אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev
School of Electrical & Computer Engineering

WiFi Based Wireless Imaging and Positioning for WSN

371-20-06

Students:
Mr. Oren Zaharia
Mr. Efi Dvir

Supervisor:
Prof. Omer Gurewitz

31/08/2020

Preface

Acknowledgements

First and foremost we would like to express our deepest gratitude and appreciation to our supervisor, Prof. Omer Gurewitz. whom above of being our adroit guide in this academic endeavor is also our friend. With our utmost respect, we couldn't have done it without you and your support. Thank you!

We would also like to thank the entire communication system engineering department teachers, staff and fellow students for a thrilling, incredibly hard yet satisfying 4 years of fun and adventure of a B.Sc diploma.

We would like to thank our families who supported us in every possible way. Your support allowed us to achieve such grate heights.

We would also each like to thank one another for the true friendship and full commitment to the success of one another and this project. It's been a long ride, but every step was awesome. Thank you my friend!

Oren wants to thanks to Onyx, that gave him moments of Atnahta (break).

About us

Mr. Oren Zaharia (28), a senior student of the Communication Systems Engineering program at the Ben-Gurion University of The Negev, Israel. Graduated practical engineers in electrical and computer engineering with communication systems major. A 4 year 8200 alumni, experienced with RF communication systems, signal engineering, and embedded programming. Among his hobbies are: Programming, bicycling, hiking, soccer, Beitar Jerusalem, history, and more.

Mr. Efi Dvir (30), a senior student of the Communication Systems Engineering program at the Ben-Gurion University of The Negev, Israel. Graduated practical engineers in electrical and computer engineering with communication systems major. A 4.5 year 8200 alumni, experienced with RF communication systems, signal engineering, and embedded programming. Among his hobbies are: IoT sensors projects, painting, cooking, music, general craftsmanship, and more.

During their B.Sc studies, Mr. Efi Dvir and Mr. Oren Zaharia attended courses such as Signals and Systems, DSP, Wireless Networks Communication, and more.

Together, Mr. Efi Dvir and Mr. Oren Zaharia are brewing beer in their free time as well as sometimes enjoying a nice cold brew at the local pub.

About this report

This report describes Mr. Efi Dvir's and Mr. Oren Zaharia's progress in their Communications Systems Engineering program's final engineering project. Throughout this document, the reader might find references to articles and books who were used as an inspiration and technical basis to our project and are not Mr. Dvir and Mr. Zaharia's implementations. This report has been made using LaTeX.

Intended audience

The report is written for the academic community that want to review and criticize our work or to be inspired or implement and learn our work. The project in a range of final project implementations from simple ideas implementations to complex ideas implementations. The report assumed that the reader has some experience and knowledge in physics, wireless sensors networks, radiofrequency, electrical wave propagation, and wireless communications. The report does not assume the experience of radar techniques, our tools, or any concrete knowledge in our implementation.

About this project

This project is Mr. Efi Dvir's and Mr. Oren Zaharia's final engineering project as a partial requirement of the B.Sc of Communications Systems Engineering degree. This project suggests a method to infer knowledge of a sensor's surrounding environment using WiFi based wireless signals and their propagation properties. By using several methods to obtain information form WiFi signals this project strives to be able to classify the sensor's location and purpose (Temperature, pressure, humidity sensor which is located in a greenhouse in the backyard), in order to be able to use this information to configure the sensor as a part of a whole sensor information network. The project includes both a theoretical algorithmic view as well as practical hardware implementation using SDRs.

Table of Contents

List of Figures

List of Tables

Glossary

802.11ac IEEE 802.11ac is a wireless networking standard in the 802.11 set of protocols. 25

BTS Base transceiver station. 10

EMI Electromagnetic interference. 33, 34

FPGA Field-programmable gate array. 24

Git Distributed version control. 25

GNU An extensive collection of free computer software licensed with GPL. 7, 8, 10, 25, 38–40, 46–48

GPL The GNU General Public License. 10

GPS Global Positioning System. 20

GSM Global System for Mobile Communications known as 2nd generation of the cellular. 10, 25

IoT Internet of Things. 2, 13, 18, 64

MIMO Multiple Input Multiple Output. 7, 14, 18, 19, 36

OpenBTS Open BTS is a software-based GSM. 25

PCB Printed circuit board. 25

PDP Power Delay Profile. 7, 15, 16

PHY Physical Layer. 20

RCS Radar Cross Section. 14

RFIC Radio-frequency integrated circuit. 24

RMA Range Migration Algorithm. 7, 8, 19, 20, 40, 42, 44–46, 48, 55, 57, 58

RSSI Received Signal Strength Indication. 5, 7, 8, 15, 20, 21, 36, 49, 51–54

SDR Software Defined Radio. 3, 7, 24, 36, 38–42, 48

USB Universal Serial Bus. 25

USRP Universal Software Radio Peripheral. 24, 25, 48, 58

UWB Ultra Wide Band. 33

WiFi Wirless Fidelity is a family of wireless networking technologies. 3, 15, 19–23, 25, 30, 33, 36, 51, 54, 63, 64

WSN Wireless Sensor Network. 64

Acronyms

AP Access Point. 5, 7, 20, 21, 30, 49, 51, 52

CW Continues wave. 5, 40, 44, 55, 64

DC Direct current. 42, 43

EM Electromagnetic. 25

IDE Integrated development environment. 26

ITU International Telecommunication Union. 51

PoC Proof of Concept. 35

RF Radio Frequency. 2, 7, 25, 33, 34, 36, 42, 58

SSID Service Set Identifier. 5, 30, 49, 52, 54

Chapter 1

Introduction

1.1 Background

Smart environment is a relatively new concept, emerging in the early 1990s, where urban residents are interacting on a constant basis with informative objects, devices, sensors, and actuators to seamlessly better their lives. These collect information and process it in order to provide intelligent insights to the end-user and assist him in his daily routine. We treat a smart environment as an intelligent agent that perceives the state of the resident and the physical surroundings using sensors and acts on the environment using controllers in such a way that the specified performance measure is optimized [CD04]. Today, the number of sensors that monitor our environment is in persistent incline. With the entry of the Internet of Things (IoT) to the common household and workplaces, it is becoming more and more demanding for the user administrator to manage the rising number of sensors and actuators under his responsibility. The total installed base of IoT connected devices is projected to amount to 75.44 billion worldwide by 2025, a fivefold increase in ten years. The IoT, enabled by the already ubiquitous Internet technology, is the next major step in delivering the Internet's promise of making the world a connected place [Dep16].

1.2 Motivation

With each new sensor added to the sensor network, there is a need to configure its preferences and functions in order to comply with the network's rules and order. Whereas a new sensor placed in its place usually needs to be manually configured with the information of its location and purpose. For example, when placing a smoke sensor in the kitchen, the sensor needs to be manually configured as the kitchen sensor in the user's system monitoring the smoke levels above the stove. If this sensor has several abilities (besides smoke detection), each with its own data feed to the information system, it may be bothersome and tedious to deploy many of these data-gathering components of the network without some

sort of automation. While exists some algorithms to organize the structure of the network's traffic flow, there is a clear lack of methods to relieve the end-user of the task of configuring each sensor in its network manually. With our knowledge and passion for wireless communication, the thought of trying to engage the wireless capabilities, commonly found in many sensors, in order to assist in the configuration tasks of the user came to mind. By giving the sensor the ability to recognize and analyze its surroundings, a large amount of information could be gathered, organized, and used in order to help reach an assumption on what configuration profile may be best suited for a sensor. Thus, helping the user, or even relieving him entirely of the task of sensor configuration.

1.3 Goals

The Primary goal of our project is to gather new side information about the structure and composition of the device's environment using the propagation and reflection properties of wireless signals already in use in basic wireless devices. This information would be useful to assist in the classification of the room, the operations intended for the sensor, and general informative data that would better with the task of configuring the device's function as part of the whole information network.

The Secondary goal of our project is to use the gathered information to deduct insights regarding the composition of the device's environment. By using machine learning tools and algorithms we would be able to classify the data that in turn will lead to the selection of a suitable configuration profile to set to the device itself. Thus, the gathered information will be used to configure the device in the user interface.

1.4 Techniques

There are several techniques in our disposal to obtain our primary goal:

Radio Frequency Imaging Using the transmission of wireless signals via MIMO antenna array in order to compose a 2D or 3D picture of the surroundings from the reflecting signals. By applying image recognition algorithms on the generated image, we would obtain environmental data to feed to the classifier.

Radar A detection system that uses radio waves to determine the range, angle, or velocity of objects. The range and angle of the device in relation to discovered objects in the vicinity would generate data. Each object discovered would carry a RCS that will be used as environmental data and feed to the classifier.

Indoor Localization Using data from the communications protocols used by the device (RSSI for example in figure 1.4.1), we would perform location triangulation survey in relation to known sources (WiFi routers for example). This location estimation would be feed to the classifier to classify in which room the device is located. We have already examine this option in our work-space at room 512/37 as seen in figure 5.1.3, where is exactly described in figure 1.4.1, where each letter describes the anchor name.

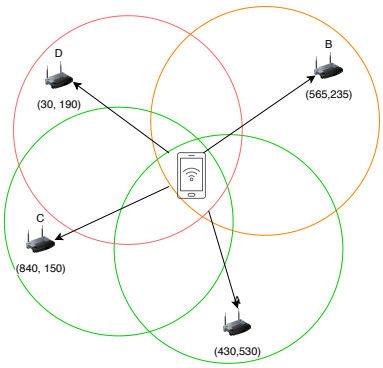


Figure 1.4.1: RSSI triangulation example

Radiation Holography Wireless data transmission systems such as WiFi or Bluetooth emit coherent light – electromagnetic waves with precisely known amplitude and phase. Propagating in space, this radiation forms a hologram – a two-dimensional wave-front encoding a three-dimensional view of all objects traversed by the light beam [HR17]. Holographic data would be feed to the classifier to classify what are the objects in the room along to the room structure and size. Figure 1.4.2 describes a holographic imaging process that generates 3D images using the microwave radiation of a WiFi transmitter experimented in [HR16] "Holography of WiFi radiation". A space with a transmitter on the left and the resulting holographic images on the right.

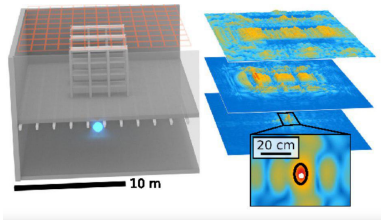


Figure 1.4.2: Holograph example

Power Delay Profile Decomposition The PDP gives the intensity of a signal received through a multipath channel as a function of time delay. Each PDP

is unique and represents the channel time response. By decomposing individual signal paths and using statistical analysis we would be able to generate data to feed to a learning machine. The figure 1.4.3 gives an example to a power delay profile, where the axis time may be in nano-seconds or coarser, the power is in dB.

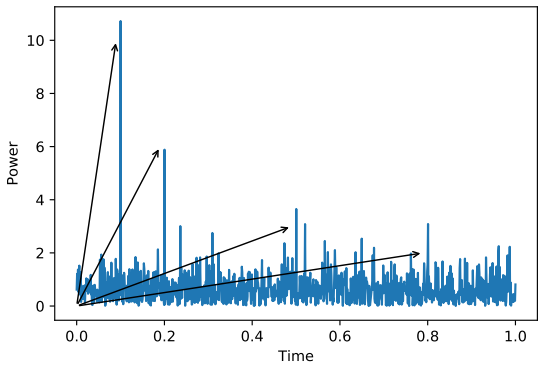


Figure 1.4.3: PDP example

There are several techniques in our disposal to obtain our secondary goal:

Auto Encoders an auto-encoder is a type of artificial neural network used to learn efficient data coding in an unsupervised manner [Wik19b] By inputting general sampled signal data, we would obtain an output of data coding that in turn can be inputted into a classifier.

Machine learning By using algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions and by relying on patterns and inference instead, we would be able to infer properties that can assist in classifying the rooms and corresponding sensor configurations to suit it.

Neural networks - artificial neural networks may be used for predictive modeling, adaptive control and applications where they can be trained via a data-set. Self-learning resulting from experience can occur within networks, which can derive conclusions from a complex and seemingly unrelated set of information [Wik19f]. By feeding the captured signals and information as a data-set to a pre-trained neural network we would derive conclusions regarding the device's environment.

Chapter 2

The Project

2.1 Theory

2.1.1 Concepts

In wireless radio communication, emitted signals experience a physical phenomenon known as multi-path propagation. A transmitted signal reaches the receiver after traveling by two or more paths. Walls and objects can reflect and scatter arriving signals as they cause changes in angle and time along the paths of the signal.

Mathematical Model [Wik19e] The mathematical model of the multi-path can be presented using the method of the impulse response used for studying linear systems.

Suppose you want to transmit a signal, ideal Dirac pulse of electromagnetic power at time 0, i.e.

$$x(t) = \delta(t) \tag{2.1.1}$$

At the receiver, due to the presence of the multiple electromagnetic paths, more than one pulse will be received, and each one of them will arrive at different times. In fact, since the electromagnetic signals travel at the speed of light, and since every path has a geometrical length possibly different from that of the other ones, there are different air traveling times (consider that, in free space, the light takes $3\mu s$ to cross a $1km$ span). Thus, the received signal will be expressed by (and seen in 2.1.1)

$$y(t) = h(t) = \sum_{n-1}^{N-1} \rho_n e^{j\phi_n} \delta(t - \tau_n) \tag{2.1.2}$$

where N is the number of received impulses (equivalent to the number of electromagnetic paths, and possibly very large), τ_n is the time delay of the generic

n^{th} impulse, and $\rho_n e^{j\phi_n}$ represent the complex amplitude (i.e., magnitude and phase) of the generic received pulse. Each pulse with its shift in time may be seen in figure 2.1.1. As a consequence, $y(t)$ also represents the impulse response function $h(t)$ of the equivalent multi-path model.

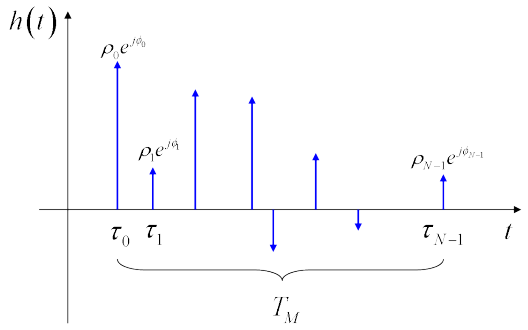


Figure 2.1.1: Multi-path impulse response

The multi-path phenomena are always present in wireless communication, especially in indoor environments. It has been a long time that we consider multi-paths as interferences, noise, or simply nuisance. Profiles of multi-paths changes from location to location, thus, multi-path channel profile works as a unique and location-specific signature. Thus, instead of being considered as a nuisance, one can design various types of analytics based on the uniqueness of the multi-path channel state information. By fully exploiting the rich multi-path information, technology can decipher the propagation environment, revealing information that is usually disregarded. Such a technology approach can enable many cutting-edge IoT applications.

The multi-path phenomena can be further exploited by using multiple-input and multiple-output (MIMO) in the transmitting as well as in the receiving device.

In MIMO systems, a transmitter sends multiple streams by multiple transmit antennas. The transmit streams go through a matrix channel which consists of all N_r paths between the N_t transmit antennas at the transmitter and N_r receive antennas at the receiver. Then, the receiver gets the received signal vectors by the multiple receive antennas and decodes the received signal vectors into the original information. A narrow-band flat fading MIMO system is modeled as

$$y = Hx + n \tag{2.1.3}$$

where y and x are the receive and transmit vectors, respectively, and H and n are the channel matrix and the noise vector, respectively [Wik19a].

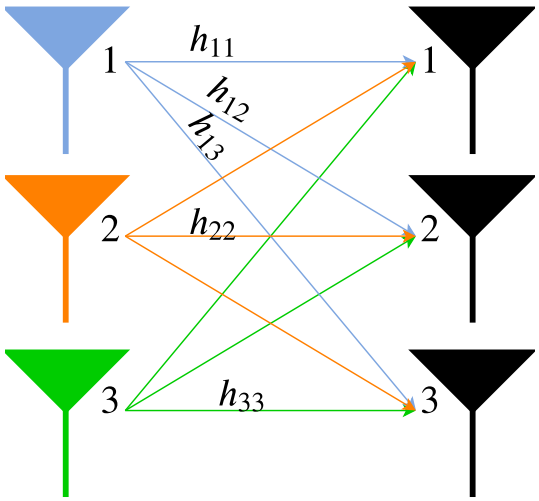


Figure 2.1.2: MIMO channel model

Each transmitting antenna (colored in blue orange and green in figure 2.1.2) is received differently in each receiving antenna (in the right side figure 2.1.2), thus, multiplying the amount of captured multi-path impulse responses. This expansion can further achieve unique spatial properties such as depth direction and other 3D information.

