NUI Galway

ANTENNA ALIGNMENT AUGMENTED REALITY VIDEO APPLICATION

Final Year Project

April 2018

Chris Cornwall -12311016
Computer Science & Information Technology

1   ACKNOWLEDGEMENTS

First and foremost, I would like to extend my sincere gratitude to my supervisor, Dr Des Chambers, for his continued guidance throughout the year - your expertise in the area of wireless link transmissions and your willingness to honestly and promptly appraise my work at every stage have been invaluable.

I would also like to thank my classmates for not only participating in the testing phase of this project, but for their unwavering support and friendship throughout the entirety of my degree.

To my other half, Lucia, thank you for the love and support you have shown me, and in particular, for all you have done to help me excel in this past year.

Finally, it would be amiss of me to exclude from these acknowledgements, the people who have made my journey through university possible. To my parents, Albert and Catherine and to my sisters, Caroline and Regina, I am forever indebted to you for all you have done for me. For the love, support (emotional and financial) and inspiration which you have, often unknowingly, provided me, I am eternally grateful.

2 CONTENTS

3   TABLE OF FIGURES

4   GLOSSARY

ISP – Internet Service Provider

AR – Augmented Reality

SDK – Software Development Kit

AREA v2 – Augmented Reality Engine Application version 2

OS – Operating System

POI – Point of Interest

GUI – Graphical User Interface

HTML5 – Hypertext Markup Language 5

PHP – Hypertext Preprocessor

AJAX – Asynchronous JavaScript And XML

JSON – JavaScript Object Notation

5 ABSTRACT

"The goal is to develop a novel smartphone application that uses augmented reality techniques to help with the alignment of highly directional microwave dish antennas. High speed fixed wireless point to point links use very high gain and directional dish type antennas that have often have a very narrow beam-width so precise alignment in both the horizontal and vertical planes can be very difficult, especially when working at height on a telecoms tower or rooftop. The application would be used to provide visual clues to help engineers with the initial alignment of the antenna as well as information on the expected signal strength based on the distance and power budget calculations. The video source could be the camera on a mobile phone or video being streamed live from a GPS enabled drone."

6 INTRODUCTION

In recent years, there has been a steady growth of wireless infrastructure across the globe (Grand View Research, 2018), fixed wireless is now an attractive solution to many businesses and homeowners, who, in previous years, may have had difficulty obtaining and maintaining a quality broadband connection.

To achieve these high speed fixed wireless connections, a signal is transmitted between an ISP and a customer via highly directional microwave antennas. Because these antennas are highly directional, the signal they transmit performs well over relatively large distances. However, as a side effect of this increased power, they must be aligned in both the horizontal and vertical plane to a high degree of accuracy in order for the receiving antenna to obtain an adequate signal.

Typically, when an ISP technician installs a highly directional microwave antenna, they use a specialist piece of equipment, commonly referred to as a spectrum analyser (Electronics Notes, 2017), to read real-time signal data. Whilst this real-time signal data can be used to correctly align the antenna to a high degree of accuracy in both the horizontal and vertical planes, it does require some initial trial and error when first approximating the correct alignment.

Currently, this initial trial and error phase cannot be avoided as there is no solution in today's market which caters for the initial alignment of wireless link antennas. However, there are a number of solutions on the market which aid with the alignment of terrestrial satellites, such as SatFinder (SatFinder, 2016) and SatFinder 3D (SatFinder, 2015). The SatFinder application offers a map view of a user's current location along with values indicating the estimated best alignment, whilst SatFinder3D takes this a step further and offers AR visual cues as well. Although the functionality of SatFinder3D in particular, is similar to the functionality of Align AR (the application described in this report), both of the aforementioned solutions fail to provide the necessary link budget analysis and method of data input (antenna locations and receiver specifications) which can be considered essential whilst performing the initial alignment of wireless link antennas and thus, crucial, in an effort to provide value to an ISP technician.

7   LITERATURE REVIEW

In an effort to conduct the relevant research for this project I have identified the following areas as being particular points of interest:

- Development of an augmented reality android application using an appropriate SDK

- Using sensor data gathered from a smartphone to accurately predict a user's location

- Basic wireless link budget analysis

For my preliminary research, I aimed to explore the topic of augmented reality. According to (Azuma *et al.,* 2001) AR allows for combining or "supplementing" real world objects with virtual objects or superimposed information. As a result, virtual objects seem to coexist in the same space with the real world. Following on from this, I focused on the current level of maturity and adoption rate of augmented reality. To this end, I have consulted Gartner's Hype cycle (*Fig.1*), which of July 2017, concludes that AR will reach the plateau of productivity in the next 5 – 10 years.
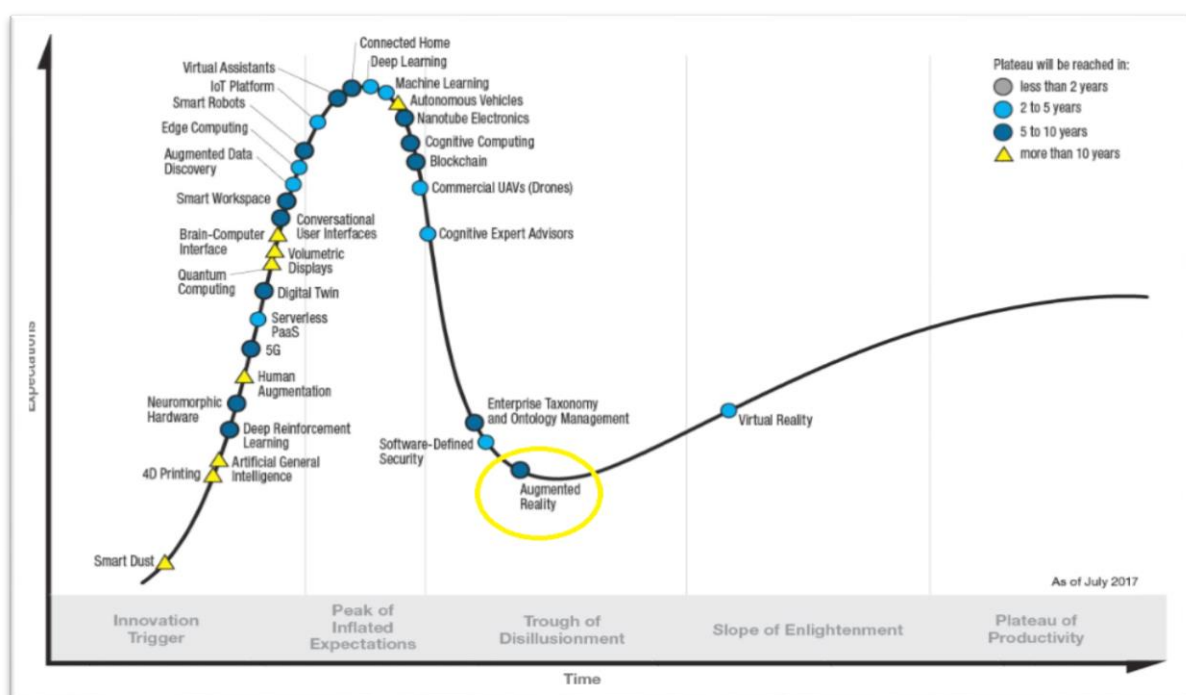

