_o^2 + x^2}\right] = \frac{x}{R(x)} = \sin\left(\frac{\theta}{2}\right) \tag{2.9}$$

Therefore,

$$\frac{d\psi}{dx} = \frac{-4\pi \sin\left(\frac{\theta}{2}\right)}{\lambda} \tag{2.10}$$

$$B_x = \frac{4 \sin\left(\frac{\theta}{2}\right)}{\lambda} \tag{2.11}$$

$$\delta x_{3dB} \approx \frac{0.89\lambda}{4 \sin\left(\frac{\theta}{2}\right)} \tag{2.12}$$

The maximum beamwidth, θ_{max}, depends on the beam pattern. As illustrated in Figure 2.3, the beam pattern for the near field (dotted) is much broader compared to that for the far field in the $R \leq \frac{2D_x^2}{\lambda}$ region, and so a conical beam cannot be assumed for that region. Note that D_x represents the physical dimension of the antenna in the along track direction.

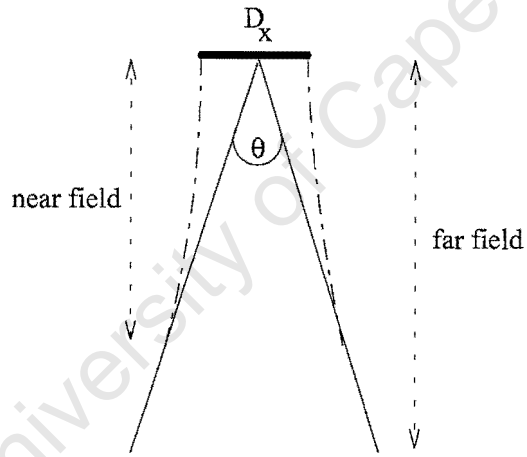

Figure 2.3: Beam pattern for near and far field.

The equations that follow are based on the assumptions that the targets are in the far field and that a conical beam is used.

For one-way propagation,

$$\sin\theta \approx \frac{\lambda}{D_x} \tag{2.13}$$

For small angle θ,

$$\sin\theta \approx \theta \approx \frac{\lambda}{D_x} \tag{2.14}$$

Therefore,

$$\delta x_{3dB} \approx \frac{0.89 D_x}{2} \tag{2.15}$$

From Equation 2.15, it is clear that the along track resolution is directly related to D_x, and so to better this resolution (decrease its value), the dimension of the antenna in the along track direction has to decrease.

Some calculations are given below to illustrate that the equations obtained in the case of far field and of small angles are reasonable for an estimate for resolutions.

Using an effective dielectric constant of 16 for the medium of propagation, an antenna bandwidth of 1600 MHz, a simulation frequency of 1600 MHz, and an antenna dimension of 5 cm, the far field region starts at

$$R = \frac{2D^2}{\lambda} = 10.7 \text{ cm} \tag{2.16}$$

Where

$$\lambda = \frac{\lambda_{freespace}}{\sqrt{\varepsilon_r}} \tag{2.17}$$

$\lambda_{freespace}$ represents the wavelength for the simulation frequency at the speed of light and λ represents the wavelength for the simulation frequency at the speed of wave propagation in the medium. For a simulation frequency of 200 MHz, the far field region starts at 1.33 cm.

Using the same parameters as above and using Equation 2.13 for any beam angle, θ becomes 1.215 radians, which is equivalent to 69.6 degrees. The along track bandwidth, from Equation 2.11, is then 48.7 cycles per meter, and this leads to a resolution of 1.83 cm from Equation 2.12. The resolution, calculated using Equations 2.14 and 2.15 for small angles, is 2.20 cm. Thus Equations 2.14 and 2.15 can be used as a crude estimate for the along track resolution.

Note that the approximations, for θ and hence the aperture length, break down as (a) the beamwidth broadens (approximation in Equation 2.14) and (b) for targets closer than the near/far field transition (Equation 2.16).

2.3.3 Across track resolution

The resolution in the across track direction, δy, is achieved via the SAR aperture synthesis technique. The resolution is given by

$$\delta y_{3dB} \approx \frac{0.89 D_y}{2} \tag{2.18}$$

where D_y is the antenna size in the y direction. The derivations and the assumptions at various stages are similar to those of the along track resolution.

2.4 Sampling issues

This section discusses factors like the sampling rate, F_s, of the signals, the along track antenna sampling spacing, Δx, and the across track antenna sampling spacing, Δy. The choice of these variables is critical to avoid aliasing. Aliasing causes the shifted replicas of the signals to overlap in the frequency domain and thus changes the shape of the signals in the time domain.

2.4.1 Sampling rate

To avoid aliasing, the sampling theorem needs to be satisfied. The sampling theorem states that a bandlimited continuous time domain signal, having a bandwidth B, can be recovered from its samples provided that the sampling rate, $F_s \geq 2B$ [18]. This formula applies for real signals. For analytic signals, the formula becomes $F_s \geq B$.

2.4.2 Along track antenna spacing

From Nyquist theorem, the antenna spacing in the along track direction, Δx, should be less than $\frac{1}{B_x}$ or $\frac{D_x}{2}$ to avoid aliasing [26]. This sample spacing is required even if only a portion of the maximum aperture is processed.

2.4.3 Across track antenna spacing

From Nyquist theorem, the antenna spacing in the across track direction, Δy, should be less than $\frac{1}{B_y}$ or $\frac{D_y}{2}$ to avoid aliasing [26].

2.5 Conceptual analysis of simulator and SAR focusing algorithm

The simulator consists of a synthesized array of antennas, positioned at a height, h, above the test field as shown in Chapter1, Figure 1.1. This synthesized array is moved in a stepwise manner across the field (x direction) and the frequency domain backscattered data is collected at each antenna position for a range of frequencies.

The SAR focusing algorithm involves the signal processing techniques required to generate the 3-D reflectivity images of scenes efficiently from their 3-D frequency domain backscattered data obtained from the simulator. These signal processing techniques include:

- A correlation-based focusing algorithm – Matched filters are generated as a function of range and these are used to focus the data.
- A block processing technique – The data is divided into blocks before processing. Each block has its distinct matched filter and hence targets are focussed more effectively.
- An amplitude compensation algorithm – The term $\frac{1}{R^2}$ in Equation 2.22 causes a point target at different ranges to have different intensities. This algorithm compensates for the decaying term.

Note that since the same simulator used to generate the signals is used as part of the image reconstruction, there is a slight possibility of cancellation of errors. However this possibility is not analyzed in this project. The mathematical modeling of the algorithm is described in the following section.

2.5.1 Description of simulator

A stepped frequency radar, transmitting continuous wave (SFCW) signals with center frequency, f_i, is used to illuminate the scene. The radar transmits coherent, pulse-to-pulse, frequency-stepped waveforms in a continuous series of bursts, with n pulses per burst. Thus a set of n echo signals are generated which are the frequency domain measurements of the reflectivity data from each burst [25].

The imaging geometry is shown in Chapter 1, Figure 1.1. The coordinate of the antenna is (x_a, y_a, z_a), where x represents the along track direction, y represents the across track direction, and z represents the depth/range direction. The antenna has a synthesized frequency bandwidth of B_t.

The antenna is moved in the y direction in a stepwise manner. A synthesized 1-D array of antennas is generated, which in turn is moved along the x direction. Thus a rectangular (2-D) planar aperture at a certain distance h above the scene is synthesized, and 3-D frequency domain backscattered data is collected.

The backscattered wavefield, V, is obtained from the equations below:

$$v = \frac{c}{\sqrt{\varepsilon_r}} \tag{2.19}$$

$$\lambda = \frac{v}{f_i} \tag{2.20}$$

$$R = \sqrt{(x_a - x)^2 + (y_a - y)^2 + (z_a - (z + h))^2} \tag{2.21}$$

$$V = \frac{\sigma(x, y, z)\exp(\frac{-j4\pi R}{\lambda})}{R^2} \tag{2.22}$$

Where

- v is the velocity of propagation in the scene medium;
- c is the speed of light;
- ε_r is the effective dielectric constant of the propagation medium;
- λ is the wavelength for each center frequency, f_i;

- (x, y, z) is the coordinate of each pixel in the scene;
- R is the distance between an antenna and a subsurface object;
- $\sigma(x, y, z)$ is the reflectivity of the subsurface object.

The magnitude of the frequency domain backscattered data is

$$\frac{\sigma(x, y, z)}{R^2} \tag{2.23}$$

and its phase is the exponential term in Equation 2.22. In order to keep the model simple, propagation losses are not considered. The SAR algorithm used to focus the backscattered data is discussed below.

2.5.2 SAR focusing algorithm

The implemented SAR focusing algorithm is a modified version of the Range Migration Algorithm, as discussed in Chapter 1, Section 1.2. The algorithm consists of several signal processing techniques that focus the backscattered wavefield to obtain the reflectivity image of the scene. The processing steps involved are illustrated in Figure 2.4 and these steps are explained in detail in the following section. The concepts of the correlation filters used are then elaborated.

Signal Processing Steps

Three-dimensional raw data is collected using a SFCW radar. The radar uses the frequency domain signatures of the targets rather than the time domain signatures, and thus by applying an inverse fast fourier transform (IFFT) in the range direction, the frequency domain backscattered data is converted to the time domain.

This data is extended in the along track direction by adding zeros on either side of the data. This ensures that the sidelobes of targets at the edge of the data block do not wrap around and appear as point targets when FFTs and IFFTs are applied to the data set. This extra portion of data is discarded once processing is completed.

The extended data is divided (in range) into blocks of certain size, specified by the user. The data closer to the antenna aperture is normally split into smaller blocks. This is because further away from the aperture, SAR focusing becomes more efficient (as the

assumptions for far field become valid) and so the point targets can be detected even in larger blocks. Even though block processing is computationally intensive, as explained in Chapter 4, it assures that all point targets are detected as each block has its distinct matched filter.

Each block of data is transformed to the frequency domain using 3-D FFTs. A 3-D matched filter, that is generated locally for the same range as the block of data, is then applied to the data in the frequency domain. The matched filter compresses the data in all three directions and hence focuses the point targets. Hamming windows are further applied to reduce the sidelobes at the expense of resolution. The data is then converted to the time domain.

The extra data, that resulted from adding zeros in the along track direction, is cancelled and an amplitude compensation algorithm is applied to the processed data.This ensures that targets with different intensities are distinguished.

All the blocks of processed data are joined together, so that the entire "scene reflectivity" data is now available. The 3-D zero padding function is applied to this data set in the frequency domain. This particular function increases (interpolates) the number of time domain samples by adding zeros to the frequency domain samples. Thus a good display of the signals is obtained. This function does not give any additional information regarding the signals.

The zero padded data in the frequency domain is transformed to the time domain using 3-D IFFTs and the scene reflectivity is visualized using 1-D and 2-D display tools.