2.1.2 Techniques Survey

Radio Frequency Imaging is not a new technology, yet it has seen little commercial success due to the cost and power consumption of a large number of antennas and radio transceivers required to build such a system. Huang, Nandakumar, and Gollakota [HNG14] describe the feasibility and limits of WiFi imaging by leveraging multi-path propagation. Their work introduces design and implementation which was able to identify objects inside a room (like a couch). Scott's thesis [Sco17] introduces 3D microwave imaging for indoor environments which involves the use of antenna arrays, operating at microwave and millimeter-wave frequencies, for capturing images of real-world objects. His work focuses on using planar antenna arrays, operating between 17 and 26 GHz, to capture three-dimensional images of people and other objects inside a room. Scott suggests 3D microwave imaging algorithms for both dense and sparse antenna arrays as well as other algorithms such as colocated range migration algorithm RMA and a MIMO range migration algorithm which assist in the evaluation of the 3D space. Scott also suggests a design of antennas for microwave imaging and uses the Vivaldi tapered slot antenna in his implementations.

We intend to apply variations on Scott's work in order to comply with our hardware limitation and aspiration to use only popular protocols such as 2.4GHz WiFi which is much lower and narrower than Scott's implementation. Yet, we

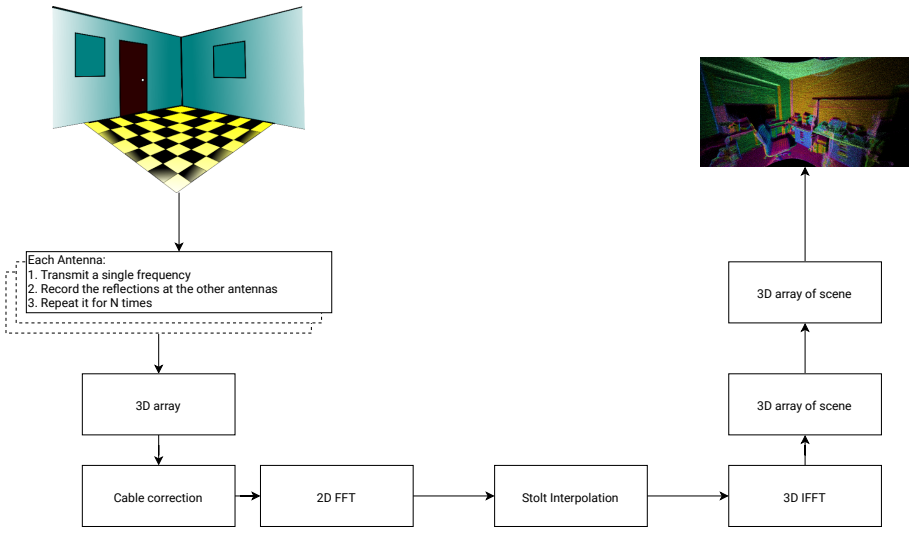


Figure 2.1.3: Block diagram for the colocated RMA

need to obtain a lower resolution than Scott in order to apply room analysis processing on the data acquired. By trying to combine Huang, Nandakumar, and Gollakota's work with Scott's we would be able to achieve the collection of the information that we strive to obtain in order to be able to apply the inference of the surrounding environment. The algorithms described in figure 2.1.3, where each block is a part in the process from raw data that had been received in its N antennas to the 3D array of the scene.

Indoor Localization is a network of devices used to locate people or objects where GPS and other satellite technologies lack precision or fail entirely. Sen, Radunovic, Choudhury and Minka [Sen+12] explore the viability of precise indoor localization using physical layer information in WiFi systems. The algorithm they suggest demonstrates localization accuracies in the granularity of 1m x 1m boxes, called spots. They show through experiments that PHY layer channel information from existing WiFi deployments can be an indicator of location. By synthesizing phase and time Lags and modeling the channel response they are able to apply clustering and classification algorithms which results in spot localization.

Huang, Zheng, Xiao and Peng [Hua+15] suggest localization based on the RSSI ranging Scope. In a RSSI based ranging algorithm, a node applies RSSI measurements to estimate its distances from the beacons, by using a known signal propagation model. Such location information is only relative to the location of all connected APs in range. Thus, for exact localization, there is a need to know where the APs are located. By combining RSSI localization and the PHY localization with other data obtained from other techniques we would

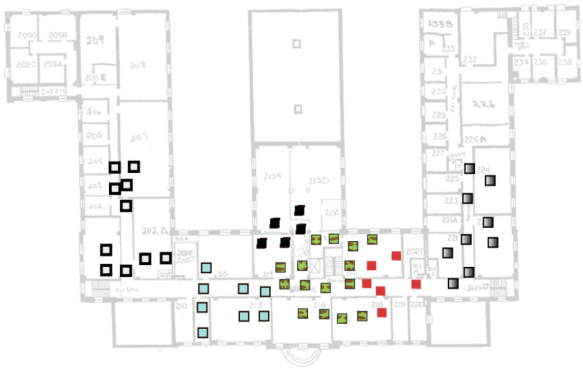


Figure 2.1.4: Engineering building floor plan. Different sets of spots shown in different colors

be able to better analyze and infer information about the device's environment. Figure 2.1.4 shows different sets of spots located by indoor localization using APs positions.

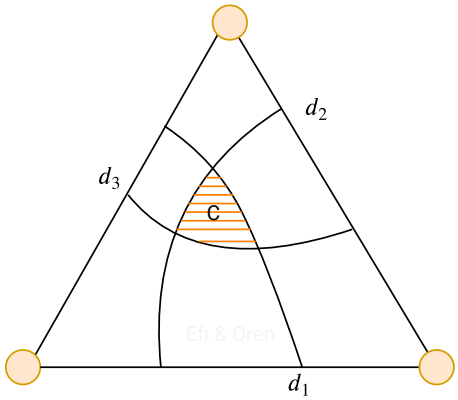


Figure 2.1.5: Region division based on the RSSI value

Holography is the science and practice of making holograms. Typically, a hologram is a photographic recording of a light field, rather than an image formed by a device without lens. [Wik19c]. In its pure form, holography requires the use of laser light for illuminating the subject and for viewing the finished hologram. Yet, Holl and Reinhard [HR16] demonstrate a scheme to record a hologram in a phase-coherent fashion and recover three-dimensional views of objects and emitters and feeding the resulting data into digital reconstruction algorithms. Figure 2.1.6 describes a holographic imaging process that generates 3D images using the microwave radiation of a WiFi transmitter experimented

in [HR16] "Holography of WiFi radiation".

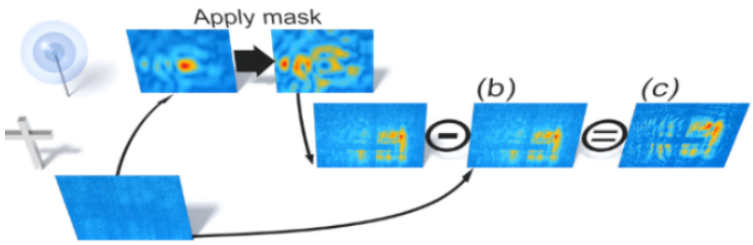


Figure 2.1.6: Reconstruction of objects using holography

Wireless Sensing Embedded in wireless signals is information on an indoor environment that is captured during radio propagation, motivating the development of emerging wireless sensing technologies. Intelligent systems have become popular recently, in that with the help of learning they are capable of comprehending an object or even the world in the way humans do. For example, researchers have spent decades on computer vision or machine vision systems that achieve a high-level understanding of digital images and videos that is comparable or even better than the human visual system. Can WiFi perceive an indoor environment? According to Liu and Wang[LW19] the answer is yes. Applying statistical analysis on a data-set of captured signals is used for centimeter-accuracy indoor positioning and tracking, biometrics and vital signs estimation, motion and speed detection, and more. We intend to apply similar methods to the data-set of captured signals in order to create a sense of the device's environment. Many methods exist to apply indoor furniture and room recognition, Varvadoukas, Giannakidou, Gomez, and Mavridis [Var+12] use internet-derived models and object context to achieve this. While there are plenty of other methods to map and reconstruct the environment like 3D point clouds [SKR11] or set of 2D slices or just range measurements, they all rely on a data set we would obtain from the wireless sensor.

2.2 Block Diagram

The following block diagram describes the inner goings of a sensor device in the stages taken to infer its surrounding environment using wireless-sensing.

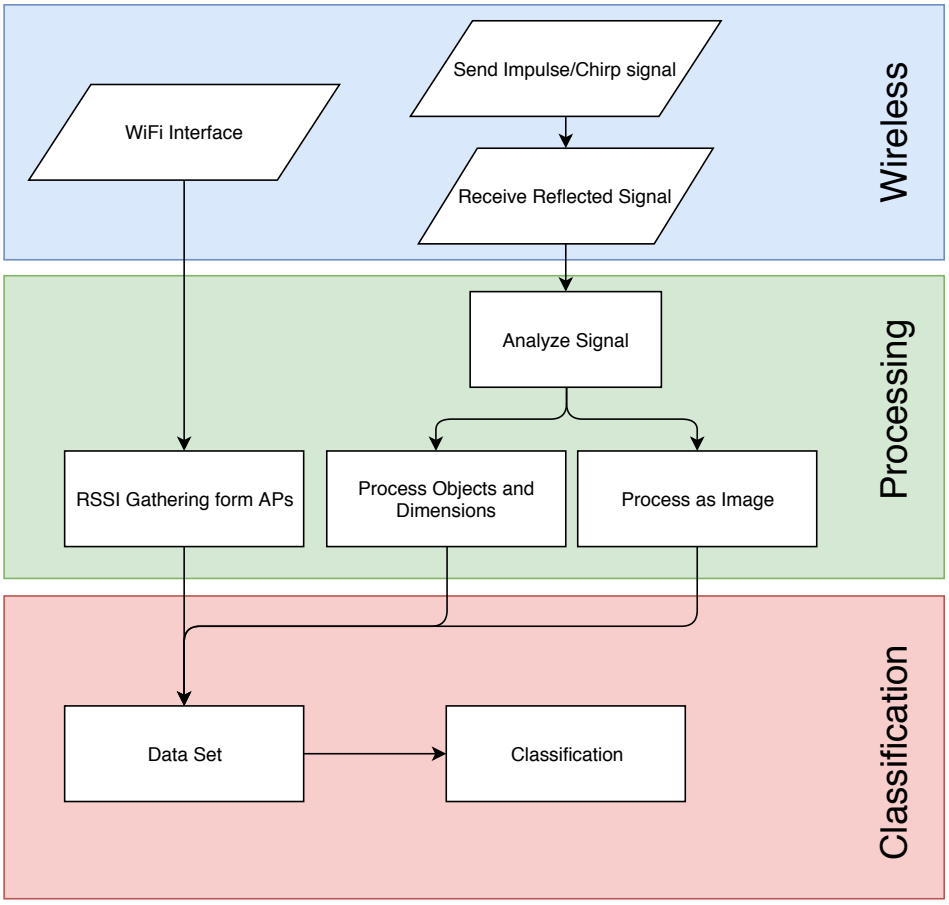


Figure 2.2.1: Block Diagram

The Block Diagram (figure 2.2.1) is divided in to three sections:

- **Wireless Section** - The transmission and and reception of Ad-Hoc and standardized WiFi protocol signals.
- **Processing Section** - analyzing the received signals and processing them to an image or objects and dimensions. Creating data set from WiFi interface.
- **Classification** - Classifying space according to the combined created data sets.

2.3 Practice

2.3.1 Constraints and solutions

Out of the proposed techniques we choose to implement Radio Frequency Imaging and RSSI Indoor Localization wheres the Holography and Wireless Sensing techniques were found less suitable to be implemented under this project constraints.
The Holography technique which requires the illumination of a room using a radio signal and recording the received signals at another point in space required grate amount of mathematical development and exploration. In fact, this technique requires the solution of the well-known inverse problem [Wik20c]. Since our problem is an inverse problem in the wave equations, it is very demanding in calculation taking all the wave scattering possibilities of the room along with the measurement system imperfections. Due to these reasons, this technique was forsaken under the scope of this project.
The Wireless Sensing techniques required us to apply statistical analysis on a data-set of captured signals, yet in order to do this, we first had to gather a data-set of signals. Due to the fact that one part of our project was dedicated to the creation of the radio system that would be used for our experiments we had a shortage of time to conduct many measurements so to create a sufficient data-set for this technique. Therefore, this technique was also forsaken under the scope of this project.

2.3.2 Hardware

Software-Defined-Radio In our project, we are using SDR as our main hardware. Software-defined radio (SDR) is a radio communication system where components that have been traditionally implemented in hardware (e.g. mixers, filters, amplifiers, modulators/demodulators, detectors, etc.) are instead implemented by means of software on a personal computer or embedded system [Wik19g][Mar03, p. xxxiii]. While the concept of SDR is not new, the rapidly evolving capabilities of digital electronics render practical many processes that were once only theoretically possible.

LimeSDR is a low cost, open source, apps-enabled (more on that later) software defined radio (SDR) platform that can be used to support just about any type of wireless communication standard.[Lim19]

Ettus USRP B210 provides a fully integrated, single-board, Universal Software Radio Peripheral (USRP^{TM}) platform with continuous frequency coverage from 70 MHz – 6 GHz. Designed for low-cost experimentation, it combines the AD9361 RFIC direct-conversion transceiver providing up to 56MHz of real-time bandwidth, an open and reprogrammable Spartan6 FPGA, and fast SuperSpeed USB 3.0 connectivity with convenient bus-power. Full support for the USRP Hardware Driver^{TM} (UHD) software allows you to immediately begin developing

with GNU Radio, prototype your own GSM base station with OpenBTS, and seamless transition code from the USRP B210 to higher performance, industry-ready USRP platforms. An enclosure accessory kit is available to users of green PCB devices (revision 6 or later) to assemble a protective steel case.[The19]

Raspberry Pi a series of small single-board computers. The last release of Raspberry Pi was in June 2019 with a 1.5GHz 64-bit quad core ARM Cortex-A72 processor. The board includes two USB 3.0 ports and support 802.11ac WiFi.

2.3.3 Programs

GNU Radio GNU Radio is a free & open-source software development toolkit that provides signal processing blocks to implement software radios. It can be used with readily-available low-cost external RF hardware to create software-defined radios, or without hardware in a simulation-like environment. It is widely used in research, industry, academia, government, and hobbyist environments to support both wireless communications research and real-world radio systems [GNU19]. The toolkit that is provided under GNU Radio is written with C/C++ and Python. The libraries that are being used with are Boost for C++ and NumPy that used Boost and compiled under C/C++ to be used with Python.

CST Studio a high-performance 3D EM analysis software package for designing, analyzing and optimizing EM components and systems.

2.3.4 Work-space

Overleaf is an online LaTeX editor that allows real-time collaboration and online compiling of projects to PDF format. Overleaf is a freely-hosted and allows:

- Track changes
- 2 collaborators per project
- Spell check

Github is a free cloud-based version software development version control using Git.

Slack is a cloud-based that provides instant messaging platform. Slack offers features that goods for incollaboration projects.

TeamGantt is a cloud-based gantt chart software can help plan your projects.

JetBrains CLion and PyCharm that are and IDEs for C/C++ and Python respectively.

MicroPython MicroPython is a lean and efficient implementation of the Python 3 programming language that includes a small subset of the Python standard library and is optimized to run on microcontrollers and in constrained environments [Mic].

2.4 Simulation

Before reaching our objective to create an image of a real scene. We need to have another step in our path, a simulation of a scene. A simulation is an action of pretending for the purpose of study, that is, by simulation we can study the algorithm, the effects of objects, mathematical theory and optimize the results of the proposed algorithm and method.

Because of the success of the simulations of the RF Imaging, we have put on hold and give up on the part of CST simulations which in retrospect would yield not too much for the RF Imaging which was our main goal.

2.4.1 Radio Frequency Imaging Simulation

To create a simulation we used a simple paint tool (similar to MS Paint of Microsoft), the output of the tool is a simple gray-scale bitmap, that is the created file is 2D-Matrix with values of 0 to 255. This file represents a delta size layer of XY in space. The simulator takes the created image file, and simulate the effect of transmitting a single Electromagnetic wave using an Isotropic antenna with a specific frequency, phase, and magnitude, the simulation calculates the reflected wave phase based on the time-delay of the image where the image is placed in known distance z. Because that a better resolution of the file results in more computing time, the simulation is computing only values that are different from zero. The simulation pretending to send a single electromagnetic wave, and calculate the reflection from all non-zero points in the bitmap file. The simulation generates a scene file that pretends to be a real measurement of a real scene that may be measured with an SDR or other RF equipment. The generated scene file is characterized by a tensor of 3 dimensions where each entry in the tensor is a complex number that represents the magnitude and the phase of the reflections of the transmitted wave.

