Cost-effective design for the measurement of man-made noise in the HF band receiver

by

Lee McQuire

Thesis submitted in fulfilment of the requirements for the degree

Master of Engineering: Electrical Engineering

in the Faculty of Engineering and Built Environment

at the Cape Peninsula University of Technology

Supervisor: Prof M.T.E. Kahn
Co-supervisor: Dr Vipin Balyan

Bellville
December 2021

DECLARATION

I, Lee McQuire, declare that the contents of this dissertation/thesis represent my unaided work and that the dissertation/thesis has not previously been submitted for academic examination towards any qualification. Furthermore, it represents my own opinions and not necessarily those of the Cape Peninsula University of Technology.

Signed **Date**

ABSTRACT

In recent years, there has been an upsurge in portable wireless devices, increasing demand for wireless data, higher transmission rates and the ever-increasing use of electromagnetic radio frequency (RF) spectrum. This has led to a scarcity of spectrum resources, therefore, to actively monitor RF spectrum use and optimise current usage models, this paper presents a unique cost-effective approach to conduct spectrum monitoring.

This paper will examine the feasibility of installing a cost-effective RF Man-Made Noise (MMN) monitoring station onto a moving vehicle. To cut down costs, the RF MMN monitoring station will comprise of commercial consumer-based products to build the prototype i.e. Software Defined Radio (SDR), a computer, and GPS that is intended to be installed on public transport. In utilizing public transport, the RF MMN monitoring system can opportunistically collect measurements while moving through a specific area. These measurements will provide valuable data about RF spectrum use over a larger area compared to a single location-based station approach.

A model of the RF MMN monitoring station is simulated in MATLAB. The emphasis on simulation models is to optimize the sample rate of measurements and maximise the accuracy and sensitivity of this approach. The sample rate is optimised by considering the spatial distance travelled on public transport. This will reduce the need for extra processing and storage overhead. To optimise the accuracy and sensitivity, an investigation is conducted on the noise floor generated by the inherent (internal) electronic noise of the system and the vehicle electronics that the system is installed in. A hardware prototype was designed to demonstrate the feasibility of the system. A pilot test run was conducted, and the measurement is plotted on a scatter plot map. The recorded measurements serve as a tool for analysing RF MMN over the geographic area in which public transport frequently travels.

The RF MMN measurement system will contribute to enhancing traditional Man-Made noise databases with the sensors moving around within public vehicles. This will assist in identifying the primary, secondary, and illegal users, confirm spectrum database accuracy and will assist in refining current propagation models within the RF band in which the samples were measured. The RF MMN monitoring system is possibly the new “eyes and ears” of the regulatory body to actively monitor spectrum use.

Keywords: Software Defined Radio; Interference mapping; Spectrum Database; Spectrum Occupancy measurement

ACKNOWLEDGEMENTS

Thank you to God for providing the strength, insight, knowledge, wisdom, and opportunity.

My sincere gratitude and appreciation to my supervisor, M.T.E. Kahn and Co-supervisor, Dr Vipin Balyan, for their invaluable guidance, enthusiastic help, and consistent encouragement throughout the entire research project.

My gratitude to my two colleagues and friends Henricus Cook and Bruce van Niekerk for motivating me to part-take in this endeavour. Thank you for supporting and motivating me through the process.

DEDICATION

I would like to dedicate this thesis to my grandmother Susan, mother Ester, my wife Charné, my two kids Gabriel and Micah for all the unconditional love, support, much patience and all encouragement.

Thank you, grandma, for always praying for me, your prayers are what leads me on the right road.

Thank you, mom, for always giving your listening ear, always supporting me, and trusting me in the decisions I have made in my life.

I would like to extend a special thanks to my wife and kids. For all the sacrifices you have made in terms of time, I know it was difficult not having your husband and daddy there, but you know I'm doing this all for our better future and always know your daddy loves you endlessly.

Charnè my flower, thank you for being my ear, my strength, and my moral compass, my life only really started when I met you.

I appreciate and love you all very much.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

Abbreviations	Definition
ADC	Analogue to Digital Converter
AGC	Automatic Gain Control
AM	Amplitude Modulation
BPSK	Binary Phase Shift Keying
CR	Cognitive Radio
CW	Continuous Wave
DAC	Digital to Analogue Converter
dB	Decibel
DGPS	Differential GPS
DSP	Digital Signal
DVB-T	Digital Video Broadcasting -Terrestrial
EMI	Electromagnetic Interference
FBO	Frequency Band Occupancy
FDM	Frequency Division Multiplexing
FFT	Fast Fourier Transform
FM	Frequency Modulation
FPGA	Field Programmable Gate Array
FSK	Frequency Shift Keying
GHz	Giga Hertz
GPS	Global Positioning System
GUI	Graphical User Interface
Hz	Hertz
I/Q	In Phase/ Quadrature complex signal
IC	Integrated Circuit
ICASA	Independent Communication Authority of South Africa
IF	Intermediate Frequency
IoT	Internet of Things
IP	Internet Protocol
ITU	International Telecommunication Union
kHz	Kilo Hertz

LNA	Low Noise Amplifier
LO	Local Oscillator
MHz	Mega Hertz
MMN	Man-Made Noise
NCC	Numerically Controlled Clock
NF	Noise Figure
PLN	Power Line Noise
PWM	Pulse Width Modulation
QPSK	Quadrature Phase Shift Keying
RBW	Resolution Bandwidth
REM	Radio Environment Map
RF	Radio Frequency
Rx	Signal Receiving
SA	Spectrum Analyzer
SDR	Software Defined Radio
SMA	Sub Miniature version of the connector
SNR	Signal-to-Noise Ratio
SO	Spectrum Occupancy
USB	Universal Serial Bus
USPR	Universal Software Radio Peripheral
VCC	Voltage Controlled clock
VCO	Voltage-Controlled Oscillator

Chapter 1 : Introduction

1.1 Background

"Indeed, the gap between wireless supply and demands widens. It is feared that a looming spectrum crisis, in which exploding demand from smartphones will soon overwhelm the wireless capacity, will occur. The problem is the lack of a new spectrum available to wireless data carriers. Smartphone data traffic is increasing so fast that if nothing is done, they will use up the available spectrum. This just affects not only smartphones but similarly all wireless devices. A huge amount of spectrum is essential for broadband use in the future as suggested in the digital agenda in Europe and the National Broadband Plan (NBP) in the USA ." (Arjoune & Kaabouch, 2019)

In recent years, there has been an upsurge in portable wireless devices, increasing demand for wireless data, higher transmission rates and the ever-increasing use of electromagnetic radio spectrum. With the uptake of technologies such as 5G, self-driving cars, LTE-M and NB-IoT (Internet of Things), more and more devices are connecting wirelessly to the internet. This movement is putting more pressure on spectrum resources and is the driving force for new innovative ways of monitoring and optimising spectrum resources.

Radio Frequency (RF) spectrum is part of the electromagnetic spectrum that is used for wireless communication, i.e., mobile cell-phone services, satellite services, television and radio broadcasting and more. RF spectrum is measured in Herts (Hz) and its frequency range is between 8.3 kHz and 3 GHz (Merriam Webster, 2017). RF spectrum is a limited natural resource that needs to be used optimally and is regulated by the International Telecommunication Union (ITU). The ITU legislation is enforced by each country's regulatory body which manages its frequency spectrum independently. The mandate of the regulatory body is to ensure that there is a rational, efficient, equitable and cost-effective use of the radio-frequency spectrum by all radio communication services (ITU-R, 2015b). The regulatory body used spectrum management to implement this mandate. Spectrum management is the overall process of administrating and regulating the use of the radio frequency spectrum on the premise of maximising spectrum efficiency, facilitating spectrum sharing, monitoring radio frequencies, minimising interference and stopping unauthorised or improper use of the radio spectrum. Radio monitoring forms part of spectrum monitoring and has an underlying function called spectrum occupancy or RF Man-made noise (MMN) measurement. In RF spectrum management, spectrum monitoring in the from of RF MMN measurement serves as the “eyes and ears” in that it provides a method of verification of frequency use.

This RF MMN measurement data can be used to identify vacant frequencies, prompt investigation when signals are present on unallocated channels and help with the planning for allocation of additional bands or assist with the re-assignment of existing bands. This is also of interest to protect the legal spectrum which is assigned to licensees at a fee. Thus, interference resulting from illegal use of spectrum can result in degraded quality of service (QoS) if not total service jamming or disruption of operations (HA Cook, MTE Kahn, 2020). RF MMN measurements are conducted by using specialised equipment that is normally found in dedicated occupancy measuring sites. The occupancy measuring sites are built in a wide array of areas to create a nationwide RF monitoring system.

However, commercially available RF monitoring systems are relatively expensive to build and has a very limited coverage area which leads to ineffective RF spectrum monitoring.

1.2 Problem Statement

As the amount of RF users has grown exponentially in the past few years, the RF monitoring environment has also become more complex. With the advances in processing, mass chip production and software-defined technology, radio equipment has become more user friendly and affordable thus is creating more RF spectrum bottlenecks that can lead to non-compliant and illegal spectrum use. Also, traditionally licensed spectrum channels are allocated for long-time spans to licence holders that may not continuously use this limited spectrum resource. This inefficient use of spectrum resources is the driving force to find more innovative technologies that will enable more efficient spectrum use (Kaabouch & Hu, 2014).

One of the ways of optimising the RF spectrum is to monitor RF spectrum use. Monitoring RF spectrum use will assist in identifying unused, underutilised, or illegally used spectrum. To monitor the RF spectrum, use an RF monitoring station are used. However, commercially available RF monitoring systems have a major disadvantage in that they are relatively expensive to build and maintain. The reason for this is that RF monitoring systems make use of highly specialised equipment that often needs to be installed in difficult to reach areas. Secondly, a great majority of the RF monitoring systems are designed to be installed in a single location and the rapid development of communication techniques has created new challenges in the deployment, design, and functional updating of the RF monitoring stations. These monitoring stations have a limited frequency range and coverage area thus there is always a need for more stations. It is a very costly venture, especially for rural areas. As RF technology evolves, similarly so must the methods of monitoring the RF spectrum.

1.3 Personal Significance

In the many years as an Inspector at the Independent Communication Authority of South Africa (ICASA), the South African frequency regulator, I found it be quite disheartening, that due to budget limitations some areas of RF monitoring did not get the detailed attention it deserves. This prompted my interest to find an alternative cost-effective means to conduct spectrum monitoring by using a combination of drive testing used in the Cell phone coverage providers to monitor network performance (my past job function) and RF monitoring used by regulatory authorities (current job function).

1.4 Objective of Research

The objective of this research is to complete a feasibility study to create a low-cost Radio Frequency (RF) Man-Made Noise (MMN) monitoring system. This RF MMN monitoring system must be portable and have the capability to be installed on any vehicle (see Figure 1.1). This proposed RF MMN monitoring system must be a cost-effective alternative to current monitoring systems. Therefore, in this thesis we will be investigating the following objectives to a greater extent:

a) The design of building a prototype of the mobile RF MMN measurement system while analysing which is the most cost-effective approach for its intended purpose.
b) To reach a clearer understanding of the unwanted effects on the observed RF noise floor induced into the RF MMN monitoring systems measurement results by its inherent (internal) electronic noise and the vehicle electronics that the system is installed in.
c) To reduce the amount of data storage required for recorded measurements collected by the RF MMN monitoring system by making use of selective sampling.
d) Conducting a Pilot test run with the RF MMN monitoring system and analysing the recording measured within the HF part of the FM band.

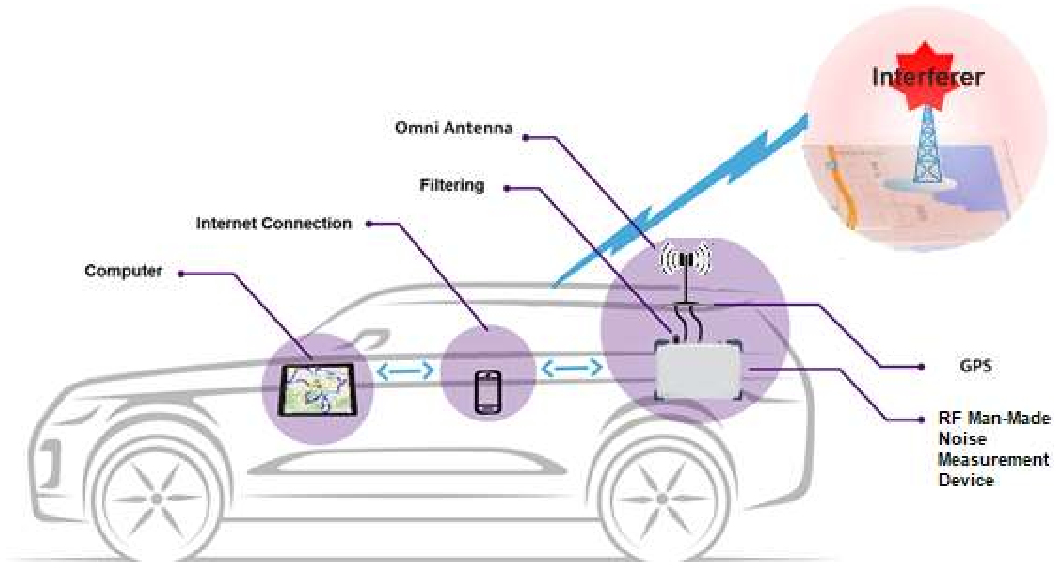


FIGURE 1.1 CONCEPTUAL INSTALLATION OF THE RF MMN MONITORING SYSTEM IN A VEHICLE

1.5 Research Design and Methodology

Resources in the form of articles, reports and papers were used to determine the best method to design, optimise, simulate, and test the RF MMN monitoring system.

- In the design phase of the RF MMN monitoring system, an investigation was conducted to find existing readily available commercial consumer-based products to build a prototype. (see Figure 1.2).
- A MATLAB simulation was conducted to determine some of the unwanted effects of the observed RF noise floor induced into the RF MMN monitoring systems measurement results by its inherent (internal) electronic noise and the vehicle electronics that the system was installed in.
- To reduce the amount of storage needed in the recording of measurement data, the sampling method investigated is spatial sampling. This will be achieved by making use of the GPS coordinates received and as the vehicle moves the distance between samples are measured. The only sample at a specified distance will be recorded thus reducing the amount of storage needed.
- A pilot test run was conducted on the FM radio band using the previously determined equipment from 1.4a and noise floor parameters determined in 1.4b while making use of the sampling method from 1.4c. The results were analysed using MATLAB.

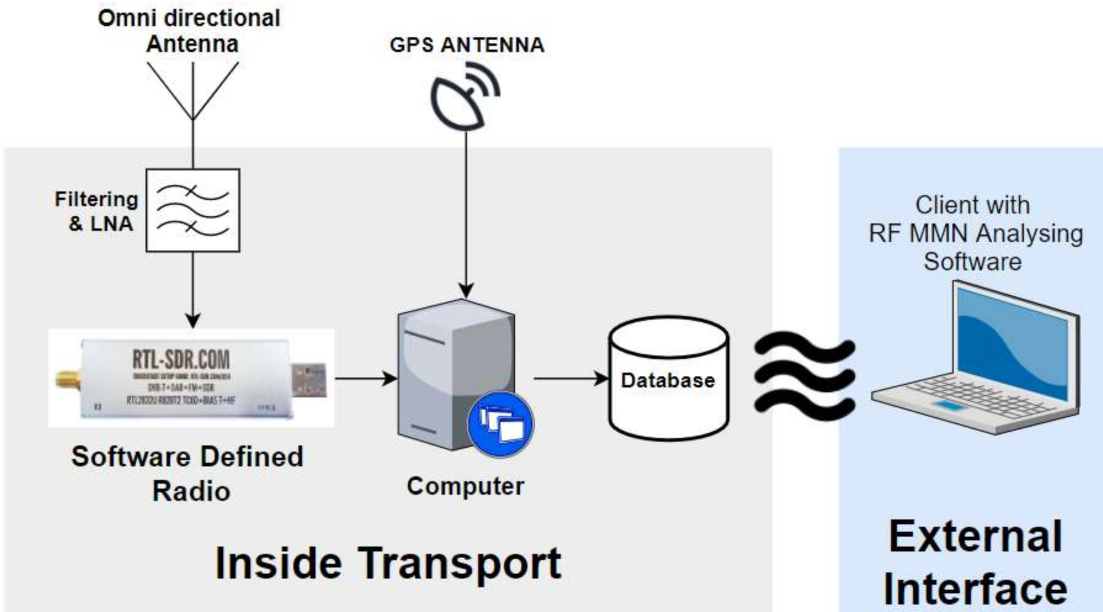


FIGURE 1.2 CONCEPTUAL DESIGN OF THE RF MMN MONITORING SYSTEM

1.6 Delineation of Research

The objective of this research is to design a cost-effective RF MMN monitoring system that can be installed in a vehicle. The following topics will not be discussed or covered.

- The design of SDR, computer and GPS module or other equipment used, as these are stand-alone units that are already commercially available.
- The test will only be conducted within the South African context thus only guidelines for this ITU region and standards will be discussed.
- This paper is only a conceptual paper therefore a commercial product will not be created.
- As this is a cost-effective approach to retrieve RF MMN measurements, the measurements are not compared to results of commercial systems nor is the accuracy of the data acquired confirmed.
- To limit the scope and complexities of the research, this paper will only cover the design of a system and will only test on the FM radio frequency band to accomplish these measurements most cost-effectively.

1.7 Significance and Contribution of Research

Measuring RF MMN is an important part of RF monitoring. RF monitoring is imperative for the reliable management of RF resources by assisting in the allocation of new spectrum, minimising the risk of improper use of RF resources and interference. Therefore, finding a more cost-effective method of conducting RF MMN spectrum measurements is vital. The RF MMN monitoring system is an alternative way to retrieve RF MMN data from the field. This can lead to a very cost-effective means of acquiring RF MMN data, especially when the measurement system is installed in existing public transport systems. As public transport moves, the RF MMN data can opportunistically be collected. The measurement data that this system generates can be used by the RF Regulatory body to better understand the usage of RF spectrum to a greater extent. The main advantage of the system would be that the RF MMN measurement can be measured more frequently, in areas where it was previously not feasible and at a fraction of the cost compared to commercial systems. The inexpensive readily available commercial components used and the means of collecting the data via public transport would be perfect for use in third world countries where financial resources are limited. The RF MMN monitoring system design can lead to future research to design a commercial setup to be used by RF Regulatory bodies.

The RF MMN measurement data retrieved can be used by the local RF Regulatory body to influence the following:

- The Allocation of New RF Spectrum
- Assist in finding illegal or Non-compliant users of the RF spectrum
- Assist in interference tracking and identification
- Influence future Regulatory Policies
- Improve the accuracy of RF spectrum databases
- Update of current propagation models etc.

Chapter 2 LITERATURE REVIEW

2.1 Introduction

In recent years, there has been an upsurge in wireless portable devices, increasing demand for wireless data, higher transmission rates and the ever-increasing use of electromagnetic radio spectrum. With the increased uptake of technologies that need radio frequency spectrum and this being a limited resource, there has been an ever-increasing need to optimise the usage of the RF spectrum. To ensure the efficient use of radio spectrum the ITU legislation is enforced by each country regulatory body. Each county regulatory body actively monitors the spectrum use and enforces compliance. RF spectrum monitoring task is performed by spectrum monitoring towers. These monitoring towers are very expensive to build, maintain and has a very limited footprint.

To mitigate this issue this paper presents a solution where a low-cost RF MMN monitoring system is to be designed to be installed in pre-existing transport systems. This proposed RF MMN monitoring system must be a cost-effective alternative to current monitoring systems.

To achieve this proposed solution the aim in this chapter, to attain an in-depth understanding of the problem at hand and cover essential existing literature to find the proper approach to implement the solution proposed. Much of the technical groundwork and information will be covered giving the reader an insight into the RF MMN monitoring regulatory, its parameters, phases, cost, and existing commercial monitoring systems. Some of the other vital topics that will be presented are RF noise sources, an introduction to RTL-SDR, why RTL-SDR was the perfect cost-effective solution and the pros and cons of using RTL-SDR.

2.2 Electromagnetism Overview

Electromagnetism is a field of study that deals with the rules and context of the interaction of magnetic and electric fields, that propagate together through space in the form of an electromagnetic wave. Electromagnetism is used in wired and wireless technologies also, including optical connections. Michael Faraday derived the law of electromagnetic induction from his experiments and was largely responsible for the discoveries in this field of study. James Clerk Maxwell unified the knowledge and used his equations to describe the interconnection of electric and magnetic fields and their properties (Mitolo & Araneo, 2019). In electromagnetism, we make use of electromagnetic waves for wireless communication and data transfer. The direction of propagation of an electromagnetic wave in space is described as the Poynting vector. The Poynting vector is described by the vector product of the magnetic field strength (B) and the electric field strength (E) component of the wave (Verhoeven & Archaeology, 2017) (see Figure 2.1**Error! Reference source not found.**).

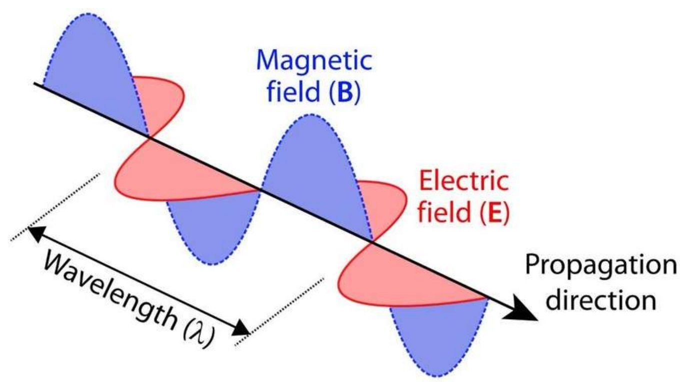


FIGURE 2.1 DEPICTION OF AN ELECTROMAGNETIC WAVE

The propagation of an electromagnetic wave through free space is significantly influenced by the frequency of the wave and the environment through which the wave passes. The elementary method of radio signal propagation is a direct wave, i.e. line of sight from the transmitter to the receiver. An obvious problem for the easy propagation of direct waves is the physical obstacle, for example in the form of mountains, landscape, water, and buildings etc. How the propagation of an electromagnetic wave will be affected depends primarily on the dimensions of the obstacle and the transmission frequency. An electromagnetic wave has the property of being reflected only by obstacles that are greater than its wavelength (Krupa, 2016). This reflected wave propagation from obstacles can add up and is called multipath propagation. Multipath propagation is illustrated in Figure 2.2**Error! Reference source not found.** (Krupa, 2016).

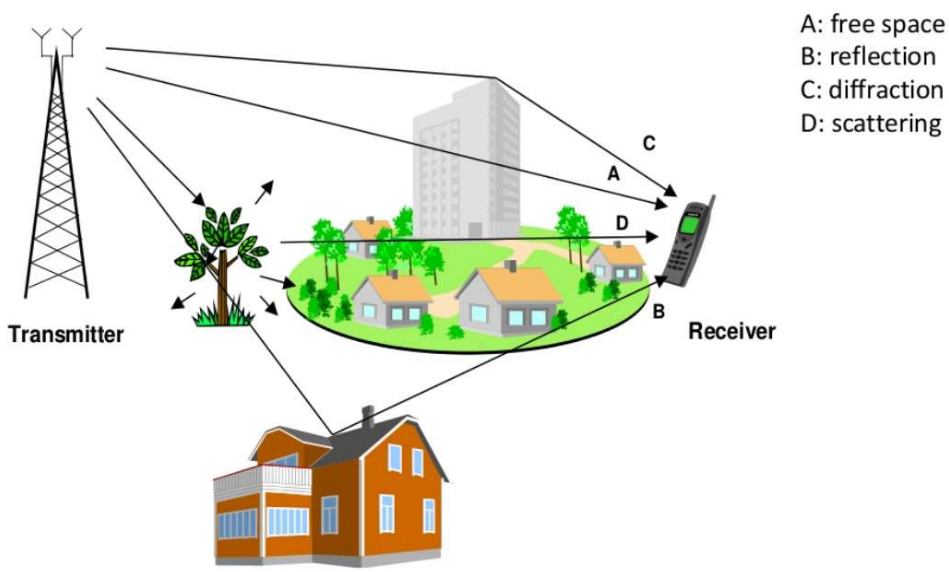


FIGURE 2.2 MULTIPATH PROPAGATION EFFECT

Radio Frequency (RF) is an electromagnetic wave that is measured in Hertz (Hz) that lies in the range from around 8.3 kHz to 300 GHz. These radio frequencies are primarily used for communication (Merriam Webster, 2017) and are regulated by the International Telecommunication Union (ITU). ITU legislation is enforced by each country's regulatory body that manages its frequency spectrum independently.

I/Q data is an abbreviated version of the English expression "In phase / Quadrature phase" and it represents signals in the time domain. "I" is the current momentary amplitude of the (real signal) and "Q" is the momentary amplitude of the signal phase-shifted -90°.
I.Q data can be represented in the Complex plane, Polar and Euler's form (Adam, 2015).

2.3 Spectrum Management

Spectrum management is the overall process of administrating and regulating the use of the radio frequency spectrum. The end goal of spectrum management is to maximise spectrum efficiency, facilitate spectrum sharing, monitor radio frequencies, minimise interference and stop unauthorised or improper use of the radio spectrum. The regulatory body, within a South African context, is ICASA (The Independent Communication Authority of South Africa). ICASA mandate is to plan, assign, control, enforce and manage frequency spectrum (ICASA, 2018).

2.3.1 Spectrum Monitoring

Spectrum monitoring serves as the ears and eyes of the spectrum management process. The purpose of spectrum monitoring is to support the spectrum management process in general, spectrum planning functions, including frequency assignment and resolve electromagnetic spectrum interference. This spectrum monitoring process involves the measuring of the radio waves and the identification of their primary parameters such as bandwidth and frequency, through a spectrum monitoring system. These spectrum monitoring systems measure and record measurements onto a spectrum database. Through the assessment of the measurement data with the spectrum database, infringements on spectrum regulations can be revealed and the discrepancies and illegal frequency use can be corrected. The relation between spectrum management and spectrum monitoring is described in Figure 2.3**Error! Reference source not found.** (Chen et al., 2016).

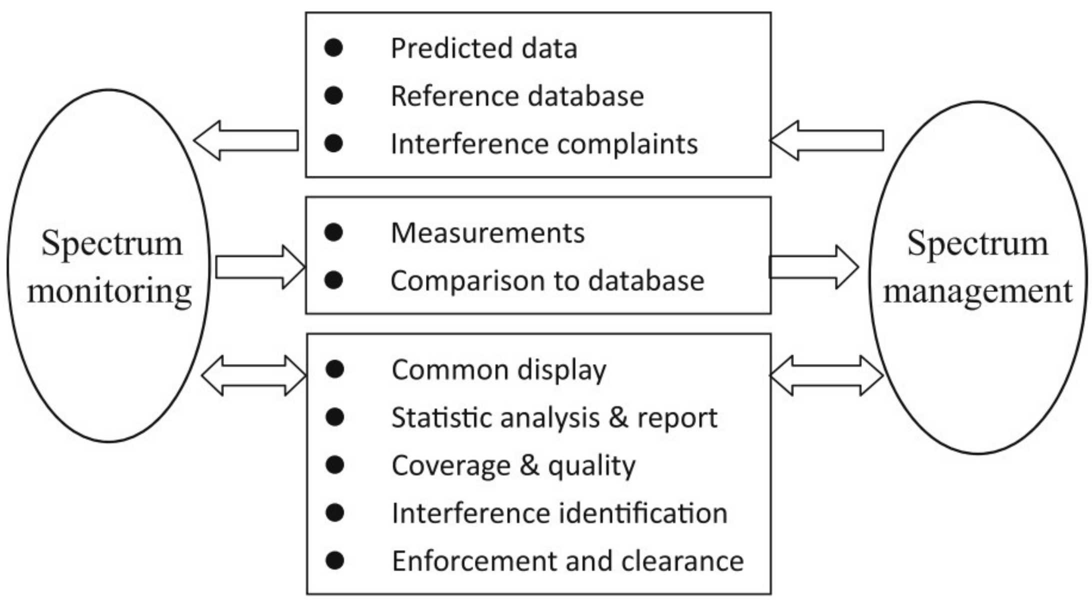


FIGURE 2.3 RELATIONS BETWEEN SPECTRUM MONITORING AND MANAGEMENT

2.3.2 Radio Monitoring

Radio Monitoring forms a vital part of spectrum management. Radio monitoring is an activity in which information is obtained about the values and parameters of the monitored radio signal (received power or voltage at the receiving antenna). Radio monitoring can be focused on monitoring the received signal power depending on the position of the receiver (measuring the coverage of the landscape by the signal), the time of reception, a combination of both, or determining the place of a transmission. Part of radio monitoring is confirming radio signal occupancy. Radio signal occupancy measurement is a collection of measurement parameters of the signal of interest, the frequency, position, and time of individual measurement (Valenta, 2017).

2.4 Spectrum Occupancy Measurements Overview

Spectrum occupancy measurement analysis aims to quantify a proportion of time in which a certain frequency band/channel is being occupied in a specific area (McHenry et al., 2006). Spectrum occupancy measurement can be described as the utilization rate of a channel or band. Spectrum occupancy measurement is defined by the ITU as the measurement of the spectrum spread over numerous various frequency channels/bands to determine whether the channel is occupied or not, and which is not necessarily separated by the same channel distance (ITU-R, 2015a). Spectrum occupancy monitoring provides a method of verification of compliance in that it allows the identification, measurement of interference sources, spectrum usage and the verification of proper operational and technical characteristics of RF signals and assist in the identification of illegal transmitters and creates data for effective future spectrum management policies (Radiocommunication Bureau, 2011). The information gathered by spectrum occupancy measurements is invaluable for frequency regulators to determine the current use of the spectrum allocation, improve the accuracy of the spectrum databases to facilitate the allocation of spectrum and assist in spectrum sharing. The occupancy data can be used to identify vacant frequencies, prompt investigation when signals are present on unassigned channels and help with the planning for allocation of additional bands or reassignment of bands.

In occupancy measurements, the signal strength is a measure of the emissions of interest above the expected threshold level. In the analysis of spectrum occupancy, there are only two-channel conditions that are likely “occupied” where the signal level exceeds the particular detection threshold and “free” where the signal level in the band/channel is high enough so that spectrum occupancy (SO) is reached by the likelihood of it being in an occupied state (ITU-R, 2012). These spectrum monitoring occupancy results can be used to determine how busy the frequency is, determine whether the frequency/channel is illegally being occupied etc. There have been many spectrum surveys conducted worldwide in different locations, that were covering both wide ranges on frequencies and specifically licenced bands. These occupancy studies such as (Valenta et al., 2009), (Lehtomaki et al., 2013), have been conducted with one device at a fixed location only focusing on the frequency and time scopes. Unfortunately, this traditional type of study usually does not provide enough information about the current utilisation of the spectrum to draw definite conclusions. The key measurement data for spectrum occupancy monitoring includes frequency, field strength, bandwidth, modulation scheme on a specific location. The regulatory body use spectrum occupancy measurements to get valuable information about efficiency of the current spectrum use. The spectrum allocations help to improve the accuracy of the spectrum databases. Spectrum database information is used to help facilitate spectrum sharing in the operational environment (Zhang et al., 2014). The information gathered forms a starting point for spectrum allocation decisions, planning, troubleshooting, radio resource management and enforcement action.

2.4.1 Summary of the History of spectrum occupancy measurements

The 1927 International Radiotelegraph Conference was the start at which different frequency bands were allocated to different service providers and users. Dedicated operational bands were defined for the interference-free operation of communication systems, satellites, broadcasting services, etc. and unlicensed bands were identified for short-range communication systems. RF is a limited resource and needs to be used as efficiently as possible. The licence requirement was added to operate within these spectrum bands (Medeisis & Holland, 2014). The compliance with these licences must be monitored and this has brought about the birth of occupancy monitoring. Currently, RF technology is evolving at a rapid pace, therefore, occupancy monitoring must also be evolving compared to its initial start. The most recent important contributions in occupancy monitoring are summarised in **Error! Reference source not found.** (Höyhtyä et al., 2016)

TABLE 2.1 IMPORTANT CONTRIBUTION IN OCCUPANCY MEASUREMENT IN THE FORM OF SURVEY AND REVIEWS

Topic of Article	Supports are specified in the articles
Interference Map Construction	Focus on measurement-based interference mapping (Yilmaz Birkan et al., 2013)
Spectrum Occupancy Modelling	Measurement-based spectrum occupancy modelling (Chen & Oh, 2016), General time-domain occupancy models (Saleem & Rehmani, 2014)
Spectrum Sensing	Survey on spectrum sensing algorithms, emphasis on dynamic spectrum access (Yücek & Arslan, 2009)
Spectrum systems and measurement methods	Procedure for measurement in the context of cognitive radio (López-Benítez & Casadevall, 2009), General spectrum measurement strategies (Matheson, 1988)

Spectrum measurement, spatial domain focus	A comprehensive approach for analysis and measurement. Spatial domain spectrum occupancy measurement using an interference map (Höyhtyä et al., 2016)

2.4.2 Spectrum Occupancy Measurement Parameters

In spectrum occupancy measurement there are certain defined parameters which Matheson (Matheson, 1988) provides some guidelines for achieving spectrum occupancy measurements. Spectrum occupancy will always be a random variable due to shifting in user activities (Ghosh et al., 2010). Spectrum occupancy measurement is divided into five dimensions: location, frequency, power, direction and time (Matinmikko et al., 2010).

2.4.2.1 Time Parameters

The time parameter is a sampling set in the time domain.
Duration of monitoring (T_s)(Eq 2-1) is the overall time throughout which the occupancy measurements are conducted. The optimum period of monitoring depends on the resolve of the occupancy measurement and prior knowledge about the radio systems utilizing the spectrum resource. In this thesis, the band to be measured contains only FM broadcast stations therefore it will be sufficient to measure the frequency/channels band only once per cycle as this is a continuous transmission.
Observation time (Tc) or measurement time is a trade-off to the scanning speed of a device.
Revisit time (T_r) is the time reserved to check all the channels to be measured (whether occupied or not occupied) and return to the starting channel (N_c). When only one frequency is measured, the revisit time is equivalent to the observation time. To calculate the total revisit time, the dead time (T_d) between sweeps needs to be considered expressed by the equation below.

$$T_s = N_c Tc + T_d \qquad \text{Eq 2-1}$$

Observation time ($TObs$)(Eq 1-2) is the time desired by the system to perform the essential measurements on one frequency, this includes any processing overheads such as setting the receiver to the desired frequency and storing the results to memory.
Sample measurement time (TM) is the definite measurement time on one frequency.

$$TObs = TM + Processing\ ovehead \qquad \text{Eq 1-2}$$

Occupancy time (TO**)(**Eq 2-2**)** is the time during which a particular frequency has a measured level beyond the threshold (noise floor). When more than one frequency is measured, a frequency cannot be continuously monitored. After the revisit time, a frequency is found to be occupied, it is presumed that it has also been occupied throughout the time in between two following measurements on that frequency.

$$TO = No\ (number\ mesurements\ with\ level\ above\ threshold) \times Tr \qquad \text{Eq 2-2}$$

2.4.2.2 Frequency

The sampling interval in the frequency domain is the **Frequency resolution.**

The **Resolution Band Width (RBW)** is the bandwidth of a sole spectrum measurement attained during a definite sampling interval in time and an exact location in the spatial domain.

Frequency band occupancy (FBO) (Eq 2-3**)** is the occupancy of an entire frequency band that counts each measured frequency and also calculates a total figure in percentage for the whole band, irrespective of the normal channel spacing. The measurement frequency resolution of a band occupancy measurement is extreme, the FBO is much lower than the frequency channel occupancy values of the channels in this band.

$$N_F = \frac{F_{stop} - F_{start}}{\Delta F} = 1000 \qquad \text{Eq 2-3}$$

The frequency channel occupancy or spectrum occupancy is the portion of the measurement time when the detection power in the band or channel surpasses the decision threshold (Sm & Spectrum, 2012).

Instantaneous frequency spectrum occupancy $Q_n(f)$ (Eq 2-4) for the n-th measurement at a frequency, f is given as below equation (3-1) where $P_{rx,n}\ (f)$ is the n-th measured power in the same band.

$$Q_n(f) = \begin{cases} 1, & if\ P_{rx,n}\ (f) \geq \gamma \\ 0, & if\ P_{rx,n}\ (f) < \gamma \end{cases} \qquad \text{Eq 2-4}$$

To obtain meaningful values, occupancy values are integrated over time.