Matched Filter and Hamming window

The effects of matched filtering and Hamming windows are described in this section [17][21]. For a received time domain signal (point target at range $r_o = \frac{ct_o}{2}$) represented by Equation 2.24, the output after applying a matched filter is given by Equations 2.25 and 2.26 in the frequency and the time domain respectively [26]. In Equation 2.24, $\delta(t)$ is the impulse response of a point target, t_o is the time taken for the signal to travel to the target and back, $\xi(t)$ is the target reflectivity profile, and ξ_o is a constant [26].

$$\xi(t) = \xi_o \delta(t - t_o) \tag{2.24}$$

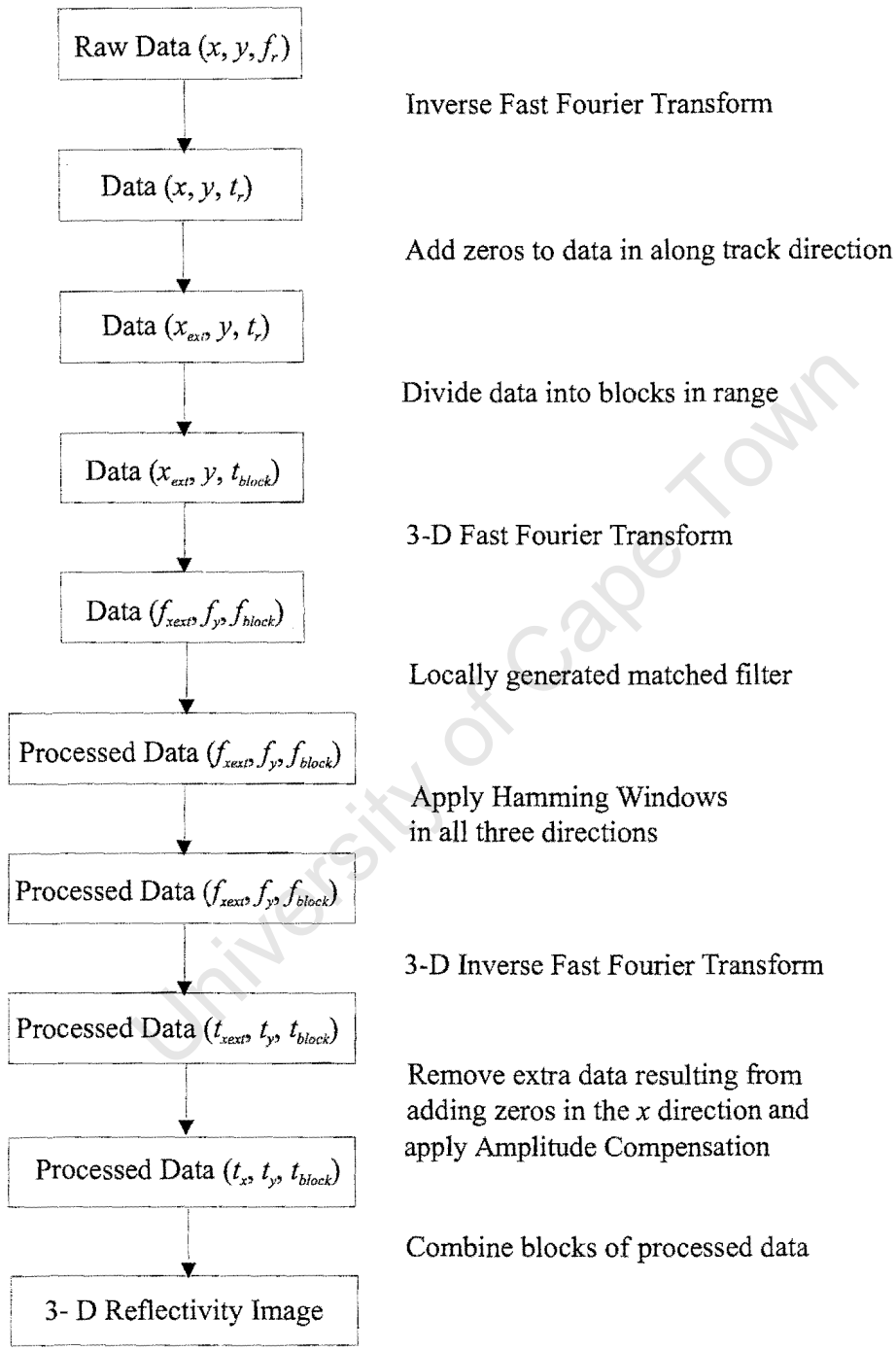

Figure 2.4: Chart showing signal processing steps.

$$V(f) = \xi(f + f_o) \cdot rect\left(\frac{f}{B}\right) \tag{2.25}$$

$$V(t) = \left[\xi(t) \cdot e^{-j2\pi f_o t}\right] \otimes Sa(\pi Bt) = \xi_o BSa(\pi B\left[t - t_o\right]) \cdot e^{-j2\pi f_o t_o} \tag{2.26}$$

The matched filter is tailored to the signal waveform to be filtered, in order to achieve maximum signal to noise ratio. However, high sidelobes are produced due to the Sa function, as shown in Equation 2.26. The output signal is displayed in the frequency and the time domain in Figure 2.5 (a) and Figure 2.5 (b) respectively. The 3dB resolution, δt_{3dB}, is also shown in the figure.

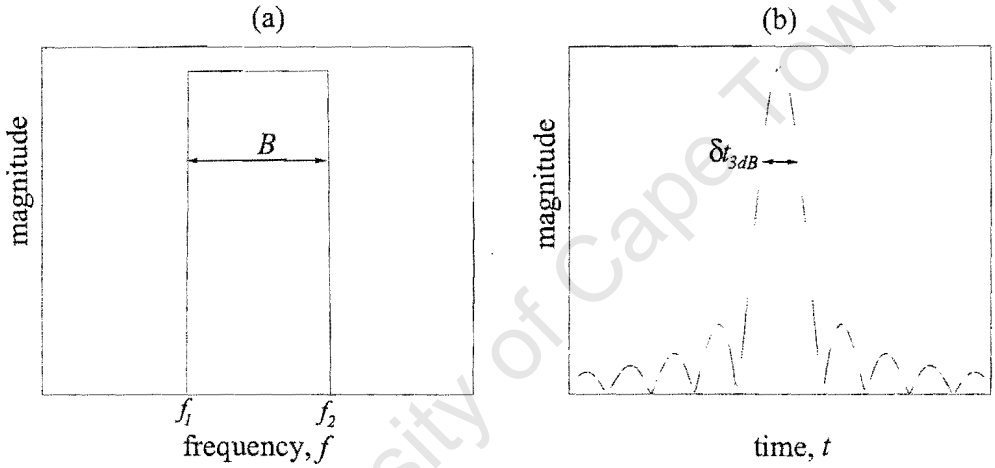

Figure 2.5: Diagrams representing signals after matched filtering, (a) frequency domain and (b) time domain.

By applying Hamming windows to the filtered signal in the frequency domain (from f_1 to f_2), the sidelobes are suppressed at the expense of resolution in the time domain. Hamming windows reduce the peak edges of the signal in the frequency domain, which in turn result in much lower sidelobes in the time domain. This is illustrated in Figure 2.6, where (a) and (b) show the frequency and time domain signals after windowing. Note that the resolution, δt_{3dB}, in Figure 2.6 (b) has deteriorated compared to that in Figure 2.5 (b).

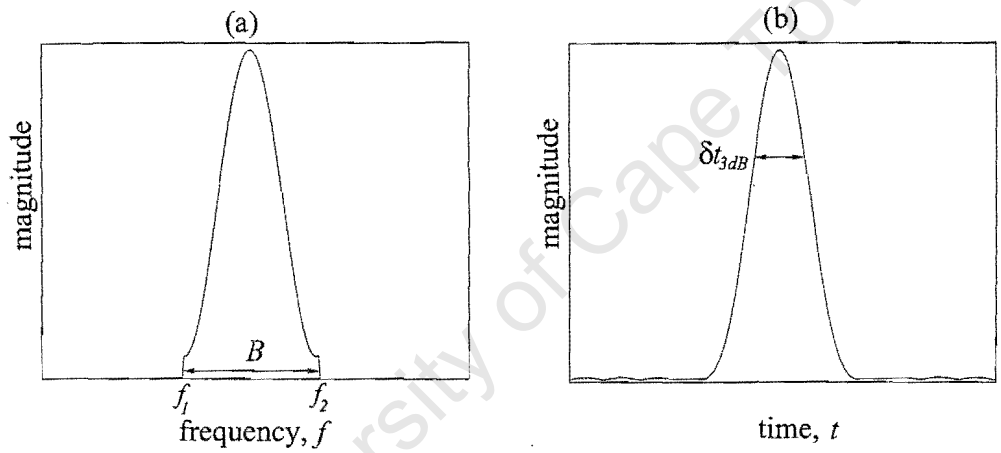

Figure 2.6: Diagrams representing signals after applying a Hamming window, (a) frequency domain and (b) time domain.

Chapter 3

Simulator Software Architecture for GPR Applications

3.1 Introduction

The theoretical concepts of a simulator to generate 3-D frequency domain backscattered data have been discussed in Chapter 2. In this chapter, the practical implementation of the simulator is explained, together with its software architecture.

The simulator model is discussed in Section 3.2. The various modules of the simulator are presented and the functions of each module are elaborated to provide an overview of the system. The program architecture is introduced in Section 3.3 and the various files and data formats are detailed. The input parameters that have to be specified by the user are clearly illustrated, and the scene model used is mentioned. The file generating the backscattered data is also discussed, and the display unit available to visualise the 3-D data is briefed.

Sections 3.4 and 3.5 summarise the input files, the operation, and the output files of the simulator.

3.2 Simulator Model

The simulator model is divided into three modules as illustrated in Figure 3.1. The functions of each module are explained below in detail, together with the code routines involved.

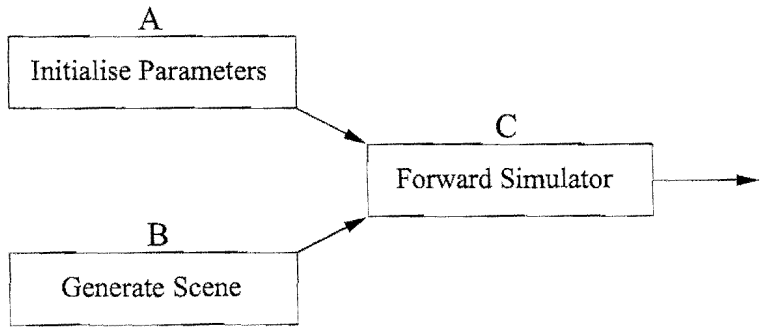

Figure 3.1: Modules showing the overall structure of the simulator.

- Module A is the setup section. All the parameters related to the antennas, the sampling spacings and the block data size are initialised here. Several options are also available regarding the type of data (frequency domain data, time domain data or their basebanded versions) desired out of the "Forward Simulator". This module is critical and has to be modified by the user according to his/her requirements before running the simulator.

- Module B incorporates the generation of the target scene. The scene dimensions and the individual pixel sizes are specified. Point targets are manually set within the scene and the scene data is stored as an ASCII file, which is then loaded into module C.

- Module C is the forward model simulator. The target scene is loaded and the frequency domain backscattered data from various antenna positions are collected as explained in Chapter 2. This raw simulated data is stored as an ASCII file, which is loaded by the focusing algorithm. Note that the raw data is also converted to the type of data the user specified in Module A and this partially processed data can be viewed using the display tools.