Figure 1 - Gartner's Hype Cycle (Gartner, 2017)

With this being the case, although the development of augmented reality technologies is currently in the "trough of disillusionment", the future looks promising as it is close to entering the "slope of enlightenment".

Whilst augmented reality's place in the gaming world was very much solidified by the success of Pokémon Go (Think Gaming, 2018), applications of AR in other areas such as tourism and engineering, have yet to have the same impact. This leads to the question "how applicable is AR in these areas? ". Specifically, because this project's goal is to apply AR in the realm of engineering but to do so using AR techniques more frequently used in the realm of tourism (i.e. location-based AR), the benefits of using AR in tourism and engineering, if any, must be examined. In 2015, Bernelind conducted an experiment in which users were given the task of navigating a route using augmented reality navigation for one run and using Google Maps for another. The goal was to ascertain which method would be more effective in this scenario. Although the results indicated that users preferred Google Maps, the metrics used to determine this result and the observations made in coming to this conclusion are of note whilst contemplating the design of any location-based augmented reality application.

To gain a better understanding of why users ranked Google Maps with a higher overall usability than the augmented reality alternative, we must first look at the requirements on which to base the statement of usability set out in this paper – user satisfaction, efficiency, effectivity and learnability. When asked questions directly relating to each of these requirements, users indicated that their experience with Google Maps was superior for a number of reasons, including lack of continuous feedback in the AR application, a feeling of "surreptitiously filming people in front" of a user and unstable markers. As my application will be used in a technical environment, the feeling of "surreptitiously filming" does not pose an issue, but the points made about continuous feedback and unstable markers are most certainly of note.

Having completed the preliminary research, I concluded that the most challenging aspect of this project was going to be the work relating to AR. In particular, accurately displaying the position and orientation of an antenna, or indeed any object presents a non-trivial technical challenge which requires the manipulation of mobile sensor, camera and GPS data. With this in mind, I conducted the next segment of my research with an emphasis on the implementation of AR applications and with an aim to uncover the methodologies behind developing a highly accurate and user-friendly AR application.

In order to successfully develop an AR application, it is important to obtain an understanding of how the kernel of such an application may operate. In "The AREA Framework for Location-Based Smart Mobile Augmented Reality Applications" (Pryss *et al.,* 2017), the design and implementation of a mobile AR kernel known as AREA v2 is described in great detail. Alongside this technical description, the challenges faced whilst implementing such a kernel and the solutions that were found are also described.

AREA v2 is a location-based AR kernel, which, like any AR application, faces domain specific challenges, namely: overlapping points of interest, OS-specific sensor data processing and the reliable display of POI markers.

The problem of overlapping POIs is a common theme amongst many research papers on the topic of location-based AR ((Bernelind, 2015), (Pryss *et al.,* 2016), (Pryss *et al.,* 2017)). Many of these papers suggest their own novel solutions to this problem – In 2017, Pryss and his fellow researchers suggested using a clustering algorithm which takes preferences from the user and then clusters points of interest, based on these preferences, into one point which is represented by a marker on the screen. This marker can then be used as a gateway to the POIs contained within it, thus enabling the user to interact with POIs that may previously have been hidden.

With regard to OS-specific sensor data processing with an aim to providing accurate and reliable POI markers, this paper notes that "regarding iOS, sensor data of the gyroscope as well as the accelerometer are used, whereas for Android sensor data of the gyroscope, accelerometer and compass of the mobile device are used to position the virtual 3D camera correctly." Whilst these nuances are naturally handled by any AR SDK and are not necessarily present in every AR engine, it is important to note this difference, as the performance of an application may degrade based on the operating system and set of circumstances at play.

Following on from this, the paper delves into issues involving "jittery" POI markers and possible sources of interference. The authors note that "when using the values of the gyroscope for a user that frequently changes the position of his Android smart mobile device, the POIs on the screen of his smart mobile device oscillate badly". They also note that "the rotation vector provided by the Android mobile OS is very precise on one hand, but it is prone to (1) frequent position changes, (2) slow position changes, and (3) magnetic interference sources on the other". Again, with these issues, some of the above may be handled by the chosen AR SDK but it is nonetheless important to take note of this so as to enable swift identification of such issues throughout the development process.