In ***frequency band occupancy β*** determination, all the channels within the whole measurement band are taken. For a single measurement device, this is calculated as equation (Eq 2-5) where N_0 is the number of measurements samples with the power level above the detection threshold and N is the full number of measurement samples over the integration time. As we will be using a single measurement device this formula will be suited.

$$\beta = \frac{N_0}{N} \quad \text{Eq 2-5}$$

2.4.2.3 Power

Power is measured via detectors that provide the measurement. Detection is the process of selecting between noise alone and signal plus noise. There is a wide array of detectors on the market including energy, matched filter, feature, and correlator detectors. Energy detectors are typically used in spectrum occupancy measurements subsequently it does not require data about the characteristics of the signal to be measured.

Decision threshold γ is an exact power level measured at the receiver input. This is used as a locus level when interpreting the real measurements. The decision threshold defines whether a channel is reflected as occupied, and this level may be fixed to a pre-defined level or algorithmically defined to a variable level. The occupancy results depend on the technique and threshold by which this is defined.

Sensitivity is the detection limit of instrumentation and is distinct by the threshold for a type of detector. Sensitivity describes the lowest SNR (signal-to-noise ratio) at which a device can detect the radio signal with sufficient reliability.

Reliability is measured with the probability of false alarms and the probability of detection. When measuring whole frequency bands, it is important to isolate emissions from adjacent channels. If the measurement bandwidth is set too high, an emission caused by a strong neighbouring channel may appear occupied giving false readings. When sweeping with a spectrum analyser, the resolution bandwidth must not be broader than 1/10 of the narrowest channel spacing in the band. To isolate from the noise floor, a sufficient signal to noise ratio (S/N) in the measurement results need to be used. For FM broadcasting, an S/N ratio of 40 dB may be used for wideband analogue transmissions (ITU-R, 2012)

2.4.3 Spectrum Occupancy Measurement Phases

To obtain meaningful data it is important to implement and design spectrum measurements properly. These measurements can be separated into 5 phases below Figure 2.4**Error! Reference source not found.** (Höyhtyä et al., 2016). We will be discussing each unique phase below:

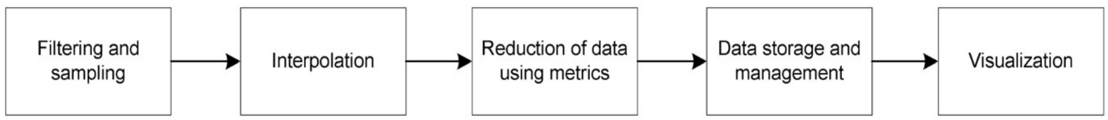


FIGURE 2.4 PHASES OF MEASUREMENT

Sampling is defined as an observation at a definite frequency, time, and location where data is being recorded. For the most accurate occupancy map, an energy measurement is continuously taken for each sample for possible locations, but there are practical limitations to this. The sampling requirement is dependent on the objective of the measurement and is limited to maintain an amount of data manageable bearing in mind the requirement to combine and distribute the information. The Nyquist sampling theorem defines that the sampling frequency essential to be at least 2 times as high as the bandwidth of the input signal. Spectrum occupancy data is obtained by associating the power values to the threshold. Spectrum occupancy is a function of the sampling intervals.
Factors that influence spectrum occupancy measurement is the resolution bandwidth, spatial resolution, which affects the number of channels, and selectivity, observation time per channel, duration of monitoring and revisit time, which affects the total measurement time. (ITU-R, 2012) The revisit time, RBW and location of sensors need to be adjusted based on the equipment that is found to be operational in the band. Below Figure 2.5**Error! Reference source not found.** (Höyhtyä et al., 2016) is sampling parameters in the frequency, time, and space domains. The measurement signal in each domain is displayed in the higher part of the figure and the lower portion defines the sampling parameters.

In the **Spectral-domain**, a resolution must be adequate not to lose important information while keeping the data amount and scanning speed to a reasonable level. The RBW of to a lesser extent than half the channel spacing is used when the centre frequency of the channel is not known (Sm & Spectrum, 2012) or generally an RBW of 1-3% of channel BW is used. With the RBW that is too small, a longer revisit time is created, and measurement time must be increased and with RBW too large the adjacent channel power might be measured leading to incorrect results. The lower the expected occupancy the added samples will be needed for the same confidence levels.

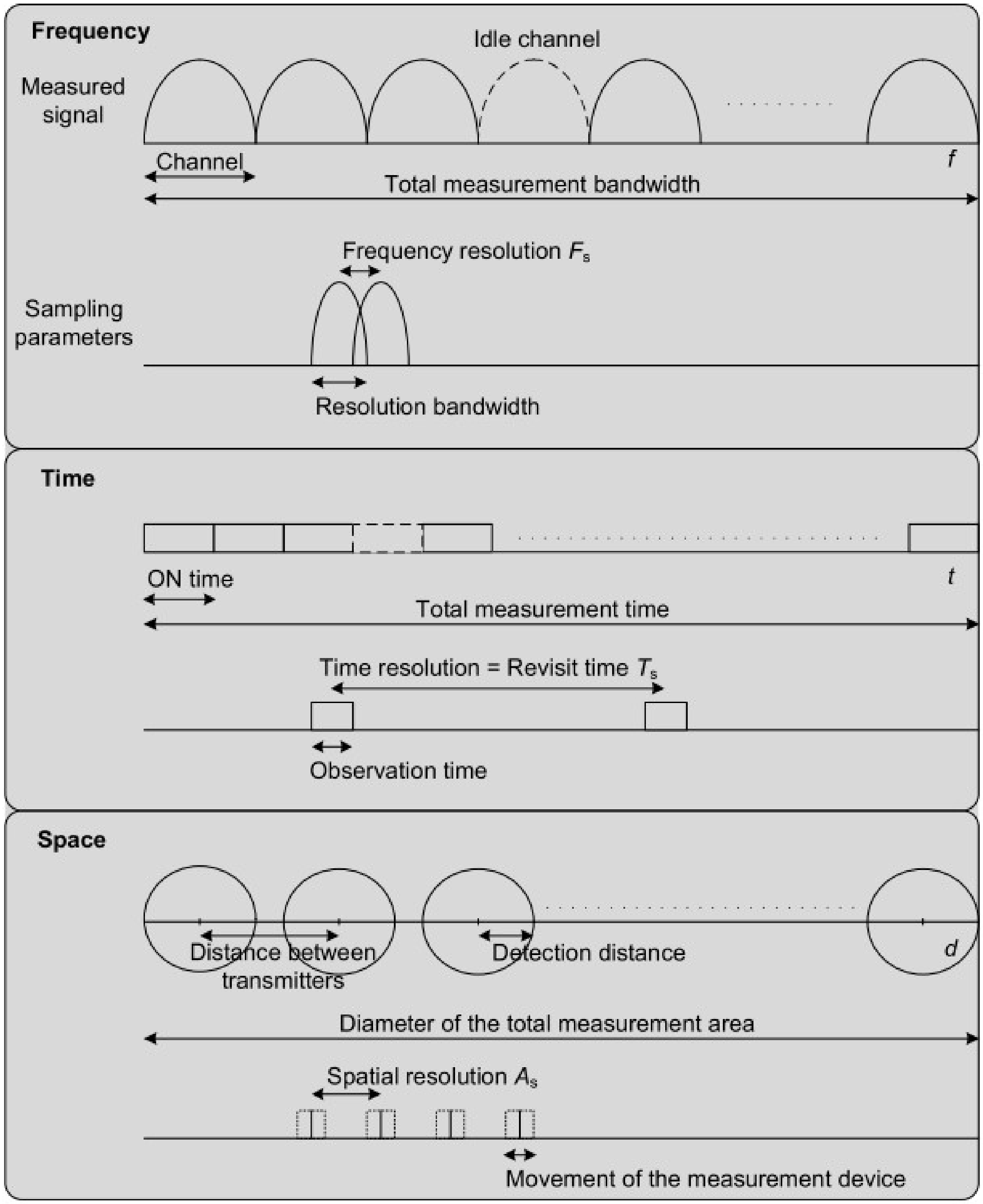


FIGURE 2.5 SAMPLING PARAMETERS IN FREQUENCY (SPECTRAL), TIME (TEMPORAL) AND SPACE (SPATIAL) DOMAINS

In **Spatial domain** measurement, the location of the transmitter and coverage is not known, thus we put the special resolution to the ability to detect a signal. The detection distance is depended on the height of the transmitter, the power and frequency used, the antenna gain used by the transmitter antenna and non-direct line-of-sight features of the environment such as buildings, mountains, water, and other obstacles. Sampling needs to be reserved at a variable distance between measurements to be able to adequately estimate the relationship between distance and variance. The spatial sampling method is split into regular sampling (see Figure 2.6), stratified and uniformly distributed point random, cluster, transect and contour sampling. For a decent spread of measurement points with no variability, regular sampling is used. Sampling can either be done asynchronous/synchronous. Synchronous sampling is when measurements from diverse sensors are mapped to a similar time scale. This allows data analytics over the same time frame using a universal clock. Asynchronous sampling does not make use of a global clock (Funck & Gühmann, 2014). The interference map defines RF power distribution in space, frequency and time domains (Kim et al., 2011). The interference map measurements are depicted in Figure 2.6 at a specific time. The black dots on the map characterize measurement points in the spatial domain for a bin frequency this is set on specific coordinates.

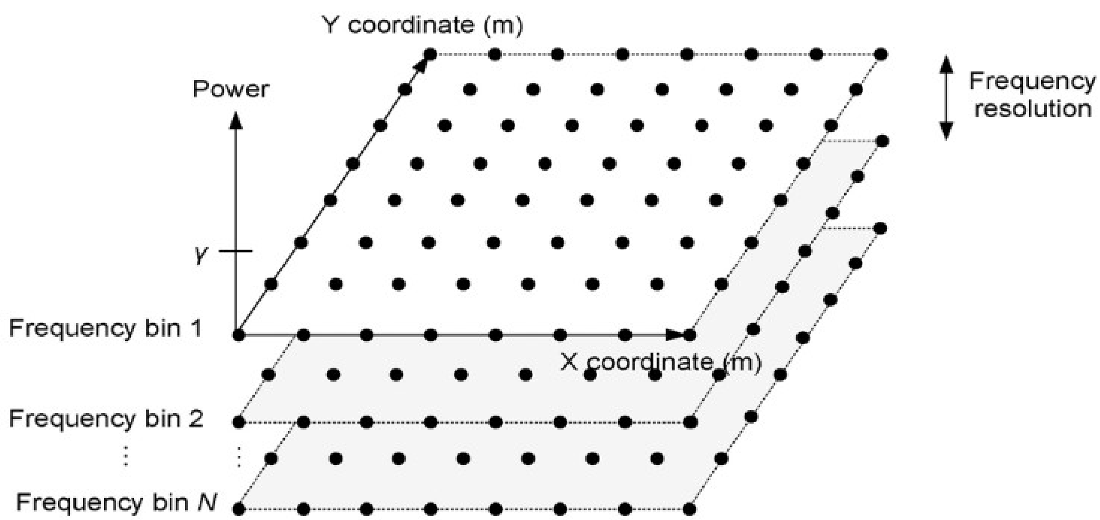


FIGURE 2.6 INTERFERENCE MAP REGULAR SAMPLING

2.4.3.2 Interpolation

Interpolation is used to increase the resolution of the information by estimating the values at a point that has not been measured. Interpolation methods are separated into local methods that only use measurements in the vicinity of the measurement point or global methods that use all obtainable data for estimations. There are more than 72 interpolation methods to create continuous spatial data over a region of interest (Li & Heap, 2011). The choice of method is mostly based on the actual data acquired, time and computer resources available and level of accuracy needed. The natural neighbour interpolation is where the value allocated at each output location is the number/value of the sample data point closest to that location. Natural neighbour interpolation establishes the value for a position as a weighted sum of the values of neighbours. Natural neighbour interpolation does not take into account the influence of the neighbouring data points at all that can lead to unexpected transitions on the interference map (Dwarakanath et al., 2013). Natural neighbour interpolation is best used for nominal data from point observations and has very low computing requirements.

2.4.3.3 Data Reduction using Metrics

Measurements can frequently provide an enormous amount of data, mainly if they are shown over an extended period. The use of mercies is to change data into a more compressed usable format as per requirement. A vital metric for spectrum measurement is channel transmission or spectrum occupancy measurement which is defined as a segment of the measurement time that the detected received signal energy power exceeds a threshold (Spaulding & Hagn, 1977). In using the spectrum occupancy metric, the quantity of required data to create an interference map can significantly be reduced. The spectrum occupancy metric expresses the occupancy as a specific location by likening the defined threshold to the received signal strength (Wang & Salous, 2011).

Setting the correct threshold is very important as untrue or false data can be recorded. Energy detection has limits in the sensitivity of the measurement outcomes there is a probability of an actual detection and false alarms being detected the closer your measurements are taken to the noise floor. In Figure 2.7 the threshold is shown where there the probability decreases of missed detections and the increased probability of false alarms (Höyhtyä et al., 2016). The threshold is set to attain a continuous false alarm rate taking the SNR wall into account (Skolnik, 2019). The detection threshold can be adeptly set based on current noise levels in the environment. The threshold of occupancy measurement using signal strength detection must be set 3 to 5 dB above the measurement noise floor to decrease the probability of false alarms or phantom occupancy.

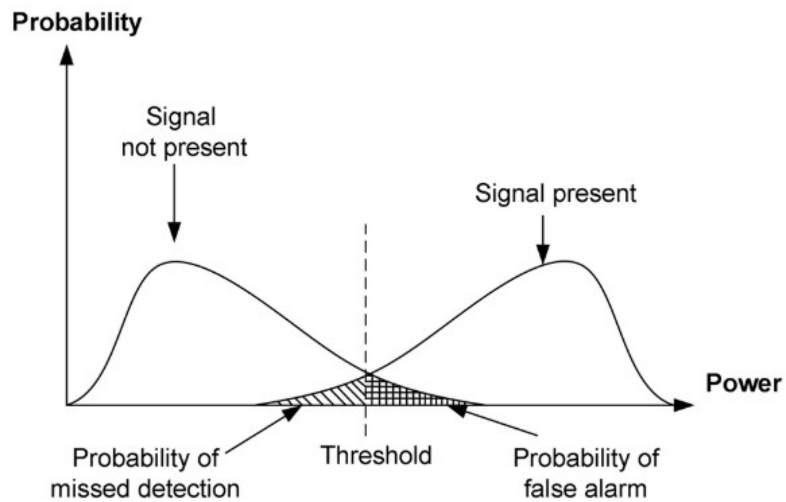


FIGURE 2.7 COMPROMISES BETWEEN FALSE ALARMS AND MISSED DETECTION WHEN SETTING UP THE THRESHOLD

2.4.3.4 Data Management and Storage

Data storage has become less costly and easier accessible in recent times. This has led to increased analysis and building models of the environment that were not possible in the past. Management and analysis of spectrum-related temporal, spatial and relative aspects of data require advanced software solutions and hardware to store process and access data. As the expansion of intelligible environment modelling for cognitive radio systems looking at spectrums occupancy needs, certain characteristic of data systems has been identified:

- The collected spectrum data from numerous geographical locations need to be stored reliably and effectively in a database system.
- All database enquiries need for a certain subdivision of data should be achieved within a reasonable reaction time and must be operator friendly.

2.4.3.5 Visualisation

Visualization helps to communicate measurement data stored in an easier to understand form that can be used by engineers, scientists, policymakers, and the public. Visualization provides a means to support new spectrums sharing and allocation. Using colourful, well-labelled charts and images to interpret spectrum monitoring data helps to communicate the data collected. These images may include time, power, space, frequency dimensions, which is all depending on the resolve of the visualization (Sharakhov et al., 2014). A REM (radio environment map) is one of these visualisations (Zhao et al., 2009). REM describes the level of interference over a location of interest within a certain frequency band and can be made up of multiple recordings over a location. A REM can be made from RF signals, policies and regulations, available services, historical experiences, and physical locations of devices.

2.4.4 Spectrum Occupancy Measurement Types

2.4.4.1 Fixed Location Measurements

Fixed location monitoring is conducted on a fixed geographical location. The locations of these measurement stations are determined by two approaches: 1) where minimal Man-made noise and radio emissions are expected, 2) or highly populated regions where many emissions are present in both cases great care must be taken not to overload the receivers with too strong signals (Sm & Spectrum, 2011).
The biggest drawback of fixed monitoring stations is that they are fixed to a specific location thus for financial reasons they cannot be built in adequate numbers. Many of these stations are automated and remotely controlled (see Figure 2.8)(a)(Bureau, 2002) & (b&c)(Various, 2020d)

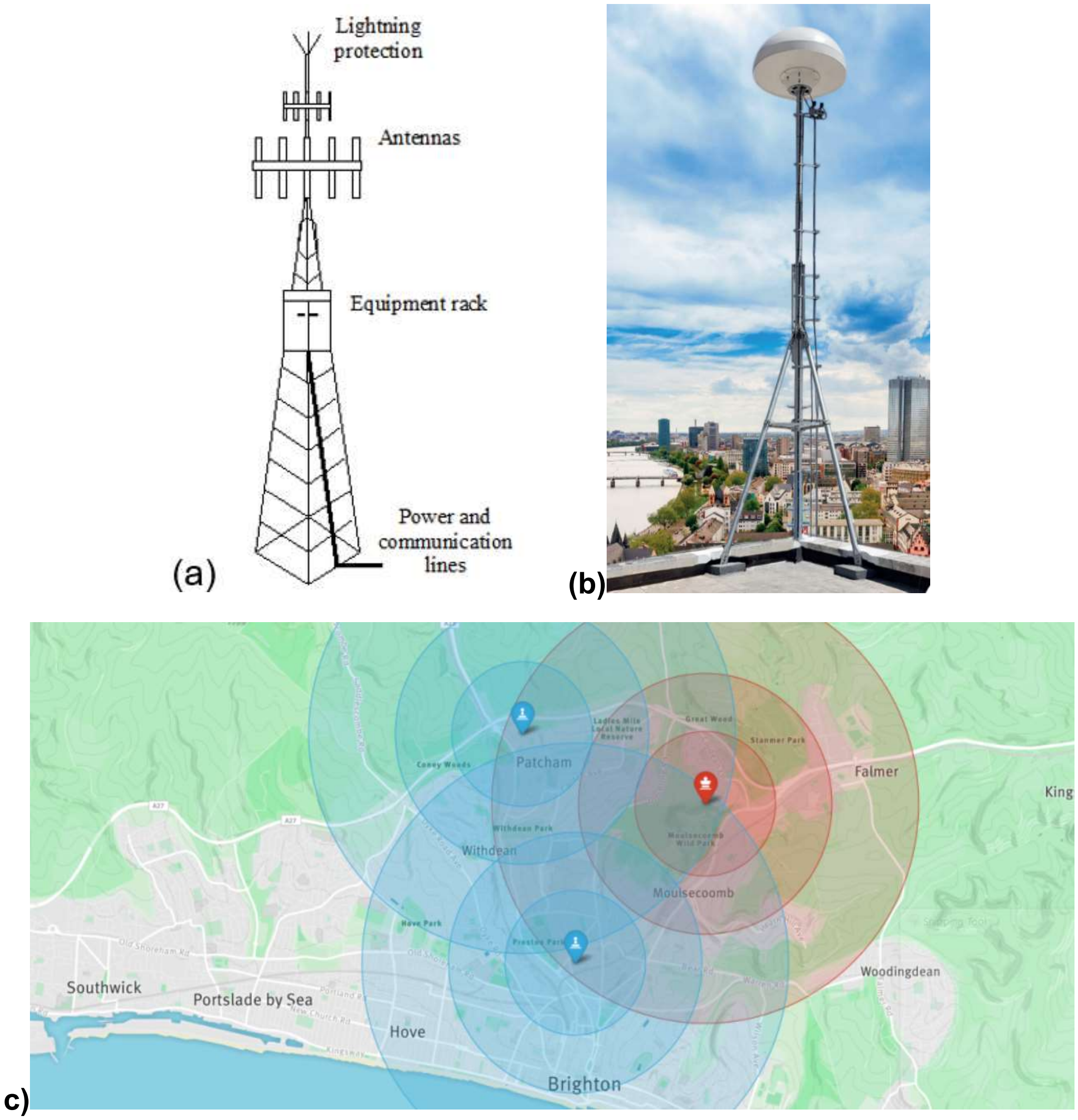


FIGURE 2.8 A) FIXED LOCATION SPECTRUM MONITORING SYSTEM B) LS TELCOM MONITORING TOWER C) LS THEORETICAL COVERAGE OF SPECTRUM MONITORING TOWERS

2.4.4.2 Vehicle-based (Semi-fixed) Measurements

Vehicle-based (Semi-fixed) measurement stations provide spectrum monitoring support which is suitable to carry out measurements close to the emitters that are characterised by high-frequency emissions or low power levels (Sm & Spectrum, 2011). Fixed monitoring stations are not dimensioned to provide the total coverage of the country regions, where vehicle-based monitoring stations can be used to act as temporary stations to improve coverage area. The information collected by mobile monitoring stations can be used for database spectrum management, to detect unlicensed emissions and/or licensing violations where it might be difficult due to topography. The fixed and vehicle-based monitoring systems can be used to perform the following functions **Error! Reference source not found.** (Sm & Spectrum, 2011):

TABLE 2.2 ELEMENTARY MEASUREMENT TASKS FOR MONITORING STATIONS

Elementary tasks	Parameters to consider performing the elementary task
Frequency measurement	–Accuracy in terms of frequency. – Range of frequencies,
Field strength and power-flux density measurement	– Frequency range, – Required accuracy in terms of level, – Specific measurements: - Measurement of antenna patterns. - Coverage measurements (measurements along a road),
Occupied bandwidth measurement	– X-dB and/or B/2 method measurements using spectrum analyser or software or monitoring receivers, – Other methods.
Modulation measurement	– Type of modulation (digital, analogue).
Spectrum occupancy including “on route” field coverage/ strength measurements and channel occupancy measurements	– Radiocommunication standard, – Channel technical specifications: bandwidth, type of modulation, spacing – Additional information to be recorded (e.g., automated identification/decoding), – Recording parameters, – Required scanning speed, – Number of meter/measurements distance based on GPS or the number of pulses/meters provided by the interface to the speedometer.
Identification measurement	– Call signs for selective calling, – Classes of emission, – Localization, – Emission mask (comparing with theoretical emission mask).
location measurement and Direction-finding	– Mapping, – Type of direction finders, – Response time – Class of accuracy,

2.4.5 Review of Commercial Occupancy Measurement Systems

COMMERCIALLY AVAILABLE RF SPECTRUM MONITORING SYSTEMS ARE ALREADY AVAILABLE AND DEPENDING ON THE PURPOSE THERE IS A GREATER COST CONNECTED TO HIGH-END DEVICES.

Table 2.3 (Gavrilovska et al., 2014) are a summary between spectrum monitoring devices and the cost and capabilities of these devices. The engineer needs to select the right monitoring system for the right scenario based on budgetary constraints.

TABLE 2.3 CLASSIFICATION COMPARISON OF SPECTRUM MEASUREMENT DEVICES

	High-Quality devices	**Mid- Quality devices**	**Low- Quality devices**
Example	Spectrum analyser and Base station	Software-defined radio and mobile terminals	Low-cost sensors and USB dongles
Price	High (Above R200 000)	Medium (R20 000 to R200 000)	Low (below R20 000)
Resolution (frequency, time, power)	High	High to medium	Low
Performance (accuracy, reliability, dynamic range)	High	High to medium	Low
On-Board processing memory and capabilities	High	High to medium	Low
Spectrum measurement software	Available	Available and Custom made	Custom made
Detection abilities	IQ detection	IQ detection	Energy detection
Power consumption	High	Medium	Low
Mobility & Large-scale deployment	Low	Medium	High

2.5 Radio Frequency Noise Sources

The range and sensitivity of radio receiving equipment are ultimately limited by the noise signal that is present in the receiving systems and measurement environment. RF noise limits the quality of the received signal that is recorded and must be taken into consideration. The ITU-R P.372-13 (ITU-R, 2015b) defines the geographical locations with their expected noise floors as:

- Rural/Remote – A vast distance away from any habitation or electrical installations e.g., in mountain regions, large forests, polar regions, oceans etc.
- Rural – Where few houses are in the area and where powerlines may be present e.g., farms etc.
- Urban – Houses with roads, with differences in density with domestic electrical equipment and commercial premises scattered in the area.
- Industrial - Retail and industrial areas having railways, manufacturing, power distribution and generation.

Radio noise that is created by natural sources is unlikely to change much. MMN (Man-made noise) is the dominant source of noise in many parts of the radio spectrum. This noise is changing with the increased use of electronic and electrical devices. Some applicable radio noise sources with their frequency range are covered in Table 2.4 (Various, 2013).

TABLE 2.4 APPLICABLE RADIO NOISE SOURCES PER FREQUENCY RANGE

Noise sources	Range of Frequency
Emissions from atmospheric gases, etc.	Above 10 GHz
Man-made noise	9 kHz to 1 GHz
Cosmic noise	4 MHz to 100 MHz
Atmospheric noise due to lightning	9 kHz to 30 MHz

2.5.1 Man-Made Noise

Man-made noise (MMN) is of particular interest because its intensity and level are dependent on human activity and can be variable in time. MMN can be created by the incidental radiator for example light dimmers, switch-mode transformers, powerlines, automotive, transport and generating facilities. Other sources are consumer electrical appliances, industrial equipment and lighting systems. The International Telecommunication Union ITU-R P.3372 defines MMN "to be the aggregated sum of all unintended emissions from multiple electrical and electronic equipment including emissions from wired telecommunication systems from several unknown sources" (ITU-R, 2015b). MMN is made of two components, one which has a Gaussian distribution, and a second component which has an impulsive character. MNN is normally made up of specific components see

Table 2.5 (Various, 2013).

TABLE 2.5 COMPONENTS OF RADIO NOISE

Noise component	Noise Properties	Noise Sources (examples)
Single carrier noise (SCN)	Bandwidth smaller than receiver bandwidth One or more distinct spectral lines Spectral power level independent of bandwidth	switched-mode power supplies, Wired computer networks
White Gaussian noise [(1)] (WGN)	Bandwidth equal to or greater than receiver bandwidth Uncorrelated electromagnetic vectors Spectral power level increases linearly with bandwidth	Power line communication networks, Computers, wired computer networks, cosmic noise
Impulsive noise (IN)	Bandwidth greater than receiver bandwidth Correlated electromagnetic vectors. Spectral power level rises with the square of bandwidth	Ignition sparks, gas lamp starters, lightning, computers, ultra-wideband devices

(1) RECOMMENDATION ITU-R SM.1753

Regulations are governing the noise generated by these devices, such as the South African National Standards (SANS) 301489-15-2004, Electromagnetic compatibility and Radio spectrum matter (ERM), Electromagnetic compatibility (EMC) standards of equipment used in South Africa.

2.5.2 Inherent (Internal) Noise

This is the noise contributions originating from the components or circuitry of the receiving system that are always present and this is all collectively described as inherent (internal) noise. The total amount of the receiver's internal noise power is commonly characterised by its noise figure in dB. The electromagnetic field density produced by radiation from such sources will be converted to an electric noise power by the receiver antenna and is sent to the input of the receiving system. External radio noise is the term used to describe this type of noise contribution at the receiver input and will add extra noise to the internal noise of the receiver (Bj, 2016). More noise can be generated by emissions from an external source.
The operating sensitivity of a radio receiving system depends on the sum of the external and internal noise power. The external noise will limit the range of the radio communication system and a receiver with a lower noise factor will not enhance the operational sensitivity.

2.5.3 Radio Frequency Interference

Interference is a subsection of noise and is defined by the ITU RR no, 1.166 as the effect of unwanted energy due to one or a combination of radiations, emissions, or inductions upon reception in a radiocommunication system, manifested by any misinterpretation, performance degradation, and/or loss of information which could be obtained in the absence of such unwanted energy.
There as 3 subsections of interference as follows (Niamey, 2015):

- Permissible interference is predicted or observed interference which complies with interference and sharing criteria.
- Accepted interference this is interference is more than permissible interference and has been agreed upon between administrations.
- Harmful interference is interference that endangers the functioning of a radio navigation service, other safety services that seriously obstructs, degrades, or repeatedly interrupts a radiocommunication service functioning as per Radio Regulations.

Interference can be very accidental at times and unforeseen. Some common causes of harmful interference due to none-compliance to regulator licensing are:

- Out of band operations/Emissions
- Non-observance of the maximum permitted power levels for spurious emissions or limits of frequency tolerances

- Operation of non-coordinated frequencies assignments (broadcasting and land mobile service)
- Non-observance of limitation within the band allocated.
- Unnecessary transmissions
- Operating with technical parameters different from those notified
- Unauthorised emissions, etc.

Interference is categorised into two bands categories, Narrowband interference is continuous modulated CW or wave (CW) signals examples of this is the clock harmonic from a digital device, adjacent and co-channel transmissions, intermodulation products, etc. This would appear on a spectrum analyser as narrow vertical lines or slightly wider modulated vertical bands related to specific frequencies. Broadband interference includes harmonics of switch-mode power supply, wireless digitally modulated systems (such as Bluetooth or Wi-Fi, or digital television). This would appear on the spectrum analyser to appear to be broad ranges of signals or an increase in the noise floor. Switch-mode power supplies and/or power line noise are the most common sources of this.

There are some common types of interference which are described below (Kenneth Wyatt, 2018):

- Co-channel Interference – where there is more than one digital harmonic (or transmitter) using or falling into, the same receive channel or band.
- Adjacent channel Interference – the transmitter functioning on an adjacent frequency energy spills over into the wanted receive channel.
- Fundamental Receiver Overload – this can be triggered by a nearby, strong, transmitter overloading the receiver front-end or other circuitry, causing interference and can even overpower the normal received signal.
- Intermodulation-Based Interference – this happens when energy from two, or more, transmitters mix to create some spurious frequencies that can land in the wanted receive channel or band. Third-order mixing products are common and typically occurs from nearby transmitters.
- Power Line Noise (PLN) is a common broadband interference issue that is typically caused by arcing on electric power lines and linked to power utility hardware. These interferences can spread from very low frequencies even below the AM broadcast band, into the HF spectrum if you are very close enough to the source, it can extend up through the UHF spectrum. This interference can sound like a harsh raspy buzz in an AM receiver.
- Switch-Mode Power Supplies are a common source of broadband interference.

- Cable Television – A signal leakage from cable television systems arise on their set channel assignments. These channels can intersect existing over-the-air radio communications channels.
- Wireless Network Interference – Interference to Bluetooth, Wi-Fi, etc. is more common, and with the abundance of mobile, medical, and household (IoT) devices by integrating Wi-Fi and other wireless forms will get worse.
- Other Transmitters:
    - Land Mobile or Two-Way Radio – Very strong interfering FM signals may result in the "capture effect" and can cause an overriding effect of the desired received signal.
    - Paging Transmitters are very powerful digitally or FM modulated transmissions that can overload a receiver. Digital paging may interfere with a wide range of receiver frequencies and will sound very raspy, like a buzzing or power saw.
    - Broadcast Transmitters interference will have modulation features comparable to their broadcasts – FM, AM, digital signals, or video carriers.

Locating radio frequency interference can be done by looking at the signal level on a spectrum analyser or recorded data. The unwanted signal is observed and followed by increasing levels in signal strength and is the most likely location of the interference source.

2.6 Software Defined Radio

In old-style hardware radios, the mathematical operation of processing and decoding of radio signals are performed making use of analogue circuitry namely: amplifier, filters, AGC or AFC circuits, mixer, a demodulator, and other supporting circuits. These hardware-based radio devices could only be modified through physical intervention which resulted in high production costs and limited flexibility and this analogue circuitry extra distortion, noise, and signal loss with each successive stage was added.

Not too long ago, computer chips became powerful enough to execute all the mathematical calculations in software, henceforth In 1991, Joseph Mitola conceived the name Software Defined Radio (SDR) (BA van Niekerk, TE Kahn, 2020). All the physically parts of SDR exist as an integrated circuit, or as a combination of discrete components, and are located on a printed circuit board. SDR differs mainly in that the AGC and AFC circuits, demodulation, and in some cases also filtering, is performed by software, inside the signal processor (DSP) or on a computer. The advantage is that the various parameters can be changed easily, such as intermediate frequency filter width and shape, automatic gain control, automatic notch filters, rejecting and identifying interference, tight filters etc.

This led to innovation to advanced SDR radios that formerly required complex analogue hardware but can be executed in software. Some of the improved radio capabilities such as waterfall displays, and wideband tuning are now available at a much lower cost due to SDR [9]. Placing the ADC and the DAC right at the antenna using high power proves to be a challenge thus in practical RF/microwave circuits it's necessary to include a hardware block see Figure 2.9 (Mitola, 2014).

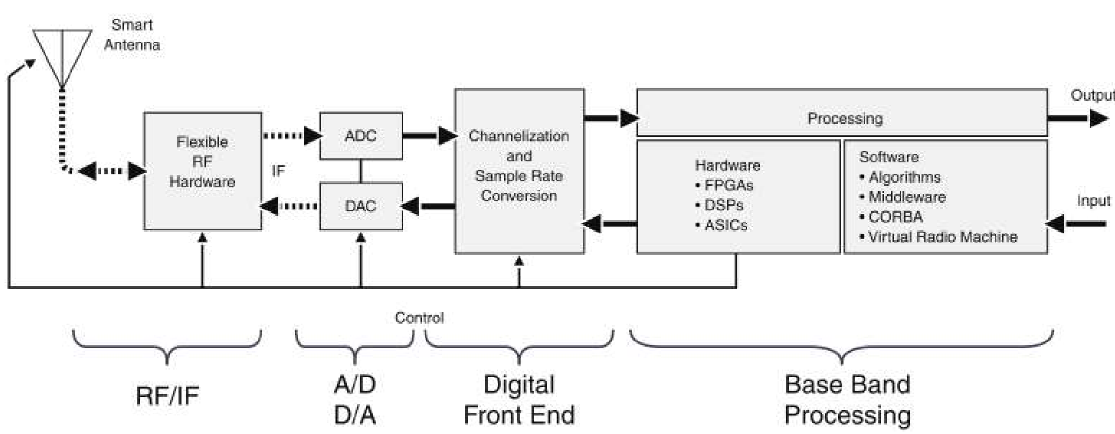


FIGURE 2.9 SDR TRANSCEIVER CONCEPT BLOCK DIAGRAM

2.6.1 RTL_SDR

The RTL-SDR is a USB sized stick that was built as a DVB-T TV (Digital HD TV) tuner. In 2012, Eric Fry a hardware hacker and Linux driver developer part of the Osmocom team which was busy creating their SDR found that the RTL2832U chip had a "test mode" which allowed it to be used as an all-purpose wideband SDR. In "test-made" the RTL2832 chip creates raw, 8-bit I/Q data samples and bypassed the DVB decoding stage and control the gain of the preamplifier inside the R820T2 circuit by modifying the drivers. By using open-source software drivers this commonly used low-priced TV dongle could then be turned into an SDR with characteristics that would have cost much, much more.

The performance of these dongles does not contest with a designed SDR, nonetheless, they perform exceptionally well for the price, and nearly all hobbyist projects that can be complete with exclusive SDRs or radios can also be completed with the RTL-SDR. This device can scan a wide range of RF frequencies between 25 MHz to 1.75 GHz. The RTL-SDR converts radio frequencies into raw I/Q samples of the spectrum that can be used as an inexpensive spectrum analyser and can with the help of software demodulate/decode FM, AM, ISM, ADS-B bands etc (see Figure 2.10). The RTL-SDR are surface mount, offers high performance, small package size, with low power consumption and is affordable. The affordability aspect of these receiver dongles attracts hardware hackers, tinkerers, ham radio enthusiasts, and anyone interested in radio frequencies. Noted that some SDR receivers also can transmit, but with relatively low power.