In a real application, the antenna parameters, the scene sizes and the medium parameters are recorded at the test field and loaded into the setup module. The wavefield data collected for the 2-D antenna aperture is slightly different from the numerical data obtained using the forward model simulator. This is because the simulator assumes:

1. A homogeneous medium from the antennas to the scatterers;
2. The wave propagation energy only decays with distance from the antenna;
3. The antenna moves in a straight line in the across and along track directions.

3.3 Program Architecture

The software architecture is discussed below, together with the functions of each file shown in Figure 3.2. This figure shows the file structure used for the simulator. Due to its ability to handle numerical simulations and its ease of implementation, "Matlab" is used as the programming language for this project.

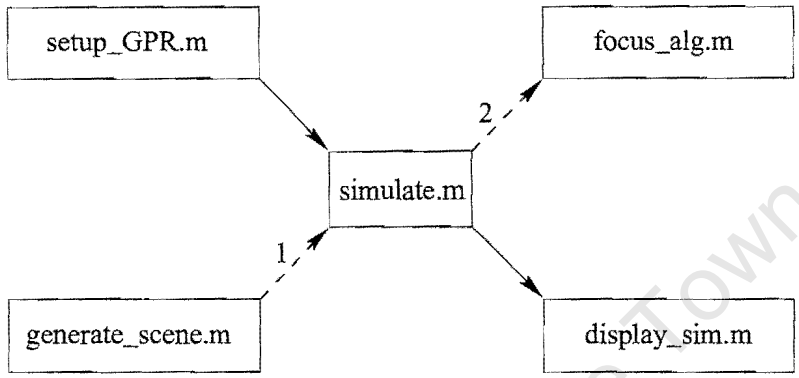

Figure 3.2: Diagram representing the file structure for the project.

The ".m" extension represents an executable Matlab file, and the dotted arrows illustrate that ASCII files are generated and stored in memory. The arrows labelled "1" and "2" indicate where the scene data is stored as a "scene.dat" file and where the simulated data is stored as a "store.dat" file respectively. This enables the user to run the simulator without having to generate the scene data. It also enables the user to apply the focusing algorithm without having to simulate the data again.

3.3.1 Setup parameters

There are various parameters that the program requires in order to run. These parameters have to be stipulated by the user, based on either a real-time or a simulated system, in the "set_GPR.m" file. A sample of the parameters is given in Appendix A. These parameters are categorised and discussed below:

Antenna

The system uses a synthesized array of antenna elements as shown in Figure 3.3. The dots on the figure represent antenna elements.

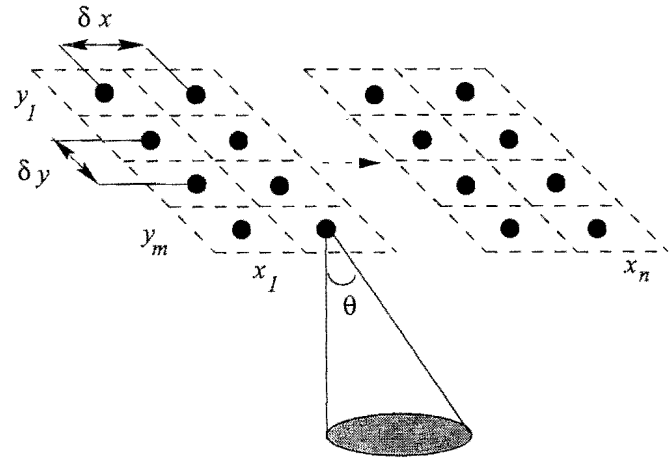

Figure 3.3: Diagram illustrating antenna spacing and beamwidth.

The starting antenna position (x_1, y_1) has to be specified, together with the number of antenna elements in the across and the along track directions. The antenna spacings, both in the across track, δy, and in the along track, δx, have to be stated. The antenna spacings in both directions are critical and have to satisfy the conditions in Chapter 2, Section 2.3.4 to avoid aliasing. Furthermore, the beamwidth, θ, of the antenna has to be set.

Type of data out of the "Forward Simulator"

The "setup_GPR.m" file allows the user to choose the type of data he/she wants out of the "simulate.m" file. The options available are :

- Frequency domain data – This is the backscattered data collected from the 2-D planar aperture.
- Time domain data – An Inverse Fast Fourier Transform (IFFT) is applied to transform the backscattered data to the time domain.
- Basebanded frequency domain data – The backscattered data is basebanded, which means that the signals are centered about the "zero" frequency. Baseband data is preferred for signal processing because the point target response is in a very convenient form: the magnitude response contains a single dominant lobe, and the phase across the main lobe is $\frac{-4\pi R}{\lambda}$.
- Basebanded time domain data – An IFFT is applied to the basebanded frequency domain data to convert the data to the time domain.

Hamming windows and zero padding

For all four choices listed above, the user can switch a Hamming window function and a zero padding function on or off. The Hamming windows reduce the sidelobes at the expense of resolution, whereas zero padding increases the sample rate in the spacial domain for better signal display. These options are made available to the user so that one can view an improved version of the signals before any processing is done.

Frequency Step

The frequency range over which the system operates is specified by the variables "Min_freq" and "Max_freq". The frequency step, $Freq\_step$, is calculated as

$$Freq\_step = \frac{v}{2Range\_max} \tag{3.1}$$

where $Range\_max$ is the maximum unambiguous range, and v is the speed of wave propagation in the medium.

Speed of propagation and the maximum unambiguous range

The medium between the antenna and the targets is assumed to be homogeneous with an effective soil dielectric constant, ε_r. The user should set this value depending on the type of soil being analysed. Note that the soil permittivity limits the wave propagation and hence the maximum penetration depth in the medium. The speed of electromagnetic wave propagation, v, is $\frac{c}{\sqrt{\varepsilon_r}} = 7.5 \cdot 10^7$ m/sec for $\varepsilon_r = 16$. The maximum unambiguous range is also defined by the user, and this in turn determines the frequency step as discussed above.

Block sizes

The block processing technique used in the SAR focusing algorithm is explained in detail in Chapter 4. The data is divided in the range direction and hence the block sizes in this direction have to be defined. This is done by the user in the "setup_GPR.m" file. The block sizes are specified in terms of the number of pixels or bins. For example, if a depth/range of 2 meters has 101 range bins, then 0.1 meter of data in the range direction will be represented by $\frac{(101-1)0.1}{2} = 5$ range bins. Note that there are two arrays labelled

"block_vect_startpt" and "block_vect_endpt" to enumerate the start and the end of the blocks respectively.

The block sizes were found empirically, by incrementally increasing the block sizes such that the resulting focussed images exhibited negligible defocussing towards the edges of the blocks. The matched filter is a range dependent function. For each block, a matched filter is computed from a point at the centre of the block. The focus degrades towards the edges of the block as the block size is increased. The maximum block size, which results in insignificant degradation in focus, is referred to as the 'depth of focus'. Although it is possible to derive crude equations for specifying the maximum block size via analysis of the phase and magnitude mismatch, in this thesis these issues were not studied in detail.

3.3.2 Scene model

The target scene is simulated numerically and it can be viewed as a rectangular box filled with pixels. The dimensions of the scene are defined, together with the pixel size. A point target within the scene is generated by setting the pixel value at the desired point to a much higher value compared to surrounding pixels. Thus one can activate as many pixels as required within the scene as shown in Figure 3.4. A sample of the code is given in Appendix B.

The scene is presented as a data cube (and not as a list of point targets) as we wish to be able to specify entire 3-D scenes consisting of many point targets.

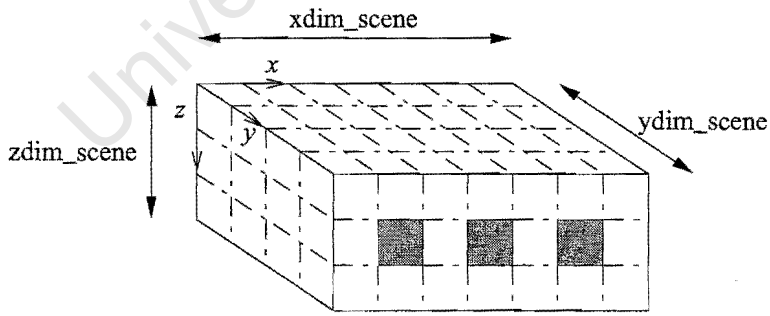

Figure 3.4: Diagram representing a scene.

Before storing the scene data in an ASCII format, the 3-D data is converted into a 2-D format, because Matlab cannot store 3-D data directly into a binary or ASCII format. The scene file is always called "scene.dat".

3.3.3 Simulated data

The backscattered data is generated by the "simulate.m" file in Figure 3.2. The scene data is loaded from the "scene.dat" file and is converted to a 3-D Matlab variable for use by this part of the program. This forward simulator has two basic purposes, namely:

1. To generate test data which is processed in the focusing algorithm;
2. To compute the point target response for the correlation filters, which in turn are used for processing data.

For each antenna position in the 2-D planar aperture, stepped frequency continuous waves are emitted. At the starting frequency, the wavefield data is calculated for each pixel in the scene using Equations 2.19 to 2.22 and summed up coherently. The frequency is then incremented by the "$Freq\_step$" variable and the process is repeated until the maximum frequency is reached. Thus a set of 3-D data is obtained in the frequency domain. The 3-D backscattered raw data is converted to a 2-D data set, which is stored in the form of an ASCII file named "store.dat". This file is loaded into the focusing algorithm.

The 3-D backscattered data is also processed in the simulator to generate the type of data as specified in the "setup_GPR.m" file. This partially processed data, called "partial_data", can be viewed in the display tool discussed below.

3.3.4 Data display

The display unit ("display_sim.m" file) allows the user to view the 3-D partially processed data, either by slicing it to get a two-dimensional data set, or by exhibiting the signals as a one-dimensional array along any orthogonal direction.

Figure 3.5 illustrates the three directions along which the data can be sliced to produce 2-D images. The user has to manually specify the values for $y_{display}$, $x_{display}$, $z_{display}$ to display the 2-D data as shown in A, B, C in Figure 3.5.

To visualize the signals as a 1-D array, the user has to stipulate the positions of any two variables in a 3-D data set. The blocks labelled A, B and C in Figure 3.6 illustrate the three possible directions along which the 1-D data can be plotted.

In A, the y and z coordinates are fixed and the x coordinate is the variable, whereas in B, the x and y coordinates are fixed and the z coordinate is the variable. In C, the x and z coordinates are fixed and the y coordinate is the variable.

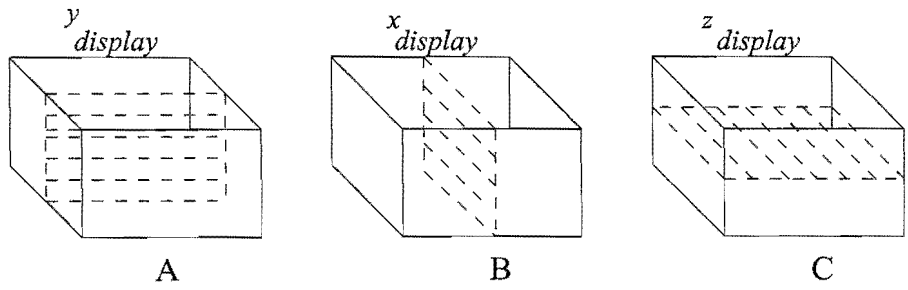

Figure 3.5: Diagram showing slices drawn across a 3-D data set.

