Guide of GPR-20 data: specification, formatting, preprocessing and processing

Universidad de los Andes
School of Engineering
Department of electric and electronic engineering

Roberto Bustamante Miller
Daniel Julián González Ramírez

June 11, 2021

Contents

Chapter 1

Introduction

Buried object/target detection is the goal of ground penetrating radar (GPR) surveys. In order to achieve proper detection of the buried targets, it is necessary to retrieve the location of said target over the x, y and z axes. This location is retrieved based on the knowledge of antenna positioning when the different A-Scans are performed and on the arrival time of the back scattered signal.

Depending in how data is captured, different procedures have to be made over it in order to end up with an understandable signal. Procedures may include data organization, metadata gathering, signal adaptation (preprocessing) and processing. The understandable signal then can be analyzed to determine the results of the survey.

This document serves as a guide on how the data (gathered from GPR-20 system and from simulation) is organized, formatted, preprocessed and processed. Chapter 2, includes an algorithm description on how to get a time domain A-Scan signal based on the frequency domain S_{21} parameter measured by the GPR, also a description on the raw data captured and the metadata required is given, for both GPR-20 surveys and simulation. Chapter 3, describes how to systematically read the data outputs previously presented and how a condensed file is created for more simple manipulation; Python code of the described method is included and explained. Chapter 4, shows how the new data file is read and plotted in different ways using Python. Chapter 5, explains the preprocessing and processing done over GPR signals in order to retrieve the final image from the subsurface; some techniques for preprocessing are presented and Kirchhoff migration algorithm (2D and 3D versions) is presented as the main processing technique, all the mentioned are presented along with its Python implementation. Finally, chapter 6 gives a summary on the data pipeline and a couple of examples for better understanding.

Chapter 2

Raw data output

2.1 Signal captured from GPR surveys

Our GPR-20 is based on a vector network analyzer (VNA) which is an instrument that measures S-parameters (relative transmission and reflection) using stepped frequency sinusoidal waves. The VNA used for the GPR-20 is a two-port VNA (Anritsu MS2026C). Each port of the VNA is connected to an ultra-wide band (UWB) antenna. The frequency range of each measurement can vary, but typically is set between 900 MHz and 6 GHz. The number of frequency steps can vary too, but typically is set as 501 (this amount of frequency points gives a good balance between frequency resolution and speed of measurement). For more details on the specifications of the used VNA please refer to the user manual at [1].

A pair of UWB antennas is connected to the VNA (one to each port). The antenna pair is mounted on a structure so that the propagation is directed downwards. The S-parameters measured with the VNA are affected by the whole network beyond the ports: connectors, antennas and the field interactions perceived by the antennas. Every object that interacts with the antenna radiated fields will scatter the fields in different directions. The antennas will perceive the scattering that propagates towards them (back scattering) and the S-parameters will be modified by it. This measurement of the S-parameters (at one point in space) is known as an A-Scan, it is originally recorded in the frequency-domain usually (and always for our system) as the S_{21} parameter.

This section of the document lays out the raw data output of the GPR system. A description of the original frequency-domain A-Scans is done. Then the inverse fast Fourier transform (IFFT) algorithm used to obtain a time-domain signal is presented. The time-domain A-Scan is described afterwards. And the metadata to consider in GPR field surveys is presented. Finally, a description of data obtained from gprMax simulations is included.

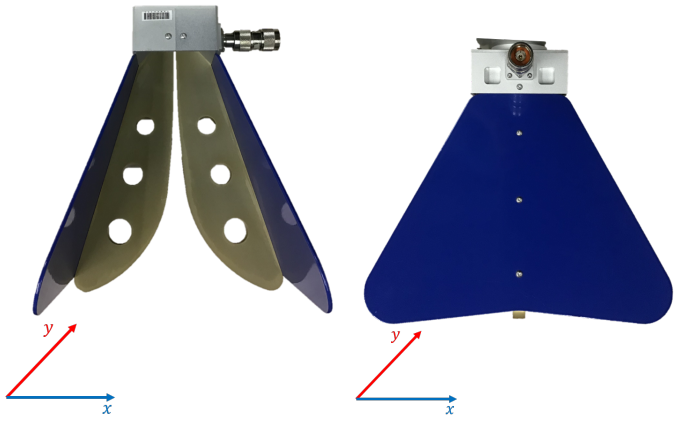
2.1.1 Frequency-domain A-Scan format and parameters

The "rawest" form of data acquired by the GPR-20 is the frequency-domain A-Scan. The data is stored in a `.csv` file that contains two columns and N + 1 rows (N being the number of frequency points of the A-Scan). The first column contains the real-part of the S_{21} parameter and the second column contains the imaginary-part of the S_{21} parameter. Other important data is saved automatically in the name of each A-Scan file that has the following syntax:

```
VNA_X*_Y*_Or*_Fs*M_Fe*M_Qf*_H*.csv
```

- `X*` tells the position (in millimeters) of where the antennas are placed over the x-axis. *Example:* `X155` *is obtained when the antennas are located at* $x = 155$ mm *from the origin.*
- `Y*` tells the position (in millimeters) of where the antennas are placed over the y-axis. *Example:* `Y310` *is obtained when the antennas are located at* $y = 310$ mm *from the origin.*
- `Or*` tells the orientation of the antennas. The value given is the axis parallel to the antenna H-plane. *Example:* `OrX` *is obtained when the ridges of the antennas can be seen from the x-axis (Figure 2.1(a)). Example:* `OrY` *is obtained when the ridges of the antennas can be seen from the y-axis (Figure 2.1(b))*

- `Fs*M` tells the starting frequency of the sweep in MHz. *Example:* `Fs600M` *is obtained when the starting frequency is* $f_s = 600$ MHz.
- `Fe*M` tells the ending frequency of the sweep in MHz. *Example:* `Fe6000M` *is obtained when the ending frequency is* $f_e = 6000$ MHz.
- `Qf*` tells the number of points of the frequency sweep. *Example:* `Qf501` *is obtained when the number of points of the frequency sweep is* $Q_f = 501$ points.
- `H*` tells the position (in millimeters) of where the antennas are placed over the z-axis in reference to the ground surface. *Example:* `H248` *is obtained when the antennas are located at* $z = 248$ mm *from the ground surface.*



(a) Antenna x orientation. (b) Antenna y orientation.

Figure 2.1: Possible orientations of the GPR-20 antennas.

An example of the data in a frequency-domain A-Scan file is shown in Table 2.1. A typical survey can produce over 25000 A-Scans, depending on the spatial resolution and surveyed area. Having a large amount of surveys means that storing all of the A-Scans can take up a considerable amount of space. The size of frequency-domain A-Scan file with a 501-points sweep is about 12 KB; if a survey of 25000 A-Scans is done with this configuration, the resulting storage space needed is close to 250 MB.

S21_real	S21_imag
-1.89E-05	-2.06E-05
3.83E-05	-1.20E-05
⋮	⋮
2.16E-04	1.06E-04

Table 2.1: Example of a frequency-domain A-Scan. File contains two columns, the one on the left contains the real part value of the S_{21} parameter and the column on the right contains the imaginary part. The title of this file is `VNA_X155_Y310_OrX_Fs600M_Fe6000M_Qf501_H248.csv`. With help of the information given in the title one can know the first frequency point swept was $f_s = 600$ MHz and the last one $f_e = 6$ GHz. The frequency steps of the sweep are $\Delta f = 10.78$ MHz ($\Delta f = [f_e - f_s]/Q_f$), therefore one can know which S_{21} values belong to each frequency point.

2.1.2 Inverse Fast Fourier Transform of VNA data

The frequency-domain A-Scan contains information about the subsurface scenario and that is why the system can be used as a GPR. A problem with the frequency-domain A-Scan is that it is not straightforward understand. Using the IFFT, the signal can be transformed into the time-domain and its meaning becomes more evident. The time-domain signal shows amplitudes at different points in time that can be used to identify buried scattering objects. Figure 2.2 shows an example on the information that an A-Scan contains both in time- and frequency-domains; it is evident that the time-domain signal is much easier to understand.

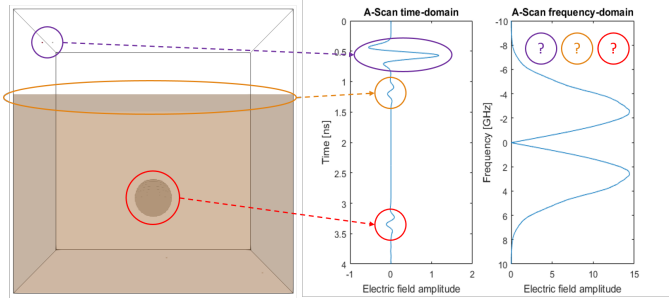


Figure 2.2: Meaning of the A-Scan in time- and frequency domain.

The VNA S_{21} parameters is given as a pass-band, one-sided, frequency-domain signal. These characteristics demand for a special implementation of the IFFT algorithm. First, the signal is zero padded for the values corresponding to $f = 0$ to $f = f_s$. After that, the signal is mirrored from the positive frequencies to the negative frequencies; the signal for the negative frequencies is the complex conjugate of the signal for the positive frequencies. Both the original and mirrored signals are then appended to form the full signal. Given that the frequency step Δf is known, the frequency vector is created the negative end frequency $-f_e$ to the positive end frequency f_e. Then signal data is organized in the way the fast Fourier transform (FFT) outputs it. The IFFT algorithm is applied over the signal. And finally, the signal is organized from the output of the IFFT. For more detail on how the organization output of FFT and IFFT is done refer to [2].