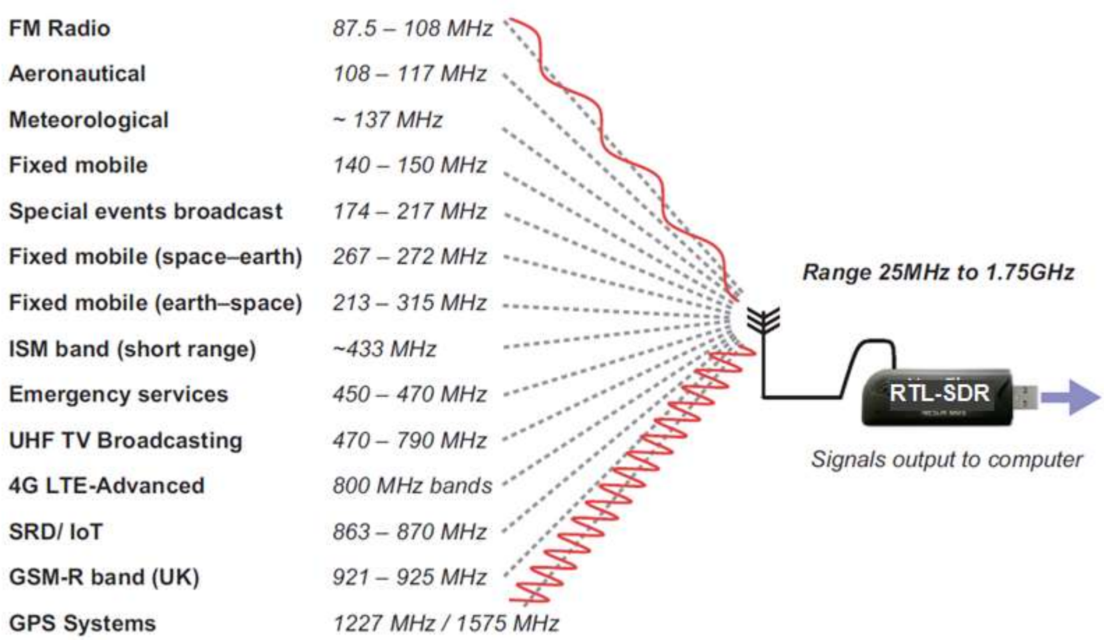


FIGURE 2.10 SIGNAL AVAILABLE IN THE RTL-SDR FREQUENCY RANGES

2.6.2 RTL-SDR Operation

The radio frequency signal received by an external antenna gets fed into the antenna input of the RTL-SDR dongle. The front end of the RTL_SDR is the same as a traditional radio where there is an attenuator that serves as protection against transients/static, switchable bandpass filters and an LNA (Low Noise Amplifier). The radio frequency signals entering the R820T tuner is down-converted to an intermediate frequency (Low-IF) using a VCO (voltage-controlled oscillator). This VCO is controlled and programmable by the RTL2832U over an I2C interface. After the AGC (active gain control) stage, the IF signal is brought down to the baseband. The traditional method of undertaking this is to pass the IF signal through an anti-alias filter. An ADC (Analog to digital converter) continuously measures the voltage at the input and sends those readings to software for processing. The ADC is clocked by a crystal at 28.8 MHz and takes measured voltage readings over 28.8 million times per second. At the output of the ADC, there is no more RF but rather a sequence of voltage readings which is fed to a dedicated chip the RTL2832U, which is where the actual DSP (Digital Signal Processing) is done. The RTL2832 chip then demodulates to baseband using quadrature numerically controlled oscillators (NCOs), i.e., a cosine and a sine oscillating at the IF frequency). This is controlled by software to finally create the raw 8-bit I/Q samples. Since computers do not have the processing power to handle the 28.8 MHz sampled rate, the SDR software filters out a portion of the 0-30 MHz that is picked up by the ADC by mixing and filtering it and then sends a reduced bandwidth signal to the computer. The smaller the measured bandwidth the SDR is sampling, the more RF samples can be sent to the computer. The performance of RTL-SDR radios is determined by the ADC which includes the sampling rate, bit depth and the receiver's dynamic range. The dynamic range in SDR-based receivers is a proportion of the smallest the highest value that can be digitally presented thus the maximum possible SNR for the ADC. Finally, these I/Q samples leave the RTL-SDR via USB 2.0 interface at a maximum RX sample rate of 2.56 Ms/s to be future processed by a computer (Mitola, 2014). See Figure 2.11 (Bob Stewart, Kenneth Barlee, 2017)

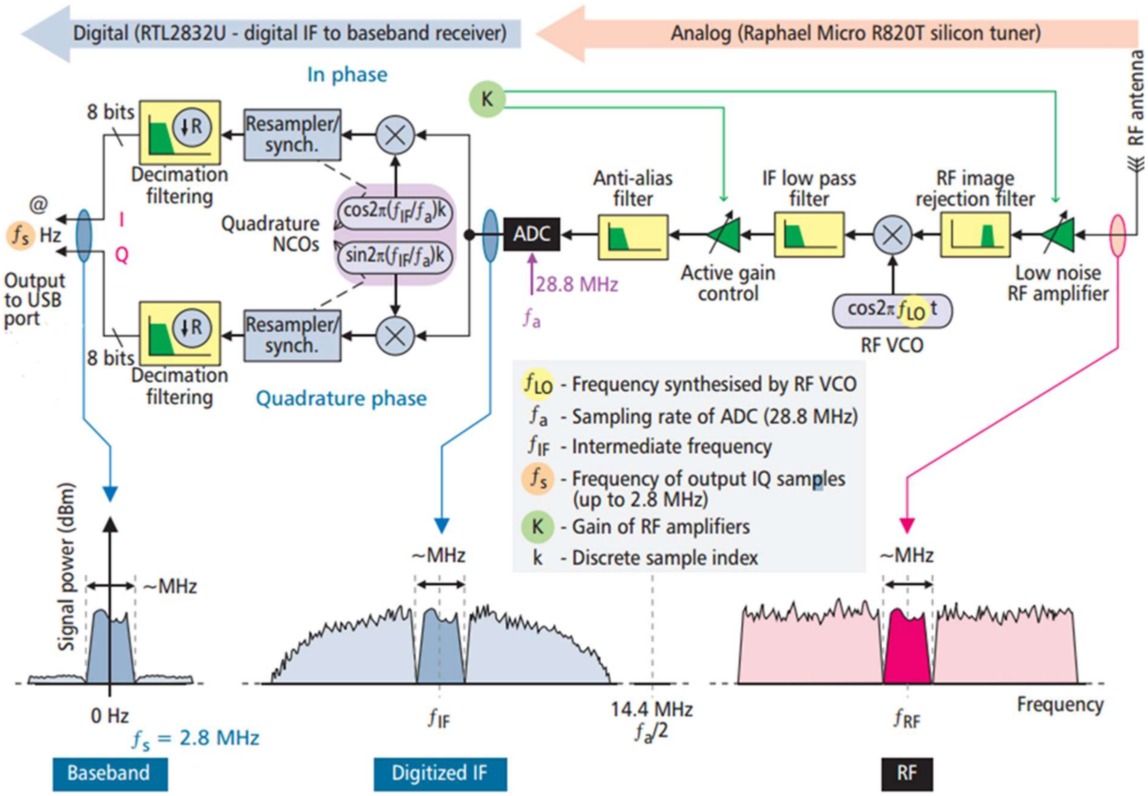


FIGURE 2.11 SIGNAL PROCESSING FLOW DIAGRAM OF THE MAIN STAGE OF RTL_SDR

2.6.3 RTL-SDR Architecture

Description Main components used in RTL-SDR see Figure 2.12 (Bob Stewart, Kenneth Barlee, 2017) and Figure 2.13 (Laufer, 2014):

- MCX or SMA Connector – for attaching the antenna to the RTL-SDR
- ESD (Electrostatic Discharge) Diode — this protects the tuner from electrostatic discharge that is originating from the SMA/MCX attached antenna
- R820T or R820T2 — this is the tuner chip that selects a portion of the RF spectrum and down converts it to an IF
- RTL2832U — is the demodulator chip that down-converts from IF to baseband then reduces the sampling rate and digitises the signal
- 28.8MHz clock crystal — delivers a reference for frequency synthesis, and is used for the generation of the local clock and oscillator
- USB 2.0 Interface — part of the RTL2832U, for the transfer of the baseband IQ data to the host computer
- The IR (Infra-Red) Sensor — is an interface for the remote control that is supplied with the device

- Serial Electronically Erasable Programmable Read-Only Memory (EEPROM) — stores USB configuration data for the device, and is connected to the RTL2832U via an I2C bus(Bob Stewart, Kenneth Barlee, 2017)

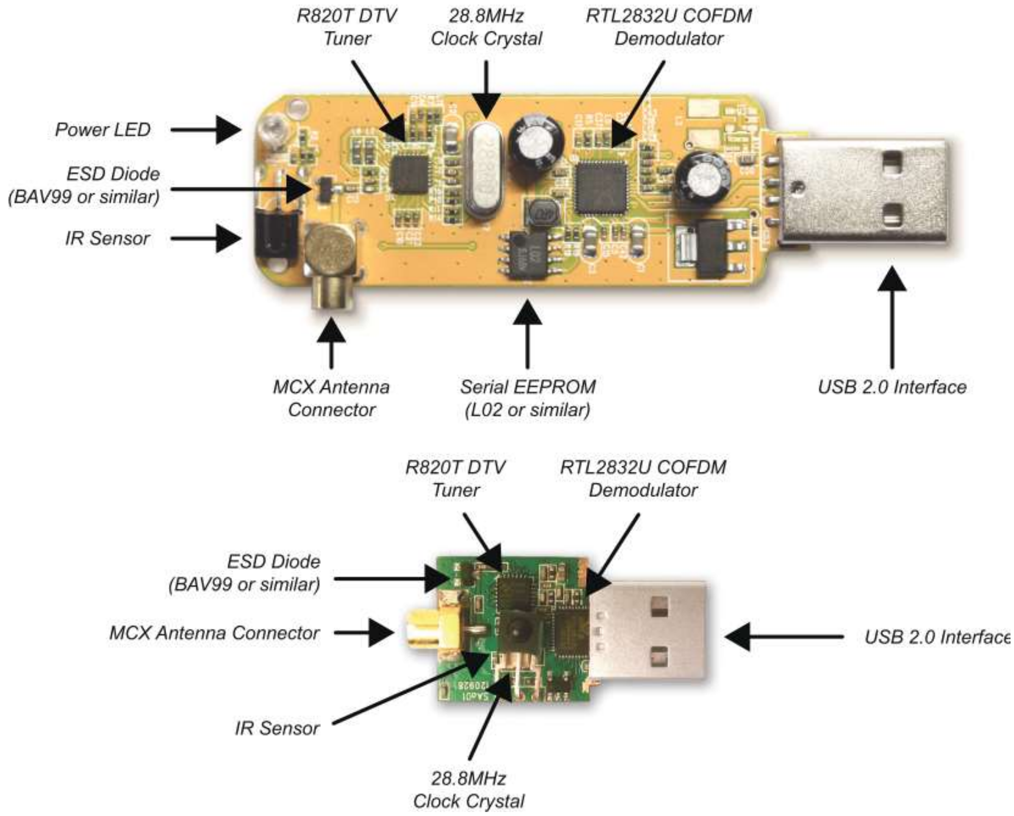


FIGURE 2.12 NOOELEC RTL-SDR MINI (TOP) AND THE RTL-SDR NANO (BOTTOM)

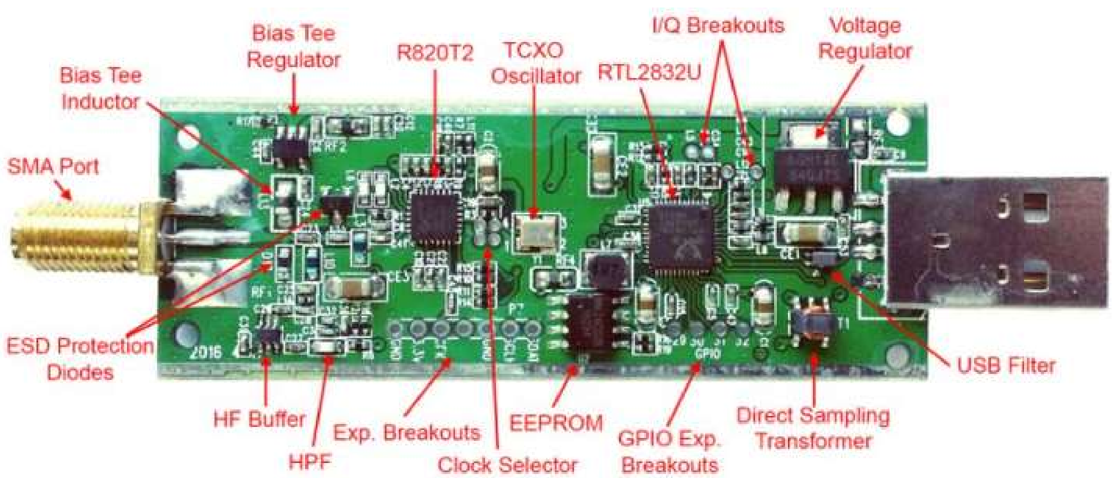


FIGURE 2.13 RTL-SDR.COM SDR

2.6.4 RTL-SDR Technical Specification

Standard parameters that all RTL-SDR share as described in Table 2.6 (Laufer, 2014)

TABLE 2.6 RTL-SDR PARAMETERS

RTL-SDR Parameters	Values
Frequency Range	22-2200 MHz
Maximum sampling rate	2.8 MS/s, theoretical 3.2 MS/s
LNA	<4.5 noise
Connectivity	USB 2.0
ADC resolution	8-bit
Bandwidth	3.2 MHz
Input impedance	50 ohm / 75 ohm
Received signal maximum power	+10 dBm
Stability of the oscillator	1 PPM

2.6.4.1 RTL-SDR ADC

ADC is a microchip that senses the analogue signal voltages and digitises the signal into a binary number. The higher the bit count the ADC has, the more precise the digitisation can be. For example, an 8-bit ADC can calculate the analogue input voltage between -127 to +127 and a 12-bit ADC can measure input voltages from -2047 to +2047. (Laufer, 2014) As with a lower bit count, the ADC will fail to recognise minor details of the analogue input, especially with weak signals and this may be lost in the digitizing process. This tends to happen if there is a strong and very weak signal nearby to one another. The RTL-SDR has a small 8-bit ADC but gives enough performance for most tasks. The dynamic range of an ADC is the range between the smallest and largest possible signal strength values. This dynamic range can be increased by making use of oversampling in software.

2.6.4.2 RTL-SDR Bandwidth

The SDR bandwidth is the section of the RF spectrum that can be viewed in real-time. The highest theoretical bandwidth of RTL-SDR is 3.2MHz (stable bandwidth of 2.8 MHz). Even though the theoretical bandwidth is higher but using this can lead to dropped samples that are not good for listening (choppy audio) and decoding signals but is fine for using RTL-SDR as a wideband spectrum analyser or as a spectrum occupancy monitoring system. Slow computer systems can cause dropped samples as well.

The RTL-SDR bandwidth (sometimes called sample rate) is 2MSPS (Mega samples per second) for 2 MHz of live data at a time. The sample rate must be more than twice the highest frequency components of the input signal, this is known as the Nyquist-Shannon or Nyquist limit. (Best, 1988)

2.6.4.3 Input Impedance

The RTL-SDR input impedance is between 50 – 75 Ohms because they were intended for use as TV hardware. The signal loss due to impedance mismatch between 50 to 75 Ohms is less than 0.177dB. Most professional equipment uses 50 Ohm connectors, cabling, and adapters.

2.6.4.4 USB interface and Current Usage

The USB 2.0 bus controller has a theoretical throughput limit of 256 Mbit/s. The USB data rate is 480 MHz, and the USB clock frequency is 48 MHz.

The R820T/2 uses roughly 60 mA, not in use and 300 mA in use. The E4000 needs approximately 170 mA in use.

2.6.5 Challenges with RTL-SDR

These RTL-SDR USB dongles are mass-produced in a very economical way and this creates problems which we will discuss in the following (Laufer, 2014):

a) RTL-SDR PARAMETERS

The signal resolutions, noise floors, and frequency accuracy of these devices are not optimal in some frequency bands, nor adequate for some applications.

b) RTL-SDR CRYSTAL TOLERANCE

The clock source (heartbeat) of the RTL-SDR circuit is the 28.8 MHz crystal. The RTL-SDR makes use of a 28.8 MHz oscillator with poor crystal tolerances which may be out by a few kilohertz up to +-150 PPM. The effect is that frequencies may have drifted from the known location. In the newer version of RTL-SDR, a Temperature-compensated crystal oscillator (TCXO) is being installed which can stabilize the frequency and is not affected by temperature changes.

c) RTL-SDR DC SPIKE

No matter where it is tuned, the R820 chip has a small spike in the centre of the spectrum. This is due to the way that R820 is designed. The removal of the DC spike with R820T usually done by the software is enough to completely remove this from GUI by selecting the IQ.

d) RTL-SDR RF SPURS

In the design of the dongle, there are unwanted (spurious) noise spikes at multiples of 28.8 MHz, there are strong spurs at multiples of the 48 MHz USB clock frequencies and the frequency of the internal oscillator and at multiples of the USB data rate of 480MHz. There is not much that can be done about this other than taking note of where they found and ignoring them. These spurs look like a single thin solid line on the waterfall/RF spectrum and sound like a single audio tone on the USB/LSB modes. Using a filter on the USB lines results in much lower USB noise at USB clock multiples and the redesigned V3 RTL-SDR dongles have resulted in fewer spurs.

e) RTL-SDR IMAGES

There are images of a strong signal every 250kHz at the bandwidth of 0.25MHz. There are images of a strong signal at every 1MHz interval at a bandwidth of 1MHz. An image is an undesirable duplication of an actual signal appearing at a frequency multiple. Images appear when the dongle becomes saturated by strong signals.

By reducing the gain, some of these images can be reduced. A larger ADC normally handles the noise floor better. Using an RF filter is advised.

f) RTL-SDR SENSITIVITY

The R820T tuner of the RTL-SDR has an average sensitivity of about -130 dBm over the range of reception see Table 2.7 (Walter, 2013).

TABLE 2.7 SENSITIVITY RATING OF R820T AND E4000

Frequency (MHz)	Sensitivity E4000 (dBm)	Sensitivity R820 (dBm)
24	n/a	-127
52	-139	-134
110	-139	-134
145	-141	-134
435	-139	-135
700	-136	-136
1000	-129	-137

g) RTL-SDR RF DRIFT

As the RTL-SDR dongle heats the crystal oscillator also do cause the received frequency of signals to move over time before it stabilises on a thermal equilibrium where a single offset value is found. The final stabilized frequency offset can be tuned by setting the PPM offset value. Typically, this is not problematic for most applications. The RTL-SDR with TCXO (Temperature Controlled Oscillators) will not show drift as oscillators contain circuitry that automatically calibrates the frequency depending on the temperature.

h) USING MORE THAN ONE DONGLE THAT HAS THE SAME SERIAL NUMBER

Some dongles share the same serial number. This created the problem that only one dongle can be plugin at a time into a PC. The serial number can be adjusted by using the EEPROM programming software for RTL2832 based DVB-T receivers named rtl_eeprom.

i) MAX AMOUNT OF DONGLES THAT USB 2.0 PORT CAN HANDLE

The theoretical limit of a USB 2.0 bus controller that has a device maximum throughput of 256 Mbit/s would be around 5 dongles. In practice, the maximum is 3 to 4 dongles at the same time. If more dongles want to be connected, a PCIe USB card can be added for more USB 2.0 controllers.

j) THE SENSITIVITY OF DEAFNESS

Another limitation of the RTL-SDR dongle is the deafness or sensitivity which decreases its ability to detect weak signals, an additional factor is that the RF front end is very open and thus is not protected and has a high noise floor.

k) RTL_SDR NOISE FLOOR

RTL-SDR internal circuitry generates its noise floor that can create false readings by using V3 dongle, there have been improvements made in comparison to the generic ones in the internal noise floor but this must still be taken note of (see Figure 2.14 (Laufer, 2014)).

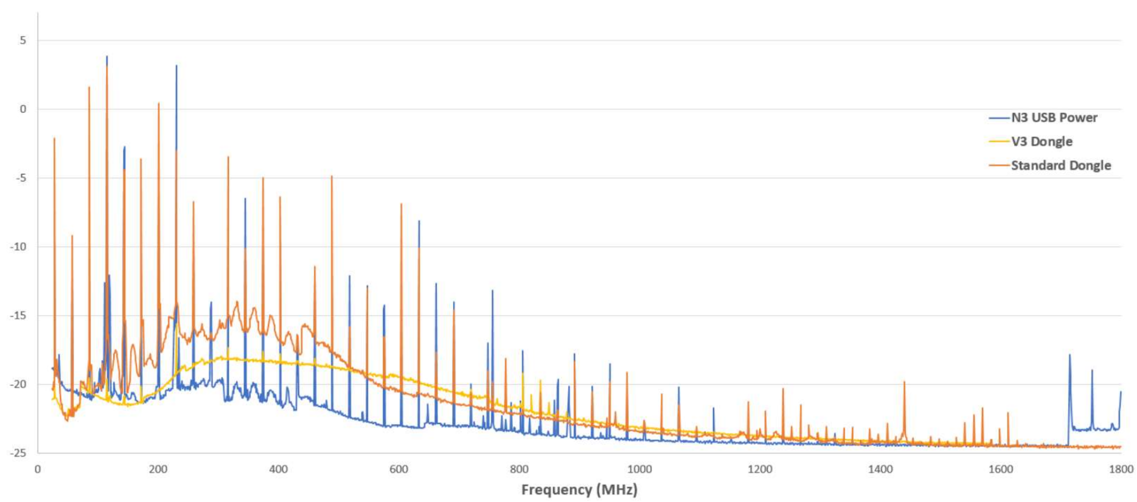


Figure 2.14 RTL-SDR Noise Floor generic vs V3 dongle measurements

2.6.6 RTL-SDR dongle Comparison

As RTL-SDR is a consumer product and most manufacturers do not provide accurate datasheets with their dongle. The question is do RTL-SDR perform the same even if they come from different manufactures? A comparison of some RTL-SDR dongles is provided in this following conference paper see Table 2.8 and Figure 2.15 (Aguilar-Gonzalez et al., 2020).

TABLE 2.8 FEATURE PROVIDED BY THE MANUFACTURER OF THE RTL-SDR

No	Commercial dongle name	Frequency range	Housing	Input	TCXO (PPM)
1	NESDR SMArt	25 MHz - 1.75 GHz	Aluminium	SMA	0.5
2	NESDR Mini 2 +	25 MHz - 1.75 GHz	Plastic	MCX	No
3	NESDR Nano 2 +	25 MHz - 1.75 GHz	Plastic	MCX	0.5
4	NEWGEN.RTL2832 SDR	25 MHz - 1.75 GHz	Aluminium	SMA	0.5
5	DVB-T+FM+DAB+SDR	500 kHz - 1.7 GHz	Plastic	SMA	1
6	NESDR Mini	25 MHz - 1.75 GHz	Plastic	MCX	No

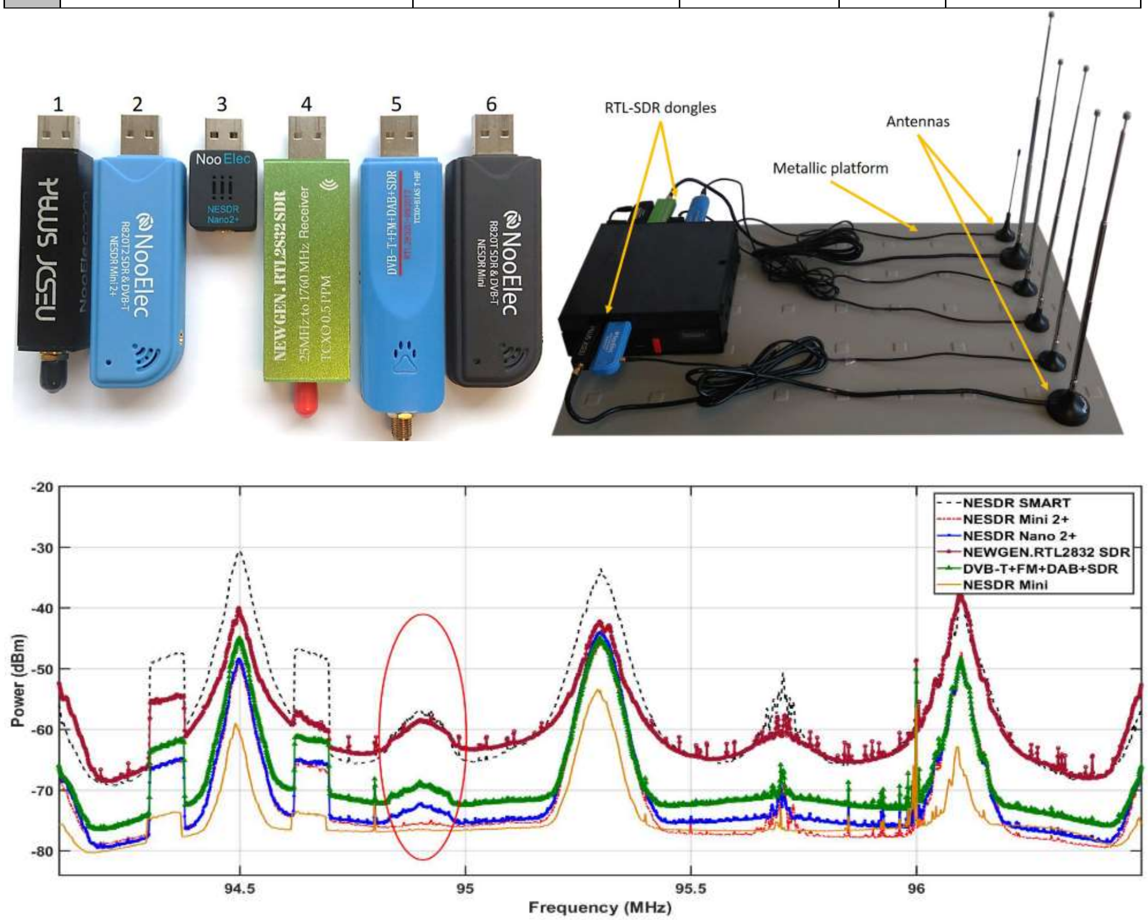


FIGURE 2.15 RTL-SDR REFERENCE DONGLES, THE TEST SETUP AND THEIR COMPARATIVE RESULTS CORRESPONDING TO FM BROADCASTING

In this thesis the writer comes to the following findings:

- From the results is that all captured signals by the SDR have similar behaviour, then the main difference among them is the gain.

-The most sensitive RTL-SDR dongles is that the dongles that come with an aluminium enclosure.

-The result showed that seemingly all dongles have similar characteristics.

2.6.8 Spectrum Analysers VS RTL-SDR in Occupancy Measurement

In the following reference document "Comparison of spectrum occupancy measurement using software-defined radio RTL-SDR with a conventional spectrum analyser approach" (Fanan et al., 2016) there is a test setup conducted to see if RTL-SDR can be used for spectrum occupancy measurement compared to a high-cost spectrum analyser (Agilent E4408B). A spectrum occupancy scan is done by using both devices connected to a range of antennas. A spectrum occupancy scan is conducted in the frequency range between 470-862 MHz and plot results compare. From the results in Figure 2.16 and Figure 2.17 (Fanan et al., 2016), there are no significant differences between the results received by both devices.

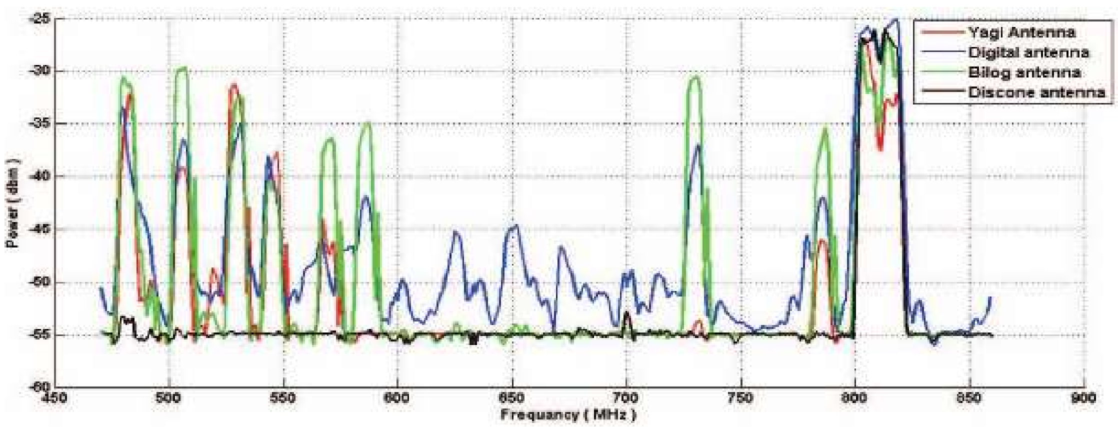


FIGURE 2.16 THE AVERAGE RECEIVED POWER VERSES FREQUENCY BAND BY USING RTL-SDR

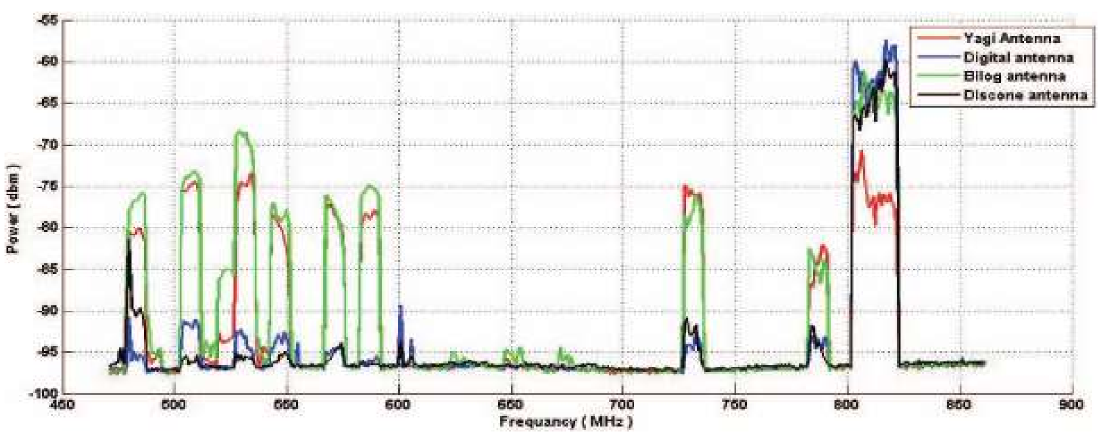


FIGURE 2.17 THE FREQUENCY VERSUS AVERAGE RECEIVED POWER BY USING A SPECTRUM ANALYSER

The paper concluded that it is possible to use the RTL-SDR dongle with minimum external hardware as a low-cost spectrum monitoring device as an alternative to the high-cost spectrum analyser.

2.7 Global Position System (GPS)

The Global Position System (GPS) determines the position according to the time difference between the arrival of GPS messages. GPS Satellites transmit a clockwise circularly polarized signal on two frequencies. Frequency one (L1) has a carrier frequency of 1575.42 MHz. Two codes are modulated based on this frequency. The coarse/acquisition (C/A) contains the data used in civil GPS receivers and precision/secure (P/Y) that is intended for military, special permit, and civilian use. Frequency two (L2) with 1227.6 MHz as its carrier, is used exclusively for accurate targeting and evaluation of ionospheric delay (McDonald, 2011).

The NMEA standards specify the format of digital communication and transmission of information between ship instrumentation of different manufacturers (GPS receivers, radio, sonar, electronic compass, radar, etc). The NMEA 1083 standard defines the requirement for the electrical properties of the signals, to form the serial communication protocol as the transmission of ASCII data, time format and text structure of the messages. NMEA 0183 allows communication at 4800 baud per second (Technology, 1996).

Selected prefixed and message content of the NMEA 0183 is:

GGA - Time, position, GPS position quality and correction data

GLL - Position (Latitude, Longitude), time and status

GSA - GPS DOP and number of active satellites

GSV - Number of visible satellites, satellite ID, elevation, azimuth, and SNR values

GRS - GPS calculation from the value given in GGA

MSS - SNR, signal strength, frequency, baud rate from the transmitter

DGPS RMC - Time, position, course, and speed

VTG speed relative to the earth

ZDA - UTC day, month, year, and local time shift

Important GPS terms:

- TTFF (Time to First Fix) refers to the period between turning on the GPS receiver and determining the first position.
- Hot start - the receiver already has the navigation data and use it immediately after switching on.
- Warm start - the GPS has valid navigation data saved from last decoded messages or obtained from another source there is still loading time needed.
- Cold start - the GPS has no navigation data available and need to get the information by extracting it from the GPS signal itself or other sources. The GPS is much slower in turning its first position.

2.8 MATLAB Simulink

MATLAB is a technical computation language used by engineers, students, scientists, and practitioners. RTL-SDR is used to obtain and sample RF signals transmitted in the frequency range of 25 MHZ to 1.7 GHz. MATLAB-Simulink is used to program receivers using primary principles of DSP algorithms. MATLAB alongside RTL-SDR is used to demodulate and process signals in a practical sense. The RTL-SDR receives live data in the Frequency domain and converts it into data. The RF signal is received at the antenna, this signal is turned into digital form via an analogue to digital converter. This digital data is quadrature-down converted by the RTL-SDR into IQ samples that are processed by the computer running MATLAB. The appropriate DSP (digital signal processing) algorithms are used by Matlab to extract the information signal and demodulate the signal to the baseband (see Figure 2.18). Signal strength measurements are recorded. The accuracy of the RF signal received is influenced by internal and external factors.

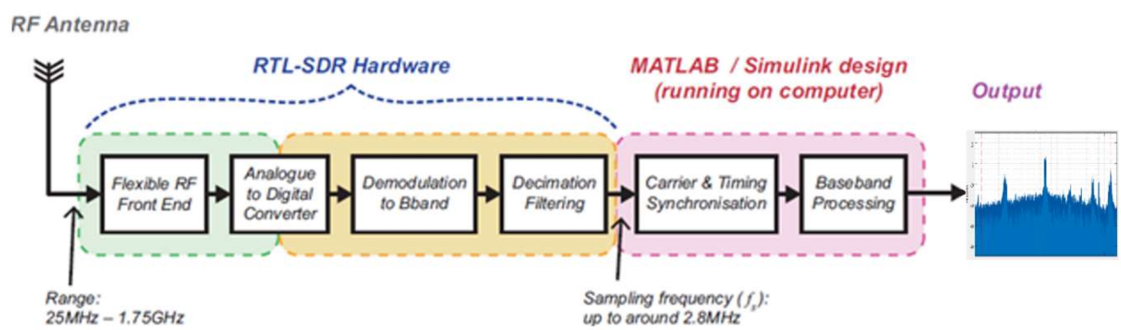


FIGURE 2.18 BLOCK DIAGRAM OF RTL-SDR INTERFACE WITH MATLAB

Chapter 3 : Design of Low-cost MMN Measurement System

3.1 Introduction

As there is an increased uptake of technologies that need radio frequency spectrum and this being a limited resource, there has been an ever-increasing need to use the radio frequency spectrum more efficiently and optimising the use of this spectrum. Each country's regulatory body actively monitors the spectrum use and enforces compliance. Radiofrequency spectrum monitoring task is performed by spectrum monitoring towers. These monitoring towers are very expensive to build, maintain and has a very limited footprint. To mitigate this issue, this paper presents a solution where a low-cost RF MMN monitoring system is to be designed to be installed in pre-existing transport systems. This proposed RF MMN monitoring system must be a cost-effective alternative to current monitoring systems.

In this chapter, we will be using the theoretical research conducted in chapter 2 to achieve the following three aims: i) Design of building a prototype of the mobile RF MMN measurement system while analysing what is the most cost-effective approach in making use of readily available commercial consumer-based products to build a prototype (3.2 & 3.3). ii) Reach a clearer understanding of the unwanted effects on the observed RF noise floor induced into the RF MMN monitoring systems measurement results by its inherent (internal) electronic noise and the vehicle electronics that the system is installed in by making use of MATLAB Simulation (3.4). iii) Investigating how to reduce the amount of data storage required for recorded measurements collected by the RF MMN monitoring system by making use of the GPS coordinates and spatial selective sampling (3.5).

3.2 Design of Low-Cost Radio Frequency MMN Measurement Station

3.2.1 Background

As the need for an RF MMN monitoring station was presented in the previous chapters. This section discusses the design of a system that has the capability of scanning and recording the RF spectrum while moving through a specific area.

The mobile monitoring station needs to adhere to the ITU standards Rec. ITU-R SM.1723-2 (Sm & Spectrum, 2011). Special considerations need to be made as the MMN monitoring system must have the capability to be installed in several public vehicles, this creates extra security, RF noise floor and accessibility issues. Careful consideration needs to be taken in the selection of equipment to obtain accurate and meaningful results and to increase the capability to detect the measured signal in the frequency band of interest and to keep the noise floor low. Since long extended time, mobility, continuous recording, and flexible designs wanted, reconfigurable RTL-SDR connected to a computer are some of the basic specifications. A suitable setup for a single measurement device consists of an SDR as well as supporting components such as filters, attenuators and LNA, which are used to adapt the dynamic range of the signals to prevent overload and increase sensitivity. To keep the cost down existing readily available commercial consumer-based products are discussed to build a prototype. A cost analysis is conducted to confirm if the prototype is built cost-effectively.