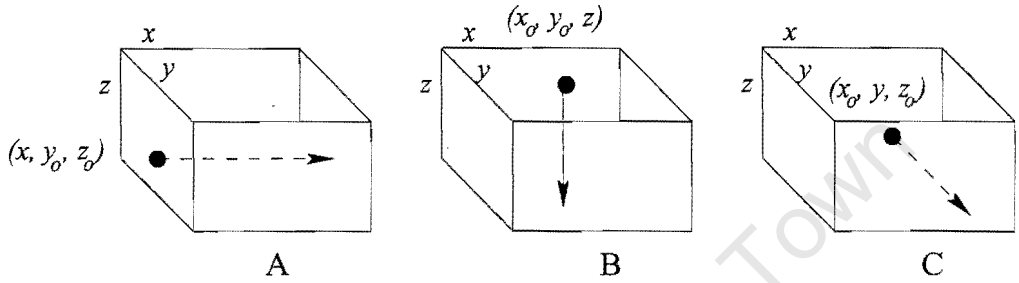

Figure 3.6: Diagram showing how a 1-D array is selected.

Both the 1-D data and the 2-D data display tools are used in Chapter 5 to illustrate the results from the simulator and the focusing algorithm.

3.3.5 Program features

This section discusses the various features that are implemented in Matlab for this project and that are used by the program files both in the simulator and the SAR focusing algorithm. These functions can be recalled and used for other similar purposes. The first five functions are used in the program, whereas the last three functions were utilised when the different modules of the code were tested.

fft3 – This three-dimensional Fast Fourier Transform was written, because Matlab can only support 1-D and 2-D FFTs with its fft and fft2 functions respectively. This 3-D FFT can transform any 3-D data set from the time domain to the frequency domain. An example is: $data\_3d\_freq = fft3(data\_3d\_time)$.

ifft3 –This three-dimensional Inverse Fast Fourier Transform was written, because Matlab can only support 1-D and 2-D IFFTs with its ifft and ifft2 functions respectively. This 3-D IFFT can transform any 3-D data set from the frequency domain to the time domain. An example is: $data\_3d\_time = ifft3(data\_3d\_freq)$.

zeropad3 – This is a three-dimensional zero padding function. The zero padding function is used to increase (interpolate) the number of time domain samples by adding zeros to the frequency domain samples. Thus the signal display in the time domain is improved. This 3-D zero padding function improves the signals in all three directions of a 3-D data set. This function requires four input variables: a 3-D data set, *xpad*, *ypad* and *zpad*. The *xpad*, *ypad* and *zpad* represent the number of zeros to be added in the x, y and z directions respectively. An example is: $data\_3d\_zeropad = zeropad3(data\_3d, xpad, ypad, zpad)$.

shift3D – This is a 3-D shifting function. It applies a phase ramp to a set of 3-D data in the frequency domain and thus the time domain data is shifted appropriately. This function requires four input parameters: a 3-D data set, Δx, Δy and Δz. The Δx, Δy and Δz represent the number of pixels by which the data is displaced in the x, y and z directions respectively. Note that the signals can be shifted by decimal figures. An example is: $data\_3d\_shift = shift3D(data\_3d, \Delta x, \Delta y, \Delta z)$.

beamwidth – The purpose of this function is to ensure that the scene pixels inside an antenna conical beam have a value of unity, and those outside of it have a value of zero. This function assumes that the beam pattern is a perfect cone, without any field deviation for near and far field objects. The parameters required for this utility are the coordinates of each pixel, each antenna position and the antenna beam angle, $beam\_angle$, in degrees. Note that the use of this function can be very computationally intensive, as for each antenna position all the pixels in the scene have to be analysed. An example is: $data\_3d\_beam = beamwidth(x_o, y_o, z_o, X_a, Y_a, beam\_angle)$, where ($x_o$, y_o, z_o) is the coordinate of a pixel and (X_a, Y_a) is an antenna position.

resol_measure – This function is used to measure the 3dB resolution of a single point target in a 1-D array of data. This utility returns the number of pixels as the result for the measured 3dB resolution, and this can be converted to meters. An example is: $data\_1D\_resol = resol\_measure(data\_1D)$.

zeropad2 –This two-dimensional zero padding function improves the signal display for a 2-D data set, in a similar methodology as *zeropad3*. The input variables required are: a 2-D data set, *xpad* and *ypad*. An example is: $data\_2d\_zeropad = zeropad2(data\_2d, xpad, ypad)$.

shift2D – This function is applied to a 2-D data set and it is similar to *shift3D* in applications. The parameters needed are: a set of 2-D data, Δx and Δy. An example is: $data\_2D\_shift = shift2D(data\_2D, \Delta x, \Delta y)$.

3.4 Program Input Files

This section provides the user with the required knowledge regarding the input of the simulator. There are two input files: "setup_GPR.m" and "generate_scene.m".

The Matlab file, "setup_GPR.m", as explained in Section 3.3.1, has various variables that have to be initiliased, and this file has to be executed before running the simulator. Appendix A provides a sample model.

The "generate_scene.m" file generates the scene data according to the model discussed in Section 3.3.2. The scene data is stored in memory as a 2-D array in an ASCII float format (scene.dat). This is recalled by the simulator and the ASCII file is converted to a 3-D Matlab variable to be used to produce frequency domain backscattered data. Note that if a set of scene data, that has previously been generated, is to be used, the "generate_scene.m" file does not have to be run again. The "scene.dat" file is recalled from memory.

3.5 Program Operation and Output Files

The forward simulator, "simulate.m" file, generates 3-D frequency domain backscattered data by using a 2-D planar aperture. The simulator has two outputs: the "store.dat" file, and the Matlab variable, "partial_data".

The raw 3-D frequency domain backscattered data is converted to a 2-D array and is stored as "store.dat", which is in ASCII float format. This file is recalled by the focusing algorithm, in order to be processed.

The "partial_data" Matlab variable is the raw data which is partially processed, according to the specifications in the setup file. This partially processed data can be viewed using the 1-D and 2-D display tools.

Chapter 4

SAR focusing algorithm

4.1 Introduction

A SAR algorithm for the reconstruction of 3-D reflectivity images from 3-D frequency domain backscattered data has already been introduced in Chapter 2. The practical implementation of this algorithm is explained in this chapter, together with its software architecture.

The SAR focusing algorithm consists of various modules that are discussed in Section 4.2. The functions of each module are described, together with its input files, output files and file formats.

Section 4.3 describes the correlation function which generates the matched filters as a function of range. The filters are used by the kernel computation algorithm to focus the backscattered data.

The various signal processing steps that are involved in the SAR algorithm are explained in Section 4.4. The different algorithms implemented (block processing algorithm and amplitude compensation algorithm) are also discussed. Section 4.5 describes the display utility available to visualize the processed data.

4.2 Model of SAR Focusing Algorithm

The SAR focusing algorithm is divided into three main modules, which are shown in Figure 4.1.

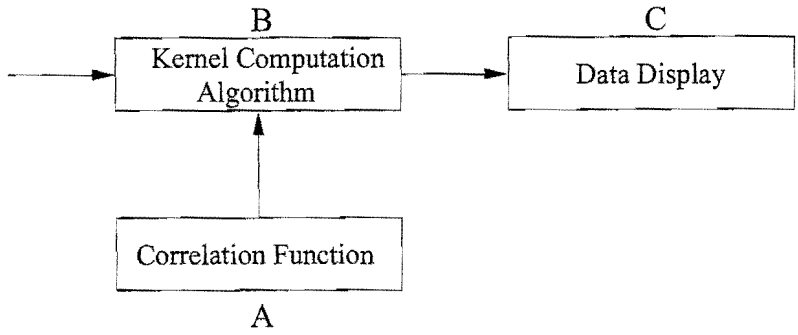

Figure 4.1: Modules showing the structure of the focusing algorithm.

- Module A is the correlation function. It consists of a matched filter that is generated as a function of range and is used by module B to focus the backscattered data.
- Module B is the kernel computation algorithm. This module was introduced in Chapter 2, Figure 2.4, as an approach to explaining systematically how the backscattered data is processed to generate reflectivity images. The ASCII file, "store.dat", from the simulator is loaded, and using the matched filter generated in the correlation function module, this data is processed. The processed data is then viewed in the display tool.
- Module C is the display utility. The processed data is visualised as 1-D and 2-D data sets to locate targets in the scene.

The files and the data formats needed for each module are given in Table 4.1. The ".dat" and ".m" extensions represent ASCII and Matlab files respectively. The Mv term in brackets means that the expression is a Matlab variable.

Module	File name	Input	Output
A	matchfilter.m	store.dat	matched_filter (Mv)
B	focus_alg.m	store.dat matched_filter (Mv)	processdata.dat
C	display_foc.m	processdata.dat	1-D and 2-D Images

Table 4.1: Input and output files for SAR focusing algorithm.

The "focus_alg.m" and "matchfilter.m" files require that the "setup_GPR.m" file, as explained in Chapter 3, be executed before they are run.

The signal processing techniques in the SAR focusing algorithm are discussed in the following sections.

4.3 Correlation Function

The correlation function is implemented in the "matchfilter.m" file shown in Table 4.1. This function generates the matched filters required to focus the backscattered data. The matched filters are generated in blocks as a function of range, and the block sizes are defined in the "setup_GPR.m" file as explained in Chapter 3. However, the block sizes are slightly modified to cater for the range curvature of a point target located at the end of the block. This is explained in detail in Section 4.4.1.

Figure 4.2 illustrates the steps involved in generating the matched filter for one block of data. Since a 3-D data set is more difficult to visualise, only a 2-D plot in the along track, x, and the range, z, is shown in the figure. The values b_o to b_5 represent the start and the end of data blocks.

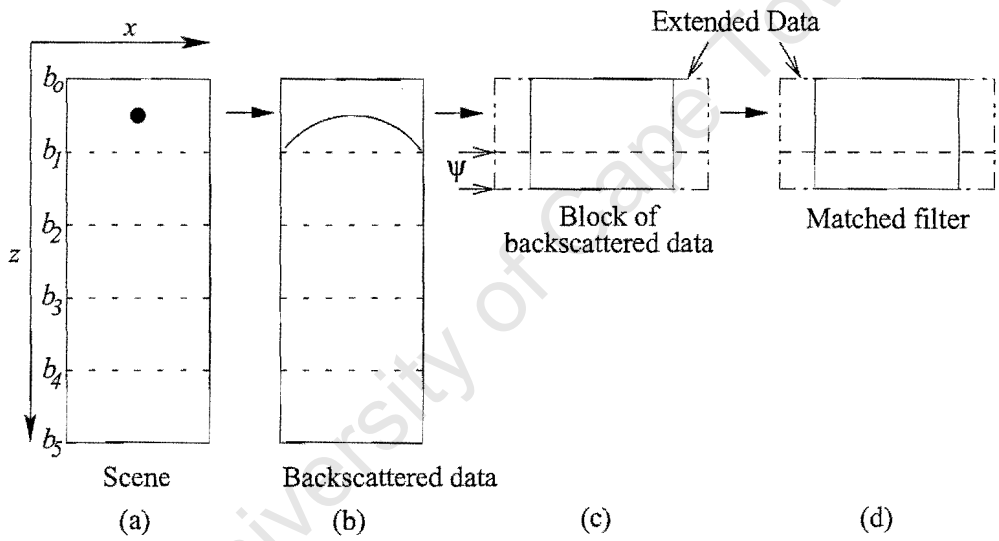

Figure 4.2: Steps involved in generating matched filter.