The following code integrate over the bitmap for dots that different from nil as given in Equation 2.4.1

$$\iint\!\!\!\square_{\text{Scene}} \exp\left(-j \cdot 2 \cdot k\sqrt{(x-x_0)^2+(y-y_0)^2+(z-z_0)^2}\right) \tag{2.4.1}$$

where (x_0, y_0, z_0) represent the position of the transiting (and receiving) antenna and (x, y, z) is the coordinate of point in the scene which has its own reflection. The integral is over all (x, y, z) in space of the scene, in the simulation case z is

constant (one layer), k is the wave-number $k = \frac{2\pi}{\lambda}$.

```
import numpy as np
def getIntegralDot2(img, k, _x, _y):
    args = np.argwhere(img)
    x_sc = img.shape[0]/Nx
    y_sc = img.shape[1]/Ny
    y = np.exp(-1j * 2* k * np.sqrt(0j +
                                    + np.power(args[:,0]*Dx/x_sc - _x, 2)
                                    + np.power(args[:,1]*Dy/y_sc - _y, 2)
                                    + np.power(Z1, 2)
                                   )
               )
    _img = img[args[:,0], args[:,1]]
    _img = _img.reshape(1, _img.shape[0])
    _img.shape
    return _img.dot(y)
```

The code is written in Python using NumPy package and implements Equation 2.4.1, see more in the appendices.

Figure 2.4.1: Ben-Gurion Logo Bitmap

Figure 2.4.1 is the given input for the simulation, and its output is the input for the RMA algorithm.

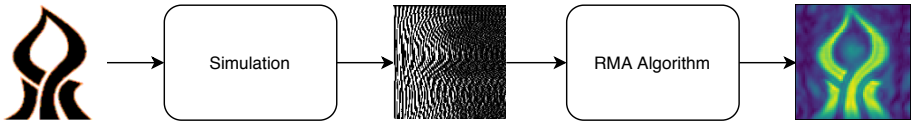


Figure 2.4.2: Simulation Flow

As may be seen in Figure 2.4.2, the input from Figure 2.4.1 is used to generate a scene that is the input for the RMA algorithm that produced the "surprising" output.

One simulated 3D scene includes fully reflective points that are located in a spherical array formation. Each point in the scene space has a ray that is transmitted to, reflected, and received directly and each of these single rays experiences no path-loss. The reconstructed images of the scene, seen in 2.4.3, shows the point array from two different perspectives (front and side views)

wherein each perspective the Z-axis is summed to create a 2D image of reflective intensity.

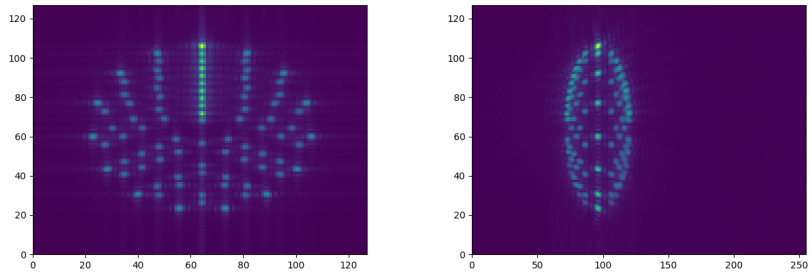


Figure 2.4.3: 3D spherical array of fully reflective points scene results (Front and side view)

Following several other scene simulations, we decided to simulate the Ben-Gurion University logo as a flat object in the middle of the scene. The simulation is of 32X32 pixels summing the Z axis of the scene. The results can be seen in 2.4.5.

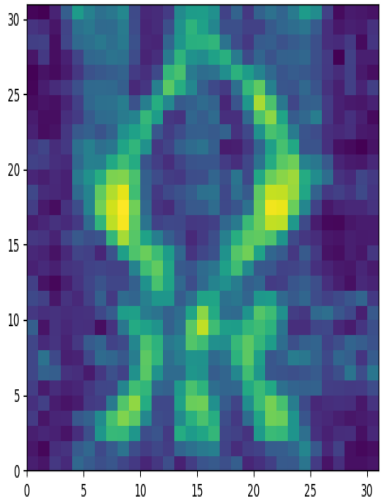


Figure 2.4.4: 32X32 Reconstructed BGU logo

2.4.1.1 Resolution

We noticed increasing the resolution of the simulation results in much better visual object recognition. The resolution of the reconstructed image correlates to the number of antennas in a sampling array. Thus, for better images, we pay with more samples and more processing of data resulting in the longer time needed to obtain an image.

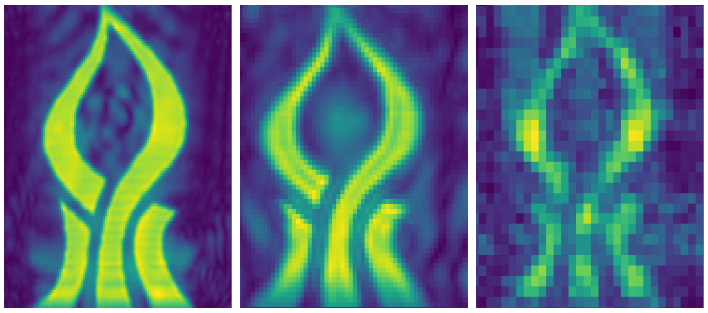


Figure 2.4.5: Reconstructed BGU logo: 32X32 positions (Right), 64X64 positions (Middle),128X128 positions(Left)

2.5 Hardware setup

In this project, we conducted implementations of the proposed methods. We created a hardware setup for each experiment that was needed to implement the method in use. The experiments can be divided into 2 objectives:

- RSSI Triangulation
- Radio Frequency Imaging

2.5.0.1 RSSI Triangulation

The required equipment for the experiments are:

1. WiFi enabled routers - Cisco-Linksys WRT54GS Wireless-G Broadband Router and others.
2. WiFi sensor node with packet parsing ability - Wemos D1 ESP8266 chip and ALPHA NETWORK 1 802.11b/g Long-Range Wireless USB Adaptor.

We placed WRT54GS routers at random places in a room in order to use them as reference anchor points in space.

Figure 2.5.1: A WRT54GS 802.11b/g Router

Each router was given a lexicographic letter (e.g A,B,C,D) as a SSID to broadcast and it's location in the room was manually measured and recorded relative to an origin point $(0m, 0m)$.

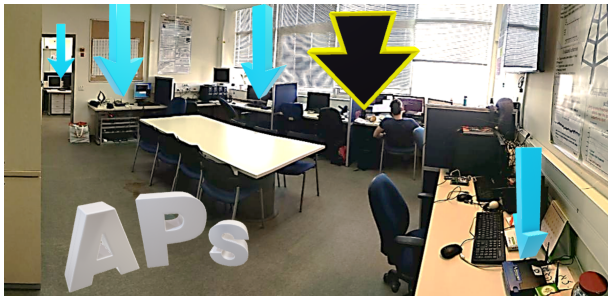


Figure 2.5.2: APs (cyan) and sensor (black) setup in Lab 512

2.5.0.2 Radio Frequency Imaging

The required equipment for the experiments are:

1. Computer include USB3 GNU-Radio environment installed.
2. SDR - In our experiments we have used B210 from Ettus.
3. Automated XY grid positioning system - In our experiments we have used a 3D printer.
4. A directional Vivaldi RF transmit and receive antenna assembly with an RF shield between them to reduce co-channel interference (CCI) .

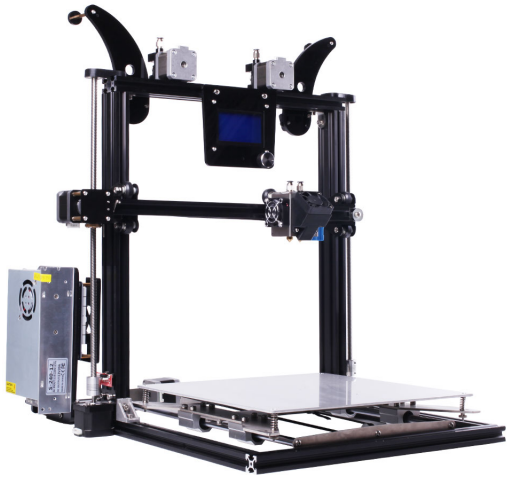


Figure 2.5.3: The ZONESTAR 3D Printer

In the following Figure 2.5.4, the antenna assembly is mounted on top of the 3D printer moving head, instead of the filament heating element, such that the antenna assembly can move along X and Y axes. The computer is connected to the 3D printer engine controller using a USB cable and over a serial connection. The computer is also connected to the B210 SDR using USB3 and tunes the frequency of the SDR transmission and using it to receive the RF signals. The receiving and transmitting is base-band represented by a stream of 2-float numbers (complex), seen in Figure 2.5.5. the received signal from the antenna is demodulated down to base-band and the down-converted signals are sampled into the I and Q (Real and Imaginary) representations of the sampled signal.

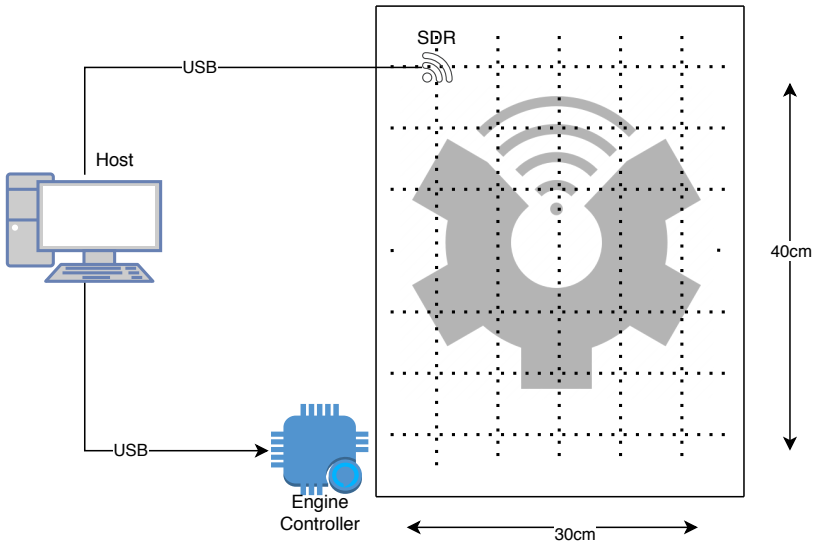


Figure 2.5.4: SAR Array Setup

It is very important to understand the need for synchronization of this part, lack of synchronization will result in wrong sampling and jitters and will add effects of a non-LTI system e.g. a non-causal system where the imaginary part from future samples combined with the present true part.

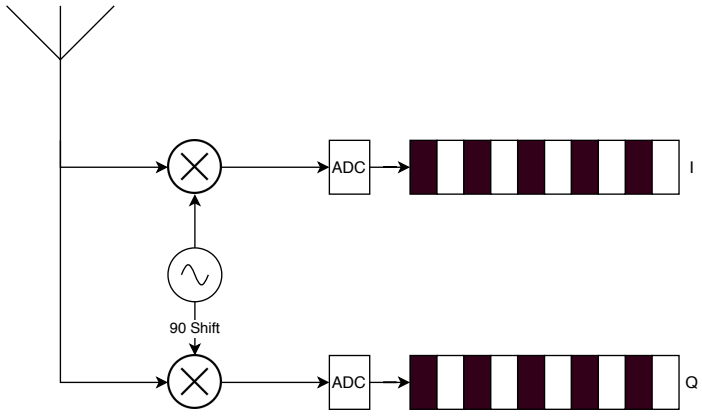


Figure 2.5.5: SDR sampling concept

The RF antenna assembly is composed of 2 Vivaldi antennas [Wik20e] which are a co-planar broadband-antenna, which is made from a solid piece of sheet metal, a printed circuit board, or from a dielectric plate metalized on one or

both sides. This assembly is located on the head of the printer that can be seen in Figure 2.5.3, which is the 3D-Printer that we have used.

Figure 2.5.6: Vivaldi antenna

The antenna we used, seen in 2.5.6, is a high gain directional wide-band antenna suitable for directional radio signal transmission and reception. Frequency range: $2.4\text{GHz} - 10.5\text{GHz}$, linear polarization with a rated gain of 7dBi and a return loss of 10dB. We found this antenna to be most suitable for UWB positioning, WiFi 2.4GHz and 5.8GHz and other common frequencies.

We placed the transmitting and receiving antennas in parallel to each other and with a 10cm RF EMI shielding [Wik20b] in between.

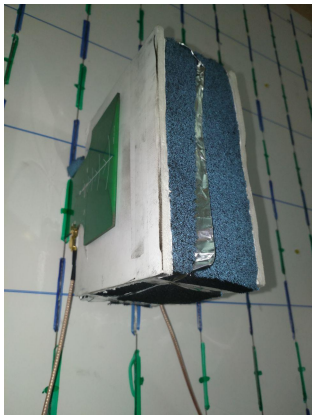


Figure 2.5.7: Antenna assembly

The EMI shielding purpose is to reduce the amount of energy picked up by the receiving antenna from the transmitting antenna at close range. Meaning to reduce the signal sampled at a virtual minimal distance of 10cm (the distance between the antennas). In order to further block the co-channel interference between the antennas we have placed an aluminum foil reflector in the middle of the RF EMI shielding. This shielding attenuation was tested and found to be greater than 10dB.

The antenna assembly was connected to the moving head of the 3D printer X,Y movement rig. The entire testing rig can be seen in 2.5.8.

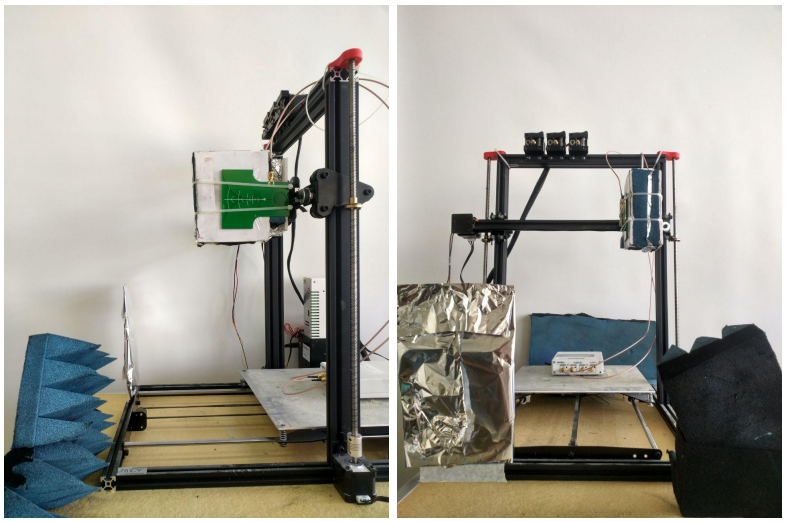


Figure 2.5.8: RF Imaging Rig

Chapter 3

Timeline & Progress

3.1 Milestones

Table 3.1: Milestones and Products

#	Name	Deadline	Hours	Measurable Outcome
1	Literature Survey ✓	05/11/2019	100	Knowledge
2	Prepartion Report ✓	12/12/2019	48	The Report itself
3	Toolkit Testing and Software Training ✓	12/12/2019	100	Ability to Work
4	GNU-Radio Gr-Radar ✓	27/12/2019	15	PoC
5	Progress Report ✓	24/01/2019	50	The Report itself
	COVID-19 TIME	03/2020 - 04/2020	COVID-19 TIME	
6	Simulation Creation ✓	05/2020	100	Simulation
7	RSSI Data Set Creation ✓	05/2020	100	Raw Data Set
8	WiFi Image Creation ✓	06/2020	200	Images
9	Optional Basic Classification ✗		Future Work	Inferences of Basic Environment
10	Optional Analysis of Data Sets ✗		Future Work	Inferences of Environment
11	Reconstructed Images ✓	08/2020	200	Recognizable Object Image
12	Create Hardware Setup for Presentation ✓	06/2020	50	Hardware Setup
13	Poster ✓	06/2020	60	A Commercial Poster
14	Presentation ✓	06/2020	60	PPT and Videos
15	Final Submission ✓	08/2020	150	Outstanding Project

3.2 Risk Management

- In retrorespect, a global pandemic such as the Novel Corona-virus 2019, a remote environment could be help.

- Some of the possible techniques are based on frequencies that we cannot generate (higher than 6GHz). We are going to use SDRs that are limited by 6GHz. Hence, this is a risk.

  In case of the embodiment of this risk, our solution is to fallback to a RSSI triangulation technique which is more implementable.

- The WiFi imaging algorithm requires a large antenna array in order to produce an image of the surrounding sensor environment. This can not be done with the equipment we have in hand. The article [Sco17] also suggests a sparse antenna array implementation but with a trade-off in resolution.
  Since we can only utilize up to 4×4 MIMO with our SDR by joining 2 devices as a single transmitter, this might be a risk. We could end up with a low-resolution image with a small amount of information to relay on.

  The solution is to put more emphasis on information gained from other methods during the surrounding sensor environment inference (such as RSSI triangulation).