3.2.2 RF MMN Monitoring System Build Model

The minimum specification of the RF MMN measurement system is a device that can measure frequency band in use via radio receiver or spectrum analyser, an appropriate antenna, GPS, PC/controller suitable post-processing software seen in Figure 3.1. Each piece of equipment has a specific function within the model discussed in Table 3.1

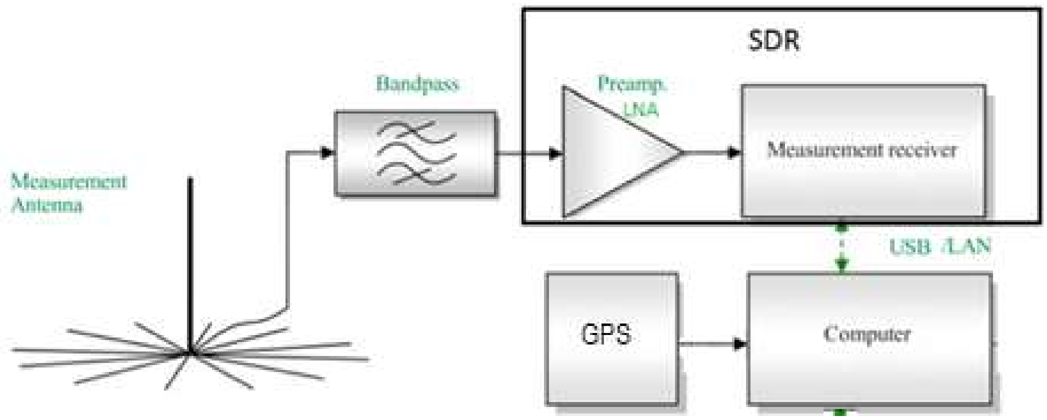


FIGURE 3.1 HARDWARE DESIGN OF PROTOTYPE SDR RECORDING SYSTEM THAT WILL BE MOUNTED IN PUBLIC TRANSPORT

TABLE 3.1 EQUIPMENT FUNCTION WITHIN BUILD MODEL

Equipment	Purpose/Function
Antenna	Converts the RF signal in the frequency domain into a voltage signal
RTL-SDR	Coverts low voltage signal into a digital form ready for processing
Computer or embedded PC	The information received from the RTL-SDR and GPS processes, stored and converted into a form that the user can use.
GPS	Receives location information and converts it to coordinates suitable for processing
Low noise amplifier (LNA)	Amplifies the low voltages received by the antenna
Pre-selector bandpass filter and Filtering components	Blocks or reduces unwanted voltage signals
Power supply/Batteries	Supplies power to all electrical equipment

3.3.3 Prototype Components Evaluation

3.3.3.1 Omni-directional Magnetic Whip Antenna

The antenna receives the radio frequency from the external environment so that this can be fed into the RTL-SDR via an SMA connector. This antenna needs to be an Omni-directional antenna with a wide frequency range that covers the frequency range of the RTL-SDR that is 0.5 to 1766 MHz. The antenna needs to be lightweight with a magnetic conductive ground plate base to be put onto the vehicle roof. The conductive metal base with a conductive ground plane improves coupling to the ground plane. The ground plane must be connected to the shield of the coax cable. This helps with blocking unnecessary induction of frequencies into the feeder in the cable running between the antenna and RTL-SDR. This silver base provides a much lower noise floor and higher signal SNR due to the better ground plane and ensures better performance. The antenna must be mounted to the rooftop, withstand speeds above 120km/h and local environmental conditions. The antenna is to be mounted on the highest point on the vehicle and the antenna feeder run needs to be kept as short as possible.

The antennas that were identified were the RTL-SDR Magnetic Whip Set. They have a heavy Duty Conductive Magnetic Whip Base with 2M RG59, 1x 9.5cm fixed metal whip (usable from 100 MHz to 2 GHz+), 1x 17cm to 1m telescopic whip (usable from 100 MHz - 400 MHz). (see Figure 3.2 and Figure 3.3 (Various, 2019).)

Figure 3.2 RTL-SDR Blog Mag Base Whip Antenna

i) ii) iii)

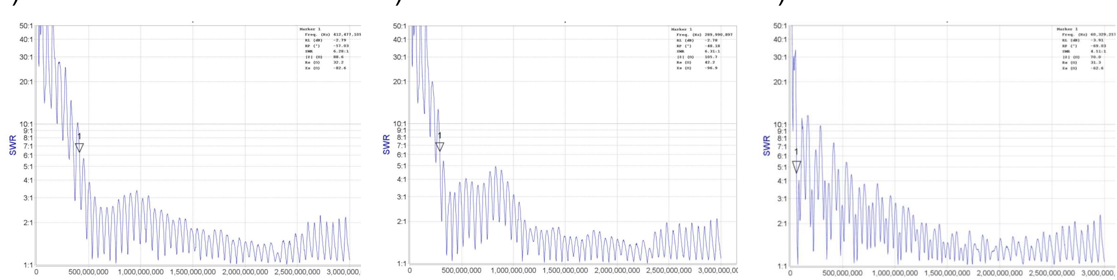


FIGURE 3.3 RTL-SDR BLOG MAG BASE WHIP ANTENNA SWR VS FREQUENCY TEST RESULTS (HZ) I) 412 MHZ, II) 29 MHZ III) 60 MHZ

3.3.3.2 Bandpass filter

One of the disadvantages of the RTL-SDR is that the RF front end is very open and has no pre-selector (see section 2.7.5). The SDR's open RF front end can create harmonics within the equipment and this can cause ghost signals which will give faulty measurements (Collier, 2017). When the gain is increased this can cause an increase in interference in the measurement readings. This interference creates a very poor performance in other bands and will make the RTL-SDR less sensitive when strong out-of-band signals are present e.g., FM and AM transmissions.

An RF filter at the front end will reduce the RTL-SDR image effect, radio interference, and improve the signal to noise ratio. The bandpass filter can significantly improve reception by attenuating (blocking) any signals that the filter is designed for.

The filter needs to have very low insertion loss in the frequency band that is desired to be measured. This insertion loss needs to be added to the RF link budget. The filter must have matching impedance to reduce losses.

The bandpass filter that is suggested is the SMAKN UAF42[1] active band-pass filter module with a high-pass, low-pass, and band-pass active filter module adjustable Q value.

3.3.3.3 Low-Noise Amplifier (LNA)

An external Low Noise amplifier assists in reducing the systems Noise Figure (NF). The preamplification reduces the noise floor by amplifying the signal before it gets to the RTL-SDR. The LNA also increases gain which helps to prevent signal losses due to long coax cable runs from the antenna and losses from filters. The internal amplifier of the RTL-SDR can cause problems as it will also amplify the inherent (internal) electronic noise (see section 3.4).

The LNA must have large gain to better overcome losses due to coax run and filters. It must have a very low NF as the NF is very important in frequencies above 1GHz. The LNA must have a high IP3 performance rating as this indicates how linear the amplifiers will operate (Laufer, 2017).

Position the LNA as close to the antenna as possible because the LNA amplifies both the signal and the noise. Do not set the gain too high as this will increase the signal strength

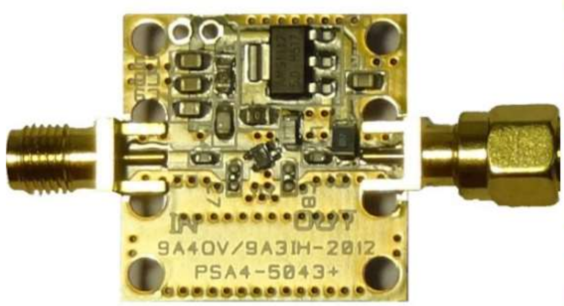
[1] https://www.amazon.com/SMAKN-low-pass-

at the dongle input and this will cause the RTL-SDR to overload, create harmonics in the system which creates bad reception.

The LNA suggested is the LNA4ALL model (see Figure 3.4) which is easily available commercially (Adam, 2015).

LNA4ALL specifications are:

- Low noise preamplifier will add 20-30db gain depending on the frequency,
- The amplifier is built around Mini-Circuits PSA4-5043+ E-PHEMT
- Ultra-Low noise MMIC amplifier operating from 50 MHz to 4 GHz,
- Has a high IP3 performance with an internal match to 50 ohms,
- The 500mA 5V voltage regulator is integrated on the board where any voltage from 6 V to 9 V DC can be used for the supply,
- the power consumption is between 55 mA and 65 mA,
- LNA4ALL has a Bias-T that send power down the cable and the power makes the pre-amplifier work,
- Less than 1 dB nominal noise figure.,
- Has a high IP3 performance with an internal match to 50 ohms,
- Power consumption is between 55 mA and 65 mA,
- Quad mounting holes for a variety of installations options,
- Female SMA connectors on the input and output for easy connectivity and insuring stable operation overall range of frequencies.



Application	Frequency MHz	Gain dB
HAM radio	28	21
HAM radio	50	22
HAM radio	70	22
Broadcast	100	22
Air traffic	120	23
HAM radio	145	23.5
HAM radio	435	23.5
DVB-T	600	22
ADB-S	1090	18
HAM radio	1296	17
SETI	1400	16
WBFM link	2200	12
HAM radio	2320	11.5

FIGURE 3.4 LNA4ALL PCB AND FREQUENCY VS GAIN STATISTICS

3.3.3.4 Software Defined Radio (SDR)

The radio frequency signal is received by an external antenna and this analogue signal gets fed into the antenna input of the SDR. The SDR is used to convert the received radio power into a digital signal called I/Q data that can be manipulated to fit our purpose. RTL-SDR can be used as an inexpensive radio receiver and outputs the I/Q data that is needed for the RF MMN monitor fit for our purpose (see section 2.6.8). For further reading of RTL-SDR see section 2.6

Choose an RTL-SDR dongle with a metal enclosure this will act as an RF shield reducing external interference that might be induced in the SDR receiver and will help with dissipation (see section 3.4). Find an RTL-SDR with a temperature compensated oscillator (TCXO) which ensures little to no drift due to temperature changes thus improving the component tolerance (see section 2.6.5). SMA connector is better as there is very low insertion loss over the frequency band. A comparison was done between RTL-SDR and certain specifications were identified that will assist with the reception of certain frequencies (section 2.6.6).

Position SDR as close to antenna and LNA as possible but still have a metre of feeder run due to the antenna and LNA picking up the RFI given off by the RTL-SDR. Shield RTL-SDR as much as possible from power supplies, especially switch mode power supplies as these tend to introduce lots of RF noise. Use a USB ferrite to assist with shielding between PC and an RTL-SDR dongle.

Due to budget constraints, the SDR that was selected is the RTL-SDR R820T2 RTL2832U 1PPM TCXO SMA SDR (see Figure 3.5 and Table 3.2)(Various, 2020a). One of the improvements made over generic brands is the Rafael Micro R820T2 tuner. The RTL-SDR has a software switchable bias-T circuit that allows you to power external devices like LNA or active antenna, Lower overall noise, and built-in direct sampling for HF reception.

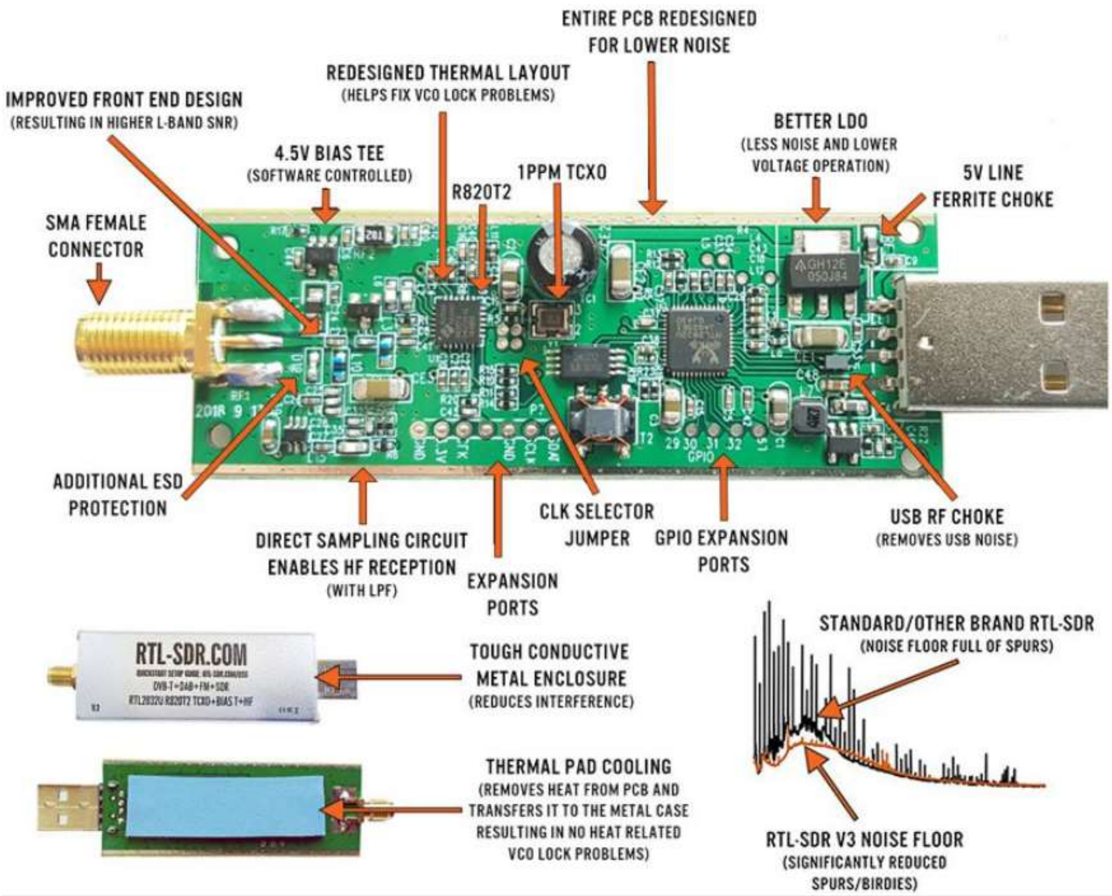


FIGURE 3.5 RTL-SDR R820T2 RTL2832U 1PPM TCXO SMA SDR(Various, 2020a)

TABLE 3.2 SUMMARY OF BASIC PARAMETERS OF RTL-SDR (R820T2 AND RTL2832U)

RTL-SDR Parameters	Values
Demodulator	Realtek RTL2832
Receiver	Realtek R820T2
Frequency Range	0.5-1766 MHz
Maximum sampling frequency	2.4 MS/s
Input Impedance	50 ohms
Maximal power of receiver signal	+10 dBm
Oscillator stability	<1 PPM TCXO
Connectivity	USB 2.0
Sensitivity (FFT: 8k, 64 QAM, CR: 7/8)	-79.5 dBm
Antenna Port	SMA female antenna port
ADC	8-bits

3.3.3.5 Computer System

The output data of the SDR and GPS data needs to be processed, recorded, and manipulated by a computer. In the design and calibration, a laptop of decent processing power is needed to record and decode I/Q data that is generated by the SDR.

The bare minimum specifications are a dual-core processor with at least 1 GB of Random-Access Memory, with a High-speed USB 2.0 or newer port and running on Windows 7 or later. Slower embedded microcontroller-based computers similar to Raspberry Pi can also be used with the Linux operating system and command line as the interface (Laufer, 2014).

The laptop that was available for the prototype is the MSI CX70 Laptop, BIOS: E1758IMS Ver:1.09, Processor: Intel® Core™ i7-4702MQ CPU @ 2.20 GHz (8 CPU), ~2.2GHz, Memory: 8192MB RAM, DirectX 11. (Various, 2020)

Software Installed: Operating system Windows 10 Pro 64-Bit, MATLAB R2020b (9.9.0.1467703) 64-Bit (win64), Simulink and Add-on installed (Various, 2020f):

- Communications System Toolbox Package for RTL-SDR Radio version 20.2.0
- Communications System Support Toolbox version 7.4
- DSP System Support Toolbox version 9.11
- Signal Processing Support Toolbox version 8.5
- Simulink version 10.2

For the public vehicle installation to make the system more cost-effective the RF MMN measuring system will be installed on a Raspberry PI4 (Various, 2020e)(see Figure 3.6). Raspberry Pi 4 is a Linux based single board card-sized computer and has enough processing power and USB 3.0 ports to support multiple SDR. The Raspberry Pi has a Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz processor, 2GB to 8GB LPDDR4-3200 SDRAM, 2 USB 3.0 ports; 2 USB 2.0 ports, etc. which will be able to process the GPS and SDR information. This low-price platform is flexible and versatile for editing, upgrading, and debugging. The operating system runs from an SD card that can be easily duplicated and backed up. The general-purpose I/O (GPIO) pins can easily be used for external peripherals for monitoring and controlling the system.

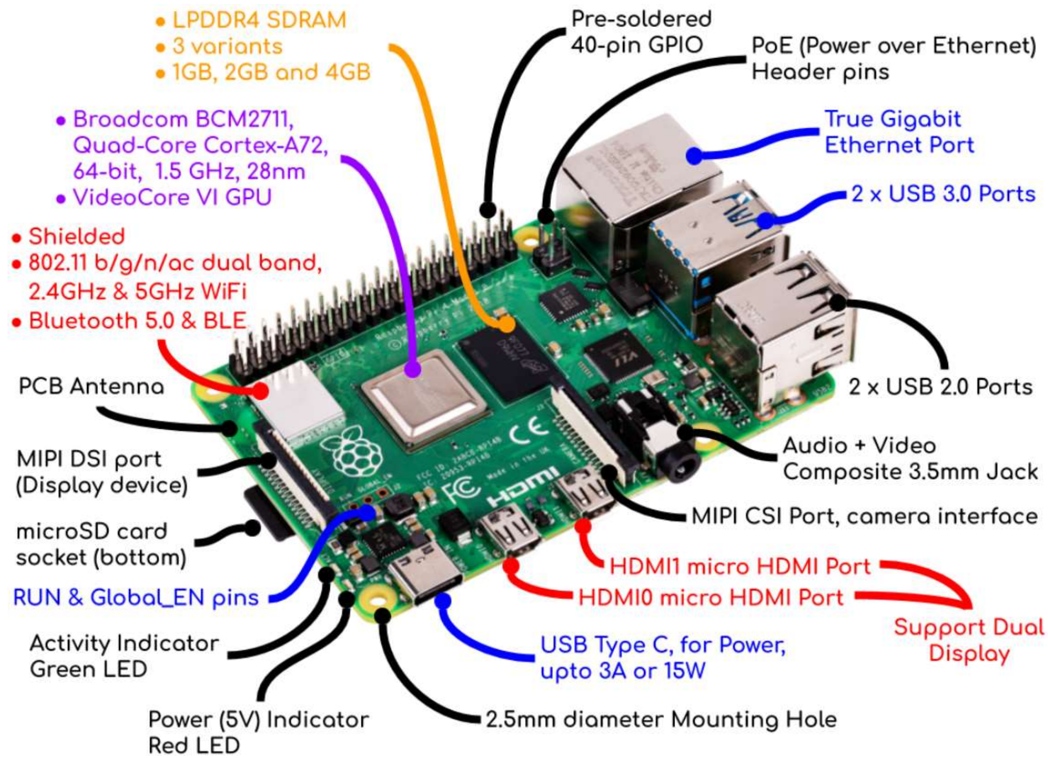


FIGURE 3.6 RASPBERRY PI 4

3.3.3.6 GPS

The location data will be received via GPS in form of longitude, latitude, altitude, standard time, date, speed, altitude, and travel direction data (see section 2.7). The GPS needs to comply with the performance requirements set out by the ITU discussed in the ITU regulations ITU-R-REC-SM.1723-2-2011 (Sm & Spectrum, 2011) for GPS receivers. The GPS needs to be magnetic and have a long enough cable so the GPS can be placed on the roof of the vehicle to improve reception.

The GPS that is used is the VK-162 G-Mouse USB GPS Dongle Navigation Module (see Figure 3.7 & Table 3.3). This GPS receiver is a low-cost high sensitivity receiver designed for mobile applications in field automation. The GPS is plugged into the computer via USB and is compatible with MATLAB and Linux software. The communication takes place following NMEA 0183 via the virtual serial line. The position, timing, velocity, elevation, and azimuth data is also available. For improved position accuracy, the unit can decode Differential GPS (DGPS).

Figure 3.7 GPS receiver Gmouse VK-162(Various, 2020c)

TABLE 3.3 PARAMETERS OF THE GPS RECEIVER GMOUSE VK-162

VK-162 GPS Parameters	Values
GPS chip Ublox	G6010/G7020
Frequency L1	1575.42 MHz
Code	C / A
Accuracy	5 m (3 m DGPS)
Coordinate system	WGS-84
Time accuracy	1 μs
Tracking sensitivity	-162 dBm
Position reading period	0.1 s
GPS protocol NMEA 0183	Version 3
Data refresh rate	1Hz-10Hz refresh rate
Data baud rate	9600
Cable	2m

3.3.3.7 RF Dummy Load

The RF dummy load will be used for calibration and noise filtering of the system (see section 3.4). A Dummy Load is a device used to simulate an antenna load and electrical load during testing. By using a dummy load as an alternative to an actual antenna, the transceiver can be configured and tested without radiating radio waves. The dummy load selected must have a similar impedance to the RTL-SDR (50-ohm) input. Using the dummy load, the port will not be open, and this can stop unwanted reflections.

The dummy load that was selected is Mcbazel Surecom 0014-0103 AHUAGO 5W load. The dummy load is a 50-ohm ± 5% load, has an SMA male connector and is stable at 5 Watt DC-6GHz[2].

3.3.3.8 Optional Components

a) **Power supply** – If the system is running off a laptop, then it is preferred to run off the laptop's battery as an inverter or laptop charger will create extra RFI. For when the system is fitted to a Raspberry Pi, the RF MMN monitoring system needs a reliable 5V low noise power supply. This power supply must be able to run off the 12-volt lighter socket fitted in all public transport. Here are some options as power supply:

- A low-cost standard 5-watt high power lighter socket 5-volt cell phone charger plug (might create unnecessary RF noise and no battery backup)
- Step down Converter 12V to 5V 3A 15W DC Buck Converter Module, DC to DC Regulator Step-Down Car Power Converter (high power, might create unnecessary RF noise and no battery backup)
- 12V portable Jump Starter with 5 Volt USB output. Because the monitoring system gets power from a battery, there is less likely to be unnecessary RF noise. The battery allows the monitoring system to have a reliable power source and to stay on for many hours after the public transport has been switched off. Thus, the GPS will not switch off and no movement data will be lost.

b) **Weatherproof case with good IP6 rating** - Protective case to keep all electronics safe when in transit and the metal case will assist with RF shielding.

c) **SD card** storage to install the Raspberry Pi Operation system.

d) **USB 3.0 External hard drive** to extend RF MMN monitoring systems range adding more storage for measurement recordings for longer monitoring sessions.

e) **Noise Filtering** – To filter the +5V supply to the mobile monitoring system by using a combination of ferrite beads and bypass capacitors to target the full spectrum of noise. Adding a ferrite chokes on the USB cable to filter out computer noise (Collier, 2017). Adding a coax choke to the coaxial cable of the antenna will stop unwanted currents in the coax shield (see Figure 3.8).

---

[2] https://www.amazon.com/Mcbazel-Surecom-0014-0103-AHUAGO-OHM/dp/B072MF2LR2/ref=sr_1_6?dchild=1&keywords=Dummy+Load+50+Ohm+sma&qid=1603826046&s=electronics&sr=1-6 28 October 2020.

FIGURE 3.8 FERRITE COAX CHOKE ADDED TO THE ANTENNA CABLE

3.3.4 RF MNN Monitoring System Budget

Since one of the main objectives of this paper is the design of a mobile deployed cost-effective RF MMN measurement system that can be safely installed in public transport, it will therefore be fitting to look at the total cost of the system (see
In the research of this paper, no commercial product of this nature has been found by the author thus no comparative cost analysis could be conducted.

Table 3.4).
To cut down on costs, only off-the-shelf components are used, and the price of the system will vary depending on frequency measured, type of SDR, filter needed, computer, storage allocation, power supply type of installation and extras used.
In the research of this paper, no commercial product of this nature has been found by the author thus no comparative cost analysis could be conducted.

TABLE 3.4 PORTABLE MONITORING SYSTEM COMPONENT LIST AND BUDGET

Component List	Price
Omnidirectional Magnetic Whip antenna(Various, 2019)	R250
Band-pass filter[3] (optional)	R800
LNA (Low Noise Amplifier)	R300
RTL-SDR R820T2 RTL2832U 1PPM TCXO SMA SDR(Various, 2020a)	R570
Raspberry Pi 4 Essentials kit (4GB Ram)(Various, 2020e)	R1400
GPS receiver Gmouse VK-162(Various, 2020c)	R 280
32GB micro-SD card	R 90
Weatherproof Case	R 200
12 Volt to 5 Volt Power supply / Jump starter (optional)	R50 - R1300
USB 3.0 External Hard Drive (optional)	R1000
50-ohm dummy load(optional)	R120
Ferrites(optional)	R50
Total cost	**R1960 - R6360**

3.3.5 Conclusion

Based on the information acquired in

In the research of this paper, no commercial product of this nature has been found by the author thus no comparative cost analysis could be conducted.

[3] https://www.amazon.com/SMAKN-low-pass-band-pass-filter-Frequency-adjustable/dp/B07DX4VFFT

Table 3.4 the main objective is achieved in finding existing readily available commercial consumer-based products to build a prototype of the cost-effective RF MMN measurement system. Based on chapter 2.4.5 &

Table 2.3 Classification comparison of Spectrum measurement devices by (Gavrilovska et al., 2014)

the proposed RF MMN measurement at a budget of between R1960 to R6360 this system keeps within the threshold of R20 000 that defines low-end devices.

3.4 Measuring Induced RF Noise into the RF MMN Monitoring System by Inherent Noise and Transportation Vehicle

3.4.1 Background

The proposed solution to the problem that was identified is to install a Low-Cost RF MMN Monitoring System into a vehicle as discussed in Chapter 1. It was determined in Chapters 2 and 3 to keep down the cost of development, consumer-based products would be used. These consumer-based products all create their own internal (Inherent) noise due to the components used. This internal noise can influence the noise floor of our monitoring system and lead to false readings and errors. For this reason, the noise floor parameters need to be calculated and simulated to improve overall performance to avoid or cancel out errors as much as possible. Secondly, the vehicle that is transporting the monitoring system also creates interference that will be inducted into the measuring equipment. This will also influence the noise floor and can cause undesirable measurement results. This external interference caused by the vehicle must also be determined. Furthermore, the internal and external unwanted interference from the public transport electronics needs to be minimized in the most cost-effective way for data integrity thus simulations might solve these issues by pre-recording the noise generated and using this information for future analysis.

The aim is to make a simulation model of the noise floor in the complete system and the following objectives have been identified to achieve this aim:

3.4.4.1 Simulate a spectrum sweep of the pre-selected frequencies

3.4.4.2 Determine inherent (internal) noise floor of the Spectrum monitoring system with spectrum sweep and safe data as CSV

3.3.4.3 Determine the RF noise generated by the vehicle plus spectrum monitoring system by doing Spectrum sweep and saving the generated measurement as CSV.

3.4.2 Model Conceptualization

A model of the RTL-SDR is created to simulate the effects of internal noise, external noise, and external circuitry of the system (see Figure 3.9)

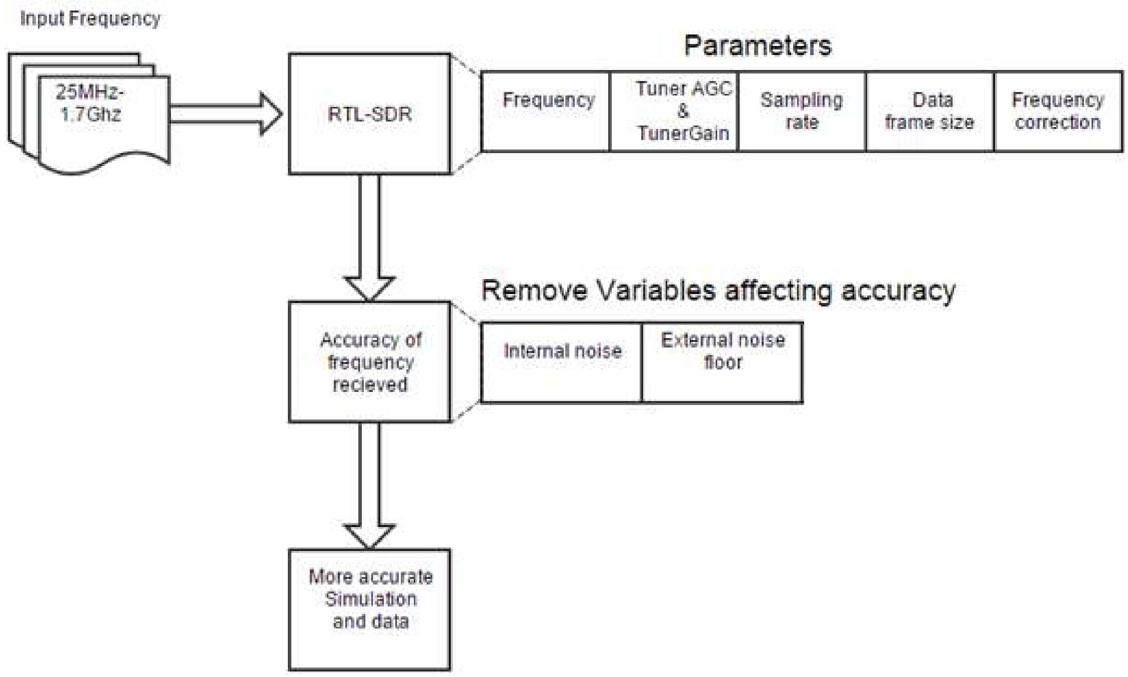


Figure 3.9 Conceptual model of parameters affecting our Radio Frequency measurement accuracy

3.4.3 Data Collection Variables

The variables for the simulation of the monitoring system:

Centre frequency (start_freq, rtlsdr_tunerfreq and stop_freq) The centre frequency of the input signal for the RTL-SDR radio in Hz. If you stipulate a value out of range for your device, you will find an error. The desired centre frequency is nonnegative scalar double-precision with the default value at 102.5 MHz. The valid range of this property hangs on the tuner chip of the RTL-SDR radio. These tuner chips we are using has a frequency range of 25MHz to 1.7GHz.

EnableTunerAGC (AGC) Turn the tuner automatic gain control (AGC) off (false) or on (true). When set to false, this property will disable the tuner AGC to provide no continuous amplitude signals. The default value is true with AGC on, an almost constant amplitude signal.

Sample Rate (rtlsdr_fs) Stipulate the preferred sample rate of the output samples of the step method, measured in Hertz (Hz), as a positive, double-precision, scalar value. The maximum ADC sample rate is 3.2 MHz. The default sample rate is 250 kHz. The maximum theoretical ADC sample rate is 3.2 MHz. To calculate the bandwidth, multiply the designated sample rate by the number of bits per sample. The amount of bits in the raw IQ stream is 8 bits for the I (In Phase component) and 8 bits for the Q (Quadrature Phase component) stream. By using the following calculation where SM (Sample Rate Ms/s) and TCPBW (TCP Bandwidth Mbit/s)

$$SM \times (I + Q) bits = TCPBW \qquad \text{Eq 3-1}$$

As an example, running at the sample rate of 2.8 Mbps we need 2.8 x (8+8) = 44.8 Mbit/s. For comparison USB 2.0 runs at a theoretical maximum of 480 Mbits/s. The RTL-2832U is the most stable at 2.8 MHz.

Output Data Type (rtlsdr_datatype) The output port supports these complex data types only: Single-precision floating-point, 16-bit signed integers, Double-precision floating-point. When you select a single or double data type, the complex values are scaled to the range of [-1,1]. The default value is double for ease of calculation and maximum accuracy.

Samples per Frame (rtlsdr_frmlen) The number of samples in a frame stipulate the number of samples in a frame for the step method to output. This value should be a scalar integer, positive that is an integer multiple of 256 and the default value is 1024. For our simulation, we are using a sample rate of 4096 for maximum resolution.

Frequency Correction (rtlsdr_ppm) The frequency correction value in ppm as an integer with a valid range at [-1e4 1e4] ppm and the default value set at 0. This parameter value is used to correct for frequency shifts that arise because of local oscillator offsets or clock rate inaccuracies. The 28.8 MHz crystal oscillator in some RTL-SDR is it is the heartbeat of the circuit (clock source). The RTL-SDR uses a 28.8 MHz oscillator which might be off by a few kHz. The effects are that frequencies may not be precisely where it expects to be. Secondly, as the dongle heats the crystal oscillator will too be affecting the received frequency of signals to wander over time before becoming stable on a single offset value. Generally, this is not a challenge for most applications. The last frequency offset can be changed by establishing the PPM offset value. For this simulation, it is not going to include this value this our PPM will be set at 0.01.

The number of frames to receive hold (nfrmhold) The number of frames to receive and is averaged over every time the RTL-SDR is returned. This number would be as big as possible, but increasing it exponentially increases the quantity of time the code takes to execute. The mean power will be returned for every spectral component and calculated. Calculating the right sample size is crucial to gaining accurate information. For a 10 kHz sweep, the script takes about +/- 2 seconds for a sample rate of 1000 number frames but in the test, there was found a deviation of less than 0.01 was between the sample rate of 3000 frames and 500 frames. Due to the wide frequency spectrum range and simulation running time our default averaged a number of frames is 500 but for better accuracy, the sample rate can be increased.

3.4.4 Simulation & Model output

In the simulation, the RTL-SDR software that we are going to use is MATLAB and SIMULINK version R2020b or later. RTL-SDR hardware support package is installed and the following add-on packages: DPS System Toolbox, Communication System Toolbox, and Signal Processing Toolbox are also installed for added functionality and flexibility. MATLAB and Simulink provide an environment where any SDR receiver algorithm can be implemented and displayed in various graphical forms.

For all experiments, the RTL2832 USB DVB-T receiver is plugged into the USB port of the laptop. Always complete a pre-check before experiments if the dongle is working by formatting the command prompt of MATLAB (see Figure 3.10). Matlab code in Appendix A-1 *rtlsdrinfo.m* for running the test.

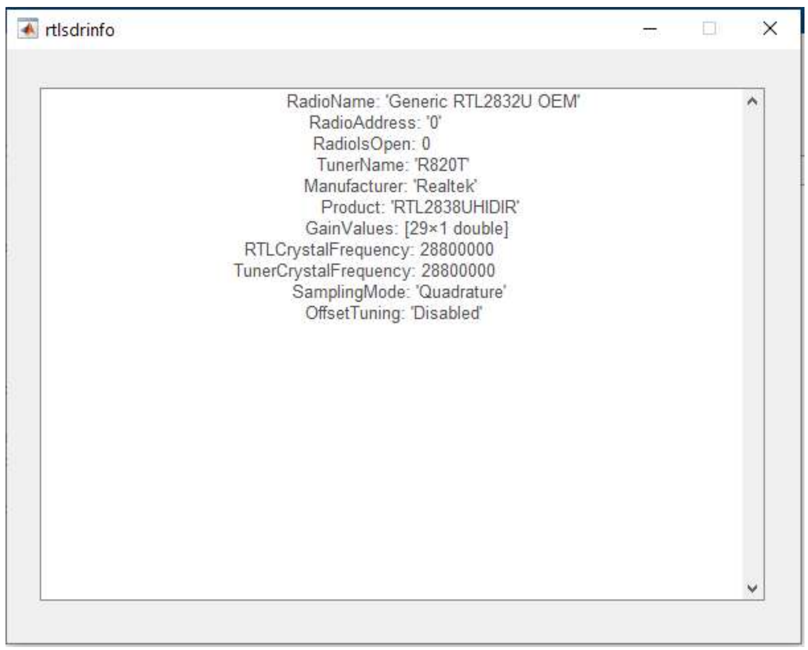


Figure 3.10 Output of Matlab command prompt confirming that RTL-SDR is working

3.4.4.1 Radio Frequency Plot (Spectrum Sweep)

RTL-SDR can plot a wide range of frequencies from 22 MHz to 1.7 GHz due to this it can be difficult to represent such a wide range frequency. The RTL-SDR also has a few parameters that need to be pre-selected and used to represent the best outcome for the RF spectrum thus the following model was created to assist in solving this issue (see Figure 3.11).