A point target, having unit reflectivity, is placed at the center of a block scene as shown in Figure 4.2(a). The scene data is generated for the entire 3-D scene. The simulator is run and the frequency domain backscattered data is produced, which is transformed to the time domain by applying an IFFT in the range (z) direction. The time domain data is shown in Figure 4.2(b). This data is then extended in the along track direction (x) by adding zeros. This is explained in detail in Section 4.4.3. The extended data is segmented according to the dimensions specified by b_o and $(b_1 + \psi)$, and this in turn is transformed to the frequency domain as shown in Figure 4.2(c). The conjugate of the segmented data in the frequency domain is the matched filter for this data block as shown in Figure 4.2(d). The purpose of ψ is to cater for the range curvature of a point target at the end (in the

range direction) of a block.

By applying this matched filter to the frequency domain data used to generate it, the point target is focussed in the time domain in all three directions. However, the point target is positioned at the zero coordinate in all three axes. The focussed point target thus needs to be shifted back to its correct position. This is achieved by multiplying the filter with an exponent function (phase ramp) in the frequency domain according to the property of the Fourier Transform for real functions [16]. This is equivalent to a shift in the time domain as illustrated by Equation 4.1 .

$$e^{-jw\tau}F(w) \Longleftrightarrow f(t-\tau) \tag{4.1}$$

The shifting is performed in all three directions using the function *shift3D*. The matched filter is stored in a 3-D array, called *matchedfilter*. The process is repeated for each block and the matched filters are accumulated in the same array. Note that the size of the *matchedfilter* array is bigger compared to the size of the backscattered data, due to the extra portion of data used for each block. The Matlab variable, *matchedfilter*, is stored in memory and is loaded by the focusing algorithm for processing.

4.4 Kernel Computation Algorithm

This section explains the steps involved in processing the backscattered data to focus targets in the scene. Various algorithms are used in the procedure and the functions of each algorithm are discussed.

4.4.1 Block processing algorithm

The block processing algorithm consists of segmenting the backscattered data into blocks for processing. This assures that all the targets in the scene are detected.

The time domain backscattered data is divided in the range direction and the block sizes are specified by the user in the "setup_GPR.m" file. The data closer to the antenna aperture is normally split into smaller blocks, because SAR focusing becomes more efficient further away from the aperture (as the assumptions for far field become valid) and so the point targets can be detected in larger blocks.

However, for each block, an extra portion of data is used during processing. This caters

for the range curvature of a point target located at the end of the block, and is illustrated in Figure 4.3. In Figure 4.3, the block sizes for this data set increase with range, as is shown by a, b and c.

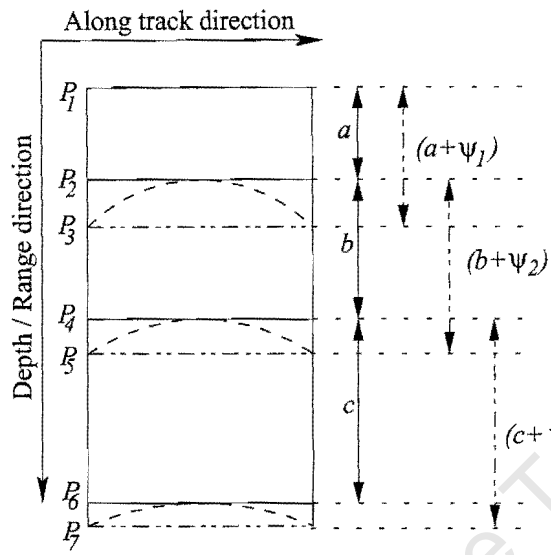

Figure 4.3: Diagram representing a slice of data in the across track direction.

The points P_1 and P_2 represent the start and the end of a block. The range curvature of a point target at a range P_2 stretches out to P_3, and so for that point target in the block to be focussed, the data from P_1 to P_3 has to be processed. This extra portion is represented by ψ, and is calculated using Equations 4.2 and 4.3:

$$R_{max} = \sqrt{(x_a - x_o)^2 + (y_a - y_o)^2 + (z_a - z_o)^2} \tag{4.2}$$

$$\psi = R_{max} - R_o \tag{4.3}$$

Where

- (x_o, y_o, z_o) is the position of a point target;
- (x_a, y_a, z_a) is the position of an antenna element;
- R_{max} is the furthest distance between any antenna element and the point target;
- R_o is the shortest distance of the point target in the range direction.

For the point target at range P_2, R_{max} is the distance given by $(a + \psi_1)$, and R_o is the distance given by a in Figure 4.3. The units of ψ, R_{max} and R_o are meters. Since all the

block sizes are specified in terms of number of pixels as explained in Chapter 3, the value of ψ is also converted to the appropriate number of pixels.

The point target at coordinate (x_o, y_o, z_o) is chosen to be at the end of the block in the range direction and at the center of the along and across directions. This is the closest approach range and the range curvature of a point target at this particular coordinate in a block spreads over a maximum number of bins compared to other positions.

The antenna element at coordinate (x_a, y_a, z_a) is chosen such that its distance to the point target is the maximum distance compared to any other antenna position.

Since the antenna has a conical beam pattern, the calculations of ψ are done for the curvature of the point target, in both the along track and the across track directions. Hence for each block, the greater value obtained for ψ is used.

4.4.2 Amplitude compensation algorithm

This algorithm ensures that a target with reflectivity σ, and having similar coordinates as the point target used for producing the matched filter, generates a peak of magnitude σ when focussed.

This is important for two main reasons:

1. Two identical objects should have the same magnitude, independent of their locations in the medium.
2. Objects having different reflectivities should be differentiated.

The amplitude compensation data, $g(x, y, z)$, is generated for each block of data. The initial steps are quite similar to the steps involved in producing the matched filter. The backscattered data is collected for a point target of unit reflectivity positioned at the center of a block. This data is segmented and processed using an appropriate matched filter. The extra portion of data represented by ψ in Figure 4.2 is then discarded. The peak value, p, of the focussed data in the time domain is measured, and all the values in the compensation algorithm for this specific block are set to $\frac{1}{p}$. The compensation data is stored in a 3-D array called *amplitude_comp*. The procedure is repeated for each block, and the compensation data is accumulated in the same array.

Each block in the *amplitude_comp* array is multiplied with its corresponding numerical data (test data) set, after the latter has been processed. Thus, a target with reflectivity σ

and positioned at the center of a block (compensation data has magnitude $\frac{1}{p}$), will have a magnitude of $\frac{\sigma \cdot p}{p} = \sigma$ as desired. This is illustrated in Figure 4.4 and Figure 4.5. Figure 4.4 shows various point targets, each having a reflectivity σ, distributed along the range direction. Without the amplitude compensation algorithm, the reflectivity values decay with distance from the antenna position. Figure 4.5 shows the effect of applying the compensation algorithm.

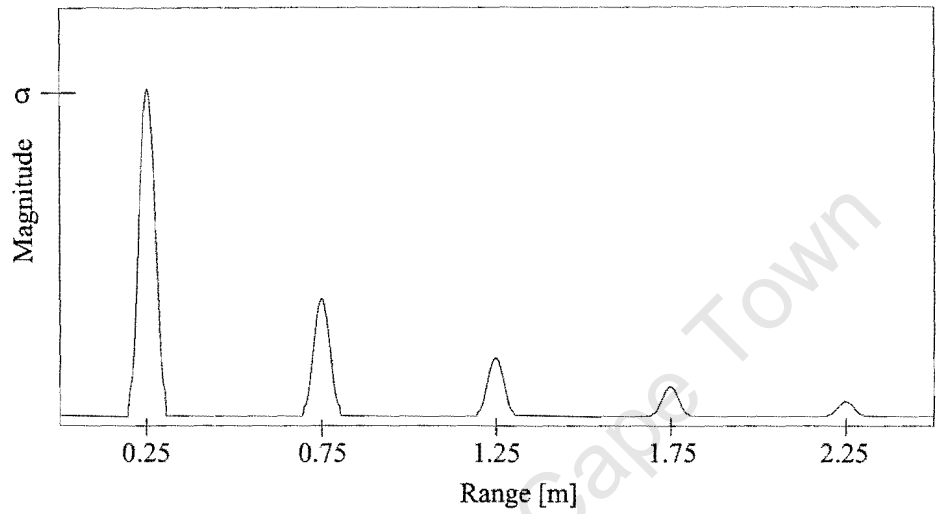

Figure 4.4: Diagram showing targets of reflectivity σ, positioned at the center of blocks along the range direction.

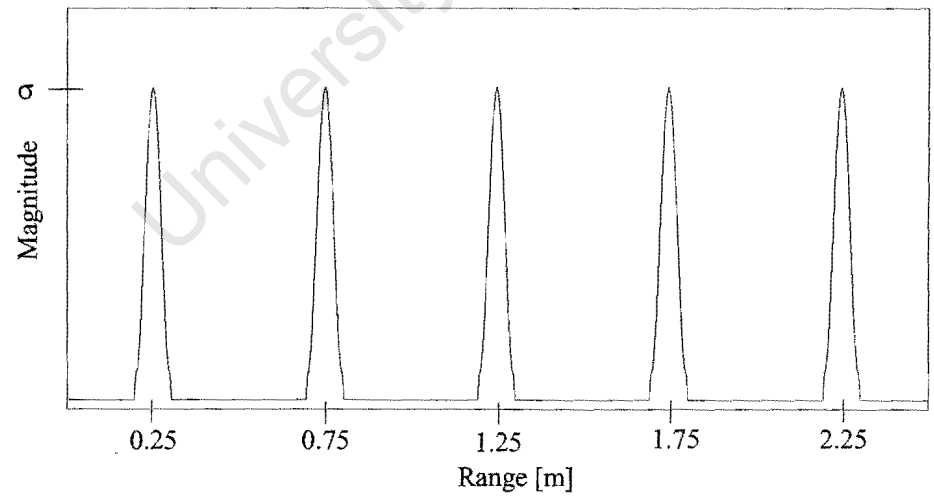

Figure 4.5: Diagram showing targets of reflectivity σ, after amplitude compensation algorithm is applied.

However, this is not the ideal method for this algorithm. A better approach would be to position a point target in the scene at a time and to collect the peak value of the focussed

data. The process is repeated for all the scene positions, and thus a compensation data set is generated which has the same dimensions as the one previously discussed. The new compensation data caters for the amplitude of the point target at each pixel, whereas the previous one caters for the amplitude of the center pixel and assumes the same value for all the neighbouring pixels in a block. The major limitation in processing all the pixels is the computational resource required.