After completing substantial research in the area of AR, I investigated the alignment of microwave antennas - The installation and alignment of microwave antennas is a complicated process involving a number of technicians and an array of equipment (Exalt, 2011). Typically, to align an antenna, tower crews generate a signal and transmit it to the receiving end at a predefined frequency. Once this basic connection is established, digital-multi-meters (or spectrum analyzers) are used to measure the received signal level. The antenna is then aligned in the horizontal plane until the highest possible received signal level is obtained and then aligned vertically until this same requirement is met. This process is typically repeated a number of times until the highest possible received signal level is obtained (Hassan *et al.,* 2011).

In conjunction with microwave antenna alignment, the topic of basic link budget analysis is also of great importance with regard to this project. Specifically, the signal to noise ratio must be calculated in order to provide a user with an estimate of the power they can expect at the receiving end. To identify the correct formulae for these calculations I have consulted the Maxim Integrated webpage (Maxim, 2011). Maxim Integrated is a Fortune 1000 company specializing in analogue and mixed-signal integrated circuits (Bloomberg, 2018). Their webpage offers a brief description of each of the following formulae and an accompanying spreadsheet which, in relation to this project, is very useful from a testing perspective.

Wavelength

$$\lambda = c/f_0$$

Effective Isotropic Radiated Power

$$EIRP = P_T + G_T$$

Free Space Path Loss (dB)

$$FSPL\ (dB) = 10\ log_{10}(\lambda/4\pi d)^2$$

RX Power Free Space Path

$$P_{RFS} = P_{ChanFS} + G_R - (C\&C\ Loss)$$

Noise Power (at RX)

$$P_n = k\ (T_{ANT} + T_E)\ BW_{RX}$$

Signal to Noise Ratio

$$SNR_{RX} = P_{RX}/P_n$$

Where:

- c = speed of light ($\approx$299792458 m/s)
- P_T = power at transmitter
- G_T = gain at transmitter
- λ = wavelength
- d = distance between receiver and antenna
- P_{ChanFS} = power at RX antenna, free space path
- G_R = Gain at receiver

- f = frequency of transmitter
- (C&C Loss) = cable & connection losses
- k = Boltzmann's constant (1.38^{-23})
- T_{ANT} = antenna temperature
- T_E = effective temperature
- BW_{RX} = receiving bandwidth
- P_{RX} = power at receiver

8   Technology Review

Many technologies have been used throughout this project. Aside from the main programming languages which were used to create the application (Java, Javascript and HTML 5), other common technologies and web development techniques such as jQuery, AJAX, PHP and MySQL were used alongside a less well-known technology – Wikitude SDK. Please see below for a brief summary of each of these technologies alongside an explanation of how they were applied to create this project.

8.1 Java

Java is a "general-purpose, concurrent, strongly typed, class-based object-oriented language" (Oracle, 2018). It is commonly used across a range of applications and is also the native language for Android development. For this project, I have used Java to retrieve and manipulate sensor and GPS data which is then passed to the Wikitude SDK.

8.2 JavaScript

JavaScript is a highly dynamic, interpreted programming language which is mainly used in conjunction with HTML (MDN, 2018). In this project, I have used JavaScript for the creation and handling of AR objects and for the overall management of the GUI.

8.3 HTML 5

HTML 5 is the standard markup language used in web pages across the globe. In relation to my project, HTML was used to render the GUI, which was then manipulated using JavaScript as mentioned above.

8.4 JQUERY

jQuery is a JavaScript library which can be used to simplify the styling and animation of HTML pages (jQuery, 2018). For this project, I used jQuery to create the basic style of the app and also to handle some of the GUI animations.

8.5 AJAX

AJAX is a set of technologies which enable the creation of highly interactive web applications. For this project, AJAX was used in the backend to send and receive data to/from a MySQL database.

8.6 PHP

"PHP is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML." (php.net, 2018). In this project, PHP was used in conjunction with MySQL to allow a user to add to, edit or delete from the antenna database.

8.7 MYSQL

MySQL is a relational database management system which is open-source and widely used. For the purpose of this project, I decided to use a MySQL database to store the locations and properties of each antenna.

8.8 WIKITUDE SDK

The Wikitude SDK is a cross-platform, mobile AR SDK which offers many different forms of AR such as geo-tagging, image-tracking and location-based AR. (Wikitude, 2017). Wikitude is available as a plugin for many platforms and its Javascript API can be used within Android Studio to develop AR apps for Android. The Wikitude SDK creates AR experiences by implementing an "augmented reality JavaScript framework, embedded in an HTML web view which sits on top of the Wikitude camera view and allows developers to control the objects in the camera view" (Sterling, 2011). This framework is known as "ARchitect" and is described in greater detail further on in this report (section 9). Alongside an abundance of awards (Wikitude, 2012) and features (Wikitude, 2015), Wikitude provides a vast array of documentation which has proved invaluable throughout the implementation of this project and which is one of the core reasons why I chose to use Wikitude instead of other SDKs such as ARmedia (AR-media, 2005) or Mixare (Mixare, 2010).