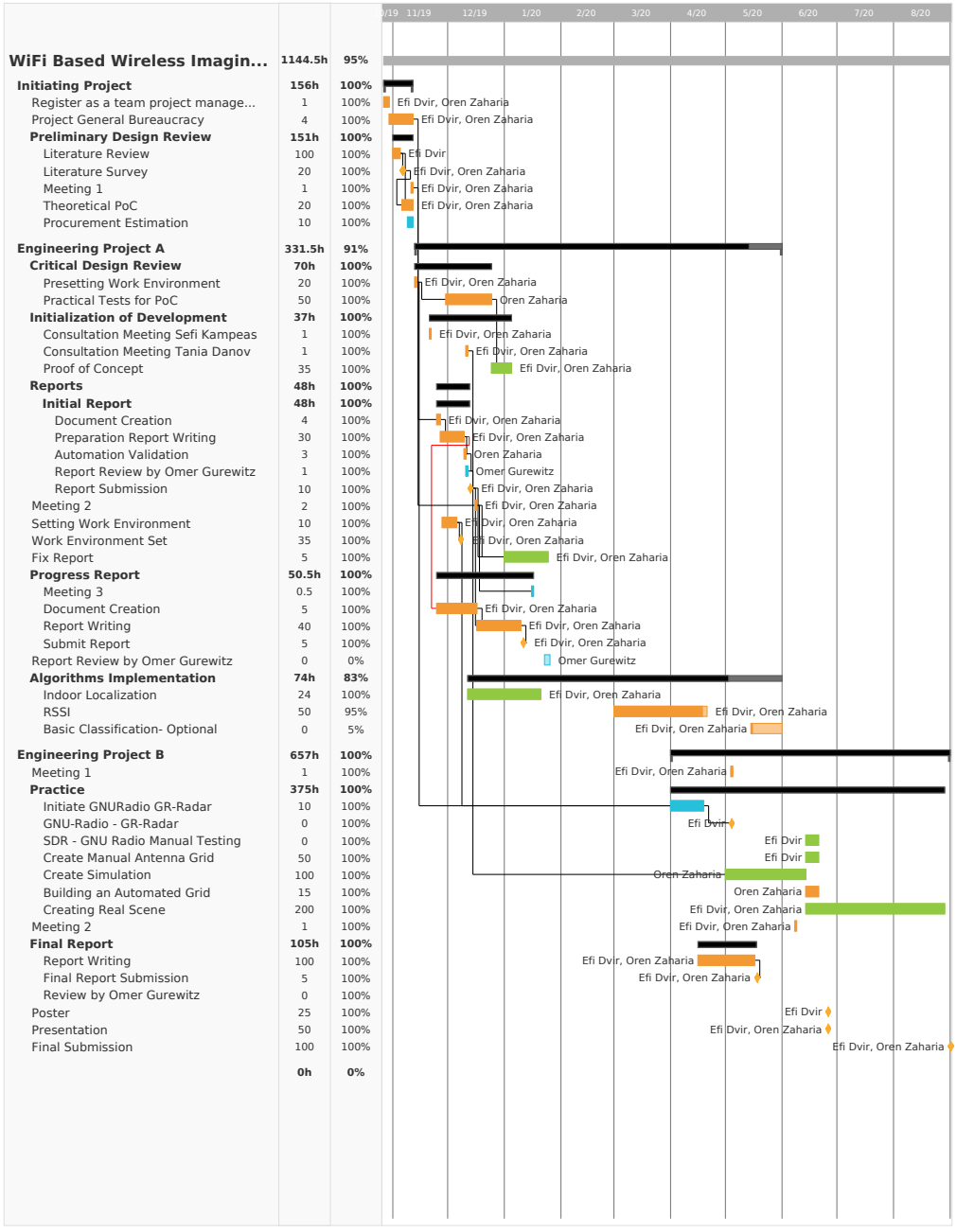
- Our project is very ambitious relative to our colleagues' final projects. Therefore, time is an influential resource, shortage of time will cause a partial implementation or lesser informative structure.

  Hence, our solution is to first implement the simpler methods and algorithms in order to gain fundamental information to be used as a data-set.

- We are limited by time and space, that is, we can use a limited amount of environments (rooms) in our data-set gathering. This might not be enough to be useful to infer other environments.

  Our solution is to simulate the algorithms using RF simulation software to train our model and gather more information using these simulations.

3.3 Gantt


teamgantt
Created with Free Edition
11/19 12/19 1/20 2/20 3/20 4/20 5/20 6/20 7/20 8/20
WiFi Based Wireless Imagin... 1144.5h 95%
Initiating Project 156h 100%
Register as a team project manage... 1 100%
Project General Bureaucracy 4 100%
Preliminary Design Review 151h 100%
Literature Review 100 100%
Literature Survey 20 100%
Meeting 1 1 100%
Theoretical PoC 20 100%
Procurement Estimation 10 100%
Engineering Project A 331.5h 91%
Critical Design Review 70h 100%
Presetting Work Environment 20 100%
Practical Tests for PoC 50 100%
Initialization of Development 37h 100%
Consultation Meeting Sefi Kampeas 1 100%
Consultation Meeting Tania Danov 1 100%
Proof of Concept 35 100%
Reports 48h 100%
Initial Report 48h 100%
Document Creation 4 100%
Preparation Report Writing 30 100%
Automation Validation 3 100%
Report Review by Omer Gurewitz 1 100%
Report Submission 10 100%
Meeting 2 2 100%
Setting Work Environment 10 100%
Work Environment Set 35 100%
Fix Report 5 100%
Progress Report 50.5h 100%
Meeting 3 0.5 100%
Document Creation 5 100%
Report Writing 40 100%
Submit Report 5 100%
Report Review by Omer Gurewitz 0 0%
Algorithms Implementation 74h 83%
Indoor Localization 24 100%
RSSI 50 95%
Basic Classification- Optional 0 5%
Engineering Project B 657h 100%
Meeting 1 1 100%
Practice 375h 100%
Initiate GNURadio GR-Radar 10 100%
GNU-Radio - GR-Radar 0 100%
SDR - GNU Radio Manual Testing 0 100%
Create Manual Antenna Grid 50 100%
Create Simulation 100 100%
Building an Automated Grid 15 100%
Creating Real Scene 200 100%
Meeting 2 1 100%
Final Report 105h 100%
Report Writing 100 100%
Final Report Submission 5 100%
Review by Omer Gurewitz 0 100%
Poster 25 100%
Presentation 50 100%
Final Submission 100 100%
0h 0%
Efi Dvir, Oren Zaharia
Efi Dvir
Oren Zaharia
Omer Gurewitz

Chapter 4

Testing & Experimentation

4.1 RF Imaging experiments

Before implementing the proposed methods we first had to prove their feasibility. Each concept involved in the method's functionality was first tested and proved to generate expected results. The following are the tests and experimentation made to show proof of concept.

4.1.1 Phase vs. Distance

In order to show that phase and distance are directly related, we constructed a basic transmitter-receiver flow chain in GNU-Radio and along with an SDR as hardware, the following experiment was made.
The transmitting and the receiving antenna were placed about one meter apart and a sine wave signal was transmitter between them in a loop-back configuration. Slowly distancing them apart while simultaneously observing the transmitted ad received sine wave. Afterward, we introduced a mediator along the path in the form of a reflective aluminum surface. As showed in 4.1.1, the object in the path was slowly moved away from the antenna array elongating the radio path distance, this resulted in a visible subtle change in phase.

Figure 4.1.1: Reflective object used for testing

4.1.2 Distance vs. Power

It is well known that radio signals traveling in space are effected by several factors and are attenuated by them. According to the free-space path loss propagation model, the distance the signal traverses is directly related to the amount of attenuation it experiences.

$$P_r = P_t \frac{G_t G_r \lambda^2}{(4\pi d)^2} \tag{4.1.1}$$

In order to better understand the power that we need to transmit we conducted an experiment in order to guarantee that the received power is within the the dynamic range limits of the Rx input of the SDR. We transmitter a signal withing a gain range between 0 and 70 and found that signals with a gain under 40 will be insufficient to traverse the entire room while a signal with again above 65 will distort the input and will cause inter-modulation. Therefore, a gain of 60 was decided as the base case of the entire implementation.

4.1.3 Phase Comparator Vs. I&Q

In order to deduce the distance from objects, the implementation had a basic requirement - detect the phase difference between the transmitted signal and the received. Since our entire implementation is based on an SDR we are limited to a mostly software implementation. Therefore, our first approach to phase detection was to using a phase comparator block as a part of the GNU-Radio flow.

We found the gr-phase-comparator implementation of a phase comparator block in GNU-Radio in `https://github.com/jettero/gr-phase-comparator` by Paul Miller aka jettero.

Figure 4.1.2: gr-phase-comparator phase comparator GNU-Radio block

This block compares the phase of two complex signal inputs and outputs the differential in phase as a float. This is done iteratively as a stream of values and the output can be wrapped in a time window to estimate the number of cycles. At a wrapped window of 1, the phase can be only in the range of $-\pi$ to π. This phase difference reading was to be combined with the relative amplitude

received from the Rx input (the sampled reflected signal) to create the input to the RMA image reconstruction.

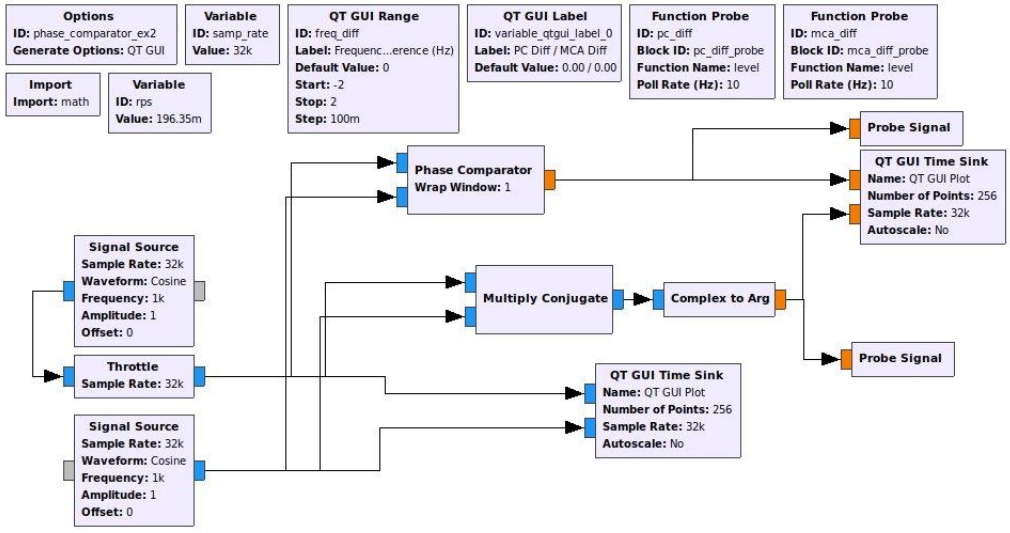


Figure 4.1.3: RF chain flow - sampling reflections

Figure 4.1.3 showes the GNU-Radio flow graph for sampling a single CW base-band signal. This is for a single position in the RMA antenna array. Using this concept we went to implement the system using an SDR, we made many tests in order to obtain the correct measurements needed for the RMA image reconstruction.

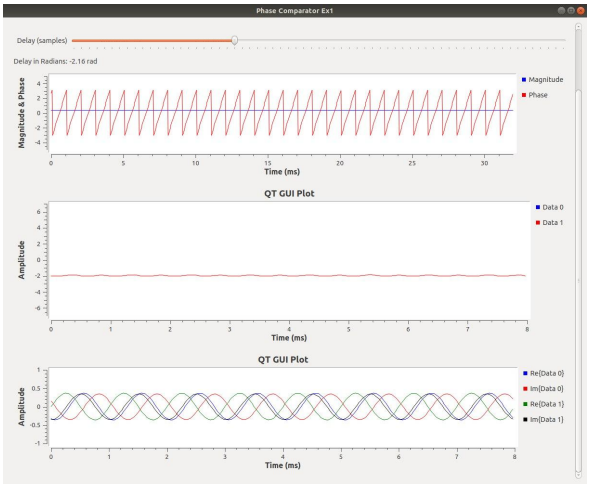


Figure 4.1.4: Phase and Magnitude live SDR mesurments

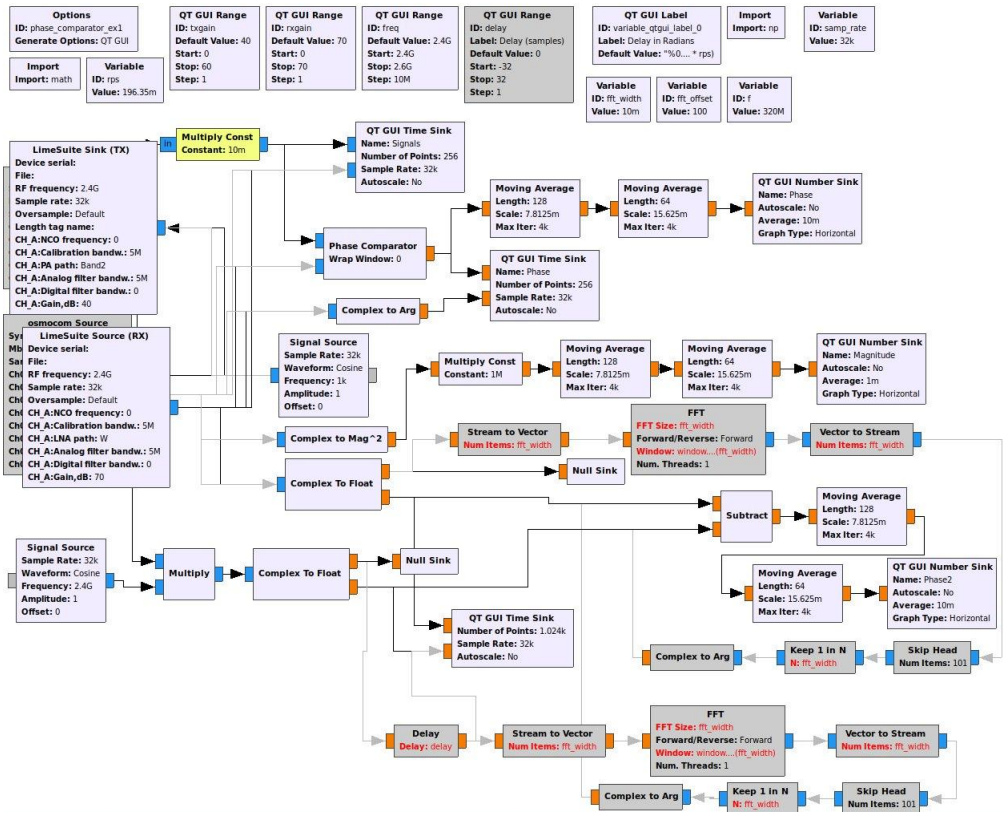


Figure 4.1.5: Tests preformed using the gr-phase-comparator

After testing the functionality of the gr-phase-comparator we came to an understanding that to implement the concept on an SDR where most functionality is implemented in software, we could use a different approach.

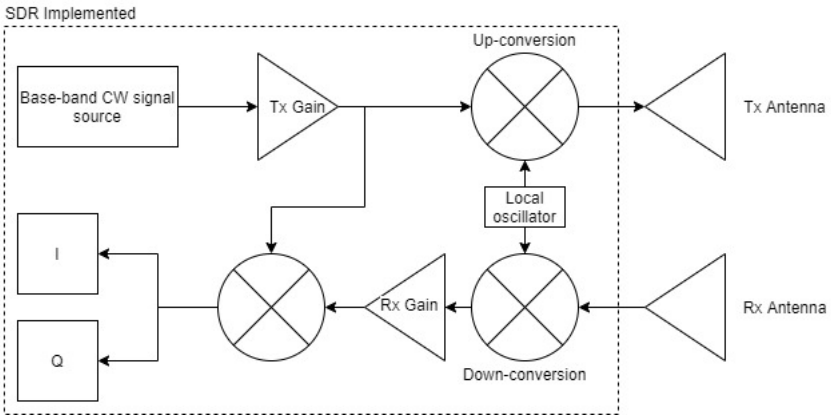


Figure 4.1.6: RMA sampling RF chain

Our implementation led us to an understanding that since the signal is up-converted and downconverted by the same oscillator withing the SDR, the domain we are working in would only be in base-band. Since both Rx and Tx signals are conversion-synced we can multiply the signals (like using a mixer) and the result would generate a base-band signal. If there are no differences in time or frequency we would obtain a 0V DC level. Where there are differences in time (and therefore in phase) we would obtain a very low-frequency shift from DC. Since our signals are already complex we could obtain the Phase and Magnitude (from the I & Q components) as required by the RMA in order to reconstruct an image.