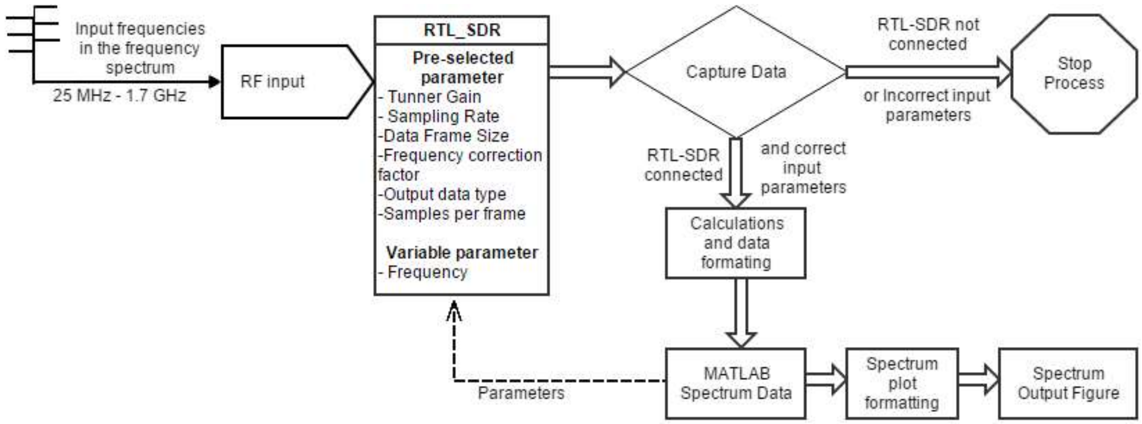


Figure 3.11 Model of Spectrum Sweep System

The spectrum sweep can be done with or without the antenna being connected to the antenna connector of the RTL-SDR. The output of spectrum sweep was obtained (see Figure 3.12) and an example of the output CSV is seen in Table 3.5

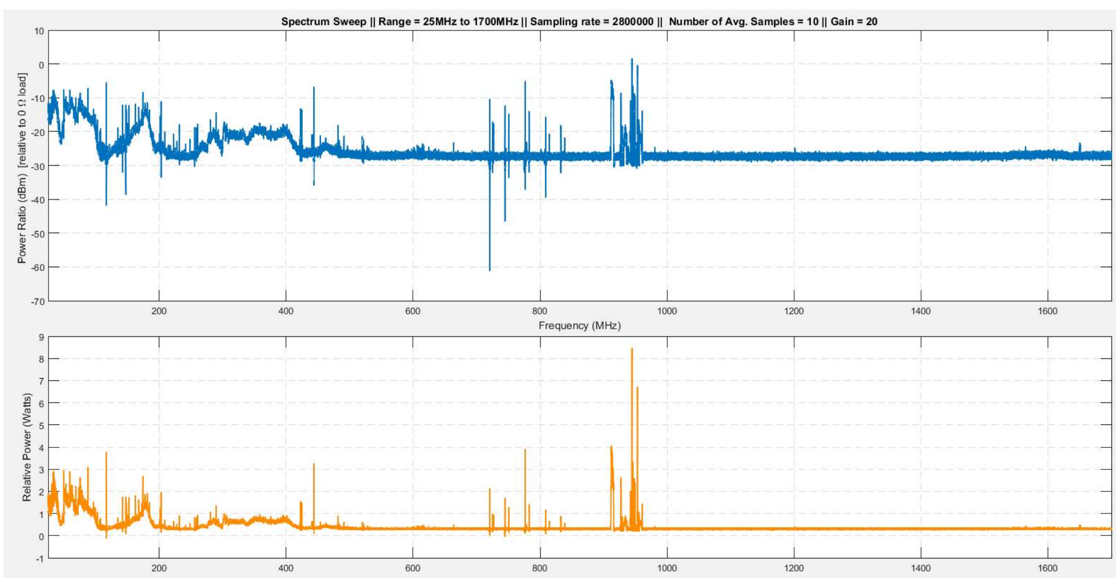


Figure 3.12 RTL-SDR Spectrum Output Plot form 25MHZ to 1.7 GHz

Table 3.5 Example of CSV output file that is generated by the spectrum sweep

Frequency (MHz)	Power Ratio (dBm)	Relative Power (Watts)
87.42	-19.249	0.77098
87.431	-18.988	0.79445
87.442	-19.152	0.77965

3.4.4.2 Inherent (internal) noise

As in the design of the RTL-SDR dongle, it will always be spurious (unwanted) signal noise spikes at multiples of 28.8 MHz, as this is created by the internal oscillator frequency and strong spurs of the USB clock frequencies at 48 MHz. There is regrettably not much that can be done about this other than to see where they are and overlook them. These spurs look like a signal thin solid line on the RF spectrum or waterfall. It will similarly sound like a single audio tone on the USB/LSB modes. At a bandwidth of 0.25 MHz and 1 MHz, there will be strong images of signals at every 250 kHz and 1 MHz interval. An image is fundamentally an unwanted duplication of a real signal being seen at multiple frequencies. The recommended equipment settings in measuring equipment noise floor(Various, 2013):

- Measurement Time - For raw data sampling run one scan if at least 0.5s length every 10s to 30s.
- Frequency range - depends fully on the chosen frequency band of measurement.
- RBW - For raw sampling, the principle dictates an RBW of half of the sampling frequency. The recommended RBW values are:
  - 30 MHz – 450 MHz at 100KHz
  - 450 MHz – 1 GHz at 300KHz
  - 1 GHz – 3 GHz at 5 MHz

The model to record this unwanted internal noise by isolating the RTL-SDR and shorting the radio frequency input terminal with a 50-ohm isolator (see Figure 3.13)

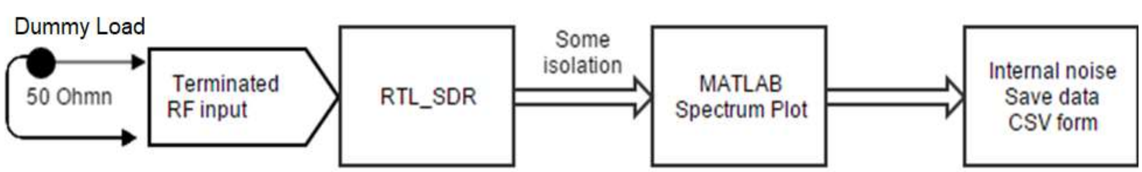


Figure 3.13 Conceptual model setup to measure inherent noise

To test the internal noise of the RTL-SDR the dongle needs to be isolated as much as possible by making the use of a USB extension cable and putting the RTL-SDR dongle as far away from the electrical apparatus as possible. To measure the system noise a manual measurement can be performed by replacing the antenna with a 50-ohm dummy load and measuring the sum of the system noise and load terminal noise(Various, 2013). This 50-ohm dummy load are connected to the antenna connector of the RTL-SDR is used to isolate external noise coming into the receiver (see Figure 3.14). It must be noted that some RTL-SDR dongles have a nominal input impedance of 75 Ohms, whereas the coupler has a impedances of 50 Ohms in the calculations there is taken into account of the difference of

1.6dB (Hb9Ajg, 2013). The following script must be opened on Matlab. rtl_specsweep_internal_noise.m with a successful run, MATLAB will provide the spectrum sweep between the selected frequencies and create a CSV file containing all the outputs of the sweep (see Figure 3.15).

Figure 3.14 Setup to test Inherent Noise

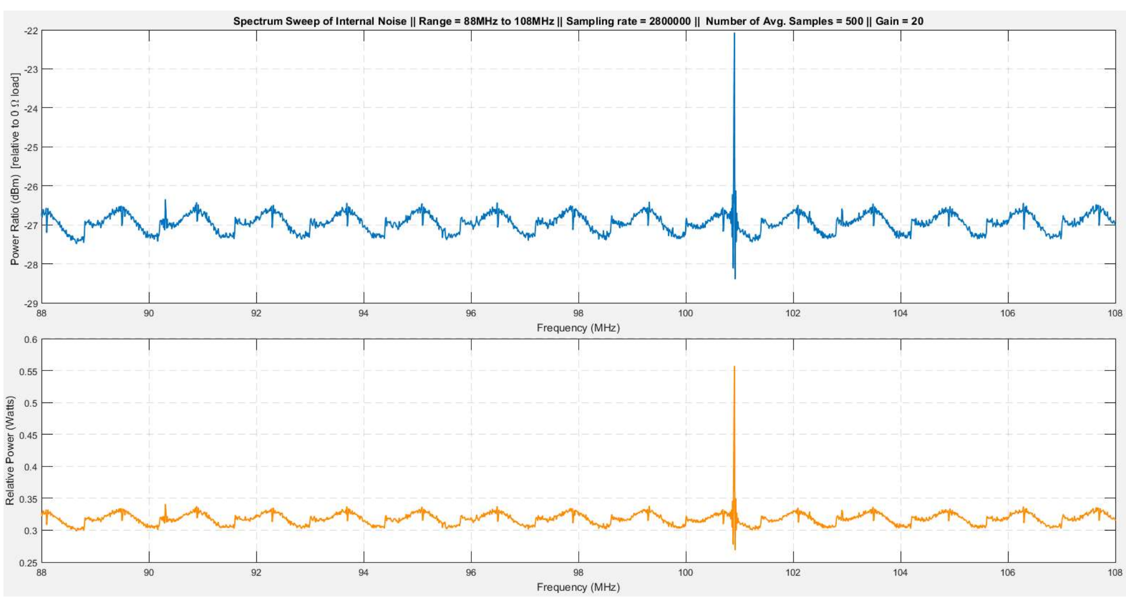


Figure 3.15 RTL-SDR internal noise spectrum output plot

3.4.4.3 External (Environmental) noise

The unwanted external environmental noise can come from anything from the controlling computer, to switch mode power supplies, the public transports electronics to any other source of noise or unwanted signal, etc.

A model is designed to determine the external noise and save this as unwanted signal data for later use (see FIGURE 3.16).

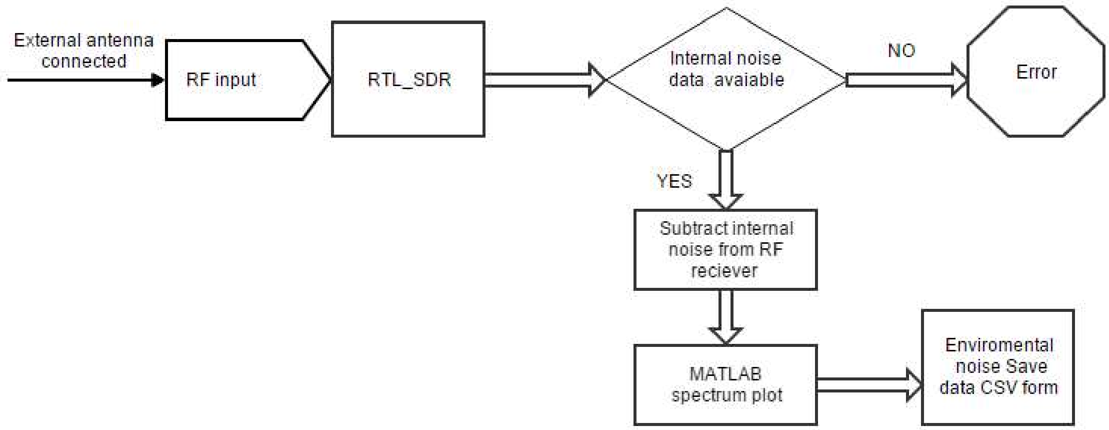


FIGURE 3.16 CONCEPTUAL MODEL SETUP FOR EXTERNAL NOISE

To test the environmental noise of the RTL-SDR the dongle is placed into the transport vehicle. Emphasis must be taken that the transport vehicle must be switched on so that the vehicle noise generated by the electronics and engine can be measured. The RTL-SDR is thus directly plugged into the USB port of the laptop. The RTL-SDR is connected to an external Omni antenna inside the vehicle (see FIGURE 3.17). Preferably the antenna not connected must not have a cable run as this allows noise picked up by the feeder/cable run.

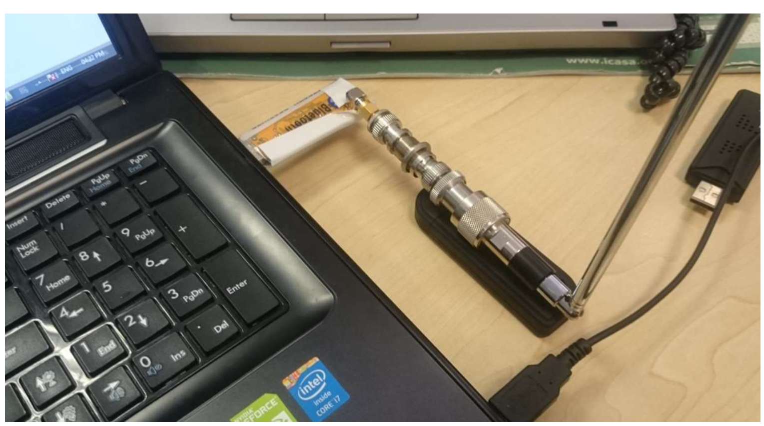


FIGURE 3.17 SETUP TO TEST EXTERNAL (TRANSPORTATION) NOISE

The following script must be opened on MATLAB. rtl_specsweep_test_enviroment.m
This script will use the previous CSV file created by the internal noise test. With a successful run, MATLAB provides the spectrum sweep between the selected frequencies and create a CSV file containing all the outputs of the sweep (see Figure 3.18).

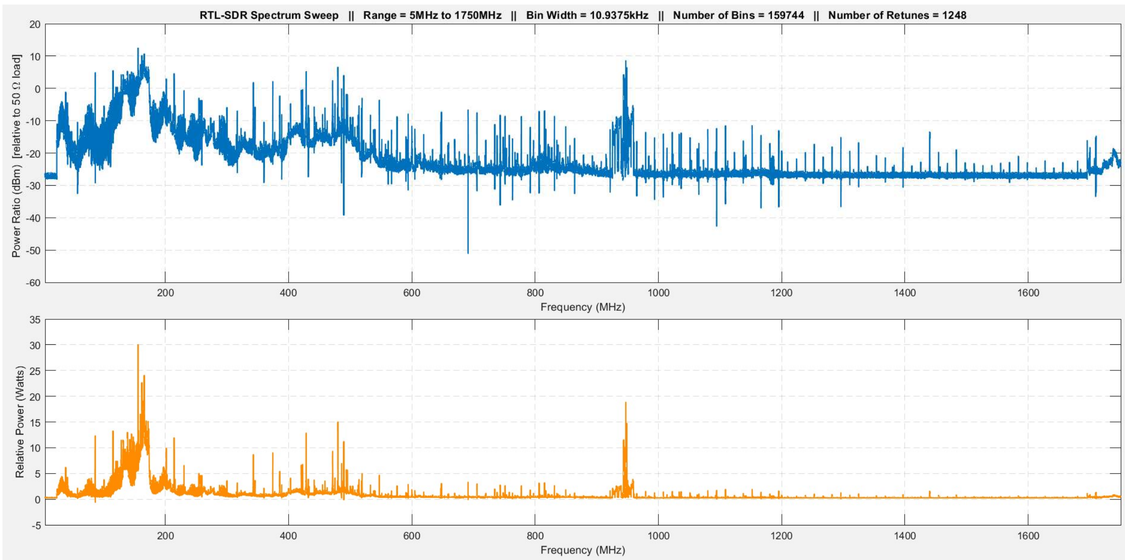
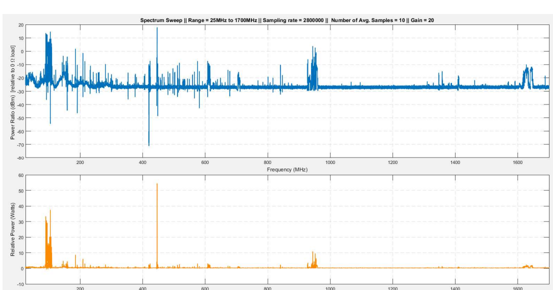


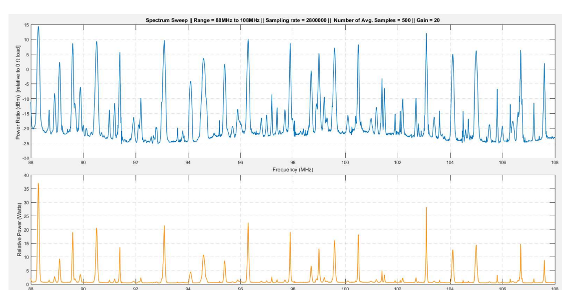
Figure 3.18 RTL-SDR Internal noise and vehicle noise spectrum output plot

3.4.5 Test Conducted in other Geographical Location

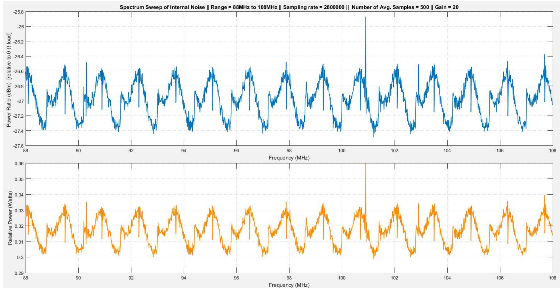
An experiment was arranged where the RTL-SDR was set up and used in a different environment. In this environment, there is more radio frequency interference, and this is the result of the test (see Figure 3.). Therefore, it must be taken into consideration where geography and in which noise floor environment the initial calibration of the noise floor is conducted as this can influence the test result.



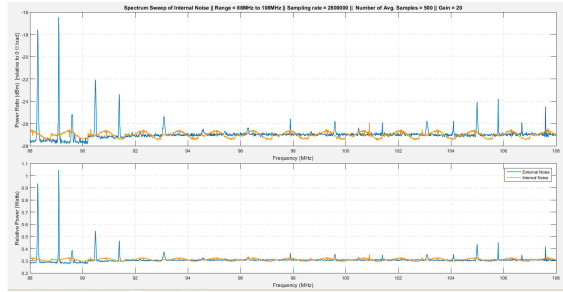
A)



B)



C)



D)

FIGURE 3.19 HIGH NOISE ENVIRONMENT A) RTL-SDR SPECTRUM PLOT FROM 25MHZ TO 1.7 GHZ, B) RTL-SDR SPECTRUM PLOT FROM 88MHZ TO 108MHZ, C) RTL-SDR INTERNAL NOISE SPECTRUM PLOT & D) RTL-SDR EXTERNAL NOISE AND INTERNAL NOISE SPECTRUM PLOT

3.4.6 Conclusion

The objective was reached as the inherent noise floor and the noise floor of the vehicle could be measured using Matlab. The inherent noise did create false reading and is greatly affected by the amount of internal gain that is used in the RTL-SDR.

The installation vehicle also created a noise floor that changed per vehicle type and was also dependent on where in the vehicle the RF MMN monitoring system was installed. The power source that is used to power the RF MMN monitoring system also created its noise as most are based on switch mode power supplies.

Much needs to be done to isolate the receiver antenna and RTL-SDR from electronics in proximity. These electronics can greatly influence measurement quality, accuracy, and reliability.

These noise floor figures can be used to increase the accuracy in measurement by creating a 10dBm guard band between the measured noise floor and actual recordings. This will significantly decrease the probability of false readings. Also, this noise floor reading can be used to decrease data storage as only reading above the noise floor will be recorded.

3.5 RF MMN Monitoring System Selective Sampling

3.5.1 Background

When the RF MMN monitoring system is active, the system will be receiving frequency, signal strength, time, and GPS coordinate data. For the most accurate results, a signal strength measurement is continuously taken for each sample for all possible locations, time, and frequency but there are practical limitations to this. To be more practical, a sample is taken as a reflection of the actual data. The sampling requirement is dependent on the objective of the measurement and is limited to maintaining a manageable amount of recorded data. The sampling needs to be taken at a known distance between measurements to be able to sufficiently approximate the relationship between distance and variance.

One of the prerequisites in the design of the RF MMN monitoring system is to keep the cost as low as possible and taking into consideration the cost of data storage, sampling needs to be significantly limited to give a good reflection at an affordable price. The plan is to install an RF MMN monitoring system into a transport vehicle. This creates extra challenges when it comes to sampling. As the transport vehicle drives at varying speeds and stops frequently, this can lead to the RF MMN monitoring station collecting data heterogeneously in density (see Figure 3.19). Therefore, to bypass this issue and save data storage costs, it was determined the best approach would be to use a spatial method of sampling.

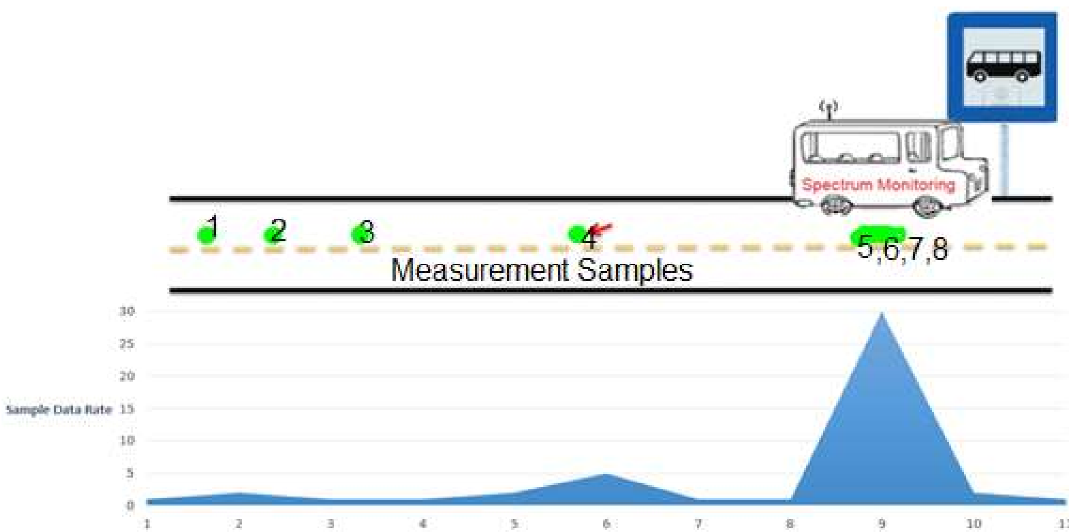


FIGURE 3.19 REPRESENTATION OF VEHICLE COLLECTING SAMPLES HETEROGENEOUSLY IN DENSITY

In Spatial domain sampling measurement, the location of the transmitter and coverage is not known, this makes it perfect in monitoring MMN. By using the Spatial sampling method, a pre-determined physical distance between samples recorded can be maintained making it much easier to represent the recorded occupancy data on an RF occupancy coverage map. By using this spatial method, no unnecessary data will be recorded on a specific location and only vital information that the operator needs can be recorded.

The spatial-domain resolution is determined in this section to adequately record data not to lose essential information by keeping the data amount and scanning speed to a reasonable level.

3.5.2 Travel Distance Calculations

In the RF MMN monitoring systems, the GPS receiver will be providing the GPS coordinates. The coordinates will be used to calculate the distance travelled between points. Some of the common methods for calculating the distance between two coordinates on the earth's surface is the Haversine formula and Euclidean distance.

Haversine (Basyir et al., 2018) as a formula to calculate the distance of a circle between longitude and latitude points assuming the radius of the earth R is 6367.45 km, the location of the two points in a spherical coordinate.

Ignoring the structure of the earth's surface (hills and valleys) is very accurate for most calculations since the ellipsoidal effect is eliminated.

$$D = 2r\ arc\left(sin\ \sqrt{sin^2\left(\frac{\theta_2 - \theta_1}{2}\right)}\right) + cos(\theta_1) \cdot cos(\theta_1) \cdot sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right) \qquad \text{Eq 3-2}$$

Where D is the distance (km), radius r = 6367.45 km, λ = longitude and θ = latitude (Basyir et al., 2018).

Euclidean method (Pramudita & Hariadi, 2019) is where the distance is calculated from two points is based on the Pythagorean Theorem. The Euclidean distance is an empirical function based on the direct distance between two free space obstacles. The value of the length of the diagonal line on the triangle.

$$D = \sqrt{((\theta_1 - \theta_2)^2 + (\lambda_1 - \lambda_2)^2)} \qquad \text{Eq 3-3}$$

Where refers D = Euclidean distance in degrees, θ= latitude and λ = longitude of the locations.

The distance D in degrees is converted into kilometres by multiplying D by 111 319 Km (1 degree).

Results of distance measurements between the Euclidean method and the Haversine Formula has a percentage ratio of the distance between the two methods of 99.89% (Maria et al., 2020).

Based on the information above, either of the techniques can be used to determine the distance. Thus, for simplicity, we will be using the Euclidean method as this is fit for our purpose.

3.5.3 Spatial Sampling Speed Factors

As there is no way to control the speed at which public transport is travelling. The only certainty is that the vehicle will come to a standstill and may travel at a maximum legal speed limit of 120 km/h on public roads within the Republic of South Africa (Transport, 2019).

$$\frac{km}{hr} = \frac{m}{km} \cdot \frac{hr}{s} \qquad \text{Eq 3-4}$$

Based on a formula to convert from kilometres per hour to meters per second, public transport would be travelling at a maximum speed of 33.333 m/s. The Gmouse VK-162 GPS receiver that is in use In this thesis has a maximum data refresh rate of 10 Hz (Various, 2020c).

Thus, the maximum theoretical distance between each GPS point would be

$$Maximum\ Legal\ speed\ Limit\ (m/s)\ /\ Maximum\ GPS\ refreshrate(Hz) = Distance(m) \qquad \text{Eq 3-5}$$

$$= \frac{33.333m}{s} * \frac{1s}{10} = 3.33\ m$$

The Calculated distance travelled based on the refresh rate of 10 Hz.

Therefore, theoretically, when the vehicle is travelling at 120 KM/h, a GPS coordinate can be placed every 3.33 meters. Unfortunately, other factors can influence this theoretical rate thus we compensate in setting the recording coordinate pin every **5 meters**. This will also be in line with our GPS as it has an accuracy of between 3 and 5 meters.

The public transport can travel at a **maximum speed of 180 KM/h** and our system will still be to put down pins accurately.

3.5.4 Reduced Sampling Rate based on Distance Travelled Calculations

To reduce the sampling rate, we will only be saving data after every 5 meters as was calculated in section 3.4.3 and to calculate the distance between the coordinates, we will be using the Euclidean method as discussed in section 3.4.4

The following equation is created based on this information.

$$Distance\ (\mathrm{D})(m) \geq \frac{\sqrt{((\theta_1 - \theta_2)^2 + \ (\lambda_1 - \lambda_2)^2)}}{111,\ 319} \qquad \text{Eq 3-6}$$

Where Distance (m) = 5 meters (selected accuracy level), coordinate 1 = (Latitude, longitude) = (λ_1, θ_1) and Coordinate 2 = (Latitude, longitude) = (λ_2, θ_2) , The distance D in degrees is converted into kilometres by multiplying D by 111 319 Km (1 degree).

Only where this formula is “false is” the coordinate is taken with its monitoring frequencies and averages signal level per frequency.

The software flow for spatial sampling (see Figure 3.20):

- First, calculate the distance between coordinate Y and the following coordinate X
- If this distance is less than D, we discard coordinate X sample information only keeping the signal levels as per frequency bin.
- Redo the calculation by calculating the distance between coordinate Y and the following coordinate X2 and if this is bigger than D, save the coordinate X2 and its corresponding signal levels are averaged per frequency bin. Coordinate X2 sample time will also be recorded with this recording.
- Now coordinate X2 is the new coordinate Y and the process restarts.

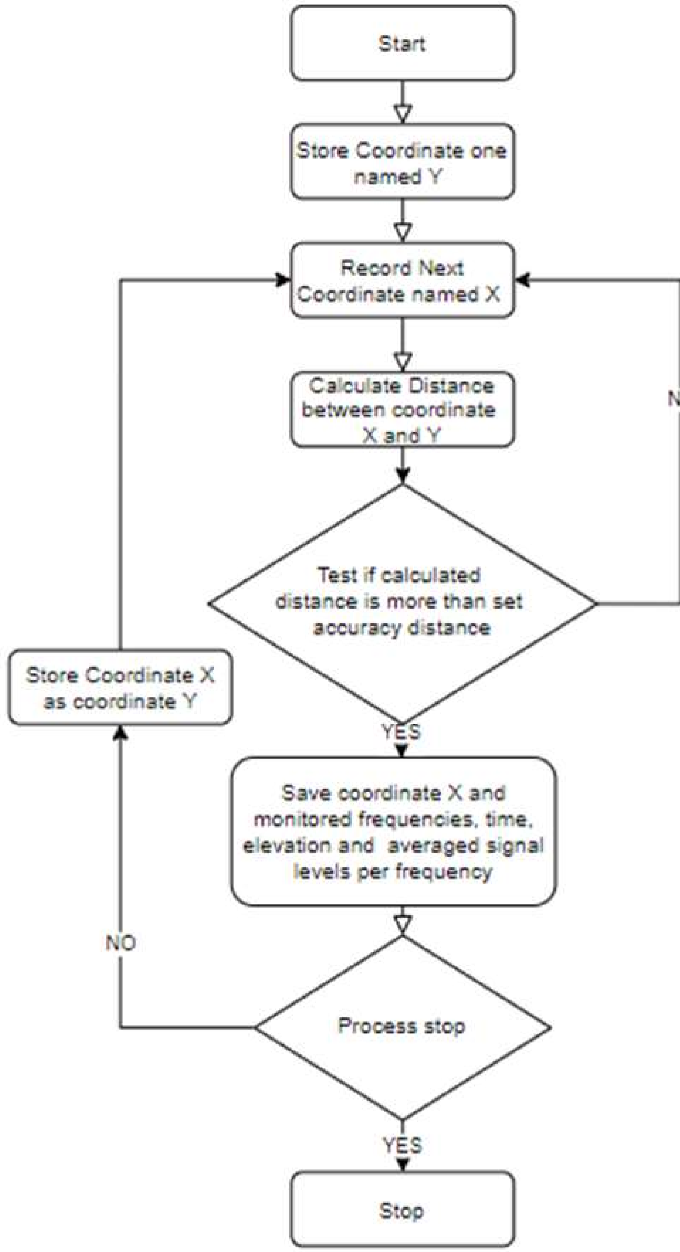


FIGURE 3.20 MODEL OF SOFTWARE FOR SPATIAL SAMPLING

3.5.5 Conclusion

As data storage can significantly increase the price of the RF MMN monitoring system using selective sampling can assist in keeping storage needs to a minimum. As public transport travels at various speeds and often stops this can cause unnecessary data to be collected hence if spatial sampling is used this problem can be avoided. By calculating the distance between coordinates samples can be taken as set distances. For our purpose, it was determined that the Euclidean method will serve our purpose and needs the least amount of processing power.

Chapter 4 : RF MMN Measurement Pilot Data Collection

4.1 MMN Monitoring Station Pilot Test Scenario

The Pilot test was conducted on an FM radio station in Malmesbury South Africa on the 03rd ^{of} December 2020. To measure the RF MMN coverage of the FM radio station in the HF part of the FM band, a pilot drive was conducted in various directions till the received signal has become impossible to differentiate from the noise floor. The pilot measurement FM Radio station has a legal ICASA Broadcasting Licence which was obtained and the following information was referenced from the licence Table 4.1, predicted coverage map and image of transmitter antenna Figure 4.1

TABLE 4.1 RADIO STATION IN MALMESBURY SPECIFICATION

Specification	Parameters
Radio Station Frequency	100.1 MHz
Regional Location	Malmesbury, South Africa
Coordinates	18E42 02 / 33S27 55
Site Height above sea level	125 meters
Antenna Height	10 meters
Power (ERP)	100 Watts
Antenna	Dipole
Antenna Horizontal Beam	Beamwidth=180; Azimuth =310
Antenna Polarisation	Vertical

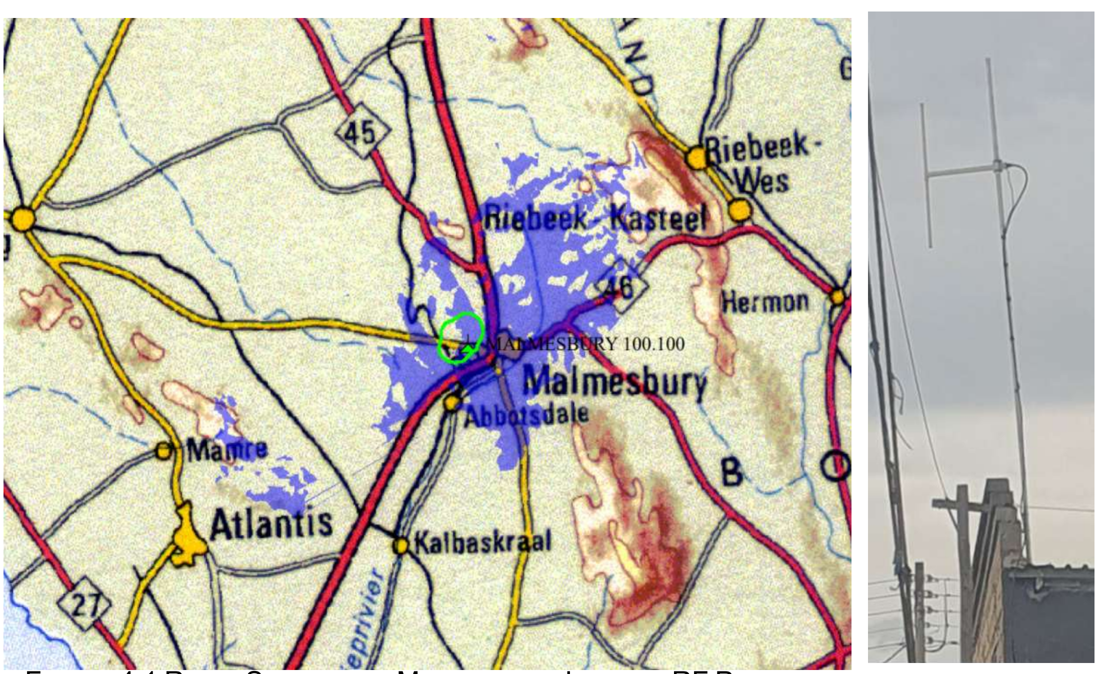


FIGURE 4.1 RADIO STATIONS IN MALMESBURY LICENCE RF PLOT OF THE RADIO STATION IN MALMESBURY AND RADIO STATION DIPOLE ANTENNA

4.2 Pilot Test RF MMN Monitoring Station Setup

The pilot test for the RF MMN monitoring system is divided into 3 main areas: The physical hardware needed and interface, the data sampling technique used and the output of received data see Figure 4.2

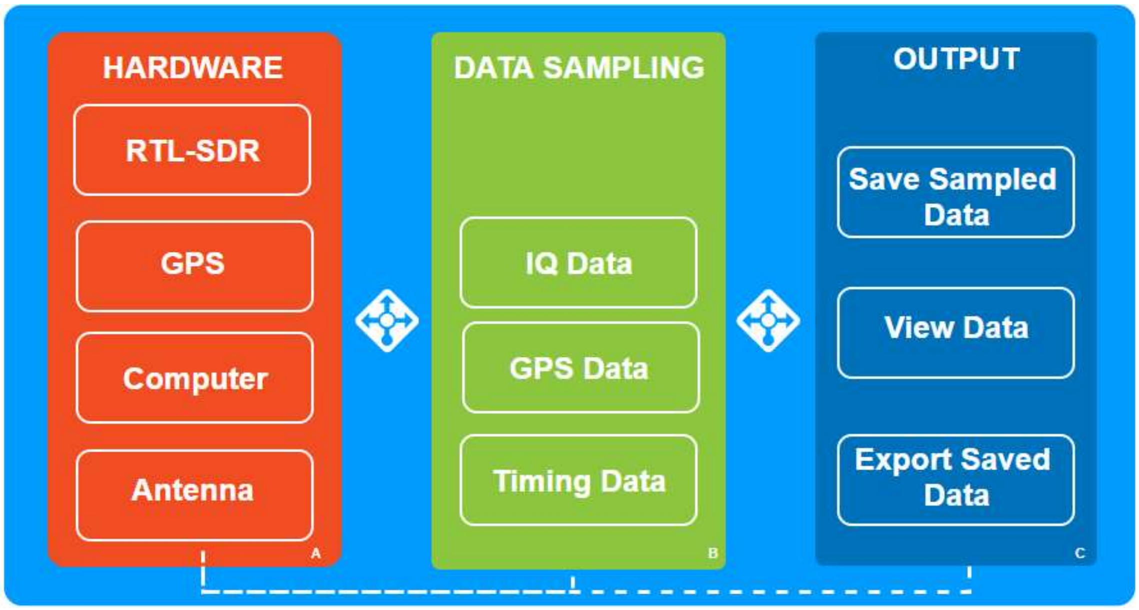


FIGURE 4.2 BLOCK DIAGRAM OF MMN MONITORING SYSTEM

4.2.1 Hardware

The Pilot test drive was conducted by making use of a 2018 Hyundai Elantra vehicle. The RF MMN monitoring system was installed in and on the vehicle as seen in Figure 4.3. The Matlab software was installed on HP Zbook 15 G3 laptop. All the MATLAB processing and data recording was conducted while the laptop operating from its internal battery, taking away the need for an external power source. By not using an external power source helped to reduce the noise floor experience within the vehicle as very noisy switch mode power supplies are used in laptop chargers. Both the GPS and RTL-SDR are plugged into the USB ports of the laptop. This USB 2.0 port serves as a power supply to the external devices and as a data transfer connection. The GPS coordinate was collected by making use of VK-162 G-Mouse USB GPS Dongle Navigation Module as discussed in chapter 3.3.3.6 and the RF signal receiver antenna that was used is the Mag Base whip antenna discussed in chapter 3.3.3.1. An additional ferrite choke was added to the coaxial of the RF receiver antenna and the coaxial length was kept to 2m, this will assist in stopping unwanted frequencies being induced into the coaxial cable. The RTL-SDR that was used in these measurements is RTL-SDR.com v.3 SDR as discussed in chapter 3.3.3.4. The was no external LNA or filter used in this Pilot test.

a) b)

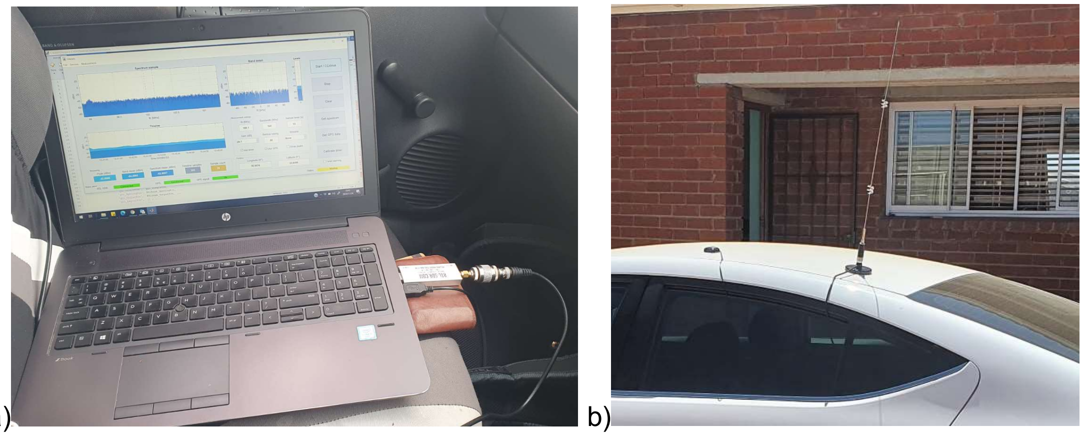


FIGURE 4.3 A) LAPTOP RUNNING MAPPING SOFTWARE & B) GPS AND RF SIGNAL RECEIVER ANTENNA ON THE ROOF OF THE VEHICLE

4.2.2 Software

4.2.2.1 Software Interface

The MMN measurement data are processed on the laptop by making use of MATLAB and Simulink. The Matlab software interface of the RF MMN monitoring system can be seen in Figure 4.4 (Valenta, 2017). The user can connect and confirm external equipment connection by using the interface. The following parameter is selected by making use of the interface e.g., Frequency of interest (f0), bandwidth, spatial sampling distance, the gain of internal LNA, etc. The user interface displays signal levels, noise floor, timestamp and GPS coordinates that are viewed in real-time by the user while conducting the measurement test. The MATLAB code is available in Appendix A-2 Frequency Mapping tool.