A Python function was developed to calculate the IFFT of different signals, both one-sided and two-sided. The function implementation can be found at `https://bit.ly/2XI1ljQ`. Below, an example of a signal similar to what can be obtained with the VNA is shown. The selected signal has an inverse Fourier transform easy to determine using the inverse Fourier transform definition and the theoretical result is compared with the result obtained from the algorithm.

Example of IFFT algorithm on signal

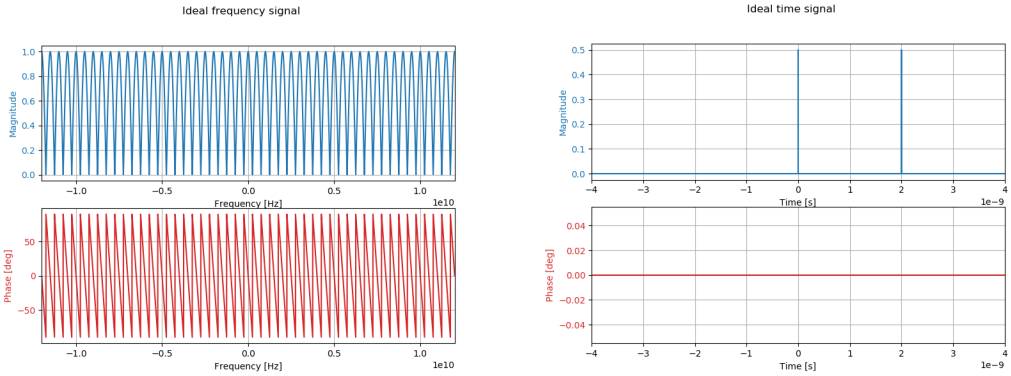
Consider the frequency-domain signal presented in Equation (2.1). Figure 2.3(a) shows the amplitude and phase of frequency-domain signal. The inverse Fourier transform of the signal can be obtained by solving the inverse Fourier integral (Equation (2.2)). Fourier transform property for time-shift (Equation (2.3)) can be used in this case to solve the integral easier. The solution for the integral is given in Equation (2.4), this is the time-domain signal representation shown in Figure 2.3(b).

$$H(f) = A \cdot \cos\left(2\pi t_0 f\right) \cdot \exp\left(-j2\pi t_0 f\right) \tag{2.1}$$

$$h(t) = \int_{-\infty}^{\infty} H(f) \cdot \exp\left(j2\pi f t\right)\, df \tag{2.2}$$

$$x(t) \overset{Fourier}{\longleftrightarrow} X(f) \quad x(t - t_0) \overset{Fourier}{\longleftrightarrow} X(f) \cdot \exp\left(-j2\pi t_0 f\right) \tag{2.3}$$

$$h(t) = \frac{A}{2} \cdot [\delta(t) + \delta(t - 2t_0)] \tag{2.4}$$



(a) Frequency-domain signal.

(b) Time-domain signal.

Figure 2.3: Complete time- and frequency-domain signals.

Now, if the system that produced the original frequency-domain response is characterized with a VNA; the VNA will give the signal shown in Figure 2.4. The signal in Figure 2.4 is a pass-band, one-sided, frequency-domain signal. Before doing the IFFT operation on the signal, it must treated as previously stated. The signal shown in Figure 2.5(a) is ready for the IFFT algorithm. Using the IFFT function the time-domain signal is obtained, result is shown in Figure 2.5(b). At first glance of Figure 2.5(b) it may seem there is a phase error in the signal, but upon closer inspection the phase shifts are of 180° and −180° which means just a change of sign in the values outside from the peaks, therefore the IFFT is determined correctly.

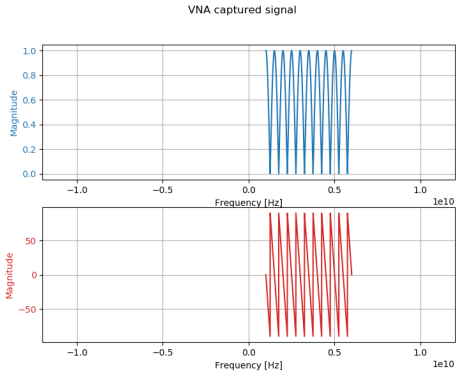
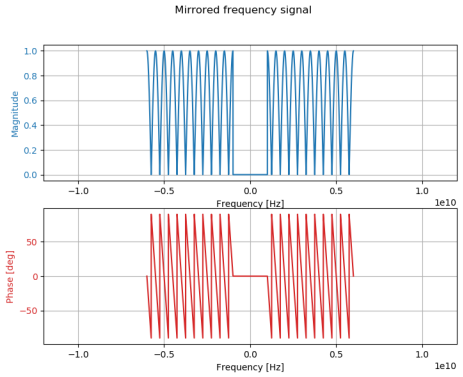
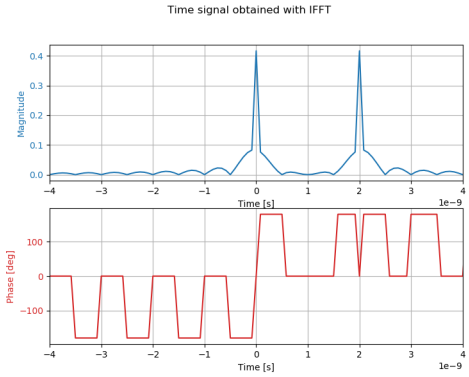


Figure 2.4: Signal captured by the VNA.



(a) Frequency-domain signal ready for the IFFT.



(b) Time-domain signal obtained from the IFFT.

Figure 2.5: Time- and frequency-domain signals.

The result of the IFFT algorithm of an actual A-Scan signal is presented in Figure 2.6. As it is expected, the signal shows an A-Scan response similar to a time-domain wavelet (broadband) source.

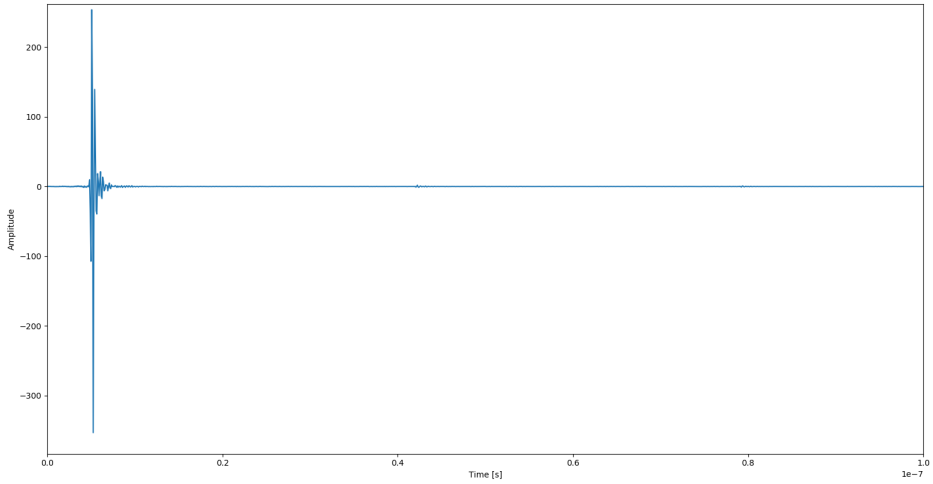


Figure 2.6: Time-Domain A-Scan obtained from the IFFT of a VNA scan.

2.1.3 Time-domain A-Scan format and parameters

After the time-domain A-Scan is obtained from the IFFT algorithm, it is saved as a `.csv` file. Description of parameters is very similar to the ones of the frequency-domain A-Scan. The file contains two columns and N + 1 rows (N being the number of time points of the A-Scan). The first column contains the real-part of the time-domain signal (IFFT of S_{21} parameter) and the second column contains the imaginary-part of the time-domain signal. Other important data is saved in the name of each A-Scan file that has the following syntax:

```
TIME_X*_Y*_Or*_Ts*u_Te*u_Qt*_H*.csv
```

The unchanged parameters from the frequency-domain data file mean the same. The parameters that changed (`Ts`, `Te` and `Qt`) are described below:

- `Ts*u` tells the starting time of the A-Scan in μs. *Example:* `Ts0u` *is obtained when the starting time is* $t_s = 0$ μs.
- `Te*u` tells the ending time of the A-Scan in μs. *Example:* `Te370u` *is obtained when the ending time is* $t_e = 370$ μs.
- `Qt*` tells the number of points of the frequency sweep. *Example:* `Qt501` *is obtained when the number of points of the frequency sweep is* $Q_t = 501$ points.

An example of the data in a time-domain A-Scan file is shown in Table 2.2. Storage space needed for time-domain A-Scan files is in the same order of magnitude as the frequency-domain A-Scan files.

A-Scan_real	A-Scan_imag
-24.04	0
-151.46	0.005
⋮	⋮
2.78E-06	-0.016

Table 2.2: Example of a time-domain A-Scan. File contains two columns, the one on the left contains the real part value of the A-Scan and the column on the right contains the imaginary part. The title of this file is `Time_X155_Y310_OrX_Ts0u_Te370u_Qt501_H248.csv`. With help of the information given in the title one can know the first point is at $t_s = 0$ μ s and the last one $t_e = 6$ μ s. The frequency steps of the sweep are $\Delta t = 0.74$ μ s ($\Delta t = [t_e - t_s]/Q_t$), therefore one can know which values belong to each time point.