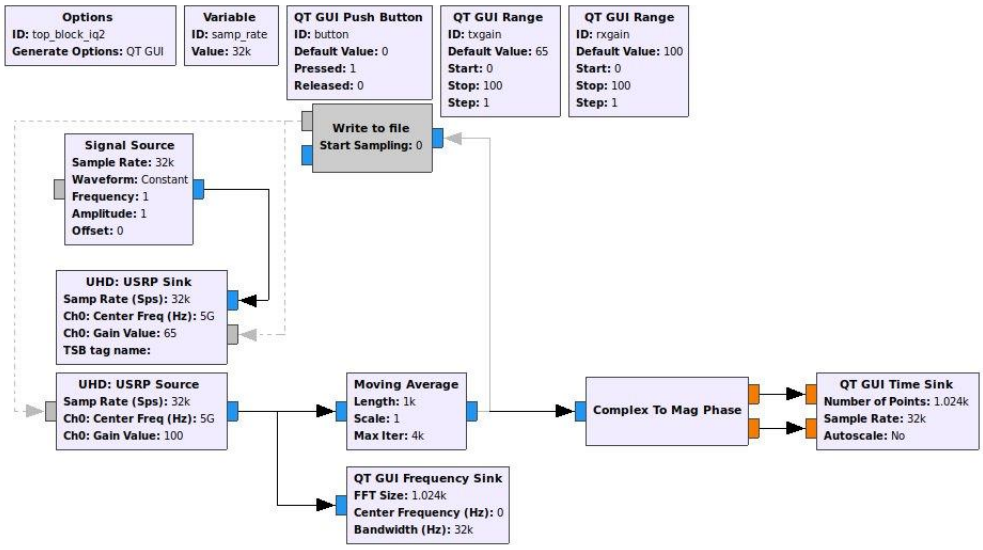


Figure 4.1.7: Simplified semi-automatic I & Q componetes mesurments

These Phase and Magnitude are simple float DC values which allow this measurement design flow to be very undemanding in processing power and sample rate.

4.1.4 The Effect of Disturbances

During most experiments, we noticed the measurement setup is very susceptible to subtle changes in the room. That is, a slight movement of objects in the room has a significant effect on the measurements taken. Therefore, we had to clear the room of any disturbances that could harm the results of our experiments. If a person would step into the room it could change the outcome of the image.

To avoid disturbances we closed lab 511 in favor of our experiment and banned all from entering during active measurement. We also pointed our rig to an "open space" (Figure 4.1.9) so it would be measuring the objects we intended and not objects around the room.

Figure 4.1.8: Lab 511 do not enter sign

Figure 4.1.9: Lab 511 "Open Space"

4.1.5 Motion

Due to the effect of disturbances, we found that we could use the same setup for identifying if there is movement in the scanned room. A person walking and moving in the vicinity of the scanner would induce a significant change and instability in the phase and power of the receiving signal for a single point in the antenna array. Thus, it is fairly easy to identify movement using the same sensing technique we introduced. To identify movement periodic samples are taken in small time intervals. If the measured phase differs significantly, something is moving.

4.1.6 CW Vs. Frequency Modulated CW

Since radar signals are usually at high frequency, it is not possible to sample at Nyquist frequency since there are not analog to digital converter that can operate at such a high sample rate. In order to sample the signals, they are down-converted to base-band so to comply with Nyquist. But with such a sample rate the time between each sample is grater that the time that a signal propagates through space. Thus, it is not possible to directly sample the signal's propagation time (from the transmission to reception).

In order to derive the propagation time, a simple trick can be used. By linearly increasing the frequency of the transmitted signal we can obtain the propagation time. The change in frequency correlates to the propagation time delay as seen in figure 4.1.10. This enables to measure the propagation time indirectly.

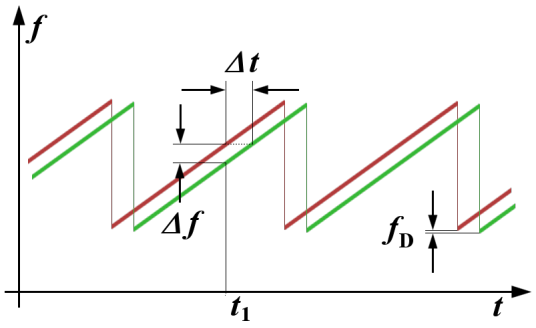


Figure 4.1.10: FMCW Principal (transmitted in red received in green)

We have made many tests with frequency-modulated CW signals, seen in figure 4.1.11, yet we found that a stepped frequency change corresponds to the linear frequency change for the RMA and is simpler to calculate. Thus, we eventually chose the continuous-wave radar [Wik20a] approach as an input for the range migration algorithm. In effect, the RMA uses different frequencies to distinguish different phase delays that correspond to the different propagation time of the transmitted signal.

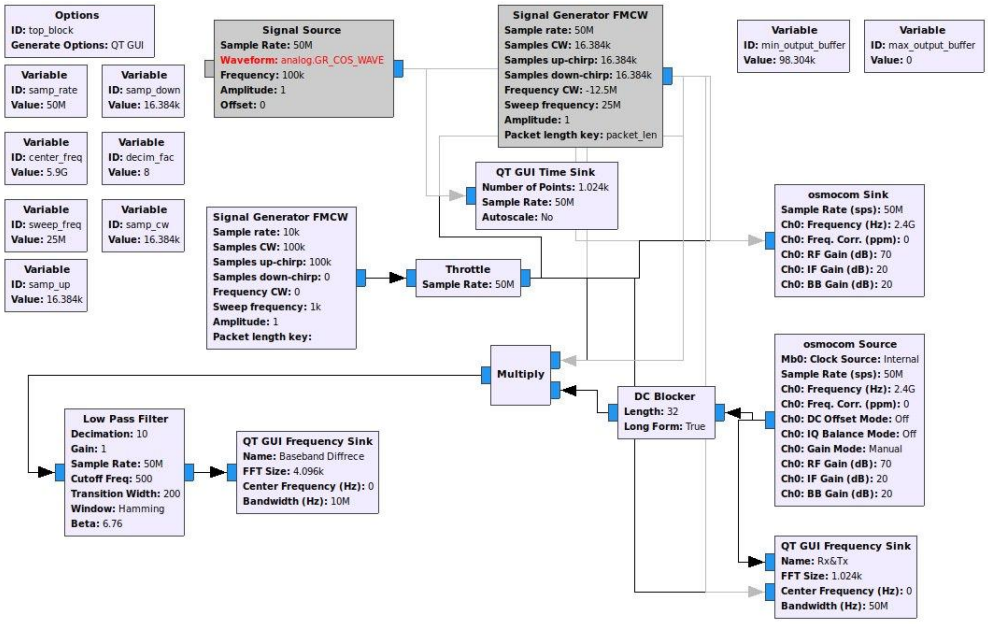


Figure 4.1.11: FMCW scanning signal

4.1.7 Manual, semi-automatic and fully automatic measurements

At the beginning of our RMA tests, in order to obtain measurements, we needed to change the position of the antenna array while changing the frequency in each position several times.

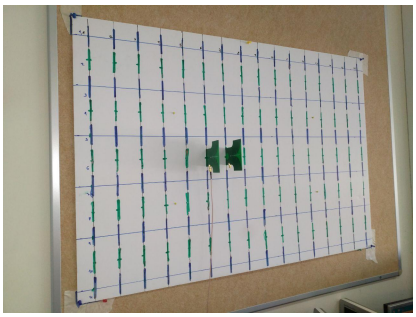


Figure 4.1.12: Manual array positioning on a foam-board

To do that we first started with manual measurements on a foam-board with 6.25cm slots spaces for the antennas to fit in (seen in figure 4.1.12). In

each position (8X8) we would change the frequency manually from 4.95GHz to 5.6GHz and write down the I and Q values of the signal in an excel file. Once completed this file was used to feed the RMA image reconstruction code.

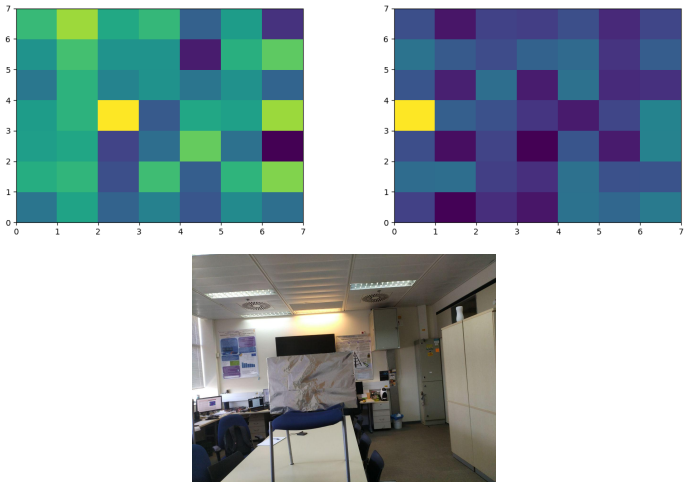


Figure 4.1.13: First RMA results

The first scans were of a rectangle aluminum reflector about 2m from the antenna array. The results in figure 4.1.13 were clearly insufficient to understand the dimensions and shape of the object yet the yellow pixel indicated that energy was indeed reflected form an object in front of the antenna array.

The manual scans were extremely laborious and took much time (about 5 hours for 64 positions and 16 frequencies). If we were to continue scanning we had to create a better scanning technique. So in the first stages of improvement, we created a semi-automated GNU-Radio script to run the frequencies in each position and log the measurements directly to an excel file.

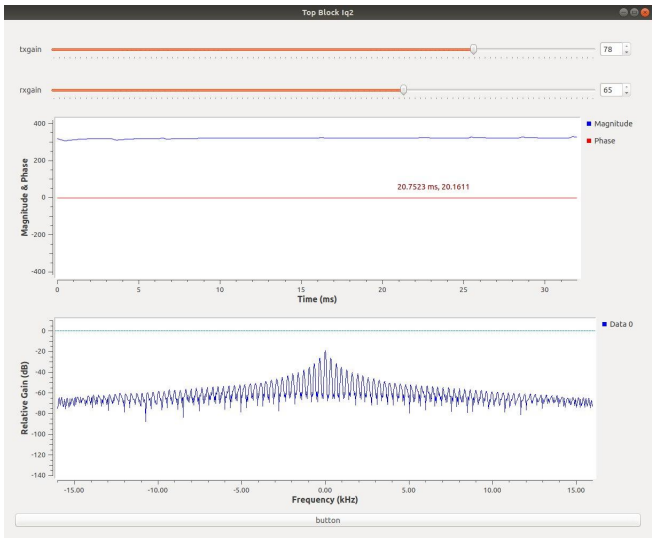


Figure 4.1.14: Semi-automatic GNU-Radio scanning

The script was written to a GNU-Radio python block (seen in 4.1.7) and with each 16 frequency changes the user had to press the GUI "Button" after chancing to the next position in the foam-board array. This the script was written to this block:

```
import numpy as np
import time
from threading import Timer
from gnuradio import gr
import pmt

flag = 0;
class blk(gr.sync_block):  # other base classes are basic_block, decim_block, interp_block
        """Embedded Python Block example - a simple multiply const"""

        def __init__(self, start_sampling=1.0):  # only default arguments here
                """arguments to this function show up as parameters in GRC"""
                gr.sync_block.__init__(
                    self,
                    name='Write to file',   # will show up in GRC
                    in_sig=[np.complex64],
                    out_sig=[np.complex64]
                )
                global flag
                self.freqs = [2.2e9,2.25e9,2.3e9,2.35e9,2.4e9,2.45e9,2.5e9,2.55e9,2.6e9,2.65e9,2.7e9
                self.nfreqs = 16
                print('global flag', flag);
                if (flag == 0):
                        self.cnt = 0

                self.boolTakeSample = False;
                self.boolTookSample = False;
                # if an attribute with the same name as a parameter is found,
                # a callback is registered (properties work, too).
                self.start_sampling = start_sampling
```

```
        self.message_port_register_out(pmt.intern('msg_out'))
        def doTimer():
                if self.boolTakeSample == True:
                        Timer(0.5, doTimer, ()).start();
                        return;
                if self.boolTookSample == False:
                        self.boolTookSample = True;
                        self.boolTakeSample = True;
                        Timer(1, doTimer, ()).start();
                        return;
                else:
                        self.boolTookSample = False;
                        self.boolTakeSample = False;

                if self.cnt == self.nfreqs - 1:
                        print("Press_QT_Button_to_Continute!");
                        while (self.start_sampling == 0):
                                continue
                        self.cnt = -1;


                self.cnt += 1;
                print(self.cnt)
                d_ChgFreq = pmt.make_dict()
                d_ChgFreq = pmt.dict_add(d_ChgFreq, pmt.intern('freq'),pmt.from_double(self.
                self.message_port_pub(pmt.intern("msg_out"), d_ChgFreq)
                Timer(2, doTimer, ()).start();
                return;

        if (flag == 0):
                Timer(1, doTimer, ()).start();
        flag = 1;


def work(self, input_items, output_items):

        def get_result():
                if self.boolTakeSample == True:
                        f = open(r'/home/lte/output2.csv', 'a+');
                        f.write('{0},'.format(input_items[0][0]));
                        print('{0},'.format(input_items[0][0]))
                        f.close()
                        self.boolTakeSample = False;
        get_result();
        output_items[0][:] = input_items[0];
        """example: multiply with constant"""
        return len(output_items[0])
```

After several semi-automatic scanning, we came to the conclusion that the RMA image reconstruction is indeed working. Therefore we went a step further to fully automatize the scanning. We constructed the final imaging rig described in 2.5.0.2, to which we had written new scripts that uses the USRP native driver instead of the GNU-Radio. We also had to write code to run the 3D printer movement between the acquirement of the measurements from the SDR. This automation allowed us to increase the number of frequencies and positions resulting in a wider range of resolutions and detection abilities. Only after reaching this point the project that we were able to focus on the object recognition and the RMA itself.

4.2 RSSI Triangulation

Triangulation is a known procedure made by many devices in order to obtain a relative position in space. Before we were to pinpoint the location of out sensor we had to tweak and test several issues in the triangulation algorithm.

4.2.1 Propagation Models

The basis of triangulation is the knowledge of the distance to several anchor points in space. But in order to obtain the distance we had to apply a propagation model to the signal's path-loss. There are many propagation models [Wik20d] to choose from, not all would result in good estimations to our indoor environment. After constructing our setup (described in 2.5.0.1) we had to vary our propagation model to better fit the results obtained. Our environment was that of an office building making the free-space path-loss give us relative errors results. The ITU model was found to be better for the testing environment in hand and was eventually combined with the simplified path-loss model. The resulting indoor localization procedure is detailed in 5.1.1.

4.2.2 RSSI, SSID and AP capabilities

All APs have different capabilities and we need to distinguish each AP from another in order to apply the triangulation algorithm. In order to obtain the information needed we had to turn to RFC5416 (`https://tools.ietf.org/html/rfc5416`). For the Tx power of an AP we found the Tx Power beacon frame in section 6.18 (`https://tools.ietf.org/html/rfc5416#section-6.18`):

```
6.18.  IEEE 802.11 Tx Power

   The IEEE 802.11 Tx Power message element value is bi-directional.
   When sent by the WTP, it contains the current power level of the
   radio in question.  When sent by the AC, it contains the power level
   to which the WTP MUST adhere.

        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |    Radio ID   |    Reserved   |        Current Tx Power       |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

This frame allows us to the Tx power and along with the RSSI calculate the path-loss for the specific Radio-ID (or SSID.

For the RSSI information at the sensor (receiver), we found to be included in each IEEE 802.11 frame and thus could be extracted.

```
   IEEE 802.11 Frame Info:  When an IEEE 802.11 frame is received from a
```

```
station over the air, it is encapsulated and this field is used to
include radio and PHY-specific information associated with the
frame.