4.4.3 Signal processing steps

This section describes the steps involved in processing the backscattered data to focus the targets in the scene. The algorithms discussed in Sections 4.3, 4.4.1 and 4.4.2 are used in the procedure.

The "store.dat" file containing the raw frequency domain backscattered data is loaded from memory and is converted to a 3-D array in Matlab. This wavefield data is basebanded and the data is transformed from the frequency domain to the time domain by applying an Inverse Fast Fourier Transform in the range direction. Note that the data is focussed in the range direction, and a point target response can be seen in that direction without further signal processing.

This time-domain data is extended in the along track direction by adding zeros to both sides of the data. The number of zeros added to either side is half the number of pixels in the along track direction. This extension ensures that the sidelobes of targets at the edge of the data block do not wrap around and appear as point targets when FFTs and IFFTs are applied to the data set.

The extended data is now segmented into blocks. Each block of data is converted to the frequency domain using 3-D FFTs and this data is applied to its corresponding matched filter in order to be compressed. The matched filter is obtained from the *matchedfilter* variable stored in memory (generated by the Correlation Function). Hamming windows are applied to the compressed data and the data set is transformed to the time domain using 3-D IFFTs.

The extra portion of data represented by ψ in Figure 4.2, and the extra portion of data resulting from adding zeros in the along track direction are discarded. The *amplitude_comp* variable, generated by the amplitude compensation algorithm, is multiplied with the focussed data as explained in Section 4.4.2, and thus the amplitudes of the targets are corrected.

The entire process is repeated for all the blocks of data, and each time the focussed data is accumulated in a 3-D array called *focus_data*. The focussed data and the scene used for generating the backscattered data are identical. The *focus_data* variable is stored as an ASCII file, "processdata.dat", which can be viewed in the display facility.

4.5 Data display

The processed data is displayed using the "display_foc.m" file. The "processdata.dat" file is loaded from memory and the data is converted to a 3-D array in Matlab. The 3-D processed data is transformed to the frequency domain where the 3-D zero padding function is applied to it. The greater the number of zeros added, the better the signal display will be, but more computational resource is required.

The zero-padded data is transformed to the time domain and viewed using 1-D and 2-D display tools.

Chapter 5

Analysis of Results

5.1 Introduction

The simulator and the SAR focusing algorithm have been implemented, and the results obtained are examined in this chapter by displaying the data as 1-D and 2-D set of graphs.

The raw 3-D frequency domain data out of the simulator for a single point target is displayed in Section 5.2. The graphs are analysed and the efficiency of the simulator to generate numerical data is discussed. The same data set is then processed by the focusing algorithm.

In Section 5.3, the results out of the focusing algorithm (for the single point target data) are displayed. The resolutions in the along track, across track and range/depth directions of the focussed target are discussed.

Futhermore, the frequency domain data, obtained for a scene having several targets, is processed. The results are examined and the ability of the SAR algorithm to focus targets are briefed.

5.2 Output from Simulator

The simulator generates the 3-D frequency domain backscattered data of a scene by synthesizing a 2-D planar aperture. The frequency range of simulation is from 200 MHz to 1600 MHz and the frequency step size is 18.8 MHz. The antenna used has a bandwidth,

β of 1600 MHz and a beamwidth, θ of 54^0.[1]

The 3-D frequency domain backscattered data set obtained by simulation for a single point target (reflectivity, σ) in a scene (scene dimensions (x, y, z) of 1m by 1m by 2m respectively) is basebanded and converted to the time domain. The partially processed data is depicted in Figure 5.1 as 1-D plots in the range/depth direction, and in Figure 5.2 as sets of 2-D slices orthogonal to the axes. The point target is located at the center of the xy-plane and at a depth, z of 0.5m.

Figure 5.1 shows 1-D plots of the backscattered data set, for the antenna positioned at the center of the across track axis and moving in the along track direction.

At each antenna position the time taken, t_{return}, for the signals, to be transmitted from the antenna to the point target and back, is proportional to the number of samples, N_r, of the peak signal in the range direction. Table 5.1 shows t_{return} for all the antenna positions in Figure 5.1.

Antenna Position(x_a, y_a, z_a)	N_{r}	t_{return} $[nanosecond]$
(0.0, 0.5, 0.0)	31	18.9
(0.1, 0.5, 0.0)	28	17.1
(0.2, 0.5, 0.0)	26	15.5
(0.3, 0.5, 0.0)	24	14.4
(0.4, 0.5, 0.0)	23	13.6
(0.5, 0.5, 0.0)	22	13.3
(0.6, 0.5, 0.0)	23	13.6
(0.7, 0.5, 0.0)	24	14.4
(0.8, 0.5, 0.0)	26	15.5
(0.9, 0.5, 0.0)	28	17.1
(1.0, 0.5, 0.0)	31	18.9

Table 5.1: Time, t_{return}, at each antenna position.

For each antenna position, the time t_{return} obtained corresponds to the theory discussed in Chapter 2.

The 3-D time domain data is now viewed as 2-D data sets, orthogonal to the across track, along track and range/depth directions respectively.

The first two plots in Figure 5.2 clearly show the range curvatures of the signals (unfocussed) in the along and across track directions. The third plot illustrates the target focussed in the range direction. Hence further signal processing is required to focus the

[1]The beamwidth is frequency dependent and the assumption that $\theta = 54^0$ (a fixed number) is not a perfect model for a broadband antenna.

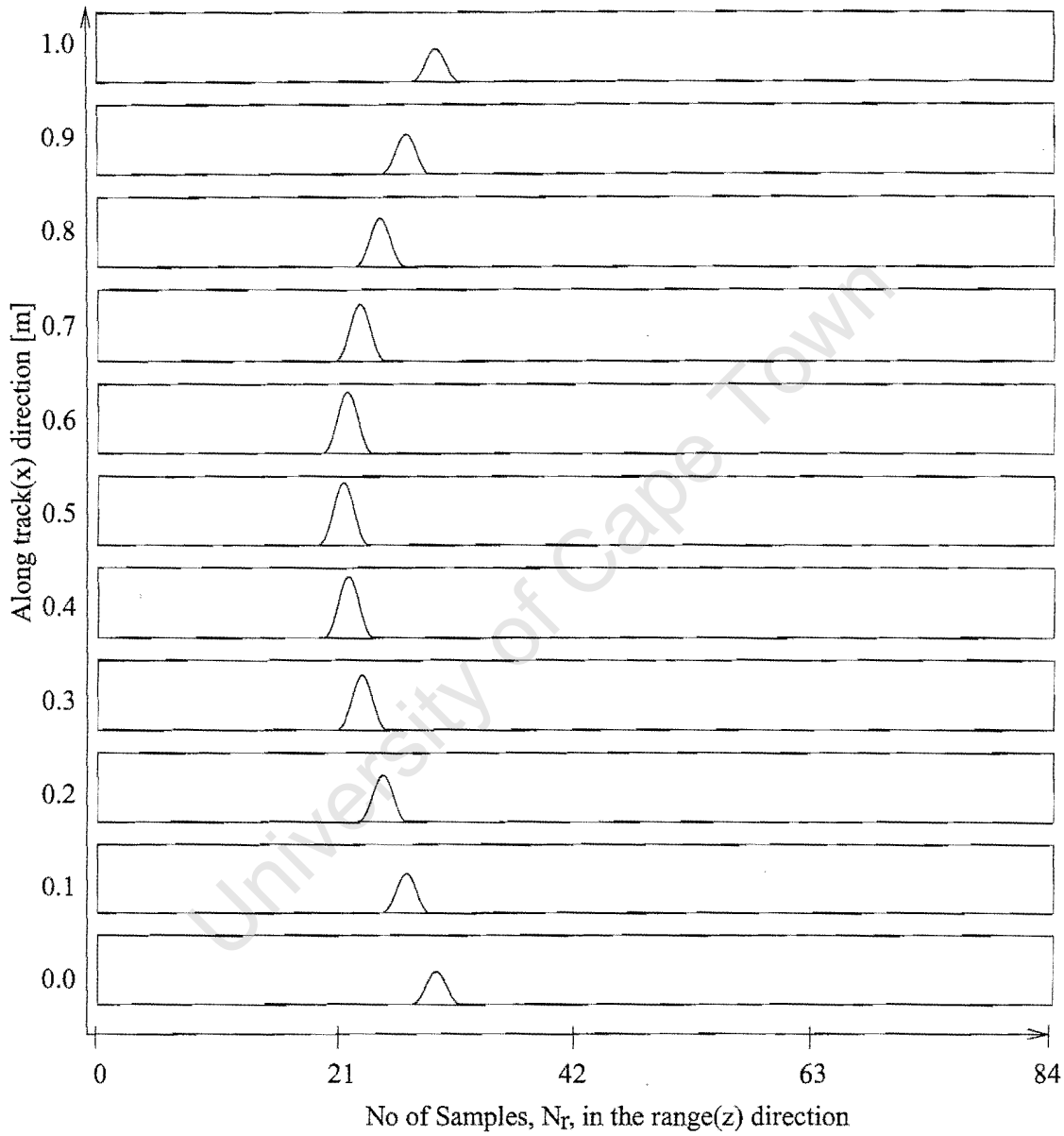

Figure 5.1: Range Curvature in the along track direction.

signals and thus locate the target.

5.3 Output from Focusing Algorithm

5.3.1 Single point target

The 3-D frequency domain data obtained by simulating a single point target is focussed using the SAR focusing algorithm. Figure 5.3(a), Figure 5.3(b) and Figure 5.3(c) show 1-D plots of the focussed data in the across track, along track and range directions respectively. The resolutions in all the three directions are given in Table 5.2.

	Dist.	Theoretical resol.	Resol. before HW	Resol. after HW
Across Track	0.50 m	1.90 cm	1.91 cm	2.10 cm
Along Track	0.50 m	1.90 cm	1.91 cm	2.12 cm
Range/Depth	0.50 m	2.34 cm	2.34 cm	2.57 cm

Table 5.2: Comparison of resolutions before and after Hamming windowing.

The point target is resolved efficiently and the resolutions in all the three directions satisfy the theoretical equations described in Chapter 2. The resolutions deteriorate slightly due to the window function.

5.3.2 Targets distributed symmetrically

Several targets, consisting of a centre target surrounded by sixteen other targets (four targets in the along track, four targets in the across track and eight targets in the range/depth direction), are placed in a scene. The scene has dimensions of ($x = 1\text{m}$, $y = 1\text{m}$, $z = 1\text{m}$) and the centre target is positioned at ($x = 0.5\text{m}$, $y = 0.5\text{m}$, $z = 0.5\text{m}$). All the targets have the same reflectivity value of σ. The frequency domain backscattered data is generated for the scene and the data is then focussed.

Figure 5.4 shows the focussed targets after application of the SAR algorithm. Three slices, orthogonal to the y, x and z axes respectively, are illustrated.

The focussed data set is an exact model of the scene. All the targets are clearly visible, each having a reflectivity value of σ. The positions of the targets are the same as those in the scene. Table 5.3 provides the positions of all the targets in the focussed data and those in the scene.