2.1.4 Metadata to consider

Metadata is all data that provides supplementary information about the measurement made with the GPR. That means that all the information aside from the A-Scan, domain parameters and position of the antennas are metadata. The identified metadata for different surveys is the following:

1. **Ground permittivity:** Value of electric permittivity $\varepsilon(f)$ of the ground where the survey is done. Ideally it should be taken at different points in the grid and the location of the measurement must be stored along with it. Also is good to have values of permittivity before and after the survey.
2. **Location of buried objects:** Location (x, y and z) buried objects placed on the survey area.
3. **Description of buried objects:** Full description of objects buried in the survey area (dimensions, materials, descriptive names, etc.).
4. **Photographs of the GPR setup, buried objects and survey surface**
5. **GPS location of the survey**

2.2 Signal captured from gprMax simulations

Another important source of GPR survey data are simulations made with simulations. The electromagnetic simulator used for most of these simulations is gprMax [3], which is based on the Finite-Difference Time-Domain algorithm. Because of the simulation algorithm, data gathered from the surveys is already given in the time-domain.

In the simulation surveys, all the scenario is built up from gprMax scripts. A ground box is created, with certain surface roughness and electric characteristics. Buried objects are placed inside the ground box, including clutter and targets. A pair of UWB antennas are placed above the ground at a certain height. Sometimes just a transmitting antennas is placed and the receiving antenna is represented by a receiver (Rx) point, this is the most common configuration used so far therefore all the explanation given in this document assumes this configuration. Receiver points are grid cells from which the electric and magnetic fields are saved for every time-step of the simulation. For more details on how to build scenarios and how the gprMax software behave, please refer to [4].

2.2.1 A-Scan format and parameters

The output file of each A-Scan taken on a simulated survey has a `.out` extension and is formatted using HDF5 format [5]. HDF5 format features groups, hierarchy, attributes and different types of data.

Note: Output files of gprMax v.4 have `.h5` file extension, when using gprMax v.4 or a later release please beware all of the mentioned `.out` extensions should be understood as `.h5` extensions.

Similar to real GPR measurements, simulated ones need the following relevant information to have a complete description of the GPR survey:

1. Antenna position in the $x-y$ plane (Green in Figure 2.8)
2. Electric field/Voltage perceived by the receiver (Blue in Figure 2.8)
3. Simulation time step (Red in Figure 2.8)
4. Total amount of simulation iterations (Purple in Figure 2.8)
5. Polarization of the transmission antenna
6. Height of the antennas in reference to the ground surface

Parameters 1 through 4 are given in the `.out` file by default, as shown in Figure 2.8. Polarization of the transmission antenna can be added to the output files with a small modification in the gprMax source code (add `grp.attrs['Polarisation'] = src.polarisation` to line 119 of `fields_output.py` file), but is not included in the output file by default. Height of the antennas in reference to the ground surface is not included in the `.out` file and it is not possible to include it.

Parameters not given in the `.out` file can be retrieved from the `.in` (input file). The `.in` file is a plain text file which contains the input commands for gprMax to create the desired scenario. A simple `.in` file is presented in Figure 2.7 to show where the Tx polarization (*blue*) and the ground surface height (*red*) are specified.

Note: when surface roughness is added to the ground is is not possible to determine the actual height at the survey points then the average of the range can be used as a good estimate.

```
#title: SimulacionPruebaAScan
#domain: 0.500 0.500 0.500
#dx_dy_dz: 0.002 0.002 0.002
#time_window: 20e-9

#material: 4 0 1 0 Soil

#waveform: ricker 10 3e9 my_ricker

#python:
from gprMax.input_cmd_funcs import *

i = (current_model_run - 1) % 67
j = int((current_model_run - 1) / 67)

hertzian_dipole('x', 0.040 + 0.006 * i, 0.060 + 0.006 * j, 0.450, 'my_ricker')
rx(0.060 + 0.006 * i, 0.060 + 0.006 * j, 0.450)
#end_python:

#box: 0 0 0 0.500 0.500 0.350 Soil
#sphere: 0.250 0.250 0.150 0.040 pec
```

Figure 2.7: gprMax input script.

```
HDF5 SimulacionPruebaAScan2.out
Group '/'
  Attributes:
    'gprMax': '3.1.5'
    'Title': 'SimulacionPruebaAScan'
    'Iterations': 5194
    'nx_ny_nz': 250 250 250
    'dx_dy_dz': 0.002000 0.002000 0.002000
    'dt': 0.000000
    'nsrc': 1
    'nrx': 1
    'srcsteps': 0 0 0
    'rxsteps': 0 0 0
  Group '/rxs'
    Group '/rxs/rx1'
      Attributes:
        'Name': 'Rx(33,30,225)'
        'Position': 0.066000 0.060000 0.450000
      Dataset 'Ex'
        Size: 5194
        MaxSize: 5194
        Datatype: H5T_IEEE_F32LE (single)
        ChunkSize: []
        Filters: none
        FillValue: 0.000000
      Dataset 'Ey'
        Size: 5194
        MaxSize: 5194
        Datatype: H5T_IEEE_F32LE (single)
        ChunkSize: []
        Filters: none
        FillValue: 0.000000
      Dataset 'Ez'
        Size: 5194
        MaxSize: 5194
        Datatype: H5T_IEEE_F32LE (single)
        ChunkSize: []
        Filters: none
        FillValue: 0.000000
      Dataset 'Hx'
        Size: 5194
        MaxSize: 5194
        Datatype: H5T_IEEE_F32LE (single)
        ChunkSize: []
        Filters: none
        FillValue: 0.000000
      Dataset 'Hy'
        Size: 5194
        MaxSize: 5194
        Datatype: H5T_IEEE_F32LE (single)
        ChunkSize: []
        Filters: none
        FillValue: 0.000000
      Dataset 'Hz'
        Size: 5194
        MaxSize: 5194
        Datatype: H5T_IEEE_F32LE (single)
        ChunkSize: []
        Filters: none
        FillValue: 0.000000
  Group '/srcs'
    Group '/srcs/src1'
      Attributes:
        'Type': 'HertzianDipole'
        'Position': 0.046000 0.060000 0.450000
```

Figure 2.8: Default parameters of gprMax output.

Each A-Scan output file has a size of about 134 KB for a simulation of roughly 5000 iterations, which is typical for the application. That means that a survey of 4000 A-Scans will take up about 530 MB of space.

2.2.2 Metadata to consider

Similar to the surveys preciously described, simulation surveys need to specify metadata to have complete understanding of the surveyed scenario. The identified metadata is the following:

1. **Ground permittivity:** Value of electric permittivity $\varepsilon(f)$ of the ground where the survey is done.
2. **Location of buried objects:** Location (x, y and z) buried objects placed on the survey area.
3. **Description of buried objects:** Full description of objects buried in the survey area (dimensions, materials, descriptive names, etc.).
4. **Rendered 3D view of the simulation scenario**

Chapter 3

Storing data in a merged file

3.1 Merging data

One aspect to take into account when dealing with GPR surveys is the availability of data on a repository. Storing complete information while using the least possible amount of space and the least possible amount of files is important to have an organized measurement repository. A small amount of files with small size means that the data can be downloaded and accessed more effectively.

As mentioned in Chapter 2, there is some minimum data and metadata needed to have a complete description of the surveyed scenarios. Having all the `.csv` or `.out` files, that a GPR survey produces, uploaded into a repository is impractical because of the size, the large amount of files and redundant data within files. Information from all of the A-Scan files can be merged into one single data file before uploading it to a repository. An A-Scan merging script was developed for this purpose. An explanation of the merged file is given in this chapter, including: format of the file, parameters it contains and hierarchical organization.

Examples of merged files are given in order to show how actual merged files look like. It is important to note that the merged files here detailed, do not contain all of the data described in Chapter 2. Metadata is managed in separate files because of format and automation concerns for creating the merged file. When uploading survey data to a repository it is important to upload the metadata files along with the merged files here described.

3.1.1 Merged file format, parameters and hierarchy

The merged survey file compresses all the A-Scans from a survey into one single file. This merged file is formatted in HDF5 format [5]. HDF5 is an easy to read format using the `h5py` library [6] in Python. It features groups, hierarchy, attributes and different types of data, which makes it vary useful for GPR surveys. Because simulated and real data differ in the way some of the parameters are presented, the merged file makes a distinction from which method the data comes from and parameters change from one another. Parameters and hierarchy are shown in Figure 3.1 and Figure 3.2, for real field surveys, and Figure in 3.3, for simulation. The Python code scripted for merging the files can be found at `https://bit.ly/3a3B3gT`.

Note: All of the values in the merged file are given in mks units (meters, seconds and Hertz).

3.1.2 Size advantages

As mentioned previously one of the main advantages of merging files is the size reduction of the data. Table 3.1 shows a size comparison between data stored in individual files against having the same amount of data in a single merged file.

Size advantages are achieved due to removing redundant data among files and due to the usage of a compression standard on the `.h5` files.