As can be seen in *Fig.2* below, Wikitude offers a Java, Objective-C and JavaScript API which, in conjunction with the appropriate plugins, enables full integration with the Unity, Xamarin, Titanium and Cordova platforms.
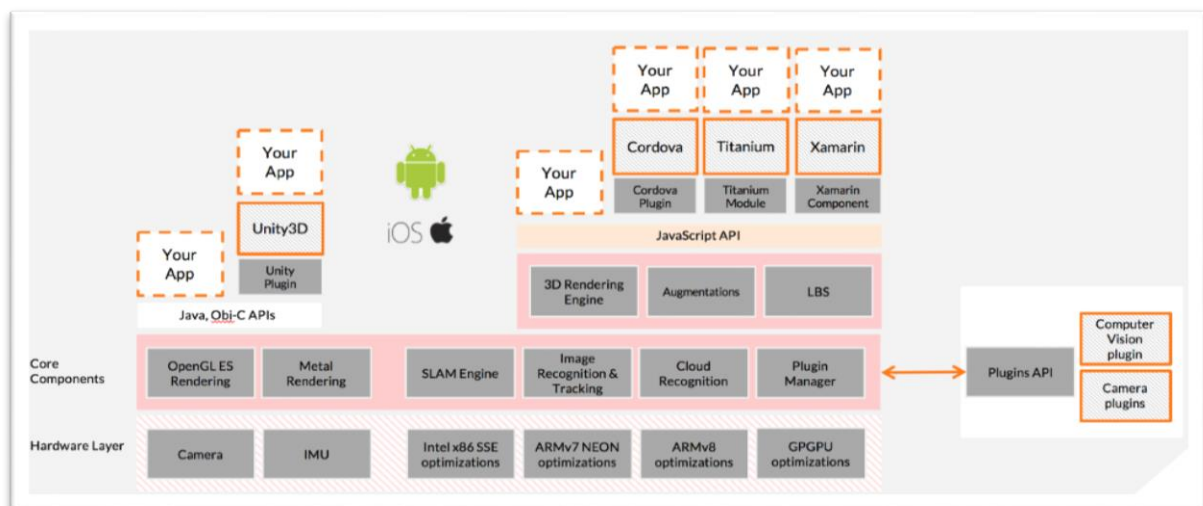

Figure 2 - Wikitude SDK Architecture (Wikitude, 2018)

9   DESIGN

The design phase of any application can be challenging, however it is particularly challenging in the realm of location-based AR. This extra layer of complexity is present because of the issue of overlapping POIs as previously mentioned. Overcoming this challenge and ensuring a high level of usability were my main priorities during the design phase of this project. The following diagrams (*Fig.3*) show my original design:
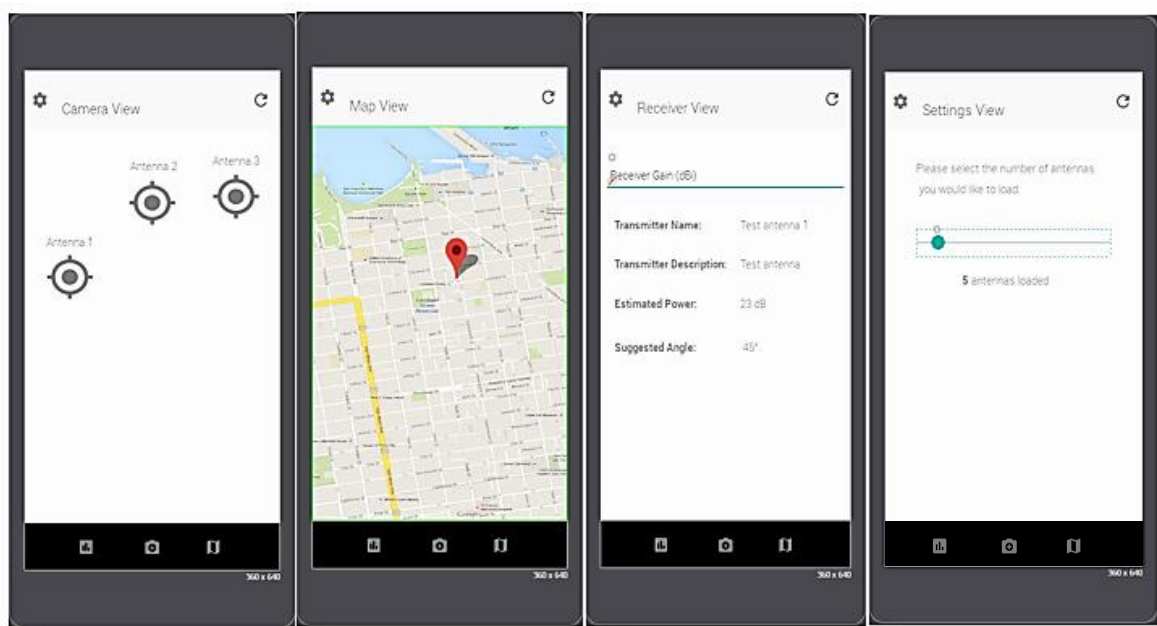

Figure 3 - Original GUI Design

This design is simplistic and very user-friendly as it contains only four screens, each of which can be navigated to using one of the three buttons found along the action bar at the bottom and the settings icon located in the top left corner. The icon in the other corner (refresh symbol) can be used to reload the markers on the screen (as seen in the camera view). This is all very intuitive and flows relatively well. However, after more consideration, I concluded that four completely different screens were unnecessary and that the above design could be condensed into one screen with multiple overlays - partially, or fully covering the camera view, as needs be.