FIGURE 4.4 RF MMN MONITORING SYSTEM SOFTWARE INTERFACE

4.2.2.2 Software Operation

The received signal strength is converted into IQ data as explained in chapter 2.6.2. This IQ data goes through Fast Fourier transforms, gets normalised and is finally converted to a decibel value (dBm) (e.g., -27.735385). The GPS receiver communicates according to the NMEA 0183 standard discussed in chapter 2.7 to MATLAB via USB 2.0 port. The received GPS data is converted to decimal form (e.g., -33.0339472 N, 18.7675183 E) to make it easier to plot within MATLAB. The received GPS coordinates are used to calculate the spatial sampling distance between points to determine which when samples will be recorded. The laptops date and time are used as the sampling time and is saved in decimal form (e.g., 738128.806605 corresponds to the date December 3, 2020, 15:21:42). The received signal strength decimal data, the sampling time and GPS longitude and latitude decimal data are combined and stored in a matrix that is a representation of one measurement. The frequency of the measurement sampling is determined by the distance the vehicle has travelled compared to the required measurement spatial sample distance. The measurement data is exported into .csv file format see an example of .csv file format Table 4.1. A row corresponds to one measurements sample.

TABLE 4.1 EXAMPLE MMN MONITORING STATION RECORDING OUTPUT IN CSV FORMAT

Latitude	Longitude	Power level	Date and Time
18.465623	-33.412550	-29.205972	738128.773250
18.466909	-33.413254	-27.735385	738128.773482
18.467945	-33.413945	-28.187192	738128.773018

The output CSV file of the measurements are imported to create an interference map graphical output (e.g. signal strength scatter map and interpolation map)(Höyhtyä et al., 2016). This interference map defines RF power distribution in the space, time and frequency domains (Kim et al., 2011). The measurement results are placed on a map image from www.openstreetmap.org. The signal strengths scatter output maps the colour represents the signal strength at a given location. For the signal strength interpolation map Natural Neighbour Interpolation (Kotulak et al., 2017) is used. Interpolation is a very computationally demanding task. It should be noted that although the interpolation map is visually attractive this is mathematical calculations of the values of the signal strength power at places where it was not measured and is not a clear indication of coverage.

4.2.2.3 Software Input Parameters

The RF MMN measurements were conducted using the following specification. As mentioned in section 3.4.5 a special sampling rate/distance of 5 meters can be set. For this test, we have selected a spatial sampling rate of 250 meters because the frequency that was tested is in the FM band where high power is normally used. With the selected sampling rate, we will get a fair coverage plot. Secondly and the most important is the lack of processing power when it comes to post-processing to plot pins on the map. It was found that one of the challenges is that the higher amount of coordinate pins the more processing power and time is needed to process the maps. Selected parameters are set out in Table 4.2.

TABLE 4.2 SELECTED RF MMN MONITORING SPECIFICATIONS PARAMETER VALUES

Specification	Parameter Value
Frequency	100.1 MHz
Measured Bandwidth	150 kHz
Internal amplifiers gain of RTL-SDR	29.7 dBm
Spatial sampling distance	250 meters

4.3 Pilot Test Results

4.3.1 Measurement of the Noise Floor

The Matlab noise floor test was conducted before conducting the pilot drive. In the noise floor test, the inherent noise and the vehicle electronics noise was recorded. In this test, the vehicle ignition was switched on, with the air conditioner, lights, and radio “ON” to make sure all RF noise that may occur within the vehicle is recorded. On the frequency, 100.1 MHz a noise floor measurement of -26.6 dBm (0.33 watts) was recorded see Figure 4.5.

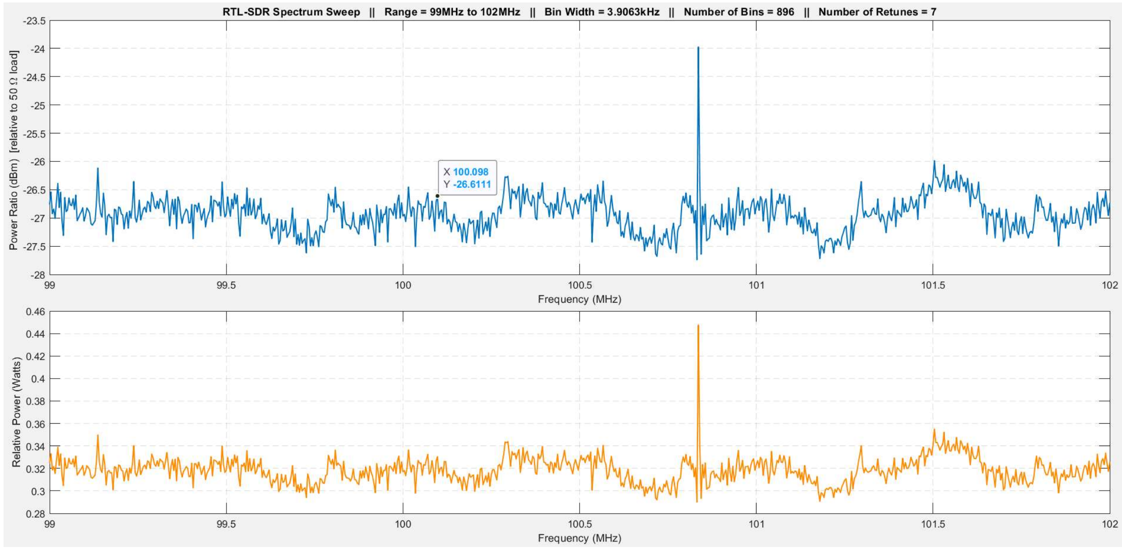


Figure 4.5 Spectrum RF Noise Floor Recording 99 MHz to 102 MHz within the Test Vehicle

4.3.2 Pilot Test Statistics

Table 4.3 represents the recorded statistics surrounding the pilot test.

Notice that the total amount of recorded data is very low bearing in mind that this is only one frequency that was monitored, and this will increase the more frequencies are simultaneously monitored. The recorded data will also increase when the spatial sampling distance is decreased.

TABLE 4.3 STATISTICS OF RF MMN MONITORING STATION PILOT DRIVE TEST OF 100.1 FM RADIO STATION IN MALMESBURY

Specification	Parameter
Duration of measurement	3 hour and 20 min
Coverage Area	1 100 Km^2
Distance covered	261 Km
Recorded Data size	61 kbits
Number of samples recorded	1044

4.3.3 Scatter and Interpolation map results

Figure 4.6 is a scatter plot of recorded measurements of the FM radio station transmitting at 100.1 MHz in Malmesbury. To view the recorded data the MATLAB code is available in APPENDIX A-4 Data Viewer.

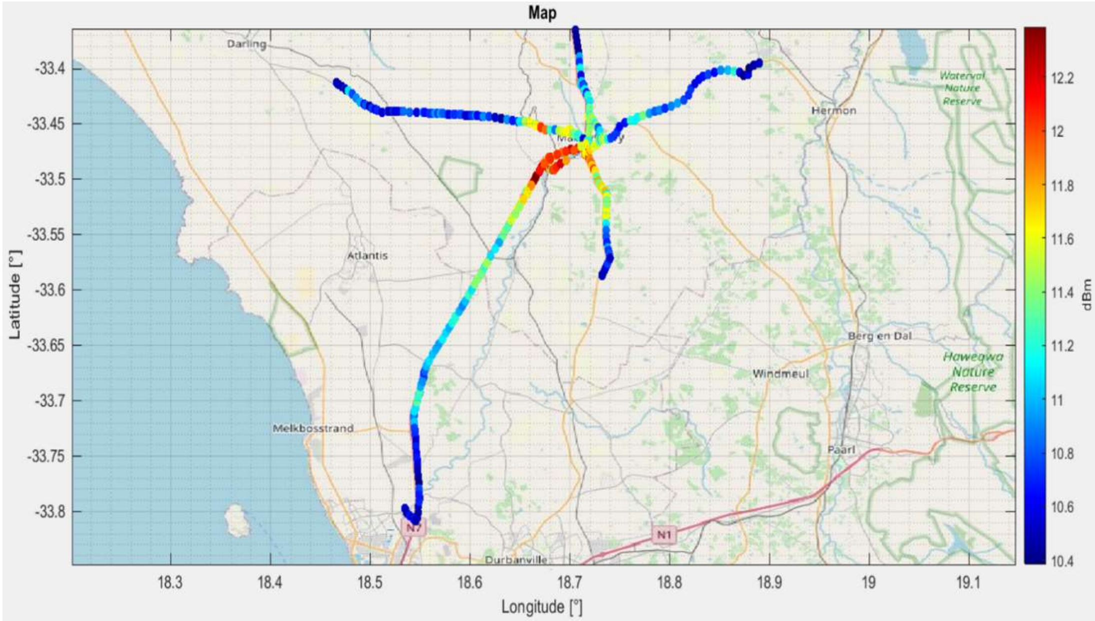


FIGURE 4.6 FM RADIO STATION IN MALMESBURY SIGNAL COVERAGE MAP - SCATTER MAP

The signal strength map was generated by making use of collected measurement samples and using natural neighbour interpolation to calculate a predicted coverage map of the FM Radio station in Malmesbury see Figure 4.7. This represents the calculated signal levels at a specific point that was not measured.

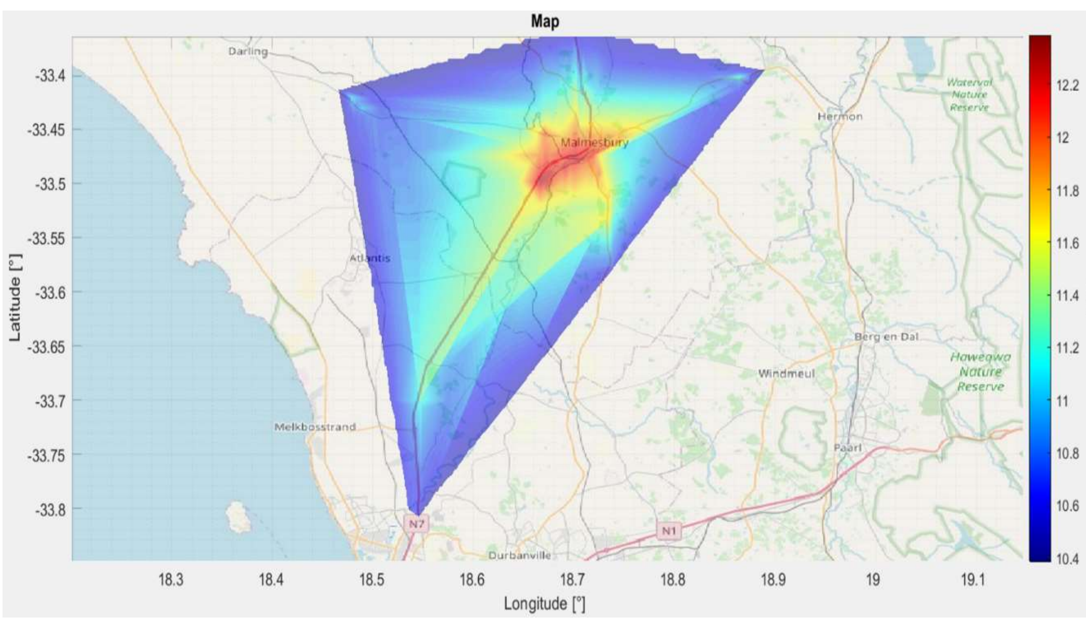


FIGURE 4.7 FM RADIO STATION IN MALMESBURY SIGNAL COVERAGE MAP - PREDICTED SIGNAL STRENGTH MAP USING NATURAL NEIGHBOUR INTERPOLATION

4.3.4 Pilot test with Noise Floor

In the initial noise floor measurement that was conducted measuring the RF noise generated within the vehicle was measured at -25.1 dBm. Recordings were collected within and below this noise floor. To increase the measurement accuracy only the measurements that are 5dBm above the equipment noise floor will be plotted, this will significantly reduce the odds of getting phantom occupancy measurements. The new measurement threshold is calculated at -20.1 dBm. By only considering measurements above this threshold noise there was a 31% reduction in data storage needs and the signal coverage plot is very similar to previous results see Figure 4.8 and Figure 4.9

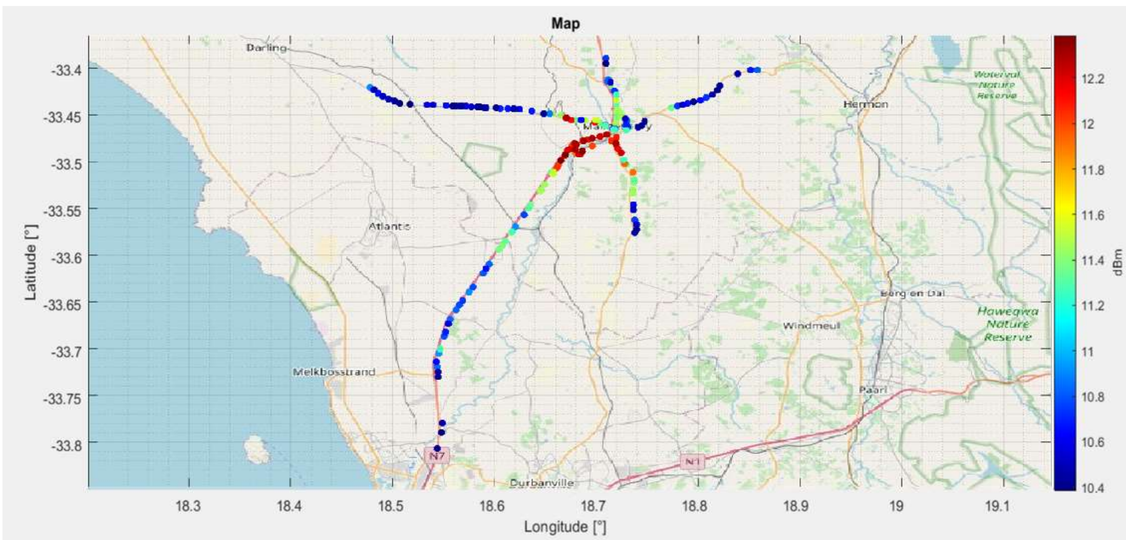


FIGURE 4.8 FM RADIO STATION IN MALMESBURY SIGNAL COVERAGE MAP – SCATTER MAP OF SIGNAL ABOVE NOISE FLOOR

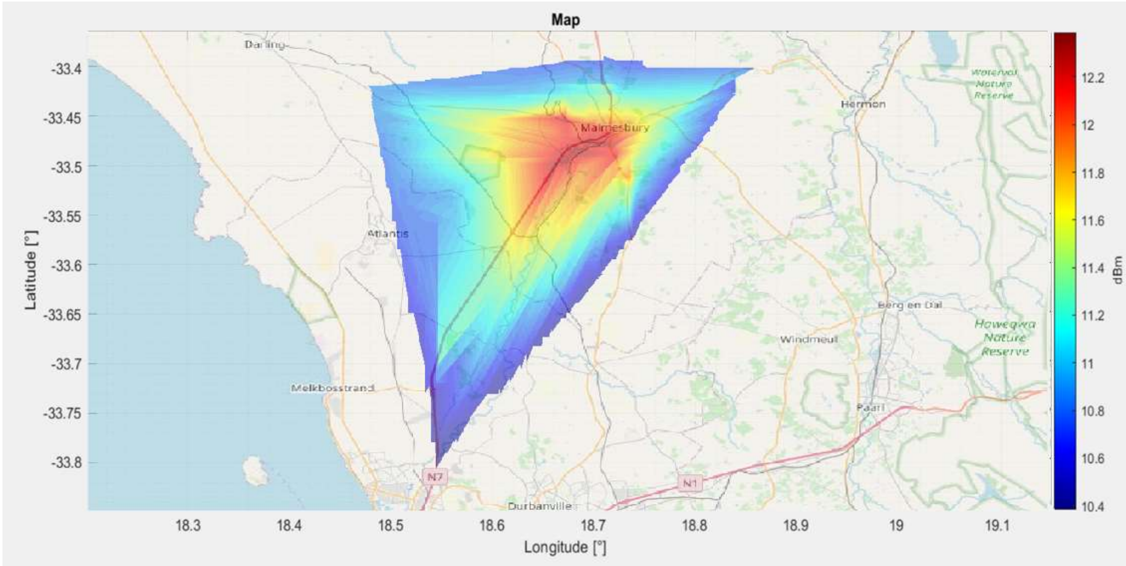


FIGURE 4.9 FM RADIO STATION IN MALMESBURY SIGNAL COVERAGE MAP - SIGNAL STRENGTH MAP USING NATURAL NEIGHBOUR INTERPOLATION ABOVE NOISE FLOOR

4.3.5 Licenced coverage Predictions vs Measured Coverage

Comparing predicted coverage vs measured coverage, proved the prediction model needs to be calibrated. Signal coverage mapping is shown to give more precise information than what was estimated with the path loss model within the licence. There is 35% more coverage for the measured result compared to the plotted results. This could create future issues in frequency allocation and may cause unwanted interference between FM radio stations.

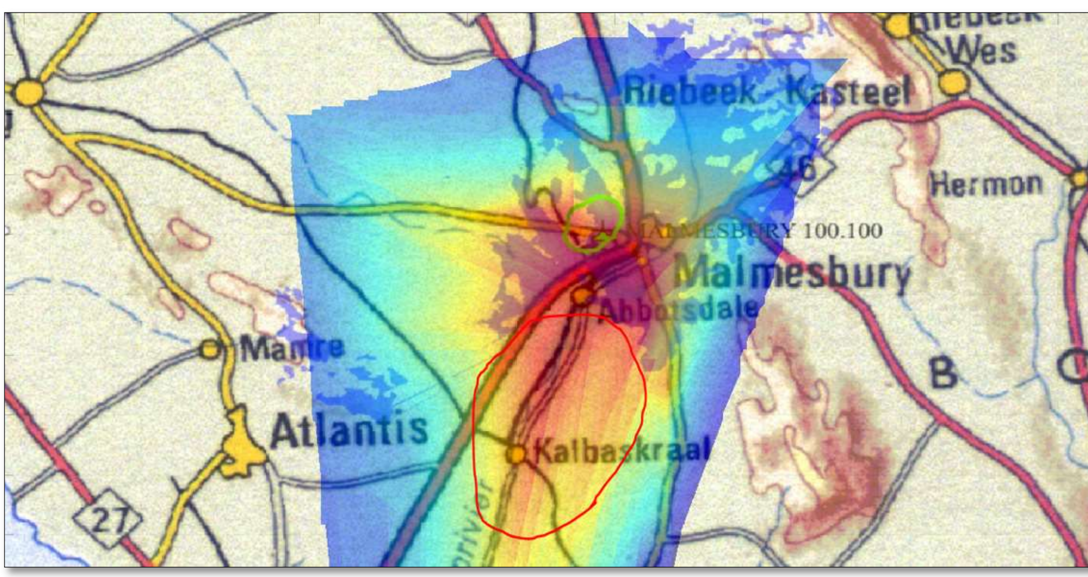


FIGURE 4.10 COMPARING PREDICTED COVERAGE VS MEASURED COVERAGE

4.4 Conclusion

The chapter aimed to conduct a pilot test of the RF MMN monitoring system on an FM radio station. This pilot test was completed using the hardware and software described in previous chapters.

The test results were recorded in CSV format. Scatter and interpolation signal strength maps were generated from these recordings. The noise floor measurements were taken into consideration and more accurate RF plots were generated. It was proven that by removing the noise floor measurement there could be a further decrease in data storage needs while providing accurate plots. The spatial sampling method worked well to reduce the data storage requirements compared to the other sampling methods. By this pilot test was proved that the current plotting software that is being used needs to be calibrated as the actual real-world signal transmitted about 35% beyond than were initially plotted. This could pose future problems in new frequency allocation and may cause unwanted interference between radio stations.

RF MMN measurement system provided enough evidence to motivate the need for mobile monitoring stations and to assist the spectrum regulatory body.

Chapter 5 CONCLUSIONS AND FUTURE RESEARCH

5.1 Conclusion

This paper presented an overview and challenges of creating a Cost-effective design for the measurement of man-made noise in the HF band receiver with emphasis on implementation in public transport.

In obtaining the four key capabilities such as being cost-effective, portable, using spatial sampling and being able to adapt to installation noise floor levels, the RF MMN monitoring system has the potential of being a disruptive force within spectrum monitoring and management. Using the RTL-SDR for the implementation of RF MMN monitoring provides a cost-effective solution and offers improved flexibility to overall spectrum monitoring. RF MMN monitoring system can exploit the spectrum white spaces or holes of licensed spectrum bands provided and assist in finding non-compliant licence holders and significantly improve the efficiency of spectrum usage. RF MMN monitoring system is the active “eyes and ears” of the regulatory body and as RF spectrum is a limited resource therefore it needs to be actively monitored.

To summarize what was presented In this thesis:

i) Supporting written literature was reviewed to define, support, and oppose some of the aims that were set out In this thesis. Much of the technical groundwork and information was covered giving the reader an insight into the RF MMN monitoring regulatory, its parameters, phases, cost, and existing commercial monitoring systems. Some of the other vital topics that were presented were RF noise sources, an introduction to RTL-SDR, why RTL-SDR was the perfect cost-effective solution and the pros and cons of using RTL-SDR.

ii) The design phase of defining the specification and finding commercial products/hardware to fit our goal of building a cost solution for RF MMN monitoring. This was accomplished at a very affordable price of under R2000 for the basic system.

(iii) The RF noise was successfully simulated in the MATLAB environment. The RF is generated by the MMN monitoring system internal electronics and the installed vehicle electronics. Our measurements/simulation revealed that the MMN monitoring system generated many ghost signals and the installation vehicle needs to be thoroughly checked as some vehicles generate more RF noise than others and without filtering and proper shielding this greatly affected the reliability, repeatability, and sensitivity of our measurements.

(iv) Spatial sampling was proven the best sampling method to reduce data storage significantly saving between 20% and 60% of data needs depending on the driving location of the vehicle.
(v) A pilot test drive on an FM radio station was conducted in Malmesbury, South Africa. MATLAB Simulink was the software environment used to conduct the pilot test. The MMN system was working, collecting, and recording RF signal strength and GPS coordinates. The information that was recorded was plotted on an interference map within MATLAB. In the findings, it was found that this monitored radio station transmitted much, much further than that was originally plotted on its broadcasting licence. The recorded coverage was about 35% more than plotted on the licence. This pilot test confirmed that spectrum use needs to be frequently tested and reinforces the need for this type of low-cost monitoring.

The challenges remain numerous, namely, dealing with the noise floor, receiver sensitivity, multipath fading, and shadowing. Limitations depend on the future standardization, usage model, and the signal receiver bandwidth. It is anticipated that improvement on the sensitivity, noise floor, and over performances can be achieved by changing antenna type, making use of better filtering, and adding an LNA. But these performances improvements will lead to additional costs. However, there are still numerous challenges to overcome before the planned approach can be efficiently used in practice, including the optimum use of data analysis, multi-dimensional sampling and management techniques, visualization, and accuracy to efficiency trade-off.

Our study supports the findings that were published in V-scope: An opportunistic wardriving approach to augmenting tv whitespace databases (Zhang et al., 2014) as there is a much-waisted spectrum. The RF MMN monitoring system also forms part of the cognitive radio function in that it confirms that additional efficient use of the RF spectrum is needed (Arjoune & Kaabouch, 2019). The RF MMN monitoring system supports the ITU mandate and assists the regulatory body to actively regulate and monitor spectrum use(ITU-R, 2015a).

In this thesis, all the objectives were achieved in design- and building a mobile deployed RF MMN monitoring system that was cost-effective, is relatively small and can be used in any vehicle. Successfully simulated and measured the RF noise generated by the RF MMN monitoring system and the installation vehicles. We mitigated data storage overhead by making the use of spatial sampling and completed the pilot test and analysed the data collected. All findings In this thesis had sufficient literature to support our findings.
The measurement system will contribute to enhancing traditional Man-Made noise databases with the sensors moving around within public vehicles. This will assist in

identifying the primary, secondary, and illegal users in the band in which the samples were measured, spectrum database accuracy confirmation and will assist in refining current propagation models.

5.2 Future Research Work

The pilot test confirmed the RF MMN monitoring system concept does work and can be effective in monitoring spectrum use, but more standardisation research is needed.

- As this paper only focussed on the ITU guideline described to South African and they are only focused on the monitoring of the FM radio band future expansion is needed to other frequency bands and regulatory zones.
- The accuracy and integration of this RF MMN monitoring system compared to other commercial systems need to be conducted.
- Proposed improvements to be established is the:
  a) Synchronization of the sample rate is a time to effectively integrate the system to results received by existing monitoring equipment (Hrach, 2019).
  b) To increase the instantaneous RF measurement bandwidth and monitor higher frequencies by making use of other commercial SDR in the market like Lime SDR (Various, 2020b).
  c) Increase the ability to monitor multiple frequencies outside of the band by using multiple SDR dongles.
  d) Improve the noise floor by making use of active noise cancelling to illuminate the effects of the transport vehicle (Various, 2018).

BIBLIOGRAPHY

Aguilar-Gonzalez, R., Prieto-Guerrero, A., Ramos, V., Santos-Luna, E. & Lopez-Benitez, M. 2020. A comparative study of RTL-SDR dongles from the perspective of the final consumer. *Digest of Technical Papers - IEEE International Conference on Consumer Electronics*, 2020-Janua(January).

Arjoune, Y. & Kaabouch, N. 2019. A comprehensive survey on spectrum sensing in cognitive radio networks: Recent advances, new challenges, and future research directions. *Sensors (Switzerland)*, 19(1).

BA van Niekerk, TE Kahn, V.Balyan. 2020. Development of a Down-Converter for a Software-Defined Radio Environment. : 255–264.

Basyir, M., Nasir, M., Suryati, S. & Mellyssa, W. 2018. Determination of Nearest Emergency Service Office using Haversine Formula Based on Android Platform. *EMITTER International Journal of Engineering Technology*.

Best, R. 1988. Digital signal processing. *tm - Technisches Messen*, 55(JG): 302–307.

Bj, S. 2016. *FFI-RAPPORT External man-made radio noise measurements*. FFI-rapport.

Bob Stewart, Kenneth Barlee, D.A.L.C. 2017. *Software Defined Radio using MATLAB and Simulink and the RTL-SDR*. www.desktopSDR.com.

Bureau, R. 2002. HANDBOOK Spectrum Monitoring. *International Telecommunication Union (ITU)*: 168.

Chen, W.T., Chang, K.T. & Ko, C.P. 2016. Spectrum monitoring for wireless TV and FM broadcast using software-defined radio. *Multimedia Tools and Applications*, 75(16): 9819–9836. http://dx.doi.org/10.1007/s11042-015-2764-5.

Chen, Y. & Oh, H.S. 2016. A survey of measurement-based spectrum occupancy modeling for cognitive radios. *IEEE Communications Surveys and Tutorials*.

Collier, P. 2017. RTL2832 Software Defined Radio. *AB9IL*. https://www.ab9il.net/software-defined-radio/rtl2832-sdr.html 27 October 2020.

Dwarakanath, R.C., Naranjo, J.D. & Ravanshid, A. 2013. Modeling of interference maps for Licensed Shared Access in LTE-advanced networks supporting Carrier Aggregation. In *IFIP Wireless Days*.

Fanan, A., Riley, N., Mehdawi, M., Ammar, M. & Zolfaghari, M. 2016. Comparison of spectrum occupancy measurements using software defined radio RTL-SDR with a conventional spectrum analyzer approach. *2015 23rd Telecommunications Forum, TELFOR 2015*: 200–203.

Funck, J. & Gühmann, C. 2014. Comparison of approaches to time-synchronous sampling in wireless sensor networks. *Measurement: Journal of the International Measurement Confederation*.

Gavrilovska, L., Atanasovski, V., Rakovic, V. & Denkovski, D. 2014. Integration of

Heterogeneous Spectrum Sensing Devices Towards Accurate REM Construction.

Ghosh, C., Pagadarai, S., Agrawal, D.P. & Wyglinski, A.M. 2010. A framework for statistical wireless spectrum occupancy modeling. *IEEE Transactions on Wireless Communications*.

HA Cook, MTE Kahn, V.B. 2020. *Radio Direction-Finding Techniques for an Unmanned Aerial Vehicle*. Micro-Electronics and Telecommunication Engineering.

Hb9Ajg. 2013. Some Measurements on DVB-T Dongles with E4000 and R820T Tuners : Image Rejection , Internal Signals , Sensitivity , Overload , 1dB Compression , Intermodulation 2 . Test Setup. , (August).

Höyhtyä, M., Mämmelä, A., Eskola, M., Matinmikko, M., Kalliovaara, J., Ojaniemi, J., Suutala, J., Ekman, R., Bacchus, R. & Roberson, D. 2016. Spectrum Occupancy Measurements: A Survey and Use of Interference Maps. *IEEE Communications Surveys and Tutorials*, 18(4): 2386–2414.

Hrach, J. 2019. *Passive emitter tracking*. Charles University. https://is.cuni.cz/webapps/zzp/detail/208913/%0Ahttps://www.rtl-sdr.com/thesis-on-locating-transmitters-with-tdoa-and-rtl-sdrs/.

ICASA. 2018. Independent Communications Authority of South Africa. https://www.icasa.org.za/ 11 September 2018.

ITU-R. 2015a. ITU-R SM.1880-1 (08/2015) Spectrum occupancy measurement and evaluation. , 1880.

ITU-R. 2015b. *Radio noise P Series Radiowave propagation*. https://www.itu.int/rec/R-REC-P.372/en.

ITU-R. 2012. Spectrum occupancy measurements and evaluation. *SM Series Report ITU-R SM.2256*, 2256(July).

Kaabouch, N. & Hu, W.C. 2014. *Handbook of research on software-defined and cognitive radio technologies for dynamic spectrum management*.

Kenneth Wyatt. 2018. Identifying and Locating Radio Frequency Interference (RFI) Interference Technology. *interferencetechnology.com*. https://interferencetechnology.com/identifying-and-locating-radio-frequency-interference-rfi/ 17 November 2020.

Kim, S.J., Dall'Anese, E. & Giannakis, G.B. 2011. Cooperative spectrum sensing for cognitive radios using kriged kalman filtering. *IEEE Journal on Selected Topics in Signal Processing*.

Kotulak, K., Froń, A., Krankowski, A., Pulido, G.O. & Henrandez-Pajares, M. 2017. Sibsonian and non-Sibsonian natural neighbour interpolation of the total electron content value. *Acta Geophysica*, 65(1): 13–28.

Krupa, S. 2016. Multichannel Fading. https://www.slideshare.net/ShreeKrupa1/multichannel-fading 29 October 2020.

Laufer, C. 2014. The hobbyist’s guide to rtl-sdr: really cheap software defined radio. , 1.

Laufer, C. 2017. The hobbyist’s guide to rtl-sdr: really cheap software defined radio. , 5.

Lehtomaki, J.J., Vuohtoniemi, R. & Umebayashi, K. 2013. On the measurement of duty cycle and channel occupancy rate. *IEEE Journal on Selected Areas in Communications*.

Li, J. & Heap, A.D. 2011. A review of comparative studies of spatial interpolation methods in environmental sciences: Performance and impact factors. *Ecological Informatics*.

López-Benítez, M. & Casadevall, F. 2009. Methodological aspects of spectrum occupancy evaluation in the context of cognitive radio. In *2009 European Wireless Conference, EW 2009*.

Maria, E., Budiman, E., Haviluddin & Taruk, M. 2020. Measure distance locating nearest public facilities using Haversine and Euclidean Methods. *Journal of Physics: Conference Series*, 1450(1).

Matheson, R.J. 1988. Strategies for spectrum usage measurements.

Matinmikko, M., Mustonen, M., Höyhtyä, M., Rauma, T., Sarvanko, H. & Mämmelä, A. 2010. Distributed and directional spectrum occupancy measurements in the 2.4 GHz ISM band. In *Proceedings of the 2010 7th International Symposium on Wireless Communication Systems, ISWCS’10*.

McDonald, K.D. 2011. GPS General Introduction - Navipedia. *esa*. https://gssc.esa.int/navipedia//index.php/GPS_General_Introduction 27 October 2020.

McHenry, M.A., Tenhula, P.A., McCloskey, D., Roberson, D.A. & Hood, C.S. 2006. Chicago spectrum occupancy measurements & analysis and a long-term studies proposal. In *ACM International Conference Proceeding Series*.

Medeisis, A. & Holland, O. 2014. State-of-the-Art in Policy and Regulation of Radio Spectrum. In A. Medeisis & O. Holland, eds. *Cognitive Radio Policy and Regulation: Techno-Economic Studies to Facilitate Dynamic Spectrum Access*. Cham: Springer International Publishing: 1–48. https://doi.org/10.1007/978-3-319-04022-6_1.