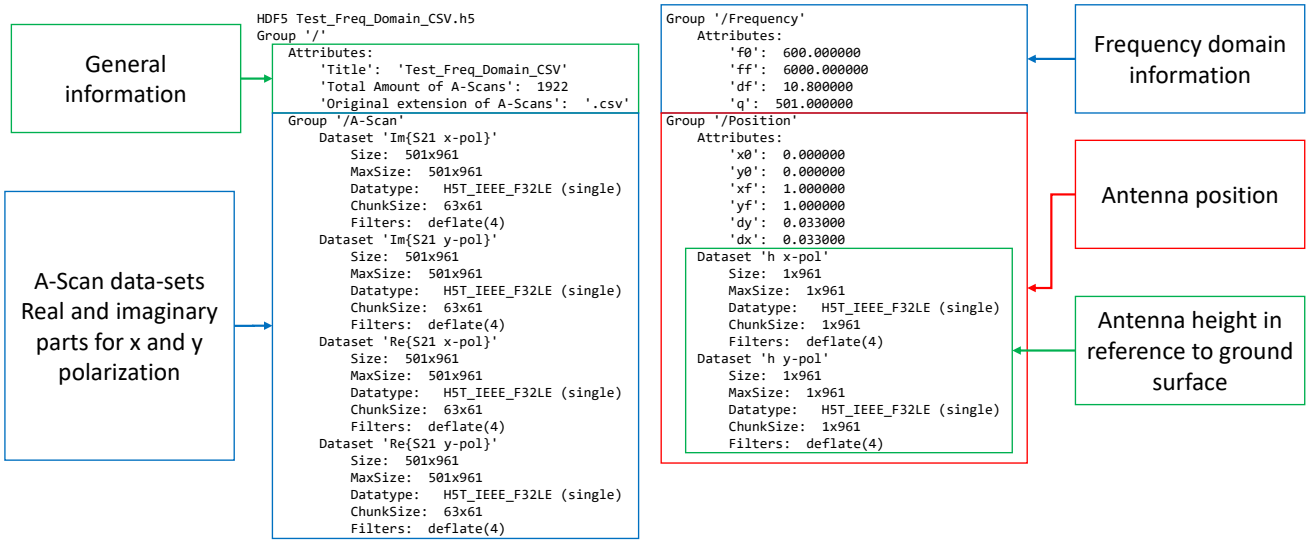


Figure 3.1: Merged file structure of a frequency domain real survey.

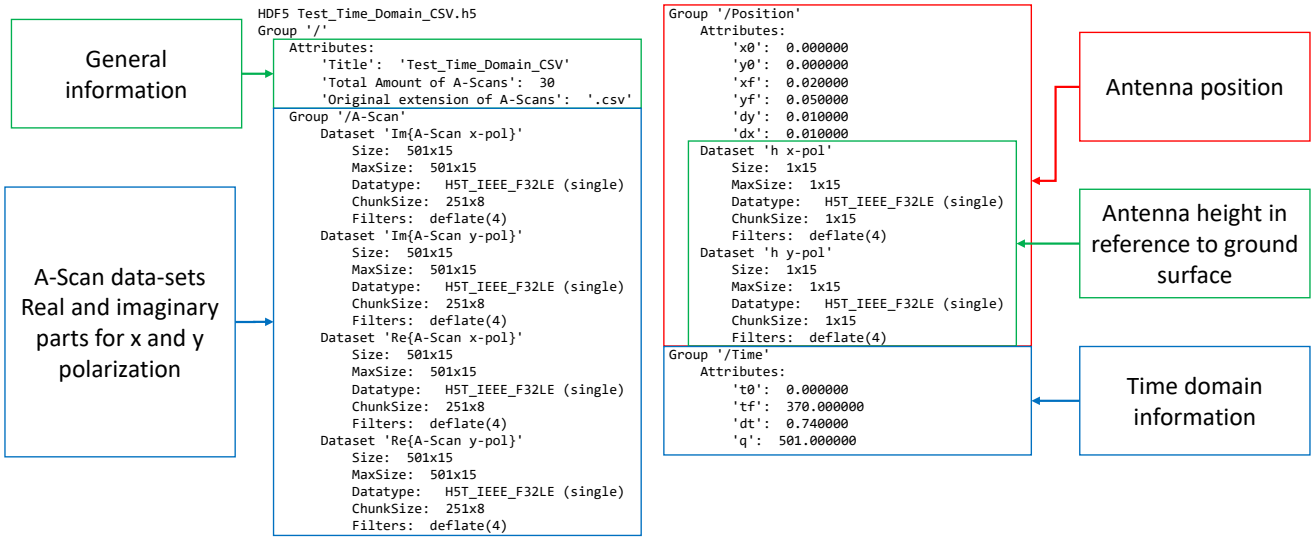


Figure 3.2: Merged file structure of a time domain real survey.

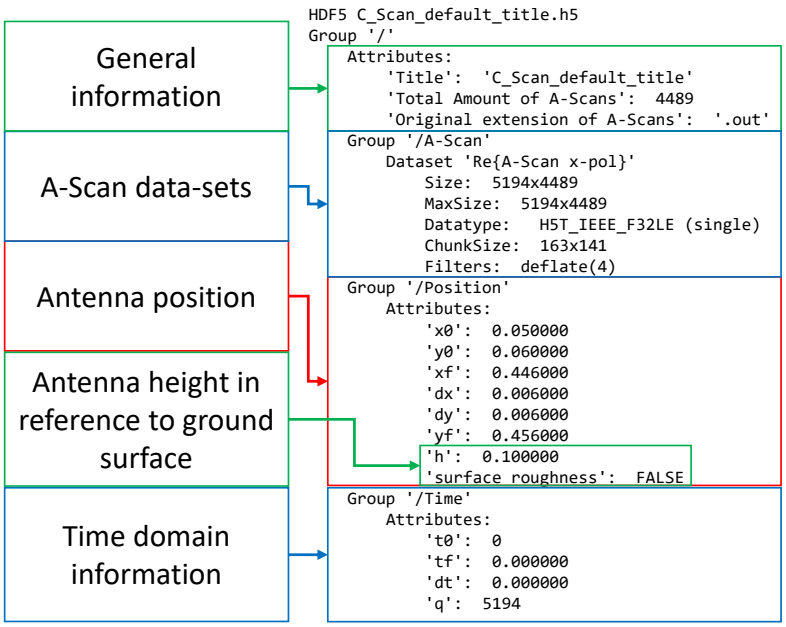


Figure 3.3: Merged file structure of a time domain real survey.

	Size of set of individual files [MB]		**Size of merged file [MB]**		**Size reduction**	
Amount of files	.csv	.out	.csv	.out	.csv	.out
50	0.97	6.64	0.2	0.89	79%	87%
100	1.95	13.28	0.37	1.74	81%	87%
200	3.9	26.56	0.72	3.47	82%	87%
400	7.81	53.12	1.4	6.82	82%	87%
800	15.6	106	2.78	13.6	82%	87%
1000	19.5	132	3.46	17	82%	87%
1500	29.2	199	5.19	25.5	82%	87%
2000	39	265	6.93	33.8	82%	87%
3000	58.5	398	10.3	50.7	82%	87%
4000	78.1	531	13.8	67.6	82%	87%

Table 3.1: Size comparison between individual file set and merged file. Files used to benchmark: `.csv` files made from 501 data-points and `.out` files made from 5192 data-points.

Chapter 4

Reading data stored in a merged file

4.1 Groups, attributes and data-sets in .h5 data files

Because the `.h5` format, data stored in the merged files is coded in a certain way. Reading the data requires an `.h5` interpreter for which either Python or Matlab can be used (among others).

One of the most important things to see from the file is which parameters are stored. Parameters stored in `.h5` files are classified into: Groups, Attributes and Data-sets. Group indicates the category under which attributes and data-sets are classified. Attributes are parameters that take a single value; it can be a boolean, integer, float or string. And data-sets are parameters that store sets of data of different kind, usually floats.

An important resource for `.h5` files is to show the hierarchical structure of data from a file, similar to what is presented in Figures 3.1, 3.2 and 3.3. It can be seen from these examples that data is arranged under certain group and classified as data-set or attribute according to its properties.

The most straightforward way to do visualize the structure of a `.h5` file is using the function `h5disp()` from Matlab, which takes as argument the name of the file to view. The file structure can also be viewed using `h5py` library, iterating through the different groups with functions `h5py.keys()` and `h5py.visit()`. Implementation of the Matlab method is given below:

```
filename = 'D:/C_Scans/C_Scan_title.h5'
h5disp(filename)
```

4.1.1 Reading attributes and data-sets

Data from the `.h5` files must be retrieved in order for it to be plotted and processed. Therefore is important to know how to read and retrieve information from attributes and data-sets. Matlab functions `h5readatt()` and `h5read()` are used to retrieve attributes and data-sets respectively; these functions take as parameters the file name and the route within file hierarchy for the parameter to be read. Reading the file in Python requires to open the file first and then reading the parameters, calling the opened file with the route within the file hierarchy for data-sets and with the method `.attrs` and the route for the attributes; an example for the Python procedure is given below (Figure 3.2 used as file):

```
import h5py
filename = 'D:/C_Scans/Test_Time_Domain_CSV.h5'
# File is opened in read ('r') mode:
file = h5py.File(filename, 'r')
# Data-set is retrieved
a_scan_im_x_pol = file['A-Scan/Im{A-Scan x-pol}']
# Attribute is retrieved
x0 = file['Position'].attrs['x0']
# Print for the first A-Scan stored in the data-set:
print(a_scan_im_x_pol[:,0])
```

4.2 Plotting data in B-Scan plots

A useful way to display A-Scan data is using the B-Scan plots. B-Scans are ensembles of A-Scan waveforms over a plane ($x-y$, $x-z$, or $y-z$). Usually B-scans given are for either the $x-z$ plane or the $y-z$ plane. The coordinate system often follows a typical convention of GPR surveys and simulations where the $z-axis$ is the axis perpendicular to the ground surface, and $x-axis$ and $y-axis$ are parallel to the ground surface.