In this new design (*Fig.4*), I have chosen to keep the same camera view as seen in *Fig.3* and to remove the navigational buttons previously found along the bottom.
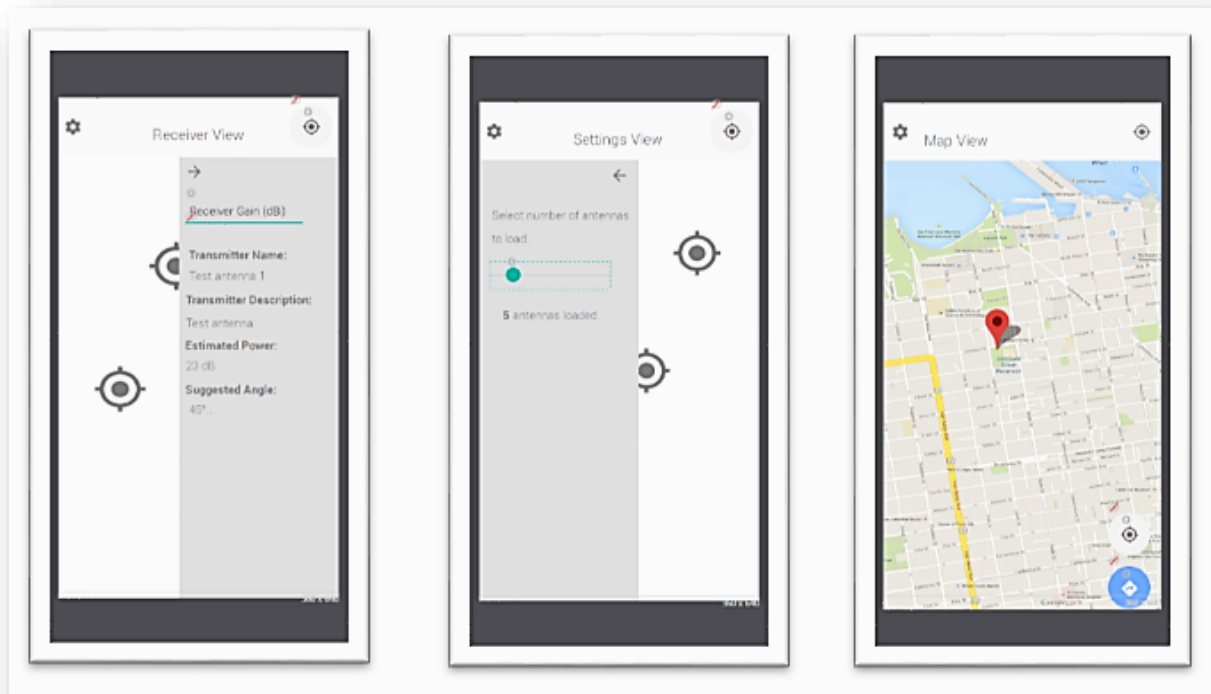

Figure 4 - Final GUI Design

Now, users can input and see signal data simply by clicking on the desired marker, which will display an overlay where data can be input/viewed. Similarly, a user can click the gear icon to display an overlay where the number of antennas loaded onto the screen can be adjusted. The map view remains much the same in this design but is accessed through the location icon on the top right of the screen which then displays a map overlay, covering the entire screen. In an effort to simplify the design, I decided to remove the reload button and instead, let any click on the screen, which is not on a button or marker, cause a reload to happen. By adapting the original design with these modifications I have fully utilized all available space and have condensed the application into one screen.

This application consists of a core, written in Java, which uses the Wikitude JavaScript API to render AR geo-objects to the screen.

In MainActivity.java the basis of the app is created, on which everything else is built. MainActivity.java gathers the relevant sensor data and also creates an "architectView". This "architectView" is part of the Wikitude Javascript API and is used to create a camera surface and to handle all sensor events (Wikitude, 2013).

Originally, I began using Unity to develop this project as it offers a whole host of UI features which are particularly suited to drawing high-quality graphics on the screen. Naturally, before beginning any development, I ensured that Wikitude supported a Unity plugin, however, I later discovered that this plugin has limited functionality and does not support Wikitude's location-based AR features. For this reason, I was forced to decide between using Unity with a different AR SDK or to revert to Android Studio and continue using the Wikitude plugin. After analyzing the other AR SDK alternatives, I decided to revert to Android Studio and continue using Wikitude. In the end, this decision was a relatively simple one as Wikitude's location-based AR is significantly more mature than other offerings and the documentation they provide is second to none.

Another major adjustment which I made early on in the development process is the use of Google Play Services fused location provider. Fused location provider is a "location API in Google Play services that intelligently combines different signals" to provide highly accurate and power efficient location information (Google, 2017). This location information is then passed to the "architectView" as latitude, longitude, altitude and accuracy.

Originally I used Android's own location service to facilitate this functionality but after experiencing significant inaccuracies and very poor power usage statistics I decided to switch to Google's fused location provider which has proved to be far more accurate and efficient.

Apart from gathering sensor data and creating the main activity, the Java code itself provides no front-end functionality to the user. The entire front-end experience is built in JavaScript and HTML, in what is commonly referred to as, the "ARchitect World". See *Fig.5* below for a complete end-to-end flow of how each component in the application interacts with each other:
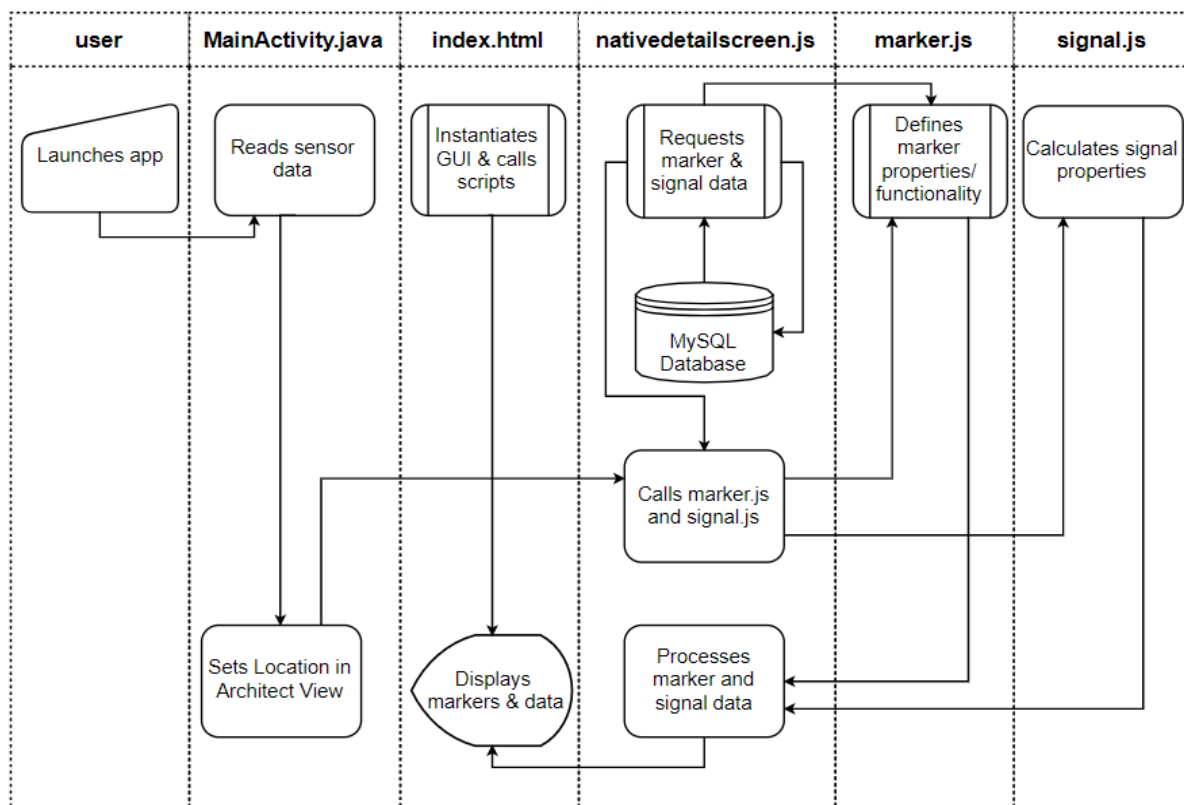