Merriam Webster. 2017. Radio Frequency | Definition of Radio Frequency by Merriam-Webster. https://www.merriam-webster.com/dictionary/radio frequency 30 May 2017.

Mitola, J. 2014. *ECE 4670 Spring 2014 Lab 6 Software Defined Radio and the RTL-SDR USB Dongle*.

Mitolo, M. & Araneo, R. 2019. A Brief History of Electromagnetism [History]. *IEEE Industry Applications Magazine*, 25(2): 7–11.

Niamey. 2015. Harmful Interference and International monitoring. *RRS-15-Africa, Niamey, 20-24 April 2015*, 5(1): 1–8. https://www.itu.int/en/ITU-R/space/workshops/2016-small-sat/Documents/RR-RoP-SSS-16.pdf 17 November 2020.

Pramudita, R. & Hariadi, F.I. 2019. Development Of Techniques to Determine Object Shifts for PCB Board Assembly Automatic Optical Inspection (AOI). In *ISESD 2018 - International Symposium on Electronics and Smart Devices: Smart Devices for Big Data*

Analytic and Machine Learning.

Radiocommunication Bureau. 2011. *Manuel Spectrum Monitoring*. Edition 20. Switzerland Geneva: ITU. https://www.itu.int/pub/R-HDB-23-2011.

Saleem, Y. & Rehmani, M.H. 2014. Primary radio user activity models for cognitive radio networks: A survey. *Journal of Network and Computer Applications*.

Sharakhov, N., Marojevic, V., Romano, F., Polys, N. & Dietrich, C. 2014. Visualizing real-time radio spectrum access with CORNET3D. In *Proceedings of the 19th International ACM Conference on 3D Web Technologies, Web3D 2014*.

Skolnik, M.I. 2019. INTRODUCTION TO RADAR SYSTEMS Second Edition-uda. *The Garden City Movement Up-To-Date*.

Sm, R.I. & Spectrum, S.M.S. 2011. Mobile spectrum monitoring unit SM Series. , 2.

Sm, R.I. & Spectrum, S.M.S. 2012. Spectrum occupancy measurements and evaluation SM Series. *International Telecommunication Union*, 2256.

Spaulding, A.D. & Hagn, G.H. 1977. On the Definition and Estimation of Spectrum Occupancy. *IEEE Transactions on Electromagnetic Compatibility*.

Technology, S. 1996. *NMEA Reference Manual*. www.SiRF.com 27 October 2020.

Transport, D. of. 2019. Road Safety Facts | Western Cape Government. *Department of Transport (National) (The Government of South Africa)*. https://www.westerncape.gov.za/general-publication/road-safety-facts 28 October 2020.

Various. 2020. Laptops - The best gaming laptop provider | MSI Global. *msi.com*. https://www.msi.com/Laptop/CX70/Specification 27 October 2020.

Adam. 2015. LNA for all: LNA for all - Low Noise Amplifier for many applications from 28 MHz to 2500 MHz. *LNA4ALL*. https://lna4all.blogspot.com/2013/04/lna-for-all-low-noise-amplifier-for.html 4 November 2020.

Various. 2019. New Product in Store: RTL-SDR Blog Magnetic Whip Antenna Set (Great for KerberosSDR Direction Finding). *RTL-SDR.COM*. https://www.rtl-sdr.com/new-product-in-store-rtl-sdr-blog-magnetic-whip-antenna-set-great-for-kerberossdr-direction-finding/ 27 October 2020.

Valenta, V. 2017. Mobile measuring workplace for monitoring the coverage of selected radio.

Valenta, V., Fedra, Z., Maršálek, R., Baudoin, G. & Villegas, M. 2009. Towards cognitive radio networks: Spectrum utilization measurements in suburb environment. In *RWS 2009 IEEE Radio and Wireless Symposium, Proceedings*.

Various. 2020a. Buy RTL-SDR Dongles (RTL2832U). *RTL-SDR.COM*. https://www.rtl-sdr.com/buy-rtl-sdr-dvb-t-dongles/ 27 October 2020.

Various. 2020b. LimeSDR - Lime Microsystems. *Lime Microsystems*. https://limemicro.com/products/boards/limesdr/ 1 December 2020.

Various. 2020c. Mini Gmouse VK-162 (GPS Receiver USB). *rhydolabz*.

https://www.rhydolabz.com/wireless-gps-c-130_186/mini-gmouse-vk162-gps-receiver-usb-p-2026.html 27 October 2020.

Various. 2018. Noise Cancellation using Linrad and the Phase Coherent Capabilities of an RSPduo. *RTL-SDR.COM*. https://www.rtl-sdr.com/noise-cancellation-using-linrad-and-the-phase-coherent-capabilities-of-an-rspduo/ 1 December 2020.

Various. 2020d. Radio Monitoring Direction Finding System. *LsTelcom*. http://www.google.com/patents?hl=en&lr=&vid=USPAT1988006&id=OKl5AAAAEBAJ&oi=fnd&dq=DIRECTION+FINDING+SYSTEM&printsec=abstract.

Various. 2020e. Raspberry Pi 4 Model B specifications – Raspberry Pi. *RASPBERRY PI FOUNDATION*. https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/?resellerType=home 27 October 2020.

Various. 2013. Recommendation ITU-R SM.1753-2 - Methods for measurements of radio noise.

Various. 2020f. RTL-SDR Support Package Hardware Setup - MATLAB & Simulink. *mathworks.com*. https://www.mathworks.com/help/supportpkg/rtlsdrradio/ug/support-package-hardware-setup.html?refresh=true 27 October 2020.

Verhoeven, G.J.J. & Archaeology, V. 2017. The reflection of two fields – Electromagnetic radiation and its role in (aerial) imaging. *AARGnews 55:*, (October 2017): 10–18.

Walter. 2013. *Some Measurements on DVB-T Dongles with E4000 and R820T Tuners: Image Rejection, Internal Signals, Sensitivity, Overload, 1dB Compression, Intermodulation*. http://uk.groups.yahoo.com/group/SDRSharp 1 November 2020.

Wang, Z. & Salous, S. 2011. Spectrum occupancy statistics and time series models for cognitive radio. *Journal of Signal Processing Systems*.

Yilmaz Birkan, H., Tugcu, T., Alagöz, F. & Bayhan, S. 2013. Radio environment map as enabler for practical cognitive radio networks. *IEEE Communications Magazine*.

Yücek, T. & Arslan, H. 2009. A survey of spectrum sensing algorithms for cognitive radio applications. *IEEE Communications Surveys and Tutorials*.

Zhang, T., Leng, N. & Banerjee, S. 2014. A vehicle-based measurement framework for enhancing whitespace spectrum databases. In *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*.

Zhao, Y., Le, B. & Reed, J.H. 2009. Network Support: The Radio Environment Map. In *Cognitive Radio Technology*.

APPENDIX A: MATLAB SCRIPTS

A-1 RTLSDR Info script

```
function varargout = rtlsdrinfo(varargin)

    gui_Singleton = 1;
  gui_State = struct('gui_Name', mfilename, ...
        'gui_Singleton', gui_Singleton, ...
        'gui_OpeningFcn', @rtlsdrinfo_OpeningFcn, ...
        'gui_OutputFcn', @rtlsdrinfo_OutputFcn, ...
        'gui_LayoutFcn', [], ...
   'gui_Callback', []);
    if nargin && ischar(varargin{1})
        gui_State.gui_Callback = str2func(varargin{1});
    end

    if nargout
        [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
    else
        gui_mainfcn(gui_State, varargin{:});
    end

    function rtlsdrinfo_OpeningFcn(hObject, eventdata, handles, varargin)

        handles.output = hObject;
        clc;

        info = sdrinfo;
        diary('text');

        diary on;
        disp(info);
        diary off;

        output = fileread('text');

        set(handles.edit1, 'string', output);

        delete('text');

        guidata(hObject, handles);

        function varargout = rtlsdrinfo_OutputFcn(hObject, eventdata, handles)

            varargout{1} = handles.output;

            function edit1_Callback(hObject, eventdata, handles)

                function edit1_CreateFcn(hObject, eventdata, handles)

                    if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'def
aultUicontrolBackgroundColor'))
                        set(hObject, 'BackgroundColor', 'white');
                    end
```

A-2 GPS Device Info script

```
function varargout = gpsdeviceinfo(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
   'gui_Singleton', gui_Singleton, ...
   'gui_OpeningFcn', @gpsdeviceinfo_OpeningFcn, ...
   'gui_OutputFcn', @gpsdeviceinfo_OutputFcn, ...
   'gui_LayoutFcn', [], ...
  'gui_Callback', []);
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
  [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
  gui_mainfcn(gui_State, varargin{:});
end

function gpsdeviceinfo_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

handles.obj1 = instrfind('Type', 'serial', 'Port', 'COM5', 'Tag', '');

if isempty(handles.obj1)
  handles.obj1 = serial('COM5');

else
  fclose(handles.obj1);
  handles.obj1 = handles.obj1(1);
end

clc;

diary('text');

diary on;
disp(handles.obj1);
diary off;

output = fileread('text');

set(handles.edit1, 'string', output);

delete('text');

fclose(handles.obj1);
flushinput(handles.obj1);
flushoutput(handles.obj1);

guidata(hObject, handles);

function varargout = gpsdeviceinfo_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;
```

```
function edit1_Callback(hObject, eventdata, handles)

function edit1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
  set(hObject, 'BackgroundColor', 'white');
end
```

A-2 Frquency_mapping_tool.m

```
function varargout = Frquency_mapping_tool(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
  'gui_Singleton', gui_Singleton, ...
  'gui_OpeningFcn', @Frquency_mapping_tool_OpeningFcn, ...
  'gui_OutputFcn', @Frquency_mapping_tool_OutputFcn, ...
  'gui_LayoutFcn', [], ...
 'gui_Callback', []);
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
  [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
  gui_mainfcn(gui_State, varargin{:});
end

function Frquency_mapping_tool_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;
handles.radiostatus = 0;
handles.gpsdeviccestatus = 0;
handles.obj1 = '';
handles.timer = 2;
handles.D = [];
handles.FRQ = 0;
handles.Windowmodevalue = 'None';
handles.offset = 30;
handles.grafmin = - 90;
handles.grafmax = 10;
handles.samplesaving = 0;
set(handles.Backupsaving, 'String', '100');
axes(handles.axes2);
grid on;
grid minor;
ylabel('dBm');
xlabel('f0 [MHz]');
title('Spectrum sample');

axes(handles.axes1);
title('Band detail');
grid on;
grid minor;
```

```
ylabel('dBm');
xlabel('f0 [kHz]');

axes(handles.axes4);
title('Timeline');
grid on;
grid minor;
ylabel('dBm');
xlabel('Time HH:MM:SS');

axes(handles.axes5);
title('Levels');
grid on;

set(handles.Status, 'BackgroundColor', 'white');
set(handles.rtlsdrstatus, 'BackgroundColor', 'red');
set(handles.gpsstatus, 'BackgroundColor', 'red');
set(handles.Gps_signal, 'BackgroundColor', 'red');
guidata(hObject, handles);
guidata(hObject, handles);

function varargout = Frquency_mapping_tool_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function Sweep_Callback(hObject, eventdata, handles)

set(handles.Status, 'String', 'Loading data');
set(handles.Status, 'BackgroundColor', 'yellow');

pause(0.01);
FrameLength = 2 * FrontEndSampleRate / 1000;
StartFrequency = - FrameLength / 4;
StopFrequency = FrameLength / 4;
handles.GetGain = get(handles.Gain, 'String');
RadioGain = str2double(handles.GetGain{get(handles.Gain, 'Value')});
Bandwidth = str2double(get(handles.Band, 'string'));
handles.samplesrate = str2double(get(handles.samplestimer, 'string'));
handles.samplesavinglimit = str2double(get(handles.Backupsaving, 'string'));
handles.timelinelimit = str2double(get(handles.Timelinelimit, 'string'));

timer_warning = get(handles.Timer_warning, 'Value');
show_peaks = get(handles.ShowPeaks, 'Value');
usegps = get(handles.UseGPS, 'Value');

if isnan(fc)
  fc = 24000000;
  handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freque
ncy');
  set(handles.f0, 'String', '24');
end
if fc < 24000000
  fc = 24000000;
  set(handles.f0, 'String', '24');
  handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freque
ncy');
else
```

```
  if fc > 1766000000
    fc = 1766000000;
    set(handles.f0, 'String', '1766');
    handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freq
uency');
  end
end

if isnan(Bandwidth)
  Bandwidth = 500;
  handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid bandwi
dth');
  set(handles.Band, 'String', '500');
end
if Bandwidth < 10
  Bandwidth = 10;
  handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid bandwi
dth');
  set(handles.Band, 'String', '10');
else
  if Bandwidth > 2200
    Bandwidth = 2200;
    handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid band
width');
    set(handles.Band, 'String', '2200');
  end
end

if isnan(handles.samplesrate)
  handles.samplesrate = 10;
  handles.h = msgbox('Invalid timer input. Use numbers only.', 'Invalid timer');
  set(handles.samplestimer, 'String', '10');
end

if isnan(handles.samplesavinglimit)
  handles.samplesavinglimit = 100;
  handles.h = msgbox('Invalid backup saving value. Must be number > 0', 'Invalid Backu
p saving value');
  set(handles.Backupsaving, 'String', '100');
end

if handles.samplesavinglimit < 0
  handles.samplesavinglimit = 100;
  handles.h = msgbox('Invalid backup saving value. Must be number > 0', 'Invalid Backu
p saving value');
  set(handles.Backupsaving, 'String', '100');
end

if isnan(handles.timelinelimit)
  handles.timelinelimit = 300;
  handles.h = msgbox('Invalid timeline limit value. Must be number > 10', 'Invalid timelin
e limit value');
  set(handles.Timelinelimit, 'String', '300');
end
```

```
if handles.timelinelimit < 10
  handles.timelinelimit = 300;
  handles.h = msgbox('Invalid timeline limit value. Must be number > 10', 'Invalid timelin
e limit value');
  set(handles.Timelinelimit, 'String', '10');
end

Bottom_Limit = handles.grafmin;
Upper_Limit = handles.grafmax;

StartFRQ = (fc / 1000 + StartFrequency) / 1000;
StopFRQ = (fc / 1000 + StopFrequency) / 1000;

Bw_sides = Bandwidth / 2;
Bw_left = (fc / 1000000) - Bw_sides / 1000;
Bw_right = (fc / 1000000) + Bw_sides / 1000;

handles.Windowmode = get(handles.Window, 'String');
handles.Windowmodevalue = handles.Windowmode{get(handles.Window, 'Value')};

Left = linspace(Bw_left, Bw_left, (Upper_Limit - Bottom_Limit));
Right = linspace(Bw_right, Bw_right, (Upper_Limit - Bottom_Limit));
Bw_line = linspace(Upper_Limit, Bottom_Limit, (Upper_Limit - Bottom_Limit));

hSDRrRx = comm.SDRRTLReceiver(...
  'CenterFrequency', fc, ...
  'EnableTunerAGC', false, ...
  'TunerGain', RadioGain, ...
  'SampleRate', FrontEndSampleRate, ...
  'SamplesPerFrame', FrameLength, ...
 'OutputDataType', 'double');

if isempty(sdrinfo(hSDRrRx.RadioAddress))
  error(set(handles.rtlsdrstatus, 'String', 'Disconnected'), set(handles.rtlsdrstat
us, 'BackgroundColor', 'red'));
  handles.radiostatus = 0;

else
  set(handles.rtlsdrstatus, 'String', 'Connected');
  set(handles.rtlsdrstatus, 'BackgroundColor', 'Green');
  handles.radiostatus = 1;
end

handles.obj1 = instrfind('Type', 'serial', 'Port', 'COM5', 'Tag', '');

if isempty(handles.obj1)
  handles.obj1 = serial('COM5');
  set(handles.gpsstatus, 'String', 'Connected');
  set(handles.gpsstatus, 'BackgroundColor', 'Green');
else
  fclose(handles.obj1);
  handles.obj1 = handles.obj1(1);
  set(handles.gpsstatus, 'String', 'Connected');
  set(handles.gpsstatus, 'BackgroundColor', 'Green');
end

handles.gpsdevicestatus = 1;
try
```

```
fopen(handles.obj1);
fclose(handles.obj1);

catch
set(handles.gpsstatus, 'String', 'Disconnected');
set(handles.gpsstatus, 'BackgroundColor', 'red');
handles.gpsdevicestatus = 0;
end

if usegps ~= 1
handles.gpsdevicestatus = 1;
end

if handles.gpsdevicestatus == 1

 if handles.radiostatus == 1
   set(handles.Status, 'String', 'Running');
   set(handles.Status, 'BackgroundColor', 'Green');

   handles.SamplesCount = 0;
   handles.X = linspace(StartFRQ, StopFRQ, FrameLength);

   drawnow();
   tic;
   set(handles.Status, 'String', 'Measuring');
   set(handles.Status, 'BackgroundColor', 'Green');
   pause(0.01);
   check = 0;
   message = 1;

   try
     check = 1;
   catch
     pause(1);

     check = 0;

   end

 hSDRrRx = comm.SDRRTLReceiver(...
     'CenterFrequency', fc, ...
     'EnableTunerAGC', false, ...
     'TunerGain', RadioGain, ...
     'SampleRate', FrontEndSampleRate, ...
     'SamplesPerFrame', FrameLength, ...
  'OutputDataType', 'double');

 handles.h = msgbox('Missing RTL-
SDR device. Check your hardware connection.', 'Missing device :(');
   beep;
   pause(3);
 end

message = 0;

end

end
```

```
if usegps == 1

  handles.obj1 = instrfind('Type', 'serial', 'Port', 'COM5', 'Tag', '');

  if isempty(handles.obj1)
    handles.obj1 = serial('COM5');
    set(handles.gpsstatus, 'String', 'Connected');
    set(handles.gpsstatus, 'BackgroundColor', 'Green');
  else
    fclose(handles.obj1);
    handles.obj1 = handles.obj1(1);
    set(handles.gpsstatus, 'String', 'Connected');
    set(handles.gpsstatus, 'BackgroundColor', 'Green');
  end

  handles.obj1.BaudRate = 4800;
  handles.obj1.InputBufferSize = 128;
  try
    fopen(handles.obj1);
  catch
    beep;
    handles.h = msgbox('Unsuccesful read.', 'Unsuccesful read from GPS device. Back
up saved.');
    fclose(handles.obj1);
    flushinput(handles.obj1);
    flushoutput(handles.obj1);
    delete(handles.obj1);
    break
  end

  try
    data = '';
    while isempty(strmatch('$GPGLL', data))
      data = fscanf(handles.obj1);
    end
  catch
    beep;
    handles.h = msgbox('Unsuccesful read.', 'Unsuccesful read from GPS device. Back
up saved.');
    fclose(handles.obj1);
    flushinput(handles.obj1);
    flushoutput(handles.obj1);
    delete(handles.obj1);
    break
  end

  fclose(handles.obj1);
  flushinput(handles.obj1);
  flushoutput(handles.obj1);
  delete(handles.obj1);
  clear handles.obj1;
end

switch handles.Windowmodevalue
  case 'Hanning'
    Filtered = iqdata .* hann(FrameLength);
  case 'Blackman'
```

```
    Filtered = iqdata .* blackman(FrameLength);
  case 'Hamming'
    Filtered = iqdata .* hamming(FrameLength);
  case 'None'
    Filtered = iqdata;

  otherwise
    Filtered = iqdata;
end

Y = fftshift(fft(Filtered, length(Filtered)) / length(Filtered));

log = (20 * log10(abs(Y)) + handles.offset);

axes(handles.axes2);
cla reset;
area(handles.X, log, 'BaseValue', - 110, 'FaceColor', [0 0.447 0.741], 'EdgeColor'
, [0 0.347 0.641]);
hold on;
plot(Left, Bw_line, '--r', Right, Bw_line, '--r');

axis([StartFRQ, StopFRQ, Bottom_Limit Upper_Limit]);
grid on;
grid minor;
ylabel('dBm');
xlabel('f0 [MHz]');
title('Spectrum sample');

if show_peaks == 1
  axes(handles.axes2);
  hold on;
  [pk, lk] = findpeaks(log, handles.X, 'MinPeakHeight', mean(log) + 10, 'MinPeak
Distance', 0.05);
  plot(lk, pk, 'o', 'Color', [0.929 0.694 0.125], 'MarkerSize', 10, 'LineWidth', 2);
end

log_size = length(log);
left = (log_size / 2) - (2 * Bw_sides);
right = (log_size / 2) + (2 * Bw_sides);
block = log(left:right);
left_cut = log(1:left);
right_cut = log(right:end);
cut_block = [left_cut; right_cut];
handles.peak = max(block);

handles.bp = mean(block);
handles.sm = mean(cut_block);

log_size = length(block);
frq2 = linspace(- log_size / 4, log_size / 4, log_size);
bpw = linspace(handles.bp, handles.bp, log_size);
pk = linspace(handles.peak, handles.peak, log_size);
sm1 = linspace(handles.sm, handles.sm, log_size);

axes(handles.axes1);
cla reset;
area(frq2, block, 'BaseValue', - 110, 'FaceColor', [0 0.447 0.741], 'EdgeColor', [
0 0.347 0.641]);

grid on;
grid minor;
```

```
ylabel('dBm');
title('Band detail');
xlabel('f0 [kHz]');
ylim([Bottom_Limit Upper_Limit]);
xlim([- Bw_sides + Bw_sides]);

axes(handles.axes5);
cla reset;
area(frq2, pk, 'FaceColor', [0 0.647 0.941], 'EdgeColor', [0 0.547 0.899], 'BaseValue', handles.grafmin);
hold on;
area(frq2, bpw, 'FaceColor', [0 0.547 0.841], 'EdgeColor', [0 0.447 0.741], 'BaseValue', handles.grafmin);
hold on;
area(frq2, sm1, 'FaceColor', [0 0.447 0.741], 'EdgeColor', [0 0.347 0.641], 'BaseValue', handles.grafmin);
title('Levels');
grid on;
ylim([Bottom_Limit Upper_Limit]);

set(handles.Peak, 'String', num2str(handles.peak));
set(handles.Bandmean, 'String', num2str(handles.bp));
set(handles.Spectrummean, 'String', num2str(handles.sm));

if usegps == 1

  [lat, data] = strtok(data, ',');
  [lat, data] = strtok(data, ',');
  [nsCardinal, data] = strtok(data, ',');
  [long, data] = strtok(data, ',');
  [ewCardinal, data] = strtok(data, ',');
  lat = str2double(lat);
  long = str2double(long);

  y_degrees = floor(lat / 100);
  y_minutes = rem(lat, 100);
  x_degrees = floor(long / 100);
  x_minutes = rem(long, 100);

  signy = 1;
  signx = 1;

  if y_degrees < 0
    signy = - 1;
  else
    if y_minutes < 0
      signy = - 1;
    end
  end
  if x_degrees < 0
    signx = - 1;
  else
    if x_minutes < 0
      signx = - 1;
    end
  end

  X = signx * (abs(x_degrees) + abs(x_minutes) / 60);
```

```
  Y = signy * (abs(y_degrees) + abs(y_minutes) / 60);

  if isnan(X)
    X = 0;
    Y = 0;
    set(handles.Gps_signal, 'BackgroundColor', 'cyan');
    set(handles.Gps_signal, 'String', 'Weak');
  else

    if isnan(Y)
      Y = 0;
      X = 0;
      set(handles.Gps_signal, 'BackgroundColor', 'cyan');
      set(handles.Gps_signal, 'String', 'Weak');
    else
      set(handles.Gps_signal, 'BackgroundColor', 'Green');
      set(handles.Gps_signal, 'String', 'Ok');
    end
  end

  set(handles.latitude, 'String', num2str(Y));
  set(handles.longitude, 'String', num2str(X));
else
  X = 0;
  Y = 0;
end
TimeValue = now;

handles.D = [handles.D;
X Y handles.peak handles.bp handles.sm TimeValue];

if handles.SamplesCount >= 1

  if handles.SamplesCount <= handles.timelinelimit
    axes(handles.axes4);
    cla reset;
    area(handles.D(:, 6), handles.D(:, 3), 'FaceColor', [0 0.647 0.941], 'EdgeCol
or', [0 0.547 0.899], 'BaseValue', handles.grafmin);
    hold on;
    area(handles.D(:, 6), handles.D(:, 4), 'FaceColor', [0 0.547 0.841], 'EdgeCol
or', [0 0.447 0.741], 'BaseValue', handles.grafmin);
    hold on;
    area(handles.D(:, 6), handles.D(:, 5), 'FaceColor', [0 0.447 0.741], 'EdgeCol
or', [0 0.347 0.641], 'BaseValue', handles.grafmin);
    title('Timeline');
    grid on;
    grid minor;
    ylabel('dBm');
    xlabel('Time HH:MM:SS');
    datetick('x', 'HH:MM:SS');
    ylim([Bottom_Limit Upper_Limit]);
    xlim([min(handles.D(:, 6)) max(handles.D(:, 6))]);
  else
    axes(handles.axes4);
    cla reset;
    area(handles.D(:, 6), handles.D(:, 3), 'FaceColor', [0 0.647 0.941], 'EdgeCol
or', [0 0.547 0.899], 'BaseValue', handles.grafmin);
```

```
    hold on;
    area(handles.D(:, 6), handles.D(:, 4), 'FaceColor', [0 0.547 0.841], 'EdgeColor', [0 0.447 0.741], 'BaseValue', handles.grafmin);
    hold on;
    area(handles.D(:, 6), handles.D(:, 5), 'FaceColor', [0 0.447 0.741], 'EdgeColor', [0 0.347 0.641], 'BaseValue', handles.grafmin);
    title('Timeline');
    grid on;
    grid minor;
    ylabel('dBm');
    xlabel('Time HH:MM:SS');
    datetick('x', 'HH:MM:SS');
    ylim([Bottom_Limit Upper_Limit]);
    xlim([handles.D(end - handles.timelinelimit, 6) handles.D(end, 6)]);
  end

end

handles.SamplesCount = handles.SamplesCount + 1;
set(handles.samplescount, 'String', num2str(handles.SamplesCount));
handles.samplesaving = handles.samplesaving + 1;

if handles.samplesaving == handles.samplesavinglimit
  handles.samplesaving = 0;

end
handles.timer = toc;
if isappdata(handles.figure1, 'stopPlot')
  break
end

waiting = handles.samplesrate - handles.timer;
set(handles.Status, 'String', 'Waiting');
set(handles.Status, 'BackgroundColor', 'yellow');

timer_enable = get(handles.TimerEnable, 'Value');

if timer_enable == 1

  if waiting < 0
    if timer_warning == 1
      handles.h = msgbox('Measurement exceeded the specified time.', 'Timer warning');
    end
    waiting = 0.1;
  end

  pause(waiting);
end

if isappdata(handles.figure1, 'stopPlot')
  break
end

set(handles.Status, 'String', 'Done');
set(handles.Status, 'BackgroundColor', 'white');
rmappdata(handles.figure1, 'stopPlot');
```

```
else
  beep;
  handles.h = msgbox('Missing RTL-
SDR device. Check your hardware connection.', 'Missing device :(');
  set(handles.rtlsdrstatus, 'String', 'Disconnected');
  set(handles.rtlsdrstatus, 'BackgroundColor', 'red');
  set(handles.Status, 'String', 'Stopped');
  set(handles.Status, 'BackgroundColor', 'red');
end

else
  if handles.radiostatus == 1
    beep;
    handles.h = msgbox('Missing GPS reciever. Check your hardware connection.', 'Missin
g device :(');
    set(handles.Status, 'String', 'Stopped');
    set(handles.Status, 'BackgroundColor', 'red');
    set(handles.gpsstatus, 'String', 'Disconnected');
    set(handles.gpsstatus, 'BackgroundColor', 'red');
    set(handles.Gps_signal, 'BackgroundColor', 'red');
    set(handles.Gps_signal, 'String', 'None');

  else
    beep;
    handles.h = msgbox('Missing RTL-
SDR and GPS reciever. Check your hardware connection.', 'Missing devices :(');
    set(handles.Status, 'String', 'Stopped');
    set(handles.Status, 'BackgroundColor', 'red');
    set(handles.gpsstatus, 'String', 'Disconnected');
    set(handles.gpsstatus, 'BackgroundColor', 'red');
    set(handles.Gps_signal, 'BackgroundColor', 'red');
    set(handles.Gps_signal, 'String', 'None');
    set(handles.rtlsdrstatus, 'String', 'Disconnected');
    set(handles.rtlsdrstatus, 'BackgroundColor', 'red');
  end
end

guidata(hObject, handles);

function latitude_Callback(hObject, eventdata, handles)

function latitude_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function longitude_Callback(hObject, eventdata, handles)

function longitude_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
```

```
set(hObject, 'BackgroundColor', 'white');
end

function Peak_Callback(hObject, eventdata, handles)

function Peak_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
set(hObject, 'BackgroundColor', 'white');
end

function edit6_Callback(hObject, eventdata, handles)

function edit6_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
set(hObject, 'BackgroundColor', 'white');
end

function edit7_Callback(hObject, eventdata, handles)

function edit7_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
set(hObject, 'BackgroundColor', 'white');
end

function f0_Callback(hObject, eventdata, handles)

function f0_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
set(hObject, 'BackgroundColor', 'white');
end

function Band_Callback(hObject, eventdata, handles)

function Band_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
set(hObject, 'BackgroundColor', 'white');
end

function Start_Callback(hObject, eventdata, handles)

function File_Callback(hObject, eventdata, handles)

function Devices_Callback(hObject, eventdata, handles)

function hwcheck_Callback(hObject, eventdata, handles)
```

```
handles.radiostatus = 0;
handles.gpsdevicestatus = 0;

set(handles.rtlsdrstatus, 'String', 'Searching');
set(handles.rtlsdrstatus, 'BackgroundColor', 'yellow');
set(handles.gpsstatus, 'String', 'Searching');
set(handles.gpsstatus, 'BackgroundColor', 'yellow');
set(handles.Status, 'String', 'Checking');
set(handles.Status, 'BackgroundColor', 'yellow');
pause(0.1);

FrameLength = 4096;
handles.StartFrequency = - 1024e3;
handles.StopFrequency = 1024e3;

hSDRrRx = comm.SDRRTLReceiver(...
  'CenterFrequency', fc, ...
  'EnableTunerAGC', true, ...
  'SampleRate', FrontEndSampleRate, ...
  'SamplesPerFrame', FrameLength, ...
 'OutputDataType', 'double');
if isempty(sdrinfo(hSDRrRx.RadioAddress))
  error(set(handles.rtlsdrstatus, 'String', 'Disconnected'), set(handles.rtlsdrstat
us, 'BackgroundColor', 'red'));
  handles.radiostatus = 0;

else
  set(handles.rtlsdrstatus, 'String', 'Connected');
  set(handles.rtlsdrstatus, 'BackgroundColor', 'Green');
  handles.radiostatus = 1;
end

handles.obj1 = instrfind('Type', 'serial', 'Port', 'COM5', 'Tag', '');

if isempty(handles.obj1)
  handles.obj1 = serial('COM5');
  set(handles.gpsstatus, 'String', 'Connected');
  set(handles.gpsstatus, 'BackgroundColor', 'Green');
else
  fclose(handles.obj1);
  handles.obj1 = handles.obj1(1);
  set(handles.gpsstatus, 'String', 'Connected');
  set(handles.gpsstatus, 'BackgroundColor', 'Green');
end

try

  fopen(handles.obj1);

catch
  set(handles.gpsstatus, 'String', 'Disconnected');
  set(handles.gpsstatus, 'BackgroundColor', 'red');
  set(handles.Gps_signal, 'BackgroundColor', 'red');
  set(handles.Gps_signal, 'String', 'None');
end
```

```
set(handles.Status, 'String', 'Done');
set(handles.Status, 'BackgroundColor', 'white');
handles.h = msgbox('Hw check complete', 'Done');
guidata(hObject, handles);

function gpsdevice_Callback(hObject, eventdata, handles)
run gpsdeviceinfo;
guidata(hObject, handles);

function rtlsdrdevice_Callback(hObject, eventdata, handles)
run rtlsdrinfo;
guidata(hObject, handles);

function Window_Callback(hObject, eventdata, handles)

function Window_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function samplestimer_Callback(hObject, eventdata, handles)

function samplestimer_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function Samples_Callback(hObject, eventdata, handles)

function Samples_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function samplescount_Callback(hObject, eventdata, handles)

function samplescount_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function SpectumSample_Callback(hObject, eventdata, handles)

set(handles.Status, 'String', 'Loading data');
set(handles.Status, 'BackgroundColor', 'yellow');
pause(0.1);

FrameLength = 2 * FrontEndSampleRate / 1000;
StartFrequency = - FrameLength / 4;
```

```
StopFrequency = FrameLength / 4;
handles.GetGain = get(handles.Gain, 'String');
RadioGain = str2double(handles.GetGain{get(handles.Gain, 'Value')});
Bandwidth = str2double(get(handles.Band, 'string'));
show_peaks = get(handles.ShowPeaks, 'Value');

if isnan(fc)
  fc = 24000000;
  handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freque
ncy');
  set(handles.f0, 'String', '24');
end
if fc < 24000000
  fc = 24000000;
  set(handles.f0, 'String', '24');
  handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freque
ncy');
else
  if fc > 1766000000
    fc = 1766000000;
    set(handles.f0, 'String', '1766');
    handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freq
uency');
  end
end

if isnan(Bandwidth)
  Bandwidth = 500;
  handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid bandwi
dth');
  set(handles.Band, 'String', '500');
end
if Bandwidth < 10
  Bandwidth = 10;
  handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid bandwi
dth');
  set(handles.Band, 'String', '10');
else
  if Bandwidth > 2200
    Bandwidth = 2200;
    handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid band
width');
    set(handles.Band, 'String', '2200');
  end
end

Bottom_Limit = handles.grafmin;
Upper_Limit = handles.grafmax;

StartFRQ = (fc / 1000 + StartFrequency) / 1000;
StopFRQ = (fc / 1000 + StopFrequency) / 1000;

handles.Windowmode = get(handles.Window, 'String');
handles.Windowmodevalue = handles.Windowmode{get(handles.Window, 'Value')};
```

```
Bw_sides = Bandwidth / 2;
Bw_left = (fc / 1000000) - Bw_sides / 1000;
Bw_right = (fc / 1000000) + Bw_sides / 1000;

Left = linspace(Bw_left, Bw_left, (Upper_Limit - Bottom_Limit));
Right = linspace(Bw_right, Bw_right, (Upper_Limit - Bottom_Limit));
Bw_line = linspace(Upper_Limit, Bottom_Limit, (Upper_Limit - Bottom_Limit));

hSDRrRx = comm.SDRRTLReceiver(...
  'CenterFrequency', fc, ...
  'EnableTunerAGC', false, ...
  'TunerGain', RadioGain, ...
  'SampleRate', FrontEndSampleRate, ...
  'SamplesPerFrame', FrameLength, ...
 'OutputDataType', 'double');

if isempty(sdrinfo(hSDRrRx.RadioAddress))
  error(set(handles.rtlsdrstatus, 'String', 'Disconnected'), set(handles.rtlsdrstat
us, 'BackgroundColor', 'red'));
  handles.radiostatus = 0;

else
  set(handles.rtlsdrstatus, 'String', 'Connected');
  set(handles.rtlsdrstatus, 'BackgroundColor', 'Green');
  handles.radiostatus = 1;
end

if handles.radiostatus == 1
  set(handles.Status, 'String', 'Running');
  set(handles.Status, 'BackgroundColor', 'Green');
  pause(0.1);
  handles.count = 0;

  handles.samplesrate = str2num(get(handles.samplestimer, 'string'));

  if ~ isempty(sdrinfo(hSDRrRx.RadioAddress))
    for count = 1:1
      check = 0;
      message = 1;

      try
        check = 1;
      catch
        pause(1);

        check = 0;

      end

   hSDRrRx = comm.SDRRTLReceiver(...
        'CenterFrequency', fc, ...
        'EnableTunerAGC', false, ...
        'TunerGain', RadioGain, ...
        'SampleRate', FrontEndSampleRate, ...
        'SamplesPerFrame', FrameLength, ...
    'OutputDataType', 'double');