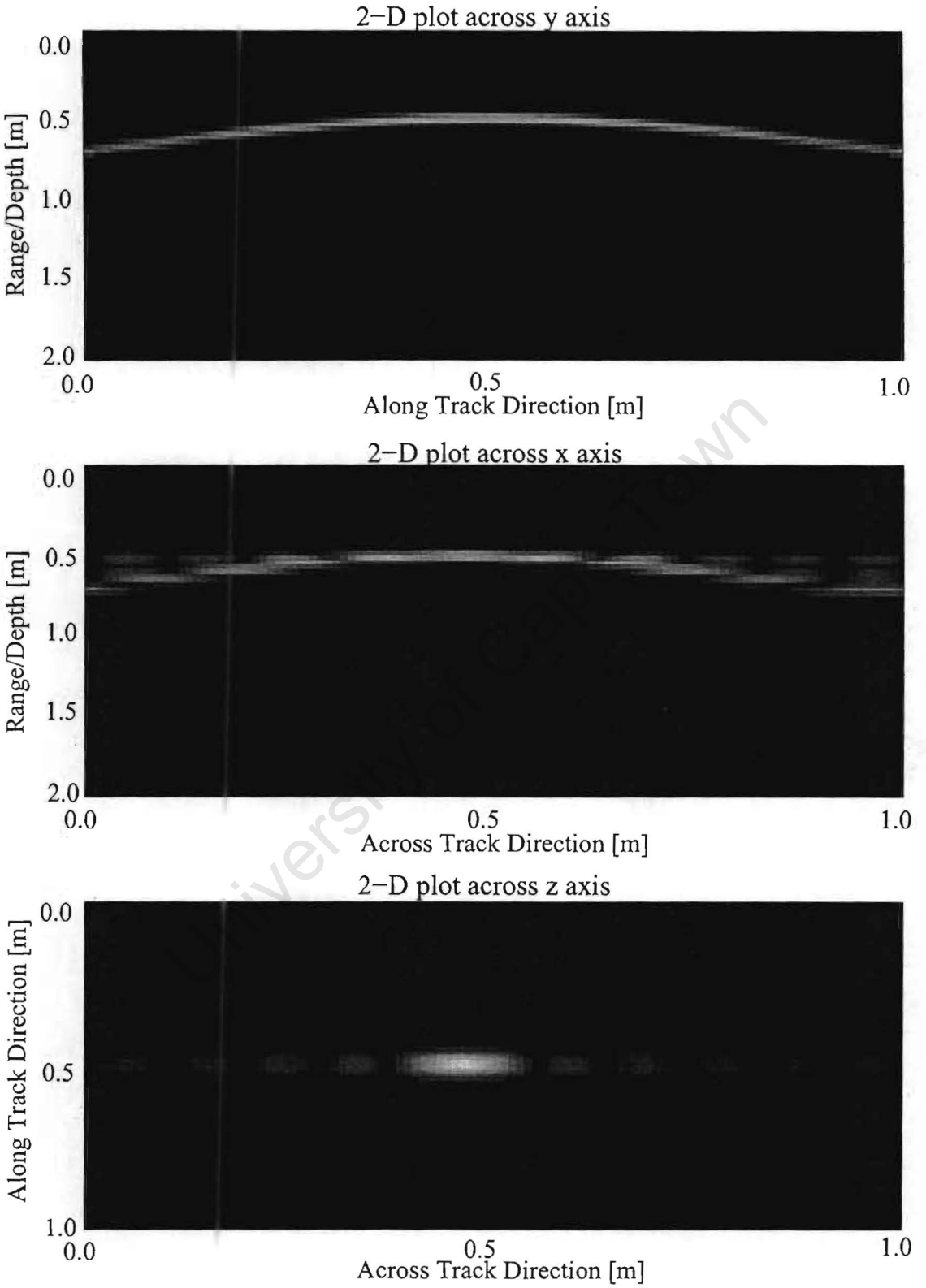

Figure 5.2: 2-D Slices of the backscattered data.

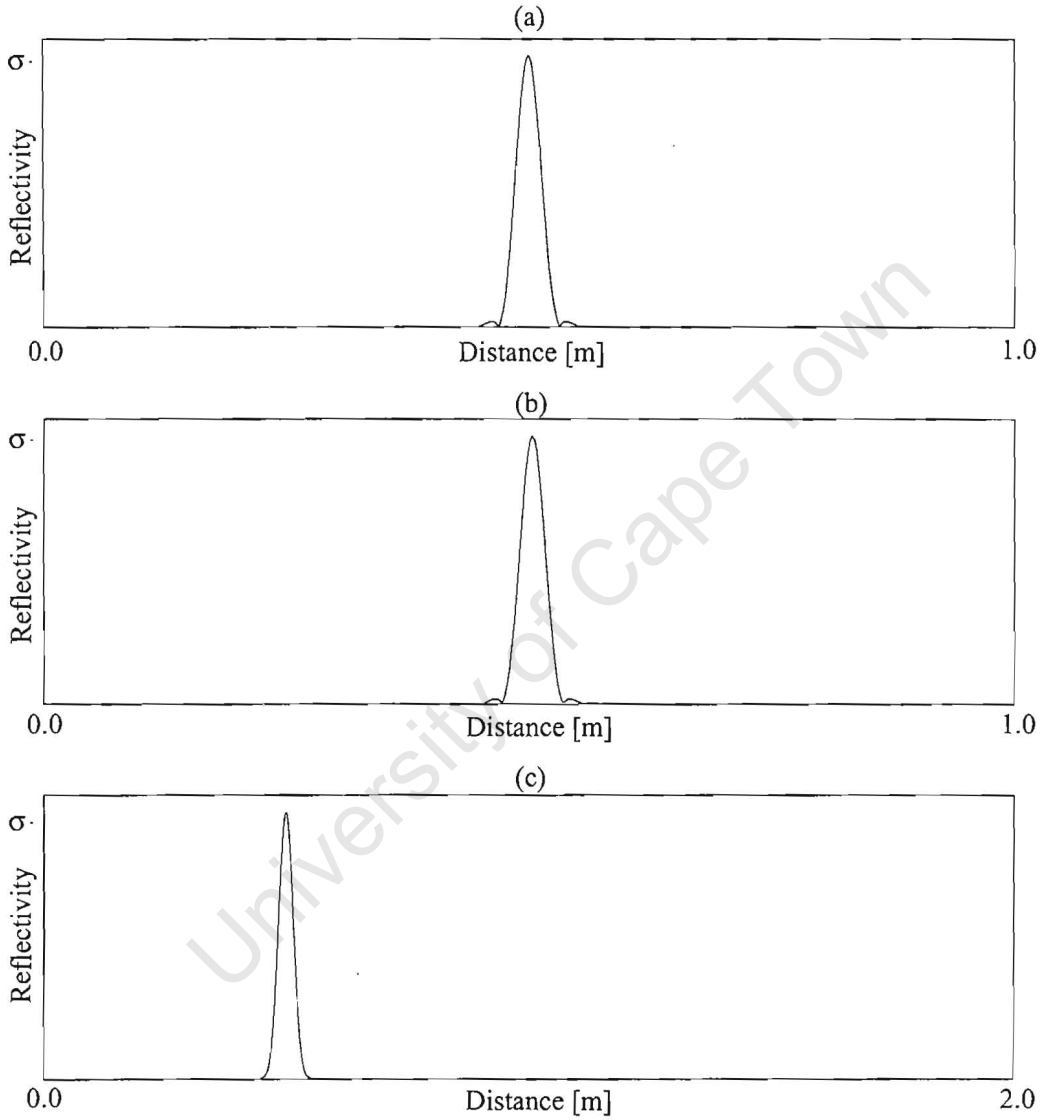

Figure 5.3: Focussed data for a single point target.

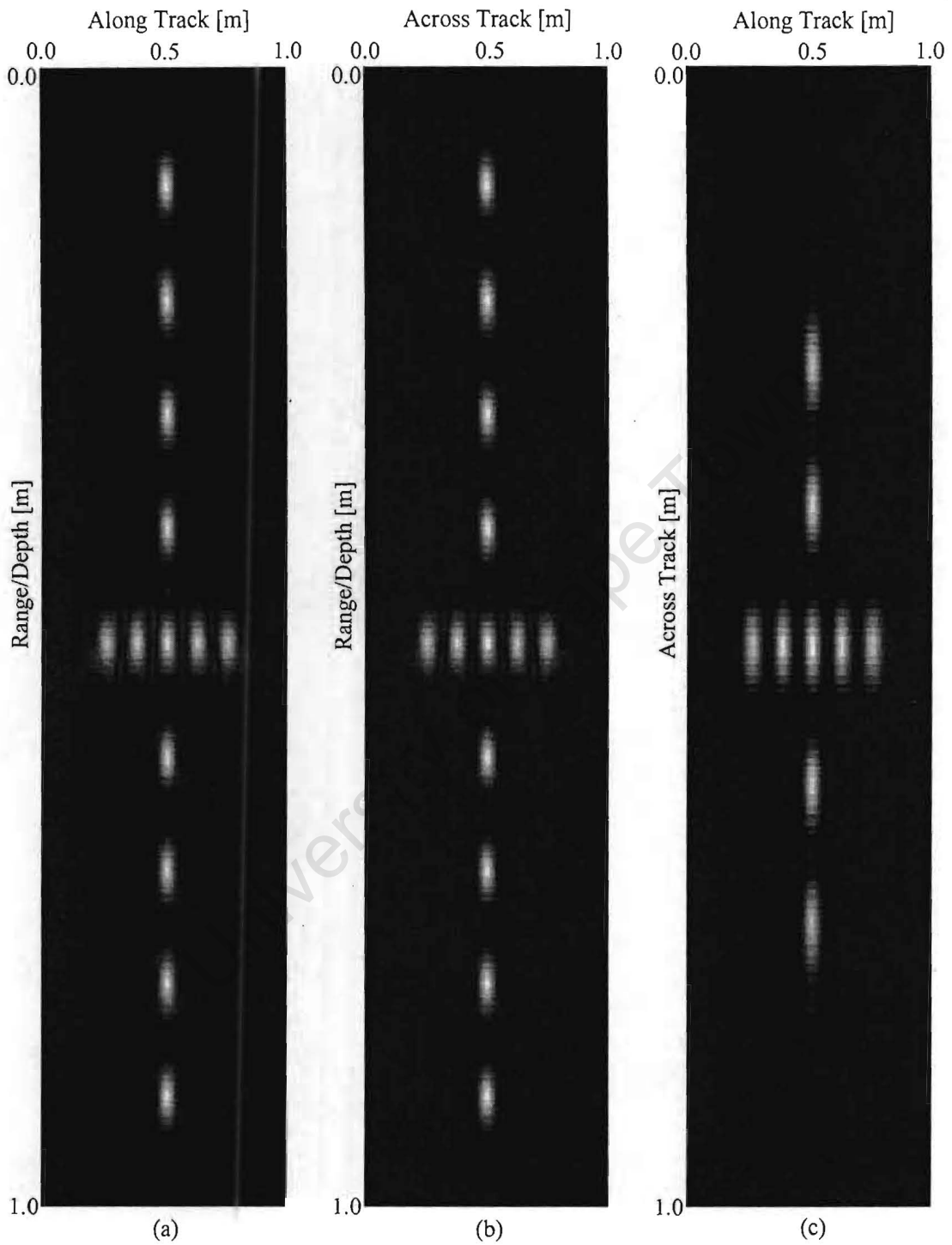

Figure 5.4: Targets focussed using SAR focusing algorithm.