Figure 5 - Architectural Flow

My ARchitect World is closely modeled off of the Wikitude example entitled "Browsing POIs" (Wikitude, 2013). In the example, the GUI is built in HTML and is manipulated using JavaScript. Likewise, my application is built using the same technique, but with a number of extra features as will be described below.

In index.html, a transparent HTML page is created over the camera view using jQuery's "mobile-transparent-ui-overlay.css". From this starting point, each JavaScript file is loaded, a breakdown of the core functionality of each of these is as follows:

10.1 NATIVEDETAILSCREEN.JS

Most of the front end display and user interaction is handled in nativedetailscreen.js, while specific marker, radar and signal properties/functionality are handled in their own scripts. In this script, a "world" variable is created which defines the AR world and the functions that control it. These functions directly influence the GUI as seen by the user and include functionality to update the values of markers as users move between locations, change the number of markers being displayed based on a user's preference and handle the GUI elements as panels are opened and closed. Some of the other functionality provided in nativedetailscreen.js is as follows:

10.1.1 Loading POI's from JSON data

As the name suggests, the "loadPoisFromJsonData" function loads POIs from a JSON file stored on an NUIG server. Each POI contains information about a given antenna, including its latitude, longitude and elevation. Using these details, a new marker is created and added to the world's marker list.

```
for (var currentPlaceNr = 0; currentPlaceNr < poiData.length; currentPlaceNr++) {

    var singlePoi = {
        "id": poiData[currentPlaceNr].id,
        "latitude": parseFloat(poiData[currentPlaceNr].latitude),
        "longitude": parseFloat(poiData[currentPlaceNr].longitude),
        "altitude": parseFloat(poiData[currentPlaceNr].altitude),
        "power": parseFloat(poiData[currentPlaceNr].power),
        "gain": parseFloat(poiData[currentPlaceNr].gain),
        "azimuth": parseFloat(poiData[currentPlaceNr].azimuth),
        "elevation":  parseFloat(poiData[currentPlaceNr].altitude),
        "frequency": parseFloat(poiData[currentPlaceNr].frequency),
        "title": poiData[currentPlaceNr].name,
        "description": poiData[currentPlaceNr].description
    };

    World.markerList.push(new Marker(singlePoi));
}
```

Figure 6 - POI JSON Data

10.1.2 Hiding Markers

The "hideMarkers" function is called when a user clicks on any marker. It loops through the "markerList" seen in *Fig.6* on the previous page and hides each one which is not currently selected thus allowing a user to focus on one antenna without unnecessary markers getting in the way. This is done using Wikitude's animation functions which are described in greater detail further on in this report (Section 9.2.2).

10.1.3 Map Initialization

The "initMap" function is used to initialize the Google Maps API. A map object is created and centered at the user's current location. Once this is done, the "initMapMarkers" function loops through the "markerList", pulls out the latitude, longitude and name of each antenna location and places the appropriate markers and labels on the map.

10.2 MARKER.JS

"marker.js" defines the properties and functionality of each individual marker. This includes the creation of geo-objects, the creation and animation of marker drawables and the setting of on-click triggers. A more detailed explanation of the creation of geo-objects, along with the creation and animation of marker drawables is as follows:

10.2.1 Creation of Geo-objects

"A GeoObject represents a virtual object bound to specific locations in the earth's 3-dimensional space" (Wikitude, 2012). In "marker.js", a geo-object is created for each antenna location consisting of a JSON object holding each possible marker drawable for that geo-object, an indicator which directs a user to its position on the screen and a radar point which represents the position of the geo-object in the radar.

10.2.2 Creation and Animation of Marker Drawables

In "marker.js" each drawable (i.e. each marker image) is assigned to its own marker drawable object which includes the image itself, properties of that image and the on-click trigger for that marker. In order to smoothly transition from one drawable