The $x-axis$ and $y-axis$ information of GPR surveys is collected by the position controlling system, this is represented by the x and y data given in the attributes of the merged file. The $z-axis$ information is given by the time-domain parameters. Time can be related to the $z-axis$ because the propagation velocity (v_p) of the electromagnetic (EM) waves is known therefore: $z = v_p \cdot t$. Because there is an A-Scan for each $x-y$ pair and each A-Scan gives an amplitude as a function of time, amplitudes can be plotted as a function of space.

Because the B-Scan plot is the ensemble of A-Scan waveforms over a plane, amplitudes in a B-Scan are plotted as a function of the different positions over that plane. This produces a 3-D plot where the plane (e.g. $x-z$) is represented by two axes and the amplitude is represented by a color of a colormap. Figure 4.1(a) shows a B-Scan for a survey done over a sphere buried in ground. This figure shows that B-Scans give more detailed information than an A-Scan to identify the objects that scatter the incident fields. Also from Figure 4.1(a) one can identify a hyperbolic pattern which is a representation of the scattered field in each position of space. A migration method can make the hyperbola tails converge and show a more accurate representation of the objects buried inside the ground, this method will be presented later on this document.

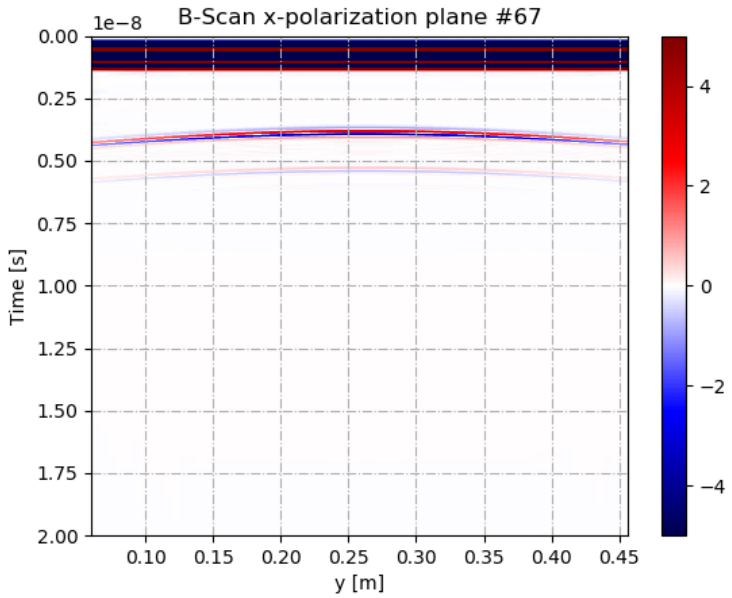
B-Scans can be plotted from the merged file using either Matlab or Python with different plotting functions. The Python code scripted for plotting B-Scans can be found at `https://bit.ly/33UdohB`. This code uses libraries `h5py` and `matplotlib`, to open the merged file and to plot the data.

4.3 Plotting data in C-Scan plots

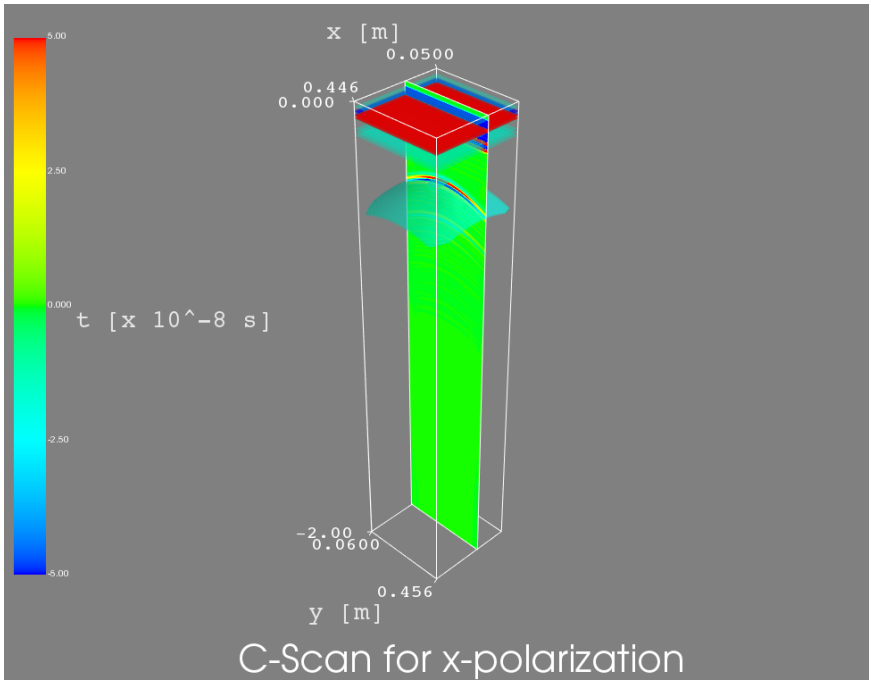
A-Scan data can also be displayed using C-Scan plots. C-Scans are ensembles of A-Scan waveforms over a volume ($x-y-z$). The building blocks for a C-Scan are the same presented previously for the B-Scan, but here amplitudes are plotted as a function of different positions over the volume. C-Scans are 4-D plots where the volume ($x-y-z$) is represented by three axes and the amplitude is represented by a color of a colormap. Figure 4.1(b) shows a C-Scan for a survey done over a sphere buried in ground. An special consideration for C-Scans is the alpha (transparency) parameter for the amplitude colors, where colors that represent lower amplitudes (near zero) are shown transparent. This plots also show hyperbolic patterns that can be focused via migration.

C-Scans can be plotted from the merged file using either Matlab or Python with different plotting functions. The Python code scripted for plotting C-Scans can be found at `https://bit.ly/2DXsF6G`. This code uses libraries `h5py`, `numpy` and `mayavi`, to open the merged file and to plot the data.

Is important to note that many times B-Scans and C-Scans are plotted as function of time instead of calculating z. This is the case for Figure 4.1.



(a) B-Scan over middle plane of a buried sphere survey.



(b) C-Scan of a buried sphere survey.

Figure 4.1: B- and C-scans of GPR survey over a buried sphere.

Chapter 5

Data pre-processing and processing

The general objective of signal processing is to obtain an image that can be interpreted properly so the targets can be located. This chapter explains some techniques used for this purpose. Before entering into details regarding pre-processing and processing, the distinction between both must be made. In this document, pre-processing refers to the techniques used to give the processing algorithm an understandable signal. And, processing refers to the migration algorithms that focus the GPR hyperbolas into points of scattering.

5.1 Data pre-processing

Raw GPR signals usually contain a lot of undesired reflections. These reflections make target detection difficult. Pre-processing techniques are different methods that can be employed to eliminate undesired reflections from the signal of interest. Figure 5.1 shows a simplified block diagram of how the GPR signals propagate. It is shown that different sources of undesired reflections are present in a GPR survey. The following are the pre-processing techniques implemented so far, for the GPR-20 project: subtraction of antenna coupling, background removal, and a special case for simulated scenarios.

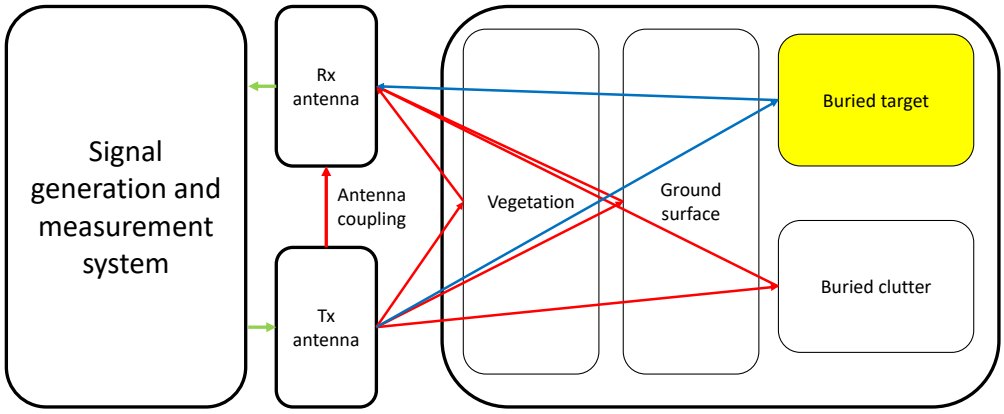


Figure 5.1: Simplified block diagram for the GPR signal path. Red arrows represent undesired reflections that are measured by the GPR. Blue arrows represent the target scattering signal path.

5.1.1 Subtraction of antenna coupling signal

Antenna coupling is referred to as the signal received by the receiving (Rx) antenna directly from the transmitting (Tx) antenna. Because both Tx and Rx are placed close together and because of the radiation pattern of the antennas, there is a direct signal path between both of them. The signal recorder from this direct path adds up to the received signal thus adding noise to the signal from which the target must be recovered.