Target Pos. (x,y,z) in scene	Target Pos. (x,y,z) in focussed data
(0.500, 0.500, 0.100)	(0.500, 0.500, 0.100)
(0.500, 0.500, 0.200)	(0.500, 0.500, 0.200)
(0.500, 0.500, 0.300)	(0.500, 0.500, 0.300)
(0.500, 0.500, 0.400)	(0.500, 0.500, 0.400)
(0.500, 0.500, 0.500)	(0.500, 0.500, 0.500)
(0.500, 0.500, 0.600)	(0.500, 0.500, 0.600)
(0.500, 0.500, 0.700)	(0.500, 0.500, 0.700)
(0.500, 0.500, 0.800)	(0.500, 0.500, 0.800)
(0.500, 0.500, 0.900)	(0.500, 0.500, 0.900)
(0.250, 0.500, 0.500)	(0.250, 0.500, 0.500)
(0.375, 0.500, 0.500)	(0.375, 0.500, 0.500)
(0.625, 0.500, 0.500)	(0.625, 0.500, 0.500)
(0.750, 0.500, 0.500)	(0.750, 0.500, 0.500)
(0.500, 0.250, 0.500)	(0.500, 0.250, 0.500)
(0.500, 0.375, 0.500)	(0.500, 0.375, 0.500)
(0.500, 0.625, 0.500)	(0.500, 0.625, 0.500)
(0.500, 0.750, 0.500)	(0.500, 0.750, 0.500)

Table 5.3: Comparison of target positions in focussed data and in scene.

Chapter 6

Conclusions and Recommendations

The following conclusions are drawn regarding the simulator and the SAR focusing algorithm.

1. The simulator generates 3-D frequency domain backscattered data (numerical data) efficiently from a 2-D planar aperture. The aperture is synthesized using a single antenna, having a bandwidth of 1600 MHz. The frequency range of simulation is from 200 MHz to 1600 MHz.

2. The 3-D data from the simulator is used to compute the point target responses for the correlation filters, which in turn are used to focus backscattered data. The simulator also generates test data for the SAR focusing algorithm.

3. The resolutions in the across track and the along track directions are achieved via SAR aperture synthesis techniques and are not limited by the physical dimensions of the antenna. The depth/range resolution is achieved via the transmission of narrowband *Stepped Frequency Continuous Wave* (SFCW) signals.

4. A SAR imaging algorithm to relocate targets in a 3-D scene has been successfully developed. The algorithm was implemented using 3-D FFT based correlation operations combined with block processing.

The following recommendations are made.

1. The SAR focusing algorithm should be used to process real data. The latter has been collected using the GPR sensors developed by the UCT Radar Remote Sensing Group.

2. The equation, used in the simulator to model the received signals, should be modified to cater for the multi-layers of the soil. The diffraction of signals at each layer and the relative dielectric constant of each layer are important issues to consider.

3. The block processing algorithm used should be replaced with a Stolt interpolation algorithm to improve the accuracy and efficiency of the focusing algorithm.

4. The code should be written in C or C++ languages, as the processing can be very computationally intensive for large amount of data. Parallel computing could also be implemented to speed up the processing time.

5. A time domain correlator could also be implemented, although it may be quite slow. Time domain correlators are very accurate.

Appendix A

Setup parameters

The code below is written in Matlab.

```
% Antenna parameter
start_antenna_x = ;          %Initialise the position of antenna in the x axis
start_antenna_y = ;          %Initialise the position of antenna in the y axis
Number_element_x = ;         % No of antenna elements in x direction
Number_element_y = ;         % No of antenna elements in y direction
delta_x = ;                  % Spacing between antenna elements in x direction
delta_y = ;                  % Spacing between antenna elements in y direction
beamwidth_antenna = ;        % Antenna beamwidth

% Frequency Range
Min_freq = ;                 % Minimum frequency for simulation
Max_freq = ;                 % Maximum frequency for simulation
effective_dielectric = ;     %value varies on type of soil (normally between 4 and 16)
Range_max = ;                % Maximum range that targets can be distinguished
speed_propagation = ;        % speed of light / (effective_dielectric)
Freq_step = ;                % speed_propagation/(2*Range_max)

%Signal Type
Signal_flag = ;              % 0, 1, 2, 3 for frequency domain, time domain,
                             % basebanded frequency domain, and basebanded
                             % time domain respectively.
HW_flag = ;                  % 0 for off and 1 for on
```

```
zeropad_flag = ;          % 0 for off and 1 for on
zeropad_factor = [ ];     % 1-D array. 3 values needed to represent number of zeros
                          % in x, y, z directions respectively

%Block sizes
block_vect_startpt = [ ]; % 1-D array for all start of data blocks
block_vect_endpt = [ ];   % 1-D array for all end of data blocks
```

Appendix B

Code for a sample scene

All the code for the scene model is written in Matlab.

```
% Define target scene boundaries
xdim_scene = 1.0;              % Scene dimensions in x (along track) direction
ydim_scene = 1.0;              % Scene dimensions in y (across track) direction
zdim_scene = Range_max;        % Scene dimensions in z (depth/range) direction
dim_scene = [xdim_scene, ydim_scene, zdim_scene];

% Define pixel sizes in scene (dx, dy and dz values respectively)
dx_scene = 0.125;
dy_scene = 0.125;
dz_scene = 0.125;
pixel_size = [dx_scene,dy_scene,dz_scene];

%Vectors to set the X, Y and Z axes for the scene
Xc= 0.0 : dx_scene : xdim_scene;
Yc= 0.0 : dy_scene : ydim_scene;
Zc= 0.0 : dz_scene : zdim_scene;
```

```
% Define array to store reflectivity values of pixels
reflectivity= zeros(length(Xc), length(Yc), length (Zc));

% Pixels are activated manually. This process can also be automated.
reflectivity(2,3,3)= 100.0;
reflectivity(3,3,3)= 100.0;
reflectivity(4,3,3)= 100.0;
reflectivity(5,3,3)= 100.0;

%Convert 3D array of reflectivity to 2D. The 'fread' command can only read 2D array
% and the 'fwrite' command can only write 2D array
reflec = zeros(length(Xc),(length(Yc)*length(Zc)) );
for a = 1: length(Xc),
reflec(a,:) = reflectivity(a,:,:);
end

% Convert the matlab file to binary or ASCII format
fid_scene = fopen('scene.dat','w');
fwrite(fid_scene,dim_scene,'real*8');
fwrite(fid_scene,pixel_size,'real*8');
fwrite(fid_scene,reflec,'real*8');
fclose('all');
```

Bibliography

[1] D. A. Ausherman, A. Kozma, J. L. Walker, H. M. Jones, and E. C. Poggio. Developments in Radar Imaging. *IEEE Transactions on Aerospace and Electronic Systems*, 20(4):363–400, July 1984.

[2] C. A. Balanis. *Antenna Theory: Analysis and Design*. John Wiley and Sons, New York, 2nd edition, 1996.

[3] C. Cafforio, C. Prati, and F. Rocca. SAR Data Focusing Using Seismic Migration Techniques. *IEEE Transactions on Aerospace and Electronic Systems*, 27(2):194–207, March 1991.

[4] W. G. Carrara, R. S. Goodman, and R. M. Majewski. *Spotlight Synthetic Aperture Radar. Signal Processing Algorithms*. Artech House, Norwood, MA, 1995.

[5] R. H. Clarke and J. Brown. *Diffraction Theory and Antennas*. Ellis Horwood Limited, Chichester, 1980.

[6] I. Cumming, F. Wong, and K. Raney. A SAR Processing Algorithm with No Interpolation. In *Proc. IEEE Geosci. Remote Sensing Symp., IGARSS'92*, pages 376–379, Clear Lake, TX, May 1992.

[7] J. C. Curlander and R. N. McDonough. *Synthetic Aperture Radar Systems and Signal Processing*. John Wiley and Sons, New York, 1991.

[8] J. P. Fitch. *Synthetic Aperture Radar*. Springer-Verlag, New York, 1988.

[9] J. Fortuny. An Efficient 3-D Near-Field ISAR Algorithm. *IEEE Transactions on Aerospace and Electronic Systems*, 34(4):1261–1269, October 1998.

[10] J. M. Horrell. *Range-Doppler Synthetic Aperture Radar Processing at VHF Frequencies*. PhD thesis, University of Cape Town, 1999.

[11] Y. Huang, Z. Ma, and S. Mao. Stepped-frequency SAR System Design and Signal Processing. In *Proc. European Conference on Synthetic Aperture Radar, EUSAR'96*, pages 565–568, Königswinter, Germany, March 1996. VDE-Verlag GMBH, Berlin and Offenbach.

[12] S. Kingsley and S. Quegan. *Understanding Radar Systems*. SciTech Publishing, Mendham, 1999.

[13] J. C. Kirk. A Discussion of Digital Processing in Synthetic Aperture Radar. *IEEE Transactions on Aerospace and Electronic Systems*, 11(3):326–337, May 1975.

[14] J. M. Lopez-Sanchez and J. Fortuny-Guasch. 3-D Radar Imaging Using Range Migration Techniques. *IEEE Transactions on Antennas and Propagation*, 48(5):728–737, May 2000.

[15] R. T. Lord and M. R. Inggs. High Resolution SAR Processing Using Stepped-Frequencies. In *Proc. IEEE Geosci. Remote Sensing Symp., IGARSS'97*, volume 1, pages 490–492, Singapore, August 1997.

[16] N. Morrison. *Introduction to Fourier Analysis*. John Wiley and Sons, New York, 1994.

[17] A. V. Oppenheim. *Applications of Digital Signal Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1978.

[18] J. G. Proakis and D. G. Manolakis. *Digital Signal Processing: Principles, Algorithms and Applications*. Prentice-Hall, 3rd edition, 1996.

[19] R. K. Raney, H. Runge, R. Bamler, I. G. Cumming, and F. H. Wong. Precision SAR processing using chirp scaling. *IEEE Transactions on Geoscience and Remote Sensing*, 32(4):786–799, July 1994.

[20] R. Stolt. Migration by Fourier transform techniques. *Geophysics*, 1(43):49–76, 1978.

[21] F. G. Stremler. *Introduction to Communication Systems*. Addison-Wesley Publishing Company, New York, 1990.

[22] J. D. Taylor, editor. *Introduction to Ultra-Wideband Radar Systems*. CRC Press, Boca Raton, Florida, 1995.

[23] F. E. Terman. *Radio Engineer's Handbook*. McGraw-Hill Book Company, New York and London, 1st edition, 1943.

[24] M. R. Vant. Synthetic Aperture Radar Signal Processing. In *Proceedings of the 1989 International Conference on Radar*, April 1989.

[25] D. R. Wehner. *High-Resolution Radar*. Artech House, Norwood, MA, 2nd edition, 1995.

[26] A. J. Wilkinson. *Lecture notes on the Radar Signal Processing Fundamentals*. University of Cape Town, South Africa, version 0.6, 2001.