The IEEE 802.11 Frame Info field has the following format:

0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     RSSI      |     SNR       |           Data Rate           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

RSSI:   Received Signal Strength Indication (RSSI) is a signed,
   8-bit value.  It is the received signal strength indication, in
   dBm.
```

Once all the information needed to apply a triangulation algorithm was found to be in had we could start our implementation.

Chapter 5

Implementations

5.1 Achieved Objectives

5.1.1 RSSI Distance Measurements

We've used two different formulas to estimate the distance of the WiFi client from its neighbors AP. The first formula is given by the ITU [Wik18]

$$L = 20\log_{10} f + N\log_{10} d + P_f\left(n\right) - 28 \tag{5.1.1}$$

where,
L the total path loss in dB.
f is the frequency in usage.
d the distance in meters.
n the number of floors between the two nodes.
P_f the floor loss penetration factor.
28 a constant that is given in the model.

Our interpretation to this model is given by

$$d = \exp\left(\log 10 \cdot \frac{L - 20\log_{10} f - P_f\left(n\right) + 28}{N}\right) \tag{5.1.2}$$

where the results of the distance d are shown in figure 5.1.1. Another model by A.Goldsmith [Gol05, p.46], simplified path-loss given by,

$$P_r = P_t + K - 10\gamma\log_{10}\left[\frac{d}{d_0}\right] \tag{5.1.3}$$

where,
P_r is the received power (RSSI).
P_t is the transmitted power.
K is a unit-less constant that depends on the antenna characteristics.

γ is the path-loss exponent.
d_0 is the refrenced distance [10-100]m.

Our interpretation to this model is given by

$$d = \exp\left(\log 10 \cdot \frac{P_t - P_r + K}{10\gamma}\right) \cdot d_0 \tag{5.1.4}$$

This implementation was done using the device branded by WeMos D1 and is given in figure 5.1.2.

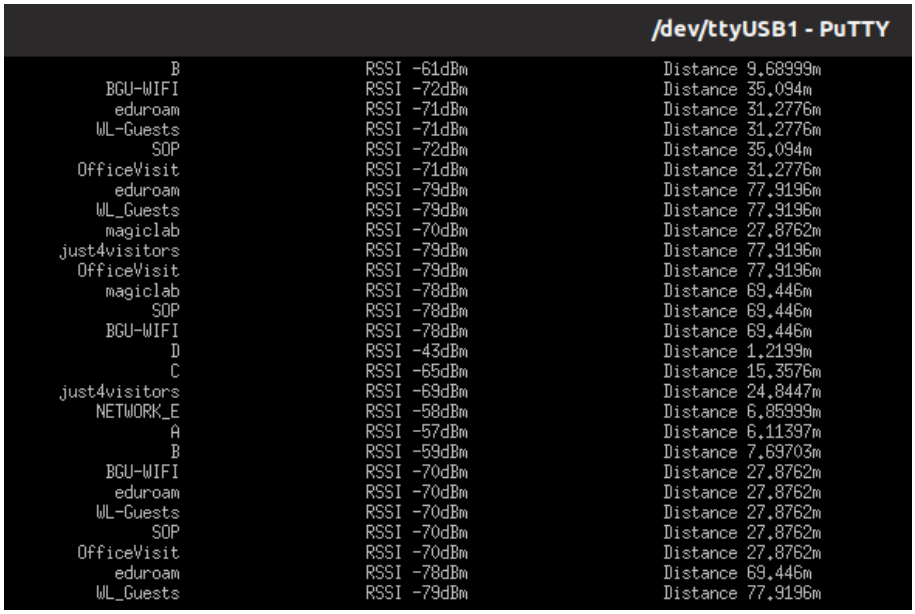


Figure 5.1.1: Results of RSSI based distances

In figure 5.1.1 appears a list of all the surrounding SSIDs broadcasted from nearby APs in range. The list includes the SSID, the RSSI and the calculated distance (using formula 5.1.4) respectively. The WeMos D1 node is running an operation system known as MicroPython, and the python code that runs it, is at appendix A.1.

Figure 5.1.2: WeMos D1 mini

5.1.2 RSSI Indoor Localization

This method described in [Wal10] and is based on the previous paragraph, where the localization is calculated by the following equation

$$\mathbf{A}\mathbf{x} = b \tag{5.1.5}$$

where,
$\mathbf{x}$ is the vector unknown sensor location (x, y) compared to the other anchors.
$\mathbf{A}$ is the matrix of the $n-1$ anchors compared to the sensor location, given by

$$A = \begin{bmatrix} 2(x_n - x_1) & 2(y_n - y_1) \\ 2(x_n - x_2) & 2(y_n - y_2) \\ \vdots & \vdots \\ 2(x_n - x_{n-1}) & 2(y_n - y_{n-1}) \end{bmatrix} \tag{5.1.6}$$

b is the vector of distances and locations of the other anchors and to the other anchors, given by

$$b = \begin{bmatrix} r_1^2 - r_n^2 - x_1^2 - y_1^2 + x_n^2 + y_n^2 \\ r_2^2 - r_n^2 - x_2^2 - y_2^2 + x_n^2 + y_n^2 \\ \vdots \\ r_{n-1}^2 - r_n^2 - x_{n-1}^2 - y_{n-1}^2 + x_n^2 + y_n^2 \end{bmatrix} \tag{5.1.7}$$

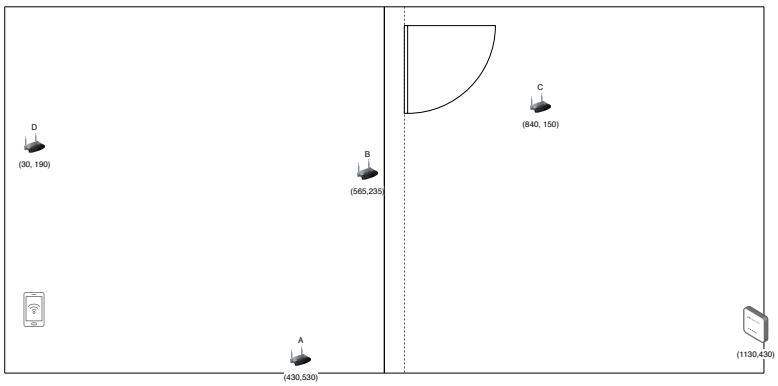


Figure 5.1.3: RSSI triangulation inside 512/37 lab

Given the figure 5.1.3 of our space, the indoor localization, is calculated using the interpreted equation

$$(x, y) = \left(\mathbf{A}^T \mathbf{A}\right)^{-1} \mathbf{A}^T b \tag{5.1.8}$$

where, the distances r_i are calculated using one of the model mentioned before, and the inverse of the matrices is calculates using the Moore-Penrose inverse methods that is pseudoinverse [Wik19d]. The Jupyter notebook code is in the appendix A.3.

We placed 4 WiFi routers in lab 512 and 521 and measured their location using a measuring meter. Each router broadcasts a SSID (A, B, C, D) which serves as an anchor point in space. The algorithm is dependent on these points to calculate the estimated location of a sensor relative to these points.

Results showed an accuracy of about 0.3m.

5.1.3 Radio Frequency Imaging

The following is a description from Simon Scott's work [Sco17] on Radio Frequency Imaging.

5.1.3.1 Range Migration Algorithm

Radio Frequency Imaging is the technology for capturing 3D images of objects and people in an indoor environment. Compared to many other imaging technologies, radio frequency imaging suffers from significantly lower resolution and the resolution is limited to half the wavelength of the carrier frequency. Therefore, microwave imaging is best suited to large indoor environments where only moderate resolution is required.

From sub-section 2.1.2, only one radio frequency imaging technique was found to be implementable in the time scope of this project and with the confinement of using a fairly standard WiFi-based radio. Therefore, we have selected to focus on the work of Simon Scott [Sco17] whose work relays on an algorithm for microwave imaging called range migration algorithm (RMA) which has its origin in seismic ground scanning. The range migration algorithm has its origins in acoustic imaging. It was first used in geological surveying applications [BRS06], where acoustic waves are used to image underground objects. It was also used in its early days for ultrasound medical imaging [Boy+71]. RMA takes into account the wavefront curvature, and can also be computed efficiently. This efficiency comes from the fact that the RMA uses the Dix approximation, i.e. only direct reflections are considered and multipathing is ignored. This approximation allows the algorithm to be expressed using Fourier transforms and computed using fast Fourier transforms (FFTs). RMA is also known as the backward-wave reconstruction algorithm [Boy+71], as it forms an image by coherently integrating the reflected wave over a synthesized aperture, and then back-projecting it into the scene. The RMA is derived here from first principles for multiple different antenna array configurations. The conventional 3-D RMA assumes a planar rectangular antenna array of colocated transmit and receive antennas, with antennas spaced less than a wavelength apart. This arrangement is the easiest to analyze and generally produces the highest resolution images.

5.1.3.2 Variables and Coordinate System

Figure 5.1.4 establishes a unified coordinate system to aid in the explanation and derivation of the different RMA variants. This coordinate system, as well as the common variables, are defined as follows:

- The antenna array lies in the xy-plane at $z = Z_0$.
- The transmitter transmits a continuous wave (CW) at frequency ω (rad/s), that is discretely stepped from ω_{min} to ω_{max}.
- $f(x, y, z)$ is the reflectively function of the scene, i.e. the image we are trying to recreate. Note that "imaging the scene" actually means finding the function f that defines how well each point in the scene reflects microwaves.
- $s(x_a, y_a, \omega)$ is the complex reflection recorded at antenna position (x_a, y_a, Z_0) and at frequency ω, when both the transmitting and receiving antennas are co-located.
- $k = \frac{\omega}{c}$ is the wavenumber of the transmitted signal.
- k_x, k_y, k_z are the spatial frequency variables of x, y, z.

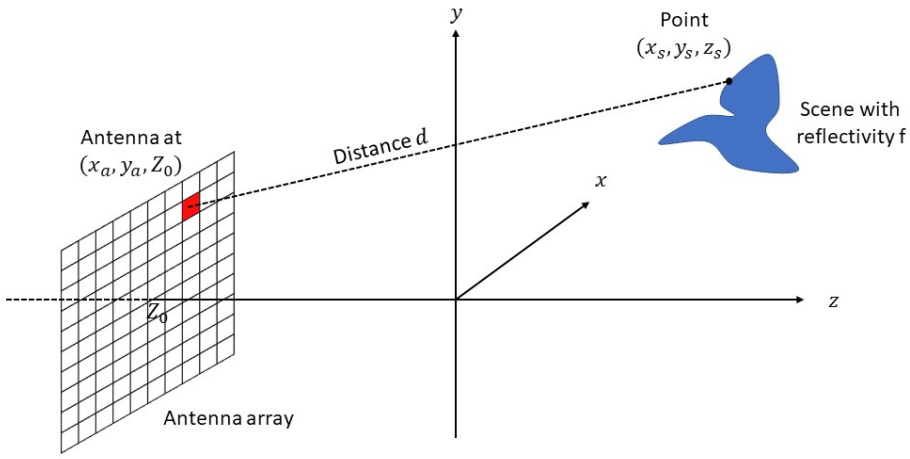


Figure 5.1.4: Scene geometry for the derivation of the range migration algorithm

Co-located Range Migration Algorithm Only two antennas in the array are active in the co-located algorithm at any one time: the transmitting antenna and its neighboring receive antenna. A continuous-wave (CW) signal is transmitted by the transmitter, which reflects off objects in the scene and this reflection is coherently recorded by the neighboring receiver. This is repeated for each frequency step from ω_{min} to ω_{max}. After recording reflections $s(x_a, y_a, \omega)$ at all frequencies, the next two antennas in the array operate as the transmit/receive pair and the process is repeated. The co-located algorithm assumes that the transmitting and receiving antennas are co-located (i.e. in the exact same position), but in practice co-location is approximated by having neighboring antennas act as a transmit/receive pair. Therefore, a common technique is to take position (x_a, y_a) as the position halfway between the neighboring antennas.

5.1.3.3 Derivation of the colocated range migration algorithm

The round-trip phase delay from transmit/receive antenna at (x_a, y_a, Z_0) to point reflector in the scene at co-ordinate (x, y, z) is:

$$2k \times d, \text{where distance } d = \sqrt{(x - x_a)^2 + (y - y_a)^2 + (z - Z_0)^2} \tag{5.1.9}$$

Attenuation effects due to path loss are ignored in the co-located range migration algorithm, as they are difficult to handle and have little effect on the resulting image quality [Xun+16]. Therefore, if the point reflector at co-ordinate (x, y, z) has reflectively $f(x, y, z)$, the response s recorded at the antenna at frequency ω will be:

$$s(x_a, y_a, \omega) = f(x, y, z)e^{-j2k\sqrt{(x-x_a)^2+(y-y_a)^2+(z-Z_0)^2}} \tag{5.1.10}$$

By regarding the scene as a collection of point reflectors, the combined reflection recorded at antenna (x_a, y_a, Z_0) is obtained by integrating 5.1.10 over the scene:

$$s(x_a, y_a, \omega) = \iiint_{scene} f(x, y, z) e^{-j2k\sqrt{(x-x_a)^2+(y-y_a)^2+(z-Z_0)^2}}\, dx\, dy\, dz \tag{5.1.11}$$

The square-root in the exponential term in 5.1.11 makes the expression difficult to invert to obtain $f(x, y, z)$. Fortunately, the exponential term describes a spherical wave, which can be expressed as a sum of plane waves [Che95], again ignoring amplitude effects:

$$e^{-j2k\sqrt{(x-x_a)^2+(y-y_a)^2+(z-Z_0)^2}} = \iint e^{-j(k_{x_a}(x-x_a)+k_{y_a}(y-y_a)+k_z(z-Z_0))}\, dk_{x_a}\, dk_{y_a} \tag{5.1.12}$$

By combining 5.1.11 and 5.1.12 and rearranging the order of the integrals, we obtain:

$$s(x_a, y_a, \omega) = \iint [\iiint_{scene} f(x, y, z) e^{-j(k_{x_a}x+k_{y_a}y+k_z z)}\, dx\, dy\, dz] \\ e^{jk_z Z_0} e^{j(k_{x_a}x_a+k_{y_a}y_a)}\, dkx_a\, dky_a \tag{5.1.13}$$

The inner triple integral represents the 3D spatial Fourier transform of $f(x, y, z)$, while the outer double integral can be expressed as the 2D inverse Fourier transform with respect to (k_{x_a}, k_{y_a}). We therefore rewrite 5.1.13 as:

$$s(x_a, y_a, \omega) = FT_{2D}^{-1}(FT_{3D}(f(x, y, z))e^{jk_z Z_0}) \tag{5.1.14}$$

Inverting the Fourier transforms, we can reconstruct the original scene using:

$$f(x, y, z) = FT_{3D}^{-1}(\Phi(FT_{2D}(s(x_a, y_a, \omega))e^{-jk_z Z_0})) \tag{5.1.15}$$

where the inner 2D Fourier transform is from (x_a, y_a) space to (k_{x_a}, k_{y_a}) space, and the outer 3D inverse Fourier transform is from $(k_x, k_y, k_z$ space to (x, y, z) space. To make the domains of these Fourier transforms compatible, $\Phi\cdot$ is the Stolt transform [Sto78] from $(k_{x_a}, k_{y_a}, \omega)$ space to (k_x, k_y, k_z) space, according to:

$$\begin{aligned} k_x &= k_{x_a} \\ k_y &= k_{y_a} \\ k_z &= \sqrt{4\frac{\omega^2}{c^2} - k_x^2 - k_y^2} \end{aligned} \tag{5.1.16}$$

The entire co-located RMA block diagram in showen in 5.1.5

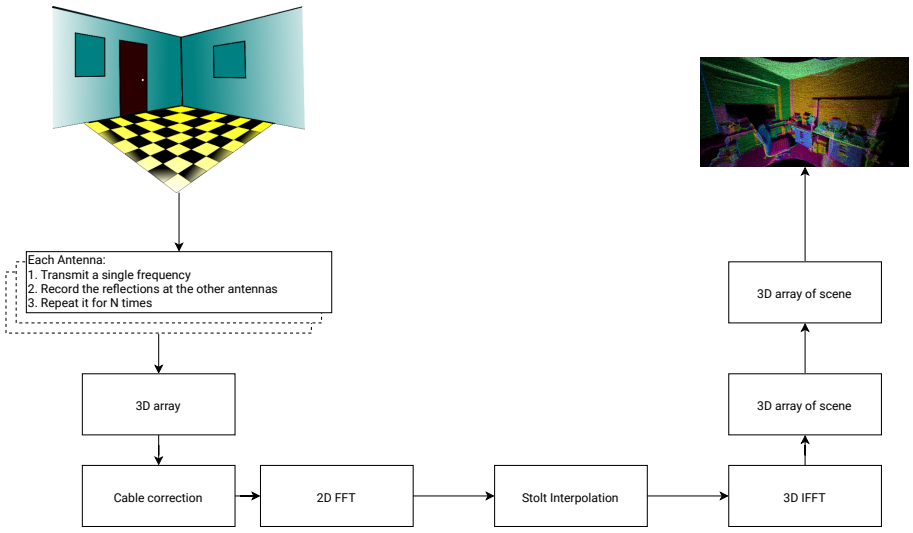


Figure 5.1.5: Block diagram for the colocated RMA

5.1.3.4 Implementation

Using the physical infrastructure described in 2.5.0.2, we initialized scans using the scanning script described in B.1 which controlled the XY-axis movement of the rig, the USRP RF transmission and reception and the logging of each value into a measurements file. Each scan had to be fed a set of co-ordinates for the structure of the antenna array (the (x_a, y_a, Z_0) points) which was created using the script shown inB.2. After obtaining the measurements file of a certain scene it was fed into image reconstruction algorithm script (shown in B.3)in which the entire Z axis was summed and normalized in order to output a 2D imaged of the reflectively in the vicinity of the scanner. Results of scans made can be seen in 5.1.3.6.

5.1.3.5 Scripts and Code Runtime

Every script and code that have been written, was written in Python, sometimes the runtime of scripts depend on hardware limitations, for example, the mechanical grid, where the velocity of the adapter on the grid is different for axis X and axis Y. After each movement, the adapter needs to rest in its place till it will stand still which results in up to 24 hours of active continuous scanning. When working with big data such as 3D tensor, methods, and algorithms such as Fast Fourier Transform that are $O(n \log n)$, as long the resolution of the image is better, it cost time, and for example, the creation of the simulated scenes, given a n_0 dots along axis X, and m_0 dots along axis Y and given k_0 frequencies, we have an input image to be simulated that is given in BMP format with n_1 and m_1, the runtime is $O(n_0 \cdot m_0 \cdot k_0 \cdot n_1 \cdot m_1)$, to create the simulated scene using Figure 2.4.1 $n_0 \cdot m_0 \cdot k_0 \cdot n_1 \cdot m_1 = 128 \cdot 128 \cdot 64 \cdot 128 \cdot 128$ that