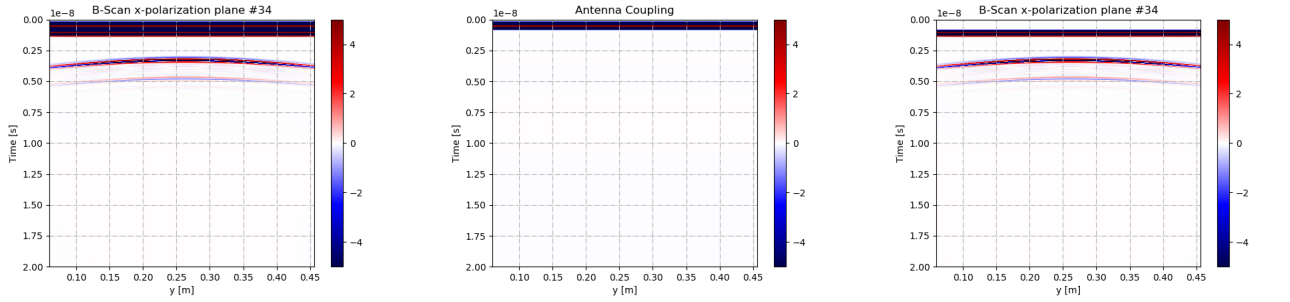
The antenna coupling signal can be subtracted, from the total reflected signal, if one has it characterized. To characterize the coupling signal, the antenna pair can be placed in an an anechoic chamber or an open-field (looking up) using the same placement that will be used in the GPR. The recorder response for the mentioned scenario will contain the antenna coupling without any other signal. Therefore the recorded response can be subtracted from all of the GPR measurements, and those measurements will then be free from the antenna coupling response.

Coupling signal subtraction can be done in the time- or the frequency-domain and the same result shall be obtained. It is more convenient to do it in the original measurement domain, in order to avoid introducing unnecessary numerical procedures. For the real GPR surveys, the subtraction is done in the frequency domain (Equation (5.1)). For the simulated GPR surveys, the subtraction is done in the time domain (Equation (5.2)).

$$S_{21}^{coupling\ free}(f) = S_{21}^{GPR}(f) - S_{21}^{antennas}(f) \tag{5.1}$$

$$h^{coupling\ free}(t) = h^{GPR}(t) - h^{antennas}(t) \tag{5.2}$$

Effect of antenna coupling subtraction can be seen with the B-Scans in Figure 5.2.



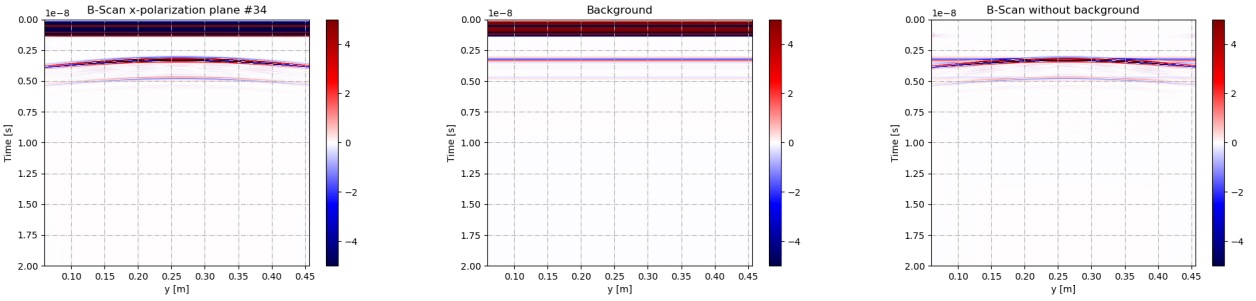
(a) Raw B-Scan. (b) Antenna coupling. (c) B-Scan without antenna coupling.

Figure 5.2: Antenna coupling in B-Scans.

5.1.2 Background removal

GPR surveys are affected by noise sources that are present throughout the whole survey, such as: antenna coupling, vegetation reflections and ground surface reflections. All these sources of noise are considered background signals because they are present in every A-Scan of the survey. Background signals are usually displayed as horizontal lines in the B-Scan and horizontal planes in the C-Scan.

Background noise signals can be removed, from the signal received by the GPR, by taking the average of all A-Scans over space and subtracting it from the original A-Scans. A graphical representation of background removal is shown in Figure 5.3. All of the horizontal lines at the top from Figure 5.3(a), which correspond to antenna coupling and ground surface reflections, are removed by subtracting the spatial average and the B-Scan in Figure 5.3(c) is obtained.



(a) Raw B-Scan. (b) Background signal. (c) B-Scan without background.

Figure 5.3: Background removal in B-Scans.

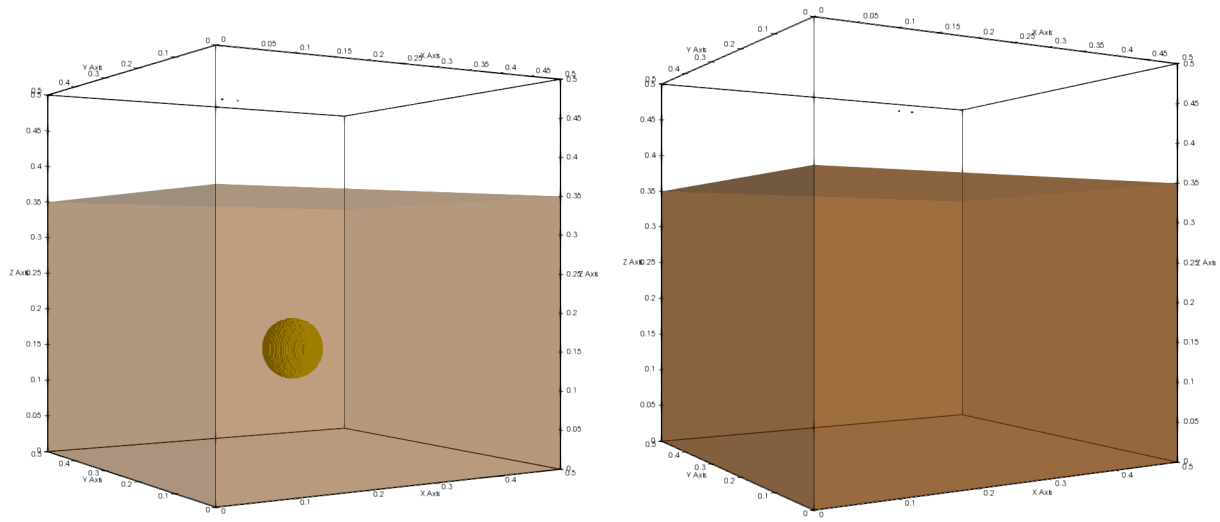
5.1.3 An special simulation case: create an scenario without the target

Simulation enables having a detailed and tuned configuration of scenarios. When simulating surveys the desired scenario can be created completely (Figure 5.4(a)) and then the target can be removed from the scenario (Figure 5.4(b)). If both of these scenarios (with and without the target) are simulated, then the noise signal can be determined for every A-Scan in the survey (noise signal being the result of the target-less scenario). The resulting noise signal can be then subtracted from the signal of the scenario with the target, resulting in noise-free A-Scans. Results of this procedure are shown in Figure 5.5.

5.2 Data processing: Kirchhoff Migration

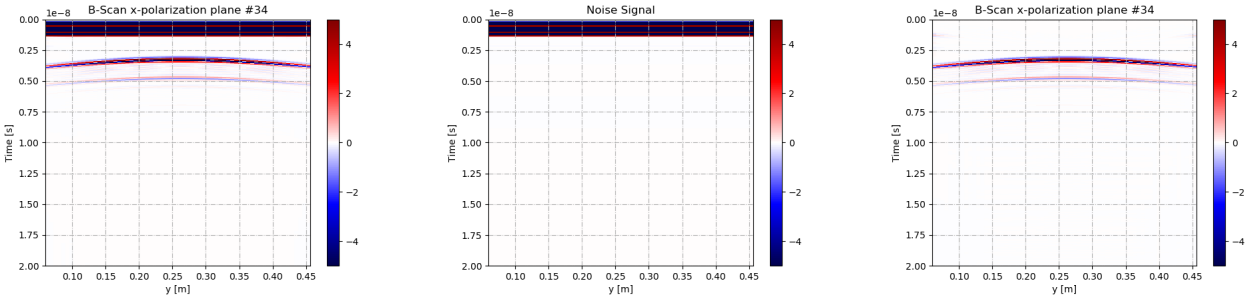
Once noise and clutter is reduced from the original A-Scans by pre-processing techniques, the resulting signals can be processed. This process is also known as imaging or migration, which focus the energy values of the original signals into discrete points. There are many migration algorithms that can be used to process the signals [7]. The migration algorithm here presented is the Kirchhoff migration algorithm [8, 9]. Migration algorithms are a tool used to solve the GPR inverse problem, which means that the staring point for the algorithm is the electric field measured by the radar and the result of the algorithm is a the geometry/material reconstruction.