```

```
handles.h = msgbox('Missing RTL-
SDR device. Check your hardware connection.', 'Missing device :(');
      beep;
      pause(3);
    end

    message = 0;

  end

end

X = linspace(StartFRQ, StopFRQ, FrameLength);

switch handles.Windowmodevalue
  case 'Hanning'
    Filtered = iqdata .* hann(FrameLength);
  case 'Blackman'
    Filtered = iqdata .* blackman(FrameLength);
  case 'Hamming'
    Filtered = iqdata .* hamming(FrameLength);
  case 'None'
    Filtered = iqdata;
  otherwise
    Filtered = iqdata;
end

Y = fftshift(fft(Filtered, length(Filtered)) / length(Filtered));

log = (20 * log10(abs(Y)) + handles.offset);

axes(handles.axes2);
cla reset;
area(X, log, 'BaseValue', - 110, 'FaceColor', [0 0.447 0.741], 'EdgeColor', [0 0.3
47 0.641]);
hold on;
plot(Left, Bw_line, '--r', Right, Bw_line, '--r');
hold on;

axis([StartFRQ, StopFRQ, Bottom_Limit Upper_Limit]);
grid on;
grid minor;
ylabel('dBm');
title('Spectrum sample');
xlabel('f0 [MHz]');

if show_peaks == 1
  axes(handles.axes2);
  hold on;
  [pk, lk] = findpeaks(log, X, 'MinPeakHeight', mean(log) + 10, 'MinPeakDistance'
, 0.05);
  plot(lk, pk, 'o', 'Color', [0.929 0.694 0.125], 'MarkerSize', 10, 'LineWidth', 2);
end

log_size = length(log);
left = (log_size / 2) - (2 * Bw_sides);
right = (log_size / 2) + (2 * Bw_sides);
block = log(left:right);
left_cut = log(1:left);
right_cut = log(right:end);
```

```
cut_block = [left_cut; right_cut];
handles.peak = max(block);

handles.bp = mean(block);
handles.sm = mean(cut_block);
log_size = length(block);

frq2 = linspace(- log_size / 4, log_size / 4, log_size);
bpw = linspace(handles.bp, handles.bp, log_size);
pk = linspace(handles.peak, handles.peak, log_size);
sm1 = linspace(handles.sm, handles.sm, log_size);

axes(handles.axes1);
cla reset;
area(frq2, block, 'BaseValue', - 110, 'FaceColor', [0 0.447 0.741], 'EdgeColor', [
0 0.347 0.641]);

grid on;
grid minor;
ylabel('dBm');
title('Band detail');
ylim([Bottom_Limit Upper_Limit]);
xlim([- Bw_sides + Bw_sides]);
xlabel('f0 [kHz]');

axes(handles.axes5);
cla reset;
area(frq2, pk, 'FaceColor', [0 0.647 0.941], 'EdgeColor', [0 0.547 0.899], 'BaseV
alue', handles.grafmin);
hold on;
area(frq2, bpw, 'FaceColor', [0 0.547 0.841], 'EdgeColor', [0 0.447 0.741], 'Base
Value', handles.grafmin);
hold on;
area(frq2, sm1, 'FaceColor', [0 0.447 0.741], 'EdgeColor', [0 0.347 0.641], 'Base
Value', handles.grafmin);
title('Levels');
grid on;
ylim([Bottom_Limit Upper_Limit]);
grid on;
ylim([Bottom_Limit Upper_Limit]);

set(handles.Peak, 'String', num2str(handles.peak));
set(handles.Bandmean, 'String', num2str(handles.bp));
set(handles.Spectrummean, 'String', num2str(handles.sm));

end
else
  warning(message('SDR:sysobjdemos:MainLoop'))
end

set(handles.Status, 'String', 'Done');
set(handles.Status, 'BackgroundColor', 'white');
else
  beep;
  handles.h = msgbox('Missing RTL-
SDR device. Check your hardware connection.', 'Missing device :(');
  set(handles.rtlsdrstatus, 'String', 'Disconnected');
  set(handles.rtlsdrstatus, 'BackgroundColor', 'red');
  set(handles.Status, 'String', 'Stopped');
```

```
  set(handles.Status, 'BackgroundColor', 'red');
end

guidata(hObject, handles);

function Position_Callback(hObject, eventdata, handles)

set(handles.Status, 'String', 'Loading data');
set(handles.Status, 'BackgroundColor', 'yellow');
pause(0.1);

handles.obj1 = instrfind('Type', 'serial', 'Port', 'COM5', 'Tag', '');

if isempty(handles.obj1)
  handles.obj1 = serial('COM5');
  set(handles.gpsstatus, 'String', 'Connected');
  set(handles.gpsstatus, 'BackgroundColor', 'Green');
else
  fclose(handles.obj1);
  handles.obj1 = handles.obj1(1);
  set(handles.gpsstatus, 'String', 'Connected');
  set(handles.gpsstatus, 'BackgroundColor', 'Green');
end

handles.gpsdevicestatus = 1;
try

  fopen(handles.obj1);
  fclose(handles.obj1);

catch
  set(handles.gpsstatus, 'String', 'Disconnected');
  set(handles.gpsstatus, 'BackgroundColor', 'red');
  handles.gpsdevicestatus = 0;
end

set(handles.Status, 'String', 'Running');
set(handles.Status, 'BackgroundColor', 'Green');
pause(0.1);
if handles.gpsdevicestatus == 1

  fopen(handles.obj1);

  handles.obj1.BaudRate = 4800;
  handles.InputBufferSize = 512;

  data = '';
  while isempty(strmatch('$GPGLL', data))
    data = fscanf(handles.obj1);
  end

  [lat, data] = strtok(data, ',');
  [lat, data] = strtok(data, ',');
  [nsCardinal, data] = strtok(data, ',');
  [long, data] = strtok(data, ',');
  [ewCardinal, data] = strtok(data, ',');
  lat = str2double(lat);
  long = str2double(long);
```

```
y_degrees = floor(lat / 100);
y_minutes = rem(lat, 100);
x_degrees = floor(long / 100);
x_minutes = rem(long, 100);

signy = 1;
signx = 1;

if y_degrees < 0
  signy = - 1;
else
  if y_minutes < 0
    signy = - 1;
  end
end
if x_degrees < 0
  signx = - 1;
else
  if x_minutes < 0
    signx = - 1;
  end
end

X = signx * (abs(x_degrees) + abs(x_minutes) / 60);
Y = signy * (abs(y_degrees) + abs(y_minutes) / 60);

handles.X_copy = 0;
handles.Y_copy = 0;

handles.X_copy = X;
handles.Y_copy = Y;

if isnan(X)
  X = 0;
  Y = 0;
else

  if isnan(Y)
    Y = 0;
    X = 0;
  end

end

set(handles.latitude, 'String', num2str(Y));
set(handles.longitude, 'String', num2str(X));

fclose(handles.obj1);
flushinput(handles.obj1);
flushoutput(handles.obj1);
delete(handles.obj1);
clear handles.obj1;

if isnan(handles.X_copy)
  set(handles.Gps_signal, 'BackgroundColor', 'cyan');
  set(handles.Gps_signal, 'String', 'Weak');
 handles.h = msgbox('Weak signal on your GPS. Position set to Lat 0°, Lon 0°.', 'Weak
signal :(');
```

```
  else

    if isnan(handles.Y_copy)
      set(handles.Gps_signal, 'String', 'Weak');
      set(handles.Gps_signal, 'BackgroundColor', 'cyan');
   handles.h = msgbox('Weak signal on your GPS. Position set to Lat 0°, Lon 0°.', 'Weak
signal :(');
    else
      set(handles.Gps_signal, 'BackgroundColor', 'Green');
      set(handles.Gps_signal, 'String', 'Ok');
    end
  end

  set(handles.Status, 'String', 'Done');
  set(handles.Status, 'BackgroundColor', 'white');

else
  beep;
  handles.h = msgbox('Missing GPS reciever. Check your hardware connection.', 'Missing
device :(');
  set(handles.gpsstatus, 'String', 'Disconnected');
  set(handles.gpsstatus, 'BackgroundColor', 'red');
  set(handles.Gps_signal, 'BackgroundColor', 'red');
  set(handles.Gps_signal, 'String', 'None');
  set(handles.Status, 'String', 'Stopped');
  set(handles.Status, 'BackgroundColor', 'red');
end

guidata(hObject, handles);

function pushbutton7_Callback(hObject, eventdata, handles)

function pushbutton8_Callback(hObject, eventdata, handles)

function pushbutton9_Callback(hObject, eventdata, handles)

function pushbutton10_Callback(hObject, eventdata, handles)

function CalibrateTimer_Callback(hObject, eventdata, handles)

set(handles.Status, 'String', 'Loading data');
set(handles.Status, 'BackgroundColor', 'yellow');
handles.count = 0;
set(handles.samplescount, 'String', num2str(handles.count));

pause(0.01);
FrameLength = 2 * FrontEndSampleRate / 1000;
StartFrequency = - FrameLength / 4;
StopFrequency = FrameLength / 4;
handles.GetGain = get(handles.Gain, 'String');
RadioGain = str2double(handles.GetGain{get(handles.Gain, 'Value')});
Bandwidth = str2double(get(handles.Band, 'string'));
show_peaks = get(handles.ShowPeaks, 'Value');

if isnan(fc)
  fc = 24000000;
```

```
  handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freque
ncy');
  set(handles.f0, 'String', '24');
end
if fc < 24000000
  fc = 24000000;
  set(handles.f0, 'String', '24');
  handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freque
ncy');
else
  if fc > 1766000000
    fc = 1766000000;
    set(handles.f0, 'String', '1766');
    handles.h = msgbox('Invalid frequency. Keep between 24 - 1766 MHz.', 'Invalid freq
uency');
  end
end

if isnan(Bandwidth)
  Bandwidth = 500;
  handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid bandwi
dth');
  set(handles.Band, 'String', '500');
end
if Bandwidth < 10
  Bandwidth = 10;
  handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid bandwi
dth');
  set(handles.Band, 'String', '10');
else
  if Bandwidth > 2200
    Bandwidth = 2200;
    handles.h = msgbox('Invalid bandwidth. Keep between 10 - 2200 kHz', 'Invalid band
width');
    set(handles.Band, 'String', '2200');
  end
end

Bottom_Limit = handles.grafmin;
Upper_Limit = handles.grafmax;

StartFRQ = (fc / 1000 + StartFrequency) / 1000;
StopFRQ = (fc / 1000 + StopFrequency) / 1000;

handles.Windowmode = get(handles.Window, 'String');
handles.Windowmodevalue = handles.Windowmode{get(handles.Window, 'Value')};

Bw_sides = Bandwidth / 2;
Bw_left = (fc / 1000000) - Bw_sides / 1000;
Bw_right = (fc / 1000000) + Bw_sides / 1000;

Left = linspace(Bw_left, Bw_left, (Upper_Limit - Bottom_Limit));
Right = linspace(Bw_right, Bw_right, (Upper_Limit - Bottom_Limit));
Bw_line = linspace(Upper_Limit, Bottom_Limit, (Upper_Limit - Bottom_Limit));

hSDRrRx = comm.SDRRTLReceiver(...
```

```
  'CenterFrequency', fc, ...
  'EnableTunerAGC', false, ...
  'TunerGain', RadioGain, ...
  'SampleRate', FrontEndSampleRate, ...
  'SamplesPerFrame', FrameLength, ...
 'OutputDataType', 'double');

if isempty(sdrinfo(hSDRrRx.RadioAddress))
  error(set(handles.rtlsdrstatus, 'String', 'Disconnected'), set(handles.rtlsdrstat
us, 'BackgroundColor', 'red'));
  handles.radiostatus = 0;

else
  set(handles.rtlsdrstatus, 'String', 'Connected');
  set(handles.rtlsdrstatus, 'BackgroundColor', 'Green');
  handles.radiostatus = 1;
end

handles.obj1 = instrfind('Type', 'serial', 'Port', 'COM5', 'Tag', '');

if isempty(handles.obj1)
  handles.obj1 = serial('COM5');
  set(handles.gpsstatus, 'String', 'Connected');
  set(handles.gpsstatus, 'BackgroundColor', 'Green');
else
  fclose(handles.obj1);
  handles.obj1 = handles.obj1(1);
  set(handles.gpsstatus, 'String', 'Connected');
  set(handles.gpsstatus, 'BackgroundColor', 'Green');
end

handles.gpsdevicestatus = 1;
try

  fopen(handles.obj1);
  fclose(handles.obj1);

catch
  set(handles.gpsstatus, 'String', 'Disconnected');
  set(handles.gpsstatus, 'BackgroundColor', 'red');
  set(handles.Gps_signal, 'BackgroundColor', 'red');
  set(handles.Gps_signal, 'String', 'None');
  handles.gpsdevicestatus = 0;
end

if handles.gpsdevicestatus == 1

  if handles.radiostatus == 1
    set(handles.Status, 'String', 'Calibrating');
    set(handles.Status, 'BackgroundColor', 'Green');
    pause(0.01);

    handles.count = 0;

    handles.samplesrate = str2num(get(handles.samplestimer, 'string'));

```

```
    for count = 1:1

      if ~ isempty(sdrinfo(hSDRrRx.RadioAddress))
        tic;
        set(handles.Status, 'String', 'Calibrating');
        set(handles.Status, 'BackgroundColor', 'Green');
        pause(0.01);
        check = 0;
        message = 1;

        try
          check = 1;
        catch
          pause(1);

          check = 0;

        end

     hSDRrRx = comm.SDRRTLReceiver(...
          'CenterFrequency', fc, ...
          'EnableTunerAGC', false, ...
          'TunerGain', RadioGain, ...
          'SampleRate', FrontEndSampleRate, ...
          'SamplesPerFrame', FrameLength, ...
      'OutputDataType', 'double');

     handles.h = msgbox('Missing RTL-
SDR device. Check your hardware connection.', 'Missing device :(');
        beep;
        pause(3);
      end

      message = 0;

    end

  end

  X = linspace(StartFRQ, StopFRQ, FrameLength);

  switch handles.Windowmodevalue
    case 'Hanning'
      Filtered = iqdata .* hann(FrameLength);
    case 'Blackman'
      Filtered = iqdata .* blackman(FrameLength);
    case 'Hamming'
      Filtered = iqdata .* hamming(FrameLength);
    case 'None'
      Filtered = iqdata;

    otherwise
      Filtered = iqdata;
  end

  Y = fftshift(fft(Filtered, length(Filtered)) / length(Filtered));

  log = (20 * log10(abs(Y)) + 30);
  axes(handles.axes2);
```

```
  cla reset;
  area(X, log, 'BaseValue', - 110, 'FaceColor', [0 0.447 0.741], 'EdgeColor', [0 0
.347 0.641]);
  hold on;
  plot(Left, Bw_line, '--r', Right, Bw_line, '--r');

  axis([StartFRQ, StopFRQ, Bottom_Limit Upper_Limit]);
  grid on;
  grid minor;
  title('Spectrum sample');
  ylabel('dBm');

  if show_peaks == 1
    axes(handles.axes2);
    hold on;
    [pk, lk] = findpeaks(log, X, 'MinPeakHeight', mean(log) + 10, 'MinPeakDistanc
e', 0.05);
    plot(lk, pk, 'o', 'Color', [0.929 0.694 0.125], 'MarkerSize', 10, 'LineWidth', 2)
;
  end

  log_size = length(log);
  left = (log_size / 2) - (2 * Bw_sides);
  right = (log_size / 2) + (2 * Bw_sides);
  block = log(left:right);
  left_cut = log(1:left);
  right_cut = log(right:end);
  cut_block = [left_cut; right_cut];
  handles.peak = max(block);

  handles.bp = mean(block);
  handles.sm = mean(cut_block);

  log_size = length(block);
  frq2 = linspace(- log_size / 4, log_size / 4, log_size);
  bpw = linspace(handles.bp, handles.bp, log_size);
  sm1 = linspace(handles.sm, handles.sm, log_size);
  pk = linspace(handles.peak, handles.peak, log_size);

  axes(handles.axes1);
  cla reset;
  area(frq2, block, 'BaseValue', - 110, 'FaceColor', [0 0.447 0.741], 'EdgeColor',
 [0 0.347 0.641]);

  grid on;
  grid minor;
  ylabel('dBm');
  title('Band detail');
  ylim([Bottom_Limit Upper_Limit]);
  xlim([- Bw_sides + Bw_sides]);

  axes(handles.axes5);
  cla reset;
  area(frq2, pk, 'FaceColor', [0 0.647 0.941], 'EdgeColor', [0 0.547 0.899], 'Base
Value', handles.grafmin);
  hold on;
  area(frq2, bpw, 'FaceColor', [0 0.547 0.841], 'EdgeColor', [0 0.447 0.741], 'Bas
eValue', handles.grafmin);
  hold on;
  area(frq2, sm1, 'FaceColor', [0 0.447 0.741], 'EdgeColor', [0 0.347 0.641], 'Bas
eValue', handles.grafmin);
```

```
  title('Levels');
  grid on;
  ylim([Bottom_Limit Upper_Limit]);

  set(handles.Peak, 'String', num2str(handles.peak));
  set(handles.Bandmean, 'String', num2str(handles.bp));
  set(handles.Spectrummean, 'String', num2str(handles.sm));
else
  warning(message('SDR:sysobjdemos:MainLoop'))
end

handles.obj1.BaudRate = 4800;
handles.obj1.InputBufferSize = 128;
fopen(handles.obj1);

data = '';
while isempty(strmatch('$GPGLL', data))
  data = fscanf(handles.obj1);
end

[lat, data] = strtok(data, ',');
[lat, data] = strtok(data, ',');
[nsCardinal, data] = strtok(data, ',');
[long, data] = strtok(data, ',');
[ewCardinal, data] = strtok(data, ',');
lat = str2double(lat);
long = str2double(long);

y_degrees = floor(lat / 100);
y_minutes = rem(lat, 100);
x_degrees = floor(long / 100);
x_minutes = rem(long, 100);

signy = 1;
signx = 1;

if y_degrees < 0
  signy = - 1;
else
  if y_minutes < 0
    signy = - 1;
  end
end
if x_degrees < 0
  signx = - 1;
else
  if x_minutes < 0
    signx = - 1;
  end
end

X = signx * (abs(x_degrees) + abs(x_minutes) / 60);
Y = signy * (abs(y_degrees) + abs(y_minutes) / 60);

if isnan(X)
  X = 0;
  Y = 0;
  set(handles.Gps_signal, 'BackgroundColor', 'cyan');
```

```
  set(handles.Gps_signal, 'String', 'Weak');
else

  if isnan(Y)
    Y = 0;
    X = 0;
    set(handles.Gps_signal, 'BackgroundColor', 'cyan');
    set(handles.Gps_signal, 'String', 'Weak');

  else
    set(handles.Gps_signal, 'BackgroundColor', 'green');
    set(handles.Gps_signal, 'String', 'OK');
  end
end

set(handles.latitude, 'String', num2str(Y));
set(handles.longitude, 'String', num2str(X));

fclose(handles.obj1);
flushinput(handles.obj1);
flushoutput(handles.obj1);
delete(handles.obj1);
clear handles.obj1;

end
time = toc;
handles.timer = round(time);

set(handles.Status, 'String', 'Done');
set(handles.Status, 'BackgroundColor', 'white');

else
  beep;
  handles.h = msgbox('Missing RTL-
SDR device. Check your hardware connection.', 'Missing device :(');
  set(handles.rtlsdrstatus, 'String', 'Disconnected');
  set(handles.rtlsdrstatus, 'BackgroundColor', 'red');
  set(handles.Status, 'String', 'Stopped');
  set(handles.Status, 'BackgroundColor', 'red');
end

else
  if handles.radiostatus == 1

    beep;
    handles.h = msgbox('Missing GPS reciever. Check your hardware connection.', 'Missin
g device :(');
    set(handles.Status, 'String', 'Stopped');
    set(handles.Status, 'BackgroundColor', 'red');
    set(handles.gpsstatus, 'String', 'Disconnected');
    set(handles.gpsstatus, 'BackgroundColor', 'red');
    set(handles.Gps_signal, 'BackgroundColor', 'red');
    set(handles.Gps_signal, 'String', 'None');

  else
    beep;
    handles.h = msgbox('Missing RTL-
```

```
SDR and GPS reciever. Check your hardware connection.', 'Missing devices :(');
    set(handles.Status, 'String', 'Stopped');
    set(handles.Status, 'BackgroundColor', 'red');
    set(handles.gpsstatus, 'String', 'Disconnected');
    set(handles.gpsstatus, 'BackgroundColor', 'red');
    set(handles.Gps_signal, 'BackgroundColor', 'red');
    set(handles.Gps_signal, 'String', 'None');
    set(handles.rtlsdrstatus, 'String', 'Disconnected');
    set(handles.rtlsdrstatus, 'BackgroundColor', 'red');

  end
end

set(handles.samplestimer, 'String', num2str(handles.timer));

guidata(hObject, handles);

function Gain_Callback(hObject, eventdata, handles)

function Gain_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgrou
ndColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function Untitled_2_Callback(hObject, eventdata, handles)

function Fast_plot_Callback(hObject, eventdata, handles)

function Save_Callback(hObject, eventdata, handles)

[filename, path] = uiputfile({'*.csv'}, 'Save as');
guidata(hObject, handles);

function End_Callback(hObject, eventdata, handles)

request = questdlg('Do you really want to quit?', ...
  'Request', ...
 'Yes', 'No', 'No');

switch request
  case 'Yes'
    close Frquency_mapping_tool;
  case 'No'
    beep;
end
guidata(hObject, handles);

function Stop_Callback(hObject, eventdata, handles)
guidata(hObject, handles);

function Data_viewer_Callback(hObject, eventdata, handles)
run Dataviewer;
guidata(hObject, handles);

function Bandmean_Callback(hObject, eventdata, handles)
```

```
function Bandmeantext_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function Peakplot_Callback(hObject, eventdata, handles)
if isempty(handles.D)
  handles.h = msgbox('No data to display. Run the measurement first.', 'No data');
else
  A = handles.D(:, 3);
  X = handles.D(:, 1);
  Y = handles.D(:, 2);

  figure;
  scatter(X, Y, [], A, 'filled');
  grid on;
  colorbar;
  title('Fast scatter plot');
  xlabel('Longitude [Y°]');
  ylabel('Latitude [X°]');
end
guidata(hObject, handles);

function Meanplot_Callback(hObject, eventdata, handles)
if isempty(handles.D)
  handles.h = msgbox('No data to display. Run the measurement first.', 'No data');
else
  A = handles.D(:, 4);
  X = handles.D(:, 1);
  Y = handles.D(:, 2);

  figure;
  scatter(X, Y, [], A, 'filled');
  grid on;
  colorbar;
  title('Fast scatter plot');
  xlabel('Longitude [X°]');
  ylabel('Latitude [Y°]');
end
guidata(hObject, handles);

function Spectrummean_Callback(hObject, eventdata, handles)

function Spectrummean_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function Bandmean_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
```

```
  set(hObject, 'BackgroundColor', 'white');
end

function Clear_Callback(hObject, eventdata, handles)

handles.D = [];
handles.peak = 0;
handles.bp = 0;
handles.sm = 0;
handles.SamplesCount = 0;
handles.samplesaving = 0;
handles.count = 0;

set(handles.Peak, 'String', num2str(handles.peak));
set(handles.Bandmean, 'String', num2str(handles.bp));
set(handles.Spectrummean, 'String', num2str(handles.sm));
set(handles.samplescount, 'String', num2str(handles.SamplesCount));

axes(handles.axes4);
cla reset;
title('Timeline');
grid on;
grid minor;
ylabel('dBm');

axes(handles.axes1);
cla reset;
title('Band detail');
grid on;
grid minor;
ylabel('dBm');

axes(handles.axes2);
cla reset;
title('Spectrum sample');
grid on;
grid minor;
ylabel('dBm');

axes(handles.axes5);
cla reset;
title('Levels');
grid on;

guidata(hObject, handles);

function None_Callback(hObject, eventdata, handles)

function Hanning_Callback(hObject, eventdata, handles)

function Blackman_Callback(hObject, eventdata, handles)

function Hamming_Callback(hObject, eventdata, handles)

function Windowmode_Callback(hObject, eventdata, handles)

function ShowPeaks_Callback(hObject, eventdata, handles)

function Timer_warning_Callback(hObject, eventdata, handles)
```

```
function TimerEnable_Callback(hObject, eventdata, handles)

function Backupsaving_Callback(hObject, eventdata, handles)

function Backupsaving_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function Timelinelimit_Callback(hObject, eventdata, handles)

function Timelinelimit_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function UseGPS_Callback(hObject, eventdata, handles)
```

A-4 Data Viewer script

```
function varargout = Dataviewer(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
   'gui_Singleton', gui_Singleton, ...
   'gui_OpeningFcn', @Dataviewer_OpeningFcn, ...
   'gui_OutputFcn', @Dataviewer_OutputFcn, ...
   'gui_LayoutFcn', [], ...
  'gui_Callback', []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end

function Dataviewer_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;
handles.data_for_plot = [];
set(handles.Map_top, 'String', '51.2353167');
set(handles.Map_bottom, 'String', '48.4134183');
set(handles.Map_left, 'String', '11.9157322');
set(handles.Map_right, 'String', '19.2386350');
handles.filename_map = '';
handles.path_map = '';
handles.D = [];

axes(handles.axes1);
cla reset;
title('Map');
grid on;
grid minor;
axes(handles.axes2);

cla reset;
title('Timeline');
grid on;
grid minor;
guidata(hObject, handles);

function varargout = Dataviewer_OutputFcn(~, eventdata, handles)

varargout{1} = handles.output;

function Untitled_1_Callback(hObject, eventdata, handles)

function Load_Callback(hObject, eventdata, handles)
handles.D = [];
axes(handles.axes1);
cla reset;
title('Map');
grid on;
grid minor;
```

```
axes(handles.axes2);
cla reset;
title('Timeline');
grid on;
grid minor;
[handles.filenamedata, handles.pathnamedata] = uigetfile('*.csv', 'Select a CSV file'
);

handles.D = csvread(fullfile(handles.pathnamedata, handles.filenamedata));
guidata(hObject, handles);

function End_Callback(hObject, eventdata, handles)

request = questdlg('Do you really want to quit?', ...
  'Request', ...
 'Yes', 'No', 'No');

switch request
  case 'Yes'
    close Dataviewer;
  case 'No'
    beep;
end
guidata(hObject, handles);

function Plot_Callback(hObject, eventdata, handles)
set(handles.Status, 'String', 'Loading');
set(handles.Status, 'BackgroundColor', 'yellow');
pause(0.01);
X = handles.D(:, 1);
Y = handles.D(:, 2);
A = handles.D(:, 3);
B = handles.D(:, 4);
C = handles.D(:, 5);
Time = handles.D(:, 6);
clear handles.data_for_plot;
cancelled = 0;
if isempty(handles.path_map)
  I = imread('cz.jpg');
  set(handles.Map_top, 'String', '51.2353167');
  set(handles.Map_bottom, 'String', '48.4134183');
  set(handles.Map_left, 'String', '11.9157322');
  set(handles.Map_right, 'String', '19.2386350');

  handles.top = str2double(get(handles.Map_top, 'String'));
  handles.bottom = str2double(get(handles.Map_bottom, 'String'));
  handles.left = str2double(get(handles.Map_left, 'String'));
  handles.right = str2double(get(handles.Map_right, 'String'));
else
  cancelled = 0;

  try
    I = imread(fullfile(handles.path_map, handles.filename_map));
  catch

    cancelled = 1;
    handles.h = msgbox('No map file choosen', 'Warning');

```

```
    end

    try
        borders = csvread(fullfile(handles.path_borders, handles.filename_borders));
    catch
        cancelled = 1;
        handles.h = msgbox('No map border file choosen', 'Warning');

    end

    if cancelled == 0
        handles.top = borders(1, :);
        handles.bottom = borders(2, :);
        handles.left = borders(3, :);
        handles.right = borders(4, :);

        set(handles.Map_top, 'String', num2str(borders(1, :)));
        set(handles.Map_bottom, 'String', num2str(borders(2, :)));
        set(handles.Map_left, 'String', num2str(borders(3, :)));
        set(handles.Map_right, 'String', num2str(borders(4, :)));
    end

end

if cancelled == 0

    [Resx, Resy] = size(I);
    x = linspace(handles.left, handles.right, Resx);
    y = linspace(handles.top, handles.bottom, Resy);
    handles.xmin = min(x);
    handles.xmax = max(x);
    handles.ymin = min(y);
    handles.ymax = max(y);

    handles.data = get(handles.Data_mode, 'String');
    handles.mode = handles.data{get(handles.Data_mode, 'Value')};

    switch handles.mode
        case 'Peak'
            handles.data_for_plot = A;

        case 'Band mean'
            handles.data_for_plot = B;

        case 'Spectrum mean'
            handles.data_for_plot = C;

        otherwise
            handles.data_for_plot = B;
    end

    checkbox = get(handles.Own_figure, 'Value');

    handles.plotmode = get(handles.plottype, 'String');
    handles.PlotMapType = handles.plotmode{get(handles.plottype, 'Value')};

    switch handles.PlotMapType
        case 'Scatter'

```

```
        if checkbox == 1
            figure(1);
            cla reset;
            imagesc(x, y, I);
            set(gca, 'Ydir', 'normal');
            hold on;
            scatter(X, Y, [], handles.data_for_plot, 'filled');
            xlim([handles.xmin handles.xmax]);
            ylim([handles.ymin handles.ymax]);
            ylabel('Latitude [°]');
            xlabel('Longitude [°]');
            c = colorbar;
            c.Label.String = 'dBm';
            colormap jet;

            figure(2);
            cla reset;
            plot(Time, handles.data_for_plot, 'r');
            title('Timeline');
            grid on;
            grid minor;
            ylabel('dBm');
            hold on;
            datetick('x', 'HH:MM:SS');
            xlim([min(handles.D(:, 6)) max(handles.D(:, 6))]);
            set(handles.Status, 'String', 'Done');
            set(handles.Status, 'BackgroundColor', 'green');

        else
            axes(handles.axes1);
            cla reset;
            imagesc(x, y, I);
            set(gca, 'Ydir', 'normal');
            hold on;
            scatter(X, Y, [], handles.data_for_plot, 'filled');
            xlim([handles.xmin handles.xmax]);
            ylim([handles.ymin handles.ymax]);
            title('Map');
            grid on;
            grid minor;
            ylabel('Latitude [°]');
            xlabel('Longitude [°]');
            c = colorbar;
            c.Label.String = 'dBm';
            colormap jet;

            axes(handles.axes2);
            cla reset;
            plot(Time, handles.data_for_plot, 'r');
            title('Timeline');
            grid on;
            grid minor;
            ylabel('dBm');
            hold on;
            datetick('x', 'HH:MM:SS');
            xlim([min(handles.D(:, 6)) max(handles.D(:, 6))]);
            set(handles.Status, 'String', 'Done');
            set(handles.Status, 'BackgroundColor', 'green');
    end
```

```
case 'Strength map'

    A1 = handles.data_for_plot;
    x1 = X;
    y1 = Y;

    [s, z] = size(A1);
    scale = 1500;
    if s > scale
      n = round(s / scale);
      A = A1(1:n:end, :);
      x2 = x1(1:n:end, :);
      y2 = y1(1:n:end, :);
    else
      A = A1;
      x2 = x1;
      y2 = y1;
    end

    V = A;
    [X1, Y1] = meshgrid(x2, y2);
    f = scatteredInterpolant(x2, y2, V);
    f.Method = 'natural';
    f.ExtrapolationMethod = 'none';
    Z = f(X1, Y1);

    if checkbox == 1
      figure(1);
      cla reset;
      imagesc(x, y, I);
      set(gca, 'Ydir', 'normal');
      hold on;
      contourf(X1, Y1, Z, 'LevelStep', 1, 'LineStyle', 'none');
      c = colorbar;
      c.Label.String = 'dBm';
      colormap jet;
      xlim([handles.xmin handles.xmax]);
      ylim([handles.ymin handles.ymax]);
      ylabel('Latitude [°]');
      xlabel('Longitude [°]');
      hold on;
      imagesc(x, y, I);
      set(gca, 'Ydir', 'normal');
      alpha(.5);

      figure(2);
      cla reset;
      plot(Time, handles.data_for_plot, 'r');
      title('Timeline');
      grid on;
      grid minor;
      ylabel('dBm');
      hold on;
      datetick('x', 'HH:MM:SS');
      xlim([min(handles.D(:, 6)) max(handles.D(:, 6))]);
      set(handles.Status, 'String', 'Done');
      set(handles.Status, 'BackgroundColor', 'green');

    else
```

```
        axes(handles.axes1);
        cla reset;
        hold on;
        imagesc(x, y, I);
        set(gca, 'Ydir', 'normal');
        title('Map');
        grid on;
        grid minor;
        ylabel('Latitude [°]');
        xlabel('Longitude [°]');
        hold on;
        contourf(X1, Y1, Z, 'LevelStep', 1, 'LineStyle', 'none');
        c = colorbar;
        c.Label.String = 'dBm';
        colormap jet;
        xlim([handles.xmin handles.xmax]);
        ylim([handles.ymin handles.ymax]);
        hold on;
        imagesc(x, y, I);
        set(gca, 'Ydir', 'normal');
        alpha(.5);

        axes(handles.axes2);
        cla reset;
        plot(Time, handles.data_for_plot, 'r');
        title('Timeline');
        grid on;
        grid minor;
        ylabel('dBm');
        hold on;
        datetick('x', 'HH:MM:SS');
        xlim([min(handles.D(:, 6)) max(handles.D(:, 6))]);
        set(handles.Status, 'String', 'Done');
        set(handles.Status, 'BackgroundColor', 'green');
      end

    end

 else
    set(handles.Status, 'String', 'Stopped');
    set(handles.Status, 'BackgroundColor', 'red');
 end

 cancelled = 0;
 guidata(hObject, handles);

 function Untitled_2_Callback(hObject, eventdata, handles)

 function Add_Callback(hObject, eventdata, handles)
 [file_name, path_name] = uigetfile('*.csv', 'Select CSV file');
 set(handles.Status, 'String', 'Loading');
 set(handles.Status, 'BackgroundColor', 'yellow');
 handles.D2 = csvread(fullfile(path_name, file_name));
 handles.D = [handles.D;
 handles.D2];
 set(handles.Status, 'String', 'Done');
 set(handles.Status, 'BackgroundColor', 'green');
 guidata(hObject, handles);
```

```
function pushbutton5_Callback(hObject, eventdata, handles)

function pushbutton6_Callback(hObject, eventdata, handles)

function pushbutton7_Callback(hObject, eventdata, handles)

function pushbutton8_Callback(hObject, eventdata, handles)

function Untitled_3_Callback(hObject, eventdata, handles)

function Untitled_4_Callback(hObject, eventdata, handles)

function Untitled_9_Callback(hObject, eventdata, handles)

function Untitled_5_Callback(hObject, eventdata, handles)

function Untitled_12_Callback(hObject, eventdata, handles)

function Untitled_7_Callback(hObject, eventdata, handles)

function Untitled_8_Callback(hObject, eventdata, handles)

function Untitled_10_Callback(hObject, eventdata, handles)

function Untitled_11_Callback(hObject, eventdata, handles)

function Untitled_13_Callback(hObject, eventdata, handles)

function Untitled_15_Callback(hObject, eventdata, handles)

function Map_top_Callback(hObject, eventdata, handles)

function Map_top_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function Map_bottom_Callback(hObject, eventdata, handles)

function Map_bottom_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
  set(hObject, 'BackgroundColor', 'white');
end

function Map_left_Callback(hObject, eventdata, handles)

function Map_left_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
  set(hObject, 'BackgroundColor', 'white');
end
```

```
function Map_right_Callback(hObject, eventdata, handles)

function Map_right_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function load_map_Callback(hObject, eventdata, handles)

[handles.filename_map, handles.path_map] = uigetfile({'*.jpeg'; '*.jpg'; '*.bmp'; '*.png'}, 'Open map file');

[handles.filename_borders, handles.path_borders] = uigetfile({'*.csv'}, 'Open csv borders file');

guidata(hObject, handles);

function save_map_Callback(hObject, eventdata, handles)

[filename, path] = uiputfile({'*.bmp'}, 'Save image as');
axes(handles.axes1);

saveas(gcf, fullfile(path, filename));

guidata(hObject, handles);

function save_timeline_Callback(hObject, eventdata, handles)

[filename, path] = uiputfile({'*.bmp'}, 'Save image as');
axes(handles.axes2);

saveas(gcf, fullfile(path, filename));

guidata(hObject, handles);

function Data_mode_Callback(hObject, eventdata, handles)

function Data_mode_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function Own_figure_Callback(hObject, eventdata, handles)

function plottype_Callback(hObject, eventdata, handles)

function plottype_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end
```