is 17179869184 which is 17 billion 179 million 869 thousand 184, and we must not forget that each iteration includes sub-iteration of Taylor Approximation because of the usage of complex math operation in each iteration, but in real time, the creation of that scene took a half hour.

Some parts of the code have been optimized to work in parallel using the built-in methods of NumPy, and trying have been made to adapt the CUDA extension for NumPy (CuPy), but most of our efforts done for the optimization of the images and not in the runtime, which can be optimized in future work.

5.1.3.6 Experimentation Results

The first object scanned was a aluminum foil rectangle (seen in Figure 5.1.6). The scan was done using 32X32 XY positions and 64 frequencies.

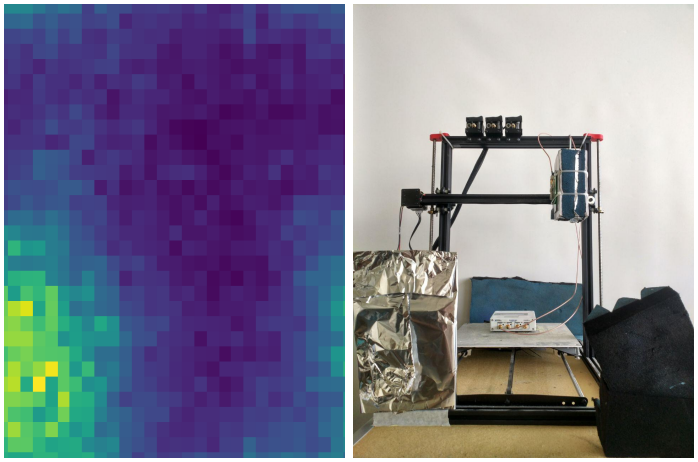


Figure 5.1.6: Aluminum foil rectangle

The bright yellow pixels depict an area where there is high reflectivity due to the aluminum reflector.

This scan was also reconstructed in 3D layers and as seen in figure 5.1.7. The layering clearly shows about 12 layers where the reflectively of the aluminum rectangle is high fallowed by layers where there is no visible relativity of an object. The layering is mirrored in the z-axis, meaning the first and last layers are the same and so forth. This is due to the FFT properties during the image reconstruction and while summing the z-axis this effect is with no effect on the result.

This aluminum rectangle reflector was used in many scans made during our experimentation period, where the displayed result is the best of them.

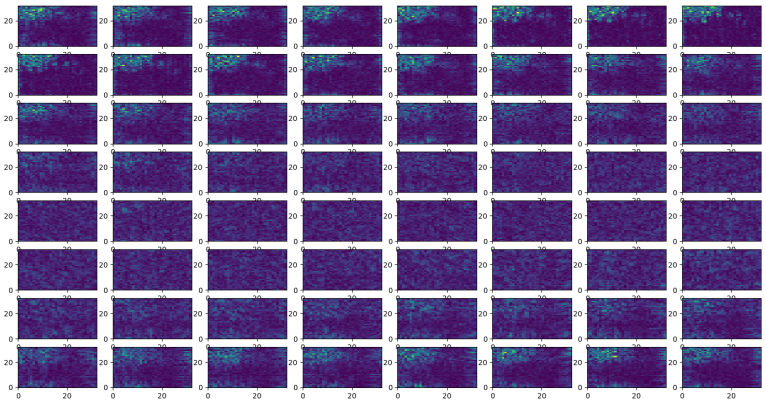


Figure 5.1.7: Layers of Aluminum rectangle reflector in Z axis

Another object scanned was an aluminum foil covered ring. the resulting reconstructed image in figure 5.1.8 shows only a part of the ring due to being located close to the scanner and parts of the ring were outside its point of view. Therefore the resulting image is clipped to the left (images are mirrored so to shoe the point of view of the scanner and not the viewer).

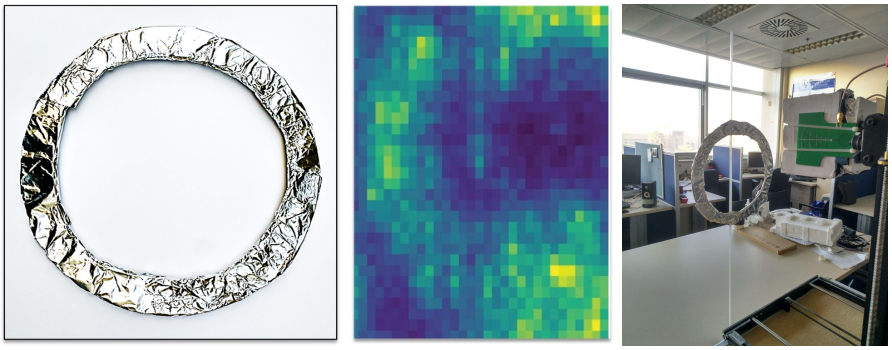


Figure 5.1.8: Aluminum foil ring

After scanning several simple objects we turned to scan a more complex shaped object - the Ben Gurion university logo.

For that, we have cut the logo out of foam board and covered it with one layer of aluminum foil on which we did most of our experiment scans trying to improve the resulting reconstructed image so it would be more understandable to the human eye.

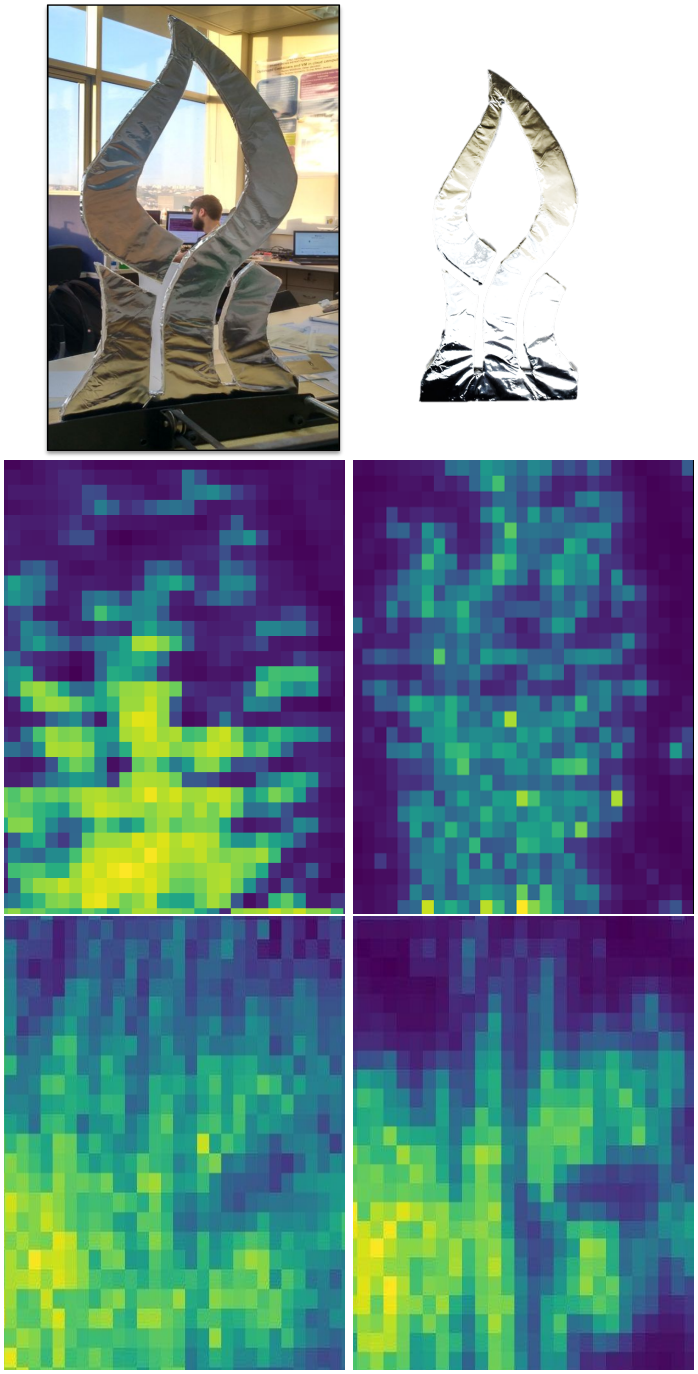


Figure 5.1.9: Aluminum foil BGU logo

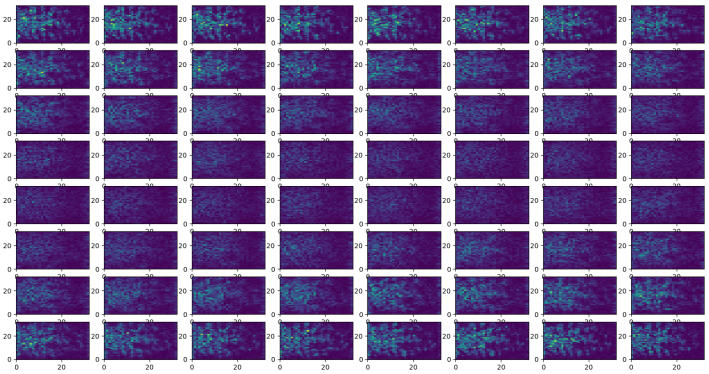


Figure 5.1.10: Layers of the BGU logo in Z axis

In total we did about 60 scans of different objects with an average 20 hours per scan. Most results gave images that were hard to identify the object scanned and therefore were left out of this report. All files including more images can be found in the project drive linked in 6.0.1

Chapter 6

References, Conclusions & Future Work

6.0.1 References

All the work made during this project is accessible in the project's Drive.

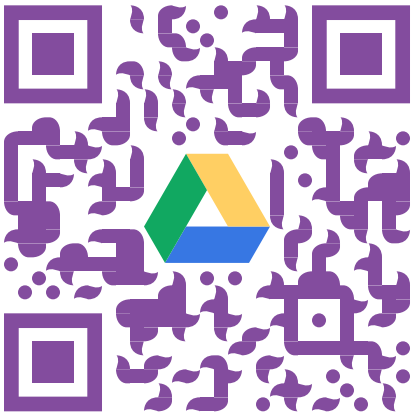


Figure 6.0.1: Scan me

6.1 Conclusions

This project has created the infrastructure for sensor positioning and imaging of its environment. We had succeeded in locating the position of the WiFi sensor node as well as create an image based on the radio abilities of its radio. This has not been done before under the limitations of WiFi radio yet was known to

be feasible[HNG14].

This project was challenging and demanding yet led to great achievements.

Though thought to be ambitious and overachieving, we had accomplished most of what we set for. Even though the imaging results and localization are imprecise and lack better resolution, they show the basic results needed to fulfill the goals of this project. We would have loved it if we could have improved the results we obtained, but this was not possible under the time and budget limits. As an example, the more isolated and precise antenna array would have let to better images. This leaves the door open for future improvements to be done in other projects that may choose to continue this work.

6.1.1 Future work

We had only scraped the edge in the field of wireless sensing. Many more proprieties of wireless transmission can be exploited in a new and inspiring way.

For the scope of this project, we set a goal to create the infrastructure for new formerly neglected information gathering from wireless sensors. Our mind was set to allow sensors to gain knowledge about their environment in order to be able to integrate themselves into a WSN without the need for intervention of a human being. As this project concludes a new range of opportunities opens up. Now, whit the ability to scan objects and rooms, a data-set could be build to later allow statistical inference regarding the recognition of the objects in sensors' vicinity.

Another future work that could fork from this project is the recognition of the movement in a room using the side results obtained during our experiments (4.1.5). One could enhance the capabilities of motion estimation in the room, including position difference, Doppler shifts, and person count, all using the Continues wave (CW) transmission system embedded in our project.

Many more statistical inferences can be made upon the WiFi radio signals. Image creation and positioning is only a fraction of the vast possibilities. Things like security applications (sensing motion), health care (sensing heart rate and fall detection), IoT applications such as sensory supplementary information (like localization and imaging), and many more.

This was only the beginning...

Chapter 7

Bibliography

Books

[Che95] W.C. Chew. *Waves and Fields in Inhomogenous Media*. IEEE Press Series on Electromagnetic Wave Theory. Wiley, 1995. ISBN: 9780780347496. URL: https://books.google.co.il/books?id=fdmbTY-F3%5C_cC.

[Mar03] Nancy Alonistioti Markus Dillinger Kambiz Madani. *Software Defined Radio: Architectures, Systems and Functions*. Wiley Series in Software Radio. Wiley, 2003. ISBN: 0470851643,9780470851647,9780470865019.

[CD04] Diane Cook and Sajal Das. *Smart Environments: Technology, Protocols and Applications (Wiley Series on Parallel and Distributed Computing)*. New York, NY, USA: Wiley-Interscience, 2004. ISBN: 0471544485.

[Gol05] Andrea Goldsmith. *Wireless Communications*. Cambridge University Press, 2005. DOI: 10.1017/CBO9780511841224.

[Wal10] Christian Poellabauer Waltenegus Dargie. *Fundamentals of Wireless Sensor Networks: Theory and Practice (Wireless Communications and Mobile Computing)*. 1st ed. 2010. ISBN: 9780470666395. URL: http://gen.lib.rus.ec/book/index.php?md5=F771E9A285613D9BE21130E07F1BA923.

[LW19] K.J.R. Liu and B. Wang. *Wireless AI: Wireless Sensing, Positioning, IoT, and Communications*. Cambridge University Press, 2019. ISBN: 9781108497862. URL: https://books.google.co.il/books?id=YvWpDwAAQBAJ.

Articles

[Sto78] R. H. Stolt. "Migration by Fourier transform". In: *Geophysics* 43.1 (Feb. 1978), pp. 23–48. ISSN: 0016-8033. DOI: 10.1190/1.1440826.

eprint: https://pubs.geoscienceworld.org/geophysics/article-pdf/43/1/23/3115559/23.pdf. URL: https://doi.org/10.1190/1.1440826.

[BRS06] G. BOLONDI, Fabio Rocca, and S. SAVELLI. "A frequency domain approach to two-dimensional migration". In: *Geophysical Prospecting* 26 (Apr. 2006), pp. 750–772. DOI: 10.1111/j.1365-2478.1978.tb01632.x.

[SKR11] Lei Shi, Sarath Kodagoda, and Ravindra Ranasinghe. "Fast indoor scene classification using 3D point clouds". In: *Proceedings of the 2011 Australasian Conference on Robotics and Automation* (Jan. 2011).

[Sen+12] Souvik Sen et al. "Spot Localization Using PHY Layer Information". In: (June 2012). URL: https://www.microsoft.com/en-us/research/publication/spot-localization-using-phy-layer-information/.

[Var+12] T. Varvadoukas et al. "Indoor Furniture and Room Recognition for a Robot Using Internet-Derived Models and Object Context". In: (Dec. 2012), pp. 122–128. ISSN: null. DOI: 10.1109/FIT.2012.30.

[HNG14] Donny Huang, Rajalakshmi Nandakumar, and Shyamnath Gollakota. "Feasibility and Limits of Wi-fi Imaging". In: SenSys '14 (2014), pp. 266–279. DOI: 10.1145/2668332.2668344. URL: http://doi.acm.org/10.1145/2668332.2668344.

[Hua+15] Yan Huang et al. "Robust Localization Algorithm Based on the RSSI Ranging Scope". In: *International Journal of Distributed Sensor Networks* 2015 (Feb. 2015), pp. 1–8. DOI: 10.1155/2015/587318.

[HR16] Philipp Holl and Friedemann Reinhard. "Holography of Wi-Fi radiation". In: *Physical Review Letters* 118 (Nov. 2016). DOI: 10.1103/PhysRevLett.118.183901.

[HR17] Philipp M. Holl and Friedemann Reinhard. "Holography of Wi-fi Radiation". In: *Physical Review Letters* 118.18 (May 2017). ISSN: 1079-7114. DOI: 10.1103/physrevlett.118.183901. URL: http://dx.doi.org/10.1103/PhysRevLett.118.183901.

[Sco17] Simon Scott. "Three-Dimensional Microwave Imaging for Indoor Environments". In: UCB/EECS-2017-191 (Dec. 2017). URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2017/EECS-2017-191.html.

References

Miscellaneous

[Dep16] Statista Research Department. *IoT: number of connected devices worldwide 2012-2015*. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. [Online]. 2016.

[Wik18] Wikipedia contributors. *ITU model for indoor attenuation — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=ITU_model_for_indoor_attenuation&oldid=857939922. 2018.

[GNU19] GNU Radio contributors. *GNU Radio - The Free & Open Source Radio Ecosystem GNU Radio*. [Online]. 2019. URL: https://www.gnuradio.org/about/.

[Lim19] LimeSDR. *LimeSDR - Lime Microsystems*. [Online]. 2019. URL: https://limemicro.com/products/boards/limesdr/.

[The19] Themis Karafasoulis & Georgios Niras. *Ettus USRP B210 — Wiki*. [Online]. 2019. URL: https://gitlab.com/librespacefoundation/sdrmakerspace/radtest/wikis/Ettus%20USRP%20B210.

[Wik19a] Wikipdia contributers. *MIMO — Wikipedia, The Free Encyclopedia*. [Online]. 2019. URL: https://en.wikipedia.org/w/index.php?title=MIMO&oldid=929449780.

[Wik19b] Wikipedia contributors. *Autoencoder — Wikipedia, The Free Encyclopedia*. [Online]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Autoencoder&oldid=929868789.

[Wik19c] Wikipedia contributors. *Holography — Wikipedia, The Free Encyclopedia*. [Online; accessed 11-December-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Holography&oldid=929908125.

[Wik19d] Wikipedia contributors. *Moore–Penrose inverse — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Moore%E2%80%93Penrose_inverse. 2019.

[Wik19e] Wikipedia contributors. *Multipath propagation — Wikipedia, The Free Encyclopedia*. [Online]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Multipath_propagation&oldid=929145405.

[Wik19f] Wikipedia contributors. *Neural network — Wikipedia, The Free Encyclopedia*. [Online]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Neural_network&oldid=925749909.

[Wik19g] Wikipedia contributors. *Software-defined radio — Wikipedia, The Free Encyclopedia*. [Online]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Software-defined_radio&oldid=930131789.

[Wik20a] Wikipedia contributors. *Continuous-wave radar — Wikipedia, The Free Encyclopedia*. 2020. URL: https://en.wikipedia.org/w/index.php?title=Continuous-wave_radar&oldid=973727344.

[Wik20b] Wikipedia contributors. *Electromagnetic shielding — Wikipedia, The Free Encyclopedia*. 2020. URL: https://en.wikipedia.org/w/index.php?title=Electromagnetic_shielding&oldid=969747192.

[Wik20c] Wikipedia contributors. *Inverse problem — Wikipedia, The Free Encyclopedia*. 2020. URL: https://en.wikipedia.org/w/index.php?title=Inverse_problem&oldid=971978242.

[Wik20d] Wikipedia contributors. *Radio propagation model — Wikipedia, The Free Encyclopedia*. 2020. URL: https://en.wikipedia.org/w/index.php?title=Radio_propagation_model&oldid=934984922.

[Wik20e] Wikipedia contributors. *Vivaldi antenna — Wikipedia, The Free Encyclopedia*. 2020. URL: https://en.wikipedia.org/w/index.php?title=Vivaldi_antenna&oldid=942916305.

[Mic] MicroPython community. *MicroPython - Python for Microcontrollers*. https://micropython.org/.

Appendices

Appendix A

Localization Codes

A.1 MicroPython code