The basic idea behind the Kirchhoff migration algorithm is to trace a hyperbola for each point of the subsurface in the scenario. The point for which the hyperbola is being traced, is considered as the hyperbola apex.The traced hyperbola is then compared against the energy of the actual signals that intersects it. All the energy over the traced hyperbola is then summed and placed in the apex point. How the hyperbola spread will be dependant of the propagation velocity of the media in the ground subsurface and the antenna positioning above the ground surface. This basic idea presented here is the underlying concept for 2D- Kirchhoff migration; for 3D Kirchhoff migration, the same concepts apply but instead of using hyperbolas for each point, hyperboliods shall be used. This concept is represented by Equation (5.3) for the 3D case and Equation (5.4) for the 2D case. Where P_{out} is the energy



(a) Scenario with target. (b) Scenario without target.

Figure 5.4: Simulated scenarios used for noise removal.



(a) Raw B-Scan. (b) Noise signal. (c) B-Scan without noise.

Figure 5.5: Noise removal for simulation cases in B-Scans.

in the migrated point (x_{out}, y_{out}, z), $theta_g$ is the angle between the signal ray going towards the migration point $(x_{in}, y_{in}, t = t_{exp})$ and a line perpendicular to the ground surface (vertical line), r is the total length of the signal ray from the antennas to the migration point, P_{in} is the input energy for the migration at the migration point, and t_{exp} is the time where the migration point is expected to be found over an A-Scan for the antennas placed at all the possible positions.

$$P_{out}\left(x_{out}, y_{out}, z\right) = \frac{1}{2\pi} \int \int \left[\frac{\cos\theta_g}{r} \frac{\partial}{\partial t} P_{in}\left(x_{in}, y_{in}, t = t_{exp}\right) \right] dx \; dy \tag{5.3}$$

$$P_{out}\left(x_{out}, z\right) = \frac{1}{2\pi} \int \left[\frac{\cos\theta_g}{\sqrt{r}} \frac{\partial}{\partial t} P_{in}\left(x_{in}, t = t_{exp}\right) \right] dx \tag{5.4}$$

The migrated image obtained by migrating the C-Scan from Figure 6.2(a) is presented in Figure 6.2(d). The migrated image in Figure 6.2(d) shows only the the image from the ground surface down. Measuring the distances from the migrated image and comparing them migrated image with the actual detection scenario (Figure 6.2), it can be seen that the depth and size of the buried object is accurately resolved.

5.2.1 Considerations and requirements for Kirchhoff Migration

An special consideration to take into account for the Kirchhoff migration in landmine detection is that antennas will be placed above the ground at a certain height (h_{ant}). One of the limitations of the migration algorithm developed for this document is that it assumes that h_{ant} is constant for all the antenna positions.

For common geophysical and GPR Kirchhoff migration, antennas are usually at ground level, therefore the path for the propagating electromagnetic fields show no diffraction from the ground surface. For landmine detection there is diffraction present at the ground surface level, meaning that the angle of the ray will change at this interface. This consideration accounts for a more complex procedure to calculate the ray length between the antennas and the migration point (r), the expected time of reflection arrival (t_{exp}) and the angles of propagation in air (θ_a) and in ground (θ_g). These angles are calculated for each combination of migration point and antenna position. The system of equations used to calculate the angles is described by Equations 5.5, where (x_p, y_p, z_p) are the coordinates of the migration point and (x_a, y_a) are the coordinates of the antenna. The ray length between antennas and migration point is calculated using Equation (5.6). And the expected time of reflection arrival is calculated using Equation (5.7), where v_p is the propagation velocity of the waves in ground and c_0 is the propagation velocity of waves in air.

$$\sqrt{(x_p - x_a)^2 + (y_p + y_a)^2} - z_p \cdot \tan\theta_g - h_{ant} \cdot \tan\left[\arcsin\left(\sqrt{\varepsilon_r} \cdot \sin\theta_g\right)\right] = 0 \tag{5.5a}$$

$$\theta_a = \arcsin\left(\sqrt{\varepsilon_r} \cdot \sin\theta_g\right) \tag{5.5b}$$

$$r = r_g + r_a = \frac{z_p}{\cos\theta_g} + \frac{h_{ant}}{\cos\theta_a} \tag{5.6}$$

$$t_{exp} = t_g + t_a = \frac{z_p}{v_p \cos\theta_g} + \frac{h_{ant}}{c_0 \cos\theta_a} \tag{5.7}$$

Propagation velocity will change significantly at the ground surface interface. Ground apparent relative permittivity is typically between 2 and 16 ($\varepsilon_r \in [2, 16]$), and propagation velocity is inversely proportional to the square root of relative permittivity ($v_p = c_0/\sqrt{\varepsilon_r}$). That is why the air trajectory must be calculated using c_0 and the ground trajectory must be calculated using v_p.

Another consideration for Kirchhoff migration is the possibility of calculating full 3D migration, using Equation (5.3), or using 2D migration for different B-Scans. 2D Kirchhoff migration is described by Equation (5.4). The main advantage of 2D Kirchhoff migration over 3D is the processing time it takes to fully migrate several B-Scans, over the processing time for 3D migration. This performance difference is based on the fact that the angles are calculated for antenna position and migration point combinations. Reducing by one dimension the antenna and points possible locations can significantly improve the amount of computations to be made. Still a 3D image can be obtained from 2D migration by staggering the resulting migrations of the B-Scans, this is known as 2.5D migration.

5.2.2 Algorithm overview

Figure 5.6 shows a flowchart of the algorithm implemented to obtain the migrated image from B- and C-Scans. The Python code written can be found at `https://bit.ly/35py3Li` and the Python code for plotting the migrated images can be found at `https://bit.ly/2GRLAkR`.

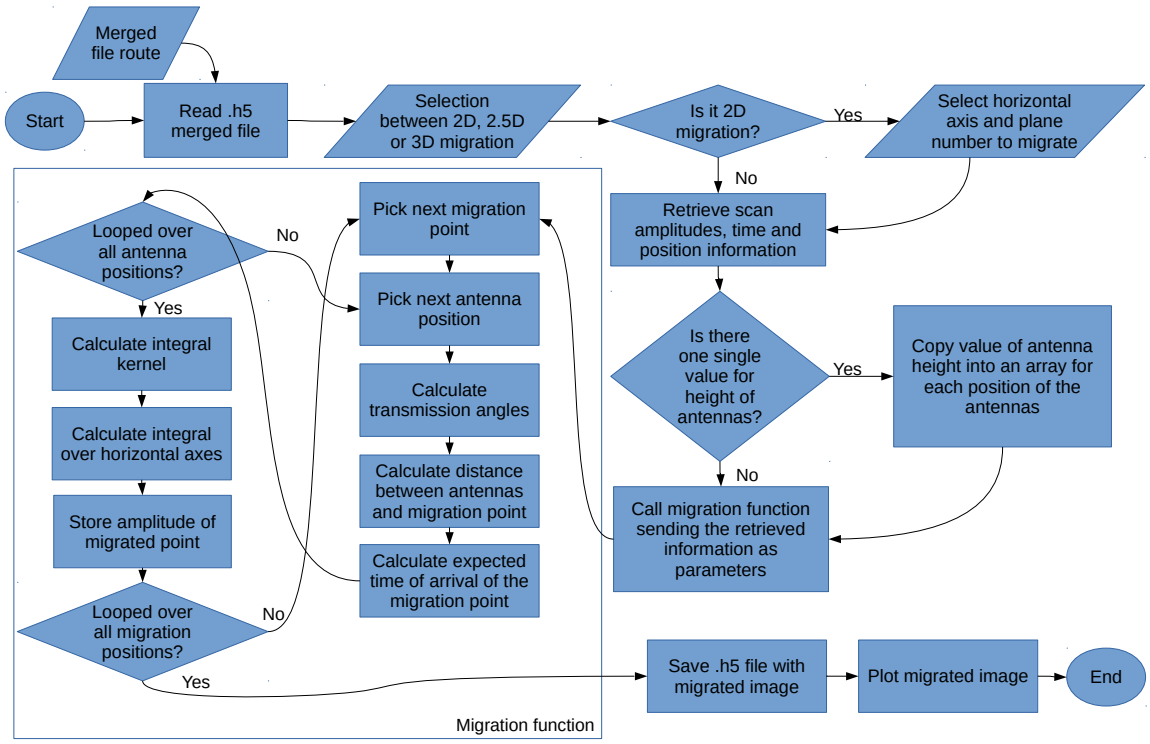
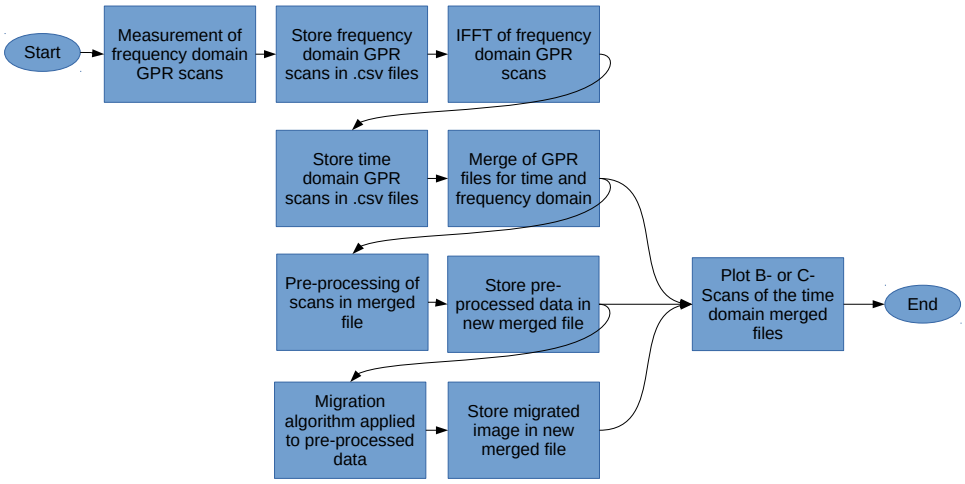


Figure 5.6: Flowchart of the Kirchhoff migration algorithm.