to the next (e.g. change marker colours etc.), Wikitude provides an animation group class. As can be seen in *Fig.7* below, after creating a number of animation properties, an animation group can be created by first specifying whether to run the animations in parallel or sequence and then passing an array of animation properties to this animation group. In the evaluation section (section 10), the changing of marker colours based on expected SNR is discussed - this functionality is achieved using the methodology described here.

```javascript
if (marker.animationGroup_idle === null) {

    // create AR.PropertyAnimation that animates the opacity to 1.0 in order to
    show the orange animation
    var showOrangeAnimation = new
        AR.PropertyAnimation(marker.markerDrawable_idle_orange, "opacity", null,
        1.0, kMarker_AnimationDuration_ChangeDrawable);
    // create AR.PropertyAnimation that animates the opacity to 0.0 in order to
    hide unnecessary drawables
    var hideSelectedDrawableAnimation = new
        AR.PropertyAnimation(marker.markerDrawable_selected, "opacity", null, 0,
        kMarker_AnimationDuration_ChangeDrawable);

    var hideIdleDrawableAnimation = new
        AR.PropertyAnimation(marker.markerDrawable_idle, "opacity", null, 0.0,
        kMarker_AnimationDuration_ChangeDrawable);

    var hideGreenAnimation = new
        AR.PropertyAnimation(marker.markerDrawable_idle_green, "opacity", null,
        0.0, kMarker_AnimationDuration_ChangeDrawable);

    // More properties defined here…

    // create AR.PropertyAnimation that animates the scaling of the title label
    var titleLabelResizeAnimationY = new AR.PropertyAnimation(marker.titleLabel,
        'scale.y', null, 1.2, kMarker_AnimationDuration_Resize, new
        AR.EasingCurve(AR.CONST.EASING_CURVE_TYPE.EASE_OUT_ELASTIC, {
          amplitude: 2.0
        }));

    marker.animationGroup_idle = new
        AR.AnimationGroup(AR.CONST.ANIMATION_GROUP_TYPE.PARALLEL,
        [hideIdleDrawableAnimation, hideGreenAnimation, showOrangeAnimation,
        hideSelectedDrawableAnimation, idleDrawableResizeAnimationX,
        selectedDrawableResizeAnimationX, titleLabelResizeAnimationX,
        idleDrawableResizeAnimationY, selectedDrawableResizeAnimationY,
        titleLabelResizeAnimationY]);
}
// Note: Code has been condensed to fit in report
```

Figure 7 - Creation and Animation of Marker Drawables Code Snippet

10.3 SIGNAL.JS

The "signal.js" script is where all the necessary calculations are performed in order to calculate a number of properties for each signal being transmitted. A breakdown of the main functionality of signal.js is as follows:

10.3.1 Link Budget Analysis

As can be seen in *Fig.8* below, to calculate the link budget between the receiver and transmitter, signal.js reads the power (dBm), frequency (MHz) and gain (dBi) of the transmitter and performs a link budget analysis using the formulae described in the literature review above. It is important to note, that whilst the formulae used in this implementation are the same as the aforementioned formulae, unit conversion is also taking place and thus modifications have been made to accommodate this.

```
// Function to calculate estimated Signal to noise ratio
function Signal(poiData, distance) {
    // If user has entered specs, read them in
    if (typeof parseInt(localStorage.getItem("rxRecGain")) == "number")
        {rxRecGain = parseInt(localStorage.getItem("rxRecGain"));}

    if (typeof parseInt(localStorage.getItem("rxConLoss"))== "number")
        {rxConLoss = localStorage.getItem("rxConLoss");}

    if (typeof parseInt(localStorage.getItem("rxCableLoss"))== "number")
        {rxCableLoss = localStorage.getItem("rxCableLoss");}
    var
        poiData = poiData;
        distance = distance;
        angle = getAngle(poiData, distance);
        // Transmitter frequency, power and gain all grabbed from input json
        frequency = poiData.frequency * 1000; // MHz
        tPower = poiData.power; // dBm
        tGain = poiData.gain; // dBi
        lightSpeed = 299792458; // m/s
        wavelength = lightSpeed/(frequency*1000000); // m
        fspl = 20*Math.log10((wavelength/(4*Math.PI*distance))); // Free space path
        loss in dB
        eirp = tPower + tGain; // Effective isotropic radiated power in dBm
        powerAtRec = eirp + fspl; // Power at receiver in dB
        rxPowerFSPL = powerAtRec + rxRecGain + rxConLoss + rxCableLoss; // Plus
        gains and losses
        // Noise at RX receiver
        boltzConst = 1.38e-23; // J/K
        rxNoiseFig = 7; // dB
        opTemp = 290; // K
        antennaTemp = 300; // K
        recBandwidth = 20; // MHz
        effectNoiseTemp = opTemp*(10^(rxNoiseFig/10)-1); //K
        rxNoisePower = 10*Math.log10(boltzConst*(antennaTemp+effectNoiseTemp)*
            (recBandwidth*1000000))+30;
        SNR = rxPowerFSPL - rxNoisePower;

        properties = {
            "rpower": powerAtRec,
            "snr": SNR,
            "angle": angle};

    return properties;
} // Note: Code has been condensed to fit in report
```

Figure 8 - Link Budget Analysis Code Snippet

10.3.2 Recommended Angle

Using the user's elevation, the transmitter's elevation and the distance between these two points, the recommended elevation angle (angle in the vertical plane) is calculated using simple trigonometry. Whilst this may be considered an over-simplified solution, given that the aim is only to provide an estimated best angle, I believe this to be sufficient.

10.3.3 Sector Filtering

In the event that a given mast has a number of sectors, each with different azimuths (i.e. antennas facing different directions), it is important to identify which sector is intended to transmit a signal to the antenna at the user's location. To enable this functionality, I have created a sector filtering function which is part of nativedetailscreen.js and which uses a function found in Signal.js, called "findCoord". "findCoord" first uses the location and azimuth of the transmitter, along with the distance from the user to calculate the co-ordinates of a point on a line originating from the transmitter, faced towards the azimuth, at the user's distance. In *Fig.9* below, if we take "I" to be the position of the ISP's mast, "U" to be the position of the user, and 1, 2, 3 and 4 to be antennas on this mast each with azimuths as stated, then "findCoord" calculates the co-ordinates of points A, B, C and D. Once these co-ordinates are calculated, they are passed into the sector filtering function which hides all but one of the markers - the markers corresponding to the points furthest away from the user are hidden (in this case, the markers representing antennas 2, 3 and 4). The implementation of "findCoord" can be seen in *Fig.10* on the next page.
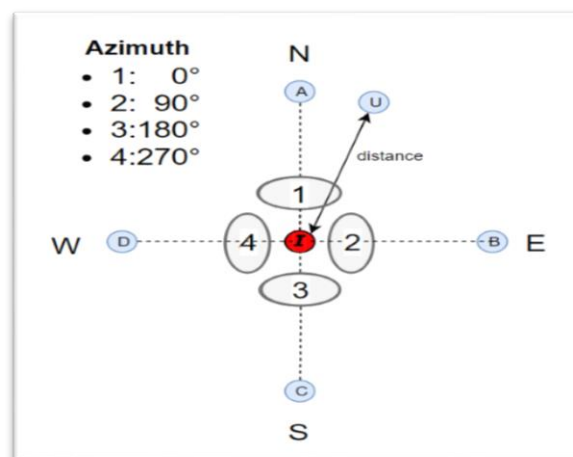