```
import network
import math
class AP:
    def __init__(self, ssid, bssid, channel, rssi, authmode, hidden):
        self.bssid = list(bssid)
        self.bbssid = bssid
        self.ssid = ssid.decode('cp1252');
        self.ssid_hebrew = ssid.decode('cp1255');
        self.channel = int(channel);
        self.rssi = rssi;
        self.authmode = authmode;
        self.hidden = hidden;
        self.distance = float(-1.0)
        self.speedoflight = 299792458;
        self.gain = 20;
    def __str__(self):
        return "AP:\tSSID {0}\tCh {1}\tRSSI {2}".format(self.ssid, self.channel, self.rssi);
    def __repr__(self):
        return self.ssid;
    def getRSSI(self):
        return self.rssi;
    def getSSID(self):
        return self.ssid;
    def getSSID_Hebrew(self):
        return self.ssid_hebrew;
    def getChannel(self):
        return self.channel;
    def getBSSID(self):
        return self.bssid;
    def getBBSSID(self):
        return self.bbssid;
    def setDistance(self, d):
        self.distance = d;
    def getDistance(self):
        return self.distance;
    def getFrequency(self):
        if (self.channel == 14):
            return 2484;
        return 2407 + 5 * self.channel;
    def getWaveLength(self):
        return float(self.speedoflight) / (self.getFrequency()*1000000);
class rssi_parser:
    def __init__(self):
        self.tx_power = 19;
        self.aps = {}
        self.nic = network.WLAN(network.STA_IF);
        self.nic.active(True);

    def parse_aps(self):
        results = self.nic.scan();
        for res in results:
            ap = AP(res[0], res[1], res[2], res[3], res[4], res[5]);
            self.aps.update({ap.getBBSSID() : ap})
    def calculate_distance_byGold(self):
        d0 = 5;
        gamma = 2;
        for ap in self.aps.values():
            K = -20*math.log(4*math.pi*d0 / ap.getWaveLength())/math.log(10)
            ap.setDistance(math.exp(math.log(10)*(-ap.gain + self.tx_power + K - ap.getRSSI())/(10*gamma))*d0)
    def calculate_distance_byITU(self):
        for ap in self.aps.values():
            ap.setDistance(math.exp(math.log(10)* ((self.tx_power - ap.getRSSI() - 20*math.log(ap.getFrequency())/math.log(10)-0 + 28)/30)));
    def show(self):
        for ap in self.aps.values():
            print("{0:>20}\t\t\tRSSI {1}dBm\t\t\tDistance {2}m".format(ap.getSSID(), ap.getRSSI(), ap.getDistance()))
```

A.2 Indoor Localization Notebook

[1]:
```
import numpy as np
from scipy.constants import *
import IPython.display
import PIL.Image
from sympy import Matrix, MatrixSymbol, Eq, Symbol,init_printing
from IPython.display import display, Math, Markdown, Latex
init_printing()
```

[2]:
```
aps = [{'ssid': 'D', 'quality': '70/70', 'signal': -24, 'x' : 0.30, 'y' : 1.90} ,
{'ssid': 'C', 'quality': '70/70', 'signal': -25, 'x' : 8.40, 'y' : 1.50} ,
{'ssid': 'B', 'quality': '70/70', 'signal': -22, 'x' : 5.65, 'y' : 2.35} ,
{'ssid': 'A', 'quality': '70/70', 'signal': -23, 'x' : 4.30, 'y' : 5.30} ]
```

[3]:
```
tx_power = 19
```

[4]:
```
f = 2415000000
def wave(freq):
    return speed_of_light / freq;
```

[5]:
```
gamma = 3.5
d0 = 10
```

[6]:
```
K = -20*np.log10(4*np.pi*d0 / wave(f))
K
```

[6]:
-60.106125923633996

[7]:
```
distances = {}
for rssi in aps:
    rssi['distance'] = np.exp(np.log(10) * (tx_power + K - rssi['signal'] )/(10␣
 ↪* gamma))*d0
    print (rssi)
```

```
{'ssid': 'D', 'quality': '70/70', 'signal': -24, 'x': 0.3, 'y': 1.9, 'distance':
3.245290250294361}
{'ssid': 'C', 'quality': '70/70', 'signal': -25, 'x': 8.4, 'y': 1.5, 'distance':
3.465971390949713}
{'ssid': 'B', 'quality': '70/70', 'signal': -22, 'x': 5.65, 'y': 2.35,
'distance': 2.8451861628068205}
{'ssid': 'A', 'quality': '70/70', 'signal': -23, 'x': 4.3, 'y': 5.3, 'distance':
3.03866005246217}
```

[8]:
```
matA = np.array([np.array([2*(aps[-1]['x'] - rssi['x']), 2*(aps[-1]['y'] -␣
 ↪rssi['y'])]) for rssi in aps[:-1:]])
matb = np.array([np.array([np.power(rssi['distance'],2) -
                           np.power(aps[-1]['distance'],2) -
                           np.power(rssi['x'],2) -
```

A.3 Indoor Localization Notebook

```
                        np.power(rssi['y'],2) +
                        np.power(aps[-1]['x'],2) +
                        np.power(aps[-1]['y'],2)]  ) for rssi in aps[:-1:]])
```

$$\mathbf{x} = \left(A^T A\right)^{\dagger} A^T b \tag{1}$$

[9]:
```
(x,y)=np.matmul(np.matmul(np.linalg.pinv(np.matmul(matA.T,matA)),matA.T),matb)
```

[10]:
```
display(Markdown(rf'$x={x[0]}$'))
display(Markdown(rf'$y={y[0]}$'))
```

$x = 4.1153215374866585$

$y = 1.9272403974566836$

[11]:
```
Latex(r'$x={0}$'.format(x))
```

[11]:

$x = [4.11532154]$

$x = x$

[ ]:
```

```

Appendix B

Imaging Codes

B.1 Radio Frequency Scanning script

```
import uhd
import numpy as np
import argparse
import time
from threading import Timer
import serial
first = True
def sendWaveForm(usrp_d, dur, freq, rate):
        ones = int(dur * rate);
        usrp_d.send_waveform(np.ones(ones), dur, freq, rate, gain =65);
def recvWaveForm(usrp_d, freq, rate, output):
        output[:] = usrp_d.recv_num_samps(100, freq, rate, gain = 100);
def sendCommand(ser, str):
        ser.write(str.encode())
def main():
        first = True;
        my_usrp = uhd.usrp.MultiUSRP("type=b200")
        s = serial.Serial(port='/dev/ttyUSB0', baudrate=115200)
        s.write('G28 X0\r\n'.encode());
        print('G28 X0\r\n');
        print(s.read_until(b'ok'))
        s.write('G28 Z0\r\n'.encode());
        print(s.read_until(b'ok'))
        coords = np.genfromtxt('coords7575_off.csv', delimiter=',')[:-1]
        coords = coords
        freqs = np.linspace(5.15e9, 5.490e9, 86).astype(np.int);
        print (coords)
        coords = [tuple(coords[i:i+2].astype(np.int)) for i in range(0, len(coords), 2)]
        #coords = coords[32*17 + 12::]
        print(coords[0])
        #exit()
        manual = False
        _isFirst = True
        last_cor = 0;
        for cor in coords:
                print(cor)
                s.write('G1 Z{0} F3600\r\n'.format(cor[0]).encode())
                s.write('G1 X{0} F10000\r\n'.format(cor[1]).encode())
                if manual:
                        input("Press Enter to continue...");
```

```
                manual = False;
        if cor[0] != last_cor:
                time.sleep(4.5)
        if _isFirst:
                _isFirst = False;
                time.sleep(30)
        time.sleep(1);
        last_cor = cor[0];
        for freq in freqs:
                a = np.empty(100, np.complex64)
                if first:
                        t_send = Timer(0, sendWaveForm, (my_usrp, 1, freq, 1e6))
                        t_recv = Timer(0.9, recvWaveForm, (my_usrp, freq, 1e6, (a)))
                else:
                        t_send = Timer(0, sendWaveForm, (my_usrp, 0.09, freq, 1e6))
                        t_recv = Timer(0.05, recvWaveForm, (my_usrp, freq, 1e6, (a)))
                t_send.start();
                t_recv.start();
                if first:
                        time.sleep(3);
                else:
                        time.sleep(0.2);
                first = False
                print(a.mean(), end=',')
                f = open('output230820_7575.csv', 'a+')
                f.write('{0},'.format(a.mean()))
                f.close()

if __name__ == "__main__":
    main()
```

B.2 Generating array coordinates

```
n = 75;
m = 75;
z = 80
x = 0
for i in range(n):
        for j in range(m):
                print('{0},{1}'.format(z,x), end=',')
                x += 4;
        z += 4;
        x = 0;
```

B.3 Image Reconstruction

```
import numpy as np
import scipy as sc
from scipy import io, interpolate
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

mat = sc.io.loadmat(r'512_scene.mat')
mat = mat['scene512']


## Parameters
c = float(299792458);           # speed of light, in m/s

# Antenna array parameters
```

```
n_ant_x = 8;
n_ant_y = 8;
delta_x = 6.25e-2
delta_y = 6.25e-2

# RF system parameters

f_carrier = 5e9;        # units are Hz
bandwidth = 600e6;        # from 10 GHz to 12 GHz
#chirp_duration = 4e-6;  # units are seconds
n_samps = float(mat.shape[2]);
delta_f = bandwidth / n_samps;
#delta_f = 250e6
# Scene parameters

range_max = 6;           # maximum range to target, in metres

Dx = delta_x;
Dy = delta_y;
f0 = float(f_carrier);
Df = delta_f;
Z1 = 2.5;


def SAR(s):
    # Compute A
    #A  = np.zeros(s.shape)
    A  = np.fft.fft2(s,axes=(0,1))
    print(np.linalg.norm(A[:,0,0]))
    Nx = A.shape[0]
    Ny = A.shape[1]
    N  = A.shape[2]
    print('FFT Done')
    kx = 2*np.pi*np.append(np.arange(0,int((Nx)/2)),  -Nx + np.arange(int((Nx)/2),Nx))/(Nx * Dx)
    ky = 2*np.pi*np.append(np.arange(0,int((Ny)/2)),  -Ny + np.arange(int((Ny)/2),Ny))/(Ny * Dy)
    # Compute D
    def D_Compute(i,j,n):
        if i < Nx/2:
            kx = i/Nx * 2*np.pi/Dx;
        else:
            kx = (i-Nx)/Nx * 2*np.pi/Dx;
        if j < Ny/2:
            ky = j/Ny * 2*np.pi/Dy;
        else:
            ky = (j-Ny)/Ny * 2*np.pi/Dy;

        k = 2*np.pi*(f0 + n*Df)/c;
        kz = np.sqrt(0j + 4*np.power(k,2) - np.power(kx,2) - np.power(ky,2))
        return A[i,j,n] * np.exp(-1j * kz * Z1)

    # Equation from SAR Seismic paper
    # t0 = abs(2*Z1/c);
    # Dijn = exp(-1j * (c*t0*k^2/2 - kz*abs(Z1))) * kz/k;
    # Dijn = exp(-1j * (- kz*abs(Z1)));% * abs(kz/k);

    Nxx, Nyy, NN = np.meshgrid(np.arange(Nx),np.arange(Ny),np.arange(N))
    D = np.vectorize(D_Compute)(Nxx, Nyy, NN)
    D = np.transpose(D, (1,0,2))

    print("D_computed");

    f_max = f0 + (N-1)*Df;
    k_min = 2*np.pi*f0/c;
    k_max = 2*np.pi*f_max/c;
    kx_max = (Nx-1)/Nx * 2*np.pi/Dx / 2;
    ky_max = (Ny-1)/Ny * 2*np.pi/Dy / 2;
    kz_min = np.sqrt(0j + 4*k_min**2 - kx_max**2 - ky_max**2)
```

```
    kz_max = np.sqrt(0j + 4*k_max**2 - 0 - 0)

    kz = np.linspace(np.real(kz_min), np.real(kz_max), N);
    Nxx, Nyy = np.meshgrid(np.arange(Nx),np.arange(Ny))
    def E_compute_row(i,j):
        _k = 0.5*np.sqrt(0j + np.power(kx[i],2) + np.power(ky[j],2) + np.power(kz,2));
        w = c*_k;
        f = w / (2*np.pi)
        n = ((c*_k)/(2*np.pi) - f0)/Df;
        return n
    EE  = np.vectorize(E_compute_row, signature='(),()->(m)')(Nxx, Nyy)
    EE = np.transpose(EE, (1,0,2))
    E = np.zeros(EE.shape)*1j
    def interp1(i,j):
        data = D[i,j,:].reshape((1,1,N))
        return sc.interpolate.interp1d(np.arange(N)+1, data[:],  kind='linear', bounds_error=False,fi
    E = np.vectorize(interp1, signature='(),()->(m)')(Nxx, Nyy)
    print("E Computed")
    # IFFT
    E = np.transpose(E, (0,1,2))
    R = np.fft.ifftn(E, axes=(2,1,0))
    R = np.flip(R, axis=0)
    print("R Computed")
    return R;

R = SAR(mat)
scene = np.linalg.norm(R, axis=(2))
X, Y = np.meshgrid(np.arange(scene.shape[0]),np.arange(scene.shape[1]))
plt.pcolormesh(X, Y, scene)
plt.show()
'''
Nxx, Nyy, NN = np.meshgrid(np.arange(scene.shape[0]),np.arange(scene.shape[1]),np.arange(scene.shape
fig = go.Figure(data=go.Isosurface(
    x=Nxx.flatten(), y=Nyy.flatten(), z=NN.flatten(), value=scene.flatten(), opacity=0.3, isomin=scer
fig.show()
'''
```