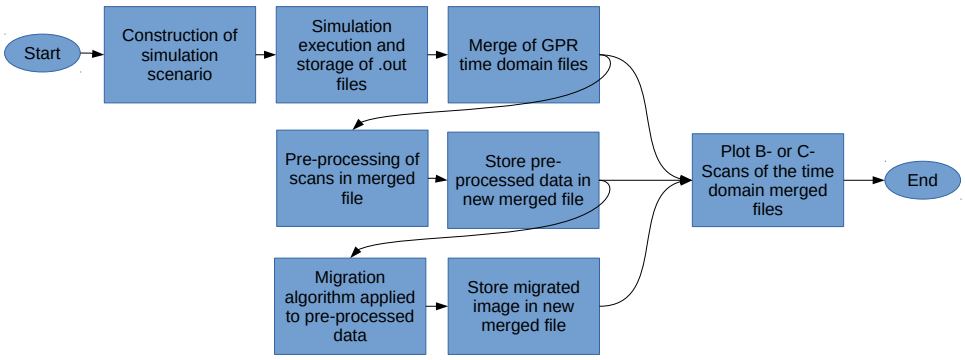
Chapter 6

Summary of the data workflow and examples

Figure 6.1(a) and Figure 6.1(b), show the data workflow for measurements made with the GPR-20 and for simulation data. Python code for all the elements of the workflow can be found at `https://bit.ly/3bLI4DZ`.



(a) GPR-20 data.

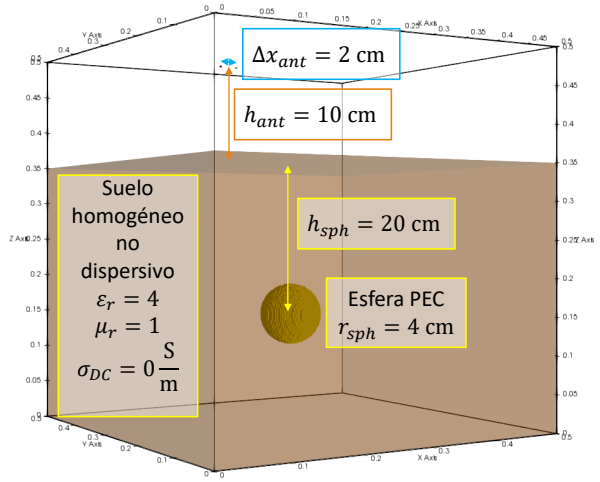


(b) Simulation data.

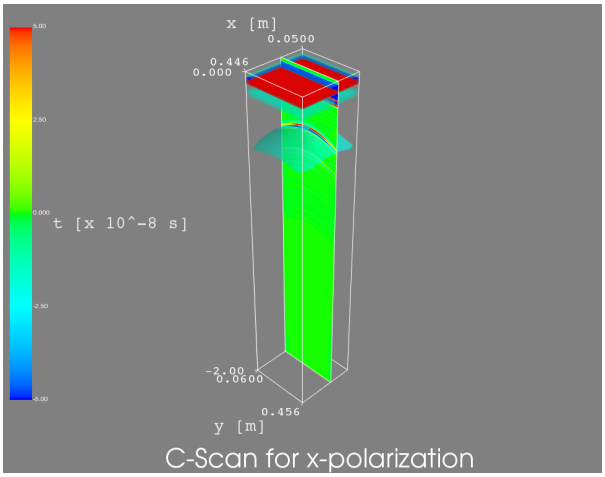
Figure 6.1: Workflows for GPR data.

Figure 6.1(b) shows the different stages of the data management procedure for a gprMax simulated scenario. It starts with the description of the actual scenario and ends with the migrated image obtained from the Kirchhoff migration.

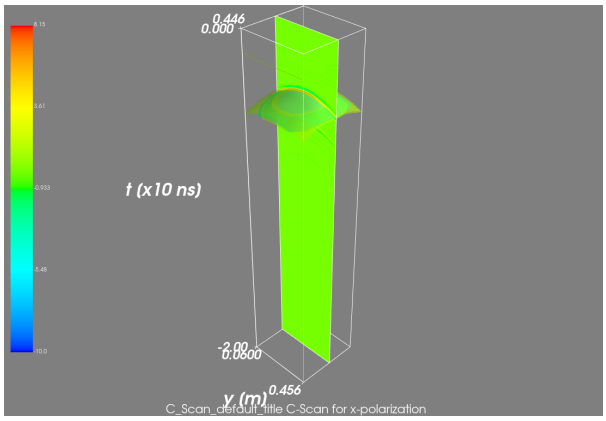
Figure 6.2 shows an example for the whole data workflow for a simulated survey. Data for this example can be found at `https://bit.ly/2FgmcEM`.



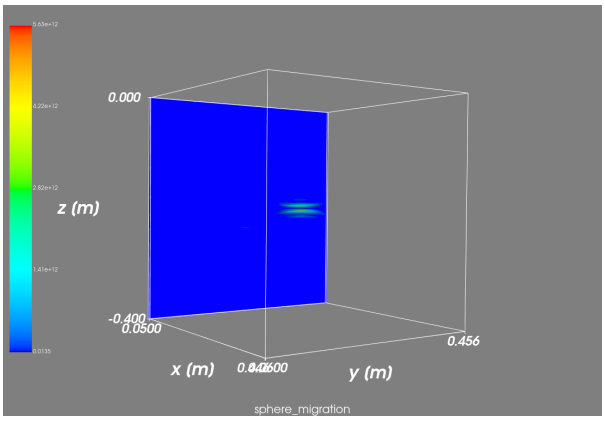
(a) Acquire GPR scans in scenario.



(b) Raw C-Scan.



(c) Pre-processed C-Scan.



(d) Migrated image.

Figure 6.2: Example of different stages of data.

Bibliography

[1] Anritsu, *User Guide: VNA Master MS20xxC*, 8 2018. Rev. M.

[2] E. O. Brigham, *The Fast Fourier Transform and its Applications*. Prentice Hall, 1988.

[3] C. Warren, A. Giannopoulos, and I. Giannakis, "gprMax: Open source software to simulate electromagnetic wave propagation for Ground Penetrating Radar," *Computer Physics Communications*, vol. 209, pp. 163–170, 2016.

[4] "gprmax user guide." `http://docs.gprmax.com/en/latest/`. Accessed: 2020-07-30.

[5] "Hdf5." `https://portal.hdfgroup.org/display/HDF5/HDF5`. Accessed: 2020-07-30.

[6] "h5py." `https://www.h5py.org/`. Accessed: 2020-07-30.

[7] C. Ozdemir, S. Demirci, E. Yigit, and B. Yilmaz, "A review on migration methods in b-scan ground penetrating radar imaging," *Mathematical Problems in Engineering*, vol. 2014, 2014.

[8] O. Yilmaz, *Seismic data analysis*. Society of exploration geophysicist, 2001.

[9] X. Feng and M. Sato, "Pre-stack migration applied to GPR for landmine detection," *Inverse Problems*, vol. 20, no. 6, 2004.

Appendix A

Change height of antennas in merged file

When making measurements with Arduino based GPR (portable and laboratory), the antenna height parameter has to be changed at the source code. Height may vary from measurement to measurement, so it is easy to forget to change the height in the source code beforehand.

When measurements are taken with a wrongly reported antenna height, a parameter from the merged file of those measurements can be modified in order to store the proper antenna height. This procedure is done through a Python script, where the proper height and the merged file to modify are reported. The script reads the file using the `h5py` library, changes the height, and stores it in a new merged file with the correct height parameter.

Script can be found at: `https://bit.ly/3zkAsDw`

Appendix B

Links to Python scripts

GitLab repository: `https://gitlab.com/desminadohumanitariouniandes/gpr-20-data-processing/-/tree/master/scripts`

OneDrive SharePoint: `https://bit.ly/3vfE2M7`

IFFT script: `https://bit.ly/3zoQ09E`

- **gpr20_ifft.py:** Contains the IFFT algorithm, examples of the IFFT, and the script for IFFT execution.
- **ifft_multiple_files.py:** Contains the script to execute the IFFT algorithm on multiple files (GPR measurements). Changing the folder for which files will be processed by the IFFT algorithm.

Measurement files merging script: `https://bit.ly/2RPDi2G`

Pre-processing and processing scripts: `https://bit.ly/3znIyLA`

- **c_scan_background_removal.py:** Contains script for removing the positional average from a merged file.
- **c_scan_background_subtraction.py:** Contains script for removing the result of a merged file from another merged file.
- **parallel_kirchhoff_migration.py:** Contains the script and algorithm for executing the Kirchhoff migration on a merged C-Scan.

Plotting scripts: `https://bit.ly/3cAMajX`

- **b_scan_plot.py:** Contains script for plotting the B-Scans of a merged file. It can plot a single B-Scan or it can plot all the possible B-Scans in one movement direction.
- **c_scan_plot.py:** Contains script for plotting the C-Scan of a merged file.
- **migrated_image_plot.py:** Contains script for plotting the migrated image stored in a merged file.