Figure 9 – Diagram of Sector Filtering Algorithm

```
function findCoord(poiData, distance){

    var
        radius = 6371e3, // radius of the earth in meters
        angDist = Number(distance) / radius, // angular distance in radians
        azimuth = Number(poiData.azimuth).toRad();
        latitude = poiData.latitude.toRad(),
        longitude = poiData.longitude.toRad();

    var newLatitude = Math.asin(Math.sin(latitude)*Math.cos(angDist) +
        Math.cos(latitude)*Math.sin(angDist)*Math.cos(azimuth));

    var newLongitude = longitude + Math.atan2(Math.sin(azimuth)
        *Math.sin(angDist)*Math.cos(latitude), Math.cos(angDist)-
        Math.sin(latitude)*Math.sin(newLatitude));

    newLongitude = (newLongitude+3*Math.PI) % (2*Math.PI) - Math.PI; // normalise
    to -180..+180°

    var coords = {
        "latitude": newLatitude.toDeg(),
        "longitude": newLongitude.toDeg()
        };

    return coords;
}
```

Figure 10 - "findCoord" Function Code Snippet

10.4 RADAR.JS

"radar.js" is a simple script which handles the creation and animation of the radar. Aside from initializing the radar, the images used to create it and its position are also defined.

10.5 SERVER-SIDE

In order to provide a useful service for ISPs, I have added a simple GUI which can be used to input the location and properties of antennas. The page itself is written in HTML and JavaScript and interacts with a MySQL database using PHP and AJAX.

As can be seen in *Fig.25* (section 10.4), this GUI consists of a simple "live table" which allows the user to view, edit and add antenna information conveniently.

Each interaction with the table is parsed into a MySQL query and sent to a database using AJAX and PHP. In addition to this, each time the table is edited, a select query is used to retrieve the entire contents of the database which is then piped into a JSON file which is stored on an NUIG server. From here, the Android application is able to read the contents of the file and create the appropriate geo-objects as explained above.

In order to provide some level of authentication to this backend, I have added a simple login screen which queries a user database using PHP. I have also used the Google login API, which allows a user to authenticate with their Google account credentials. This adds a convenient login method for a user and also allows me to keep track of who has logged in and at what time via the Google developer console, thus enabling me to keep track of the changes a particular user has made.

11 TESTING & EVALUATION

11.1 TESTING

I have tested this application in the areas of performance, accuracy and usability. The following is a brief summary of the procedures performed and the results obtained in each of these categories (the results obtained from each of these tests are discussed in section 11.2).

11.1.1 Performance

In order to test the performance of this application, I installed it on two devices. The first device is an LTE Axion 7 Mini smartphone running Android version 6.0.1 and the second is a Lenovo TB-8504F tablet running Android version 7.1.1. To test the performance of the application on each of these, I downloaded System Panel 2 (SystemPanel2, 2018) on both devices, killed all unnecessary processes on each device and performed a typical use-case in the application for approximately five minutes at a time. I then performed these same steps for the Snapchat application, as this application offers similar functionality (AR, map view, camera etc.) and so is an appropriate application to draw comparisons with. The following are the results obtained:
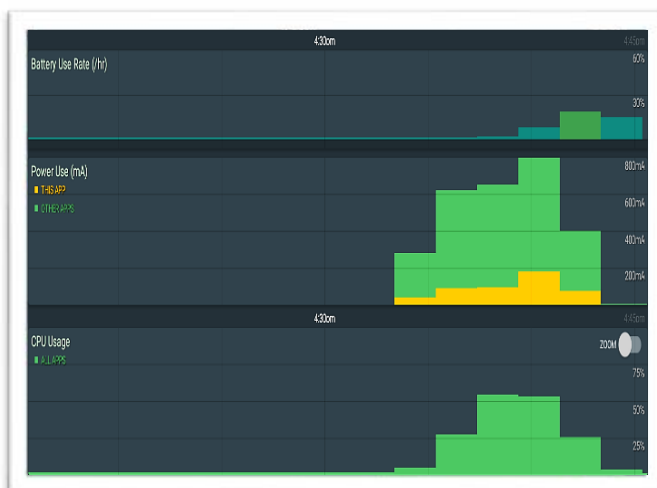

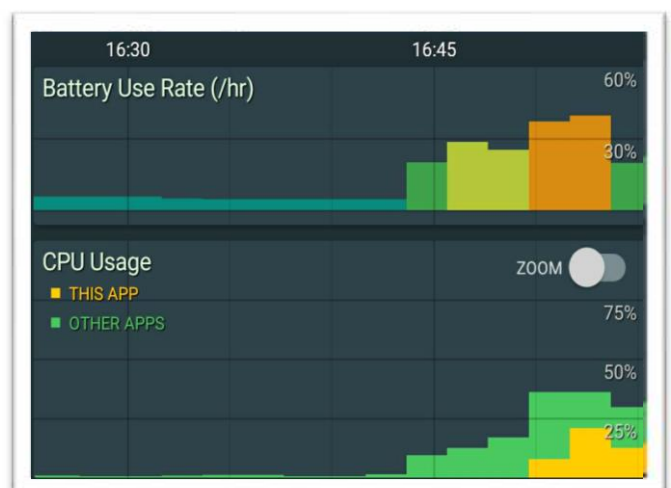*Figure 11 (a) – Tablet: Battery and CPU Usage for Align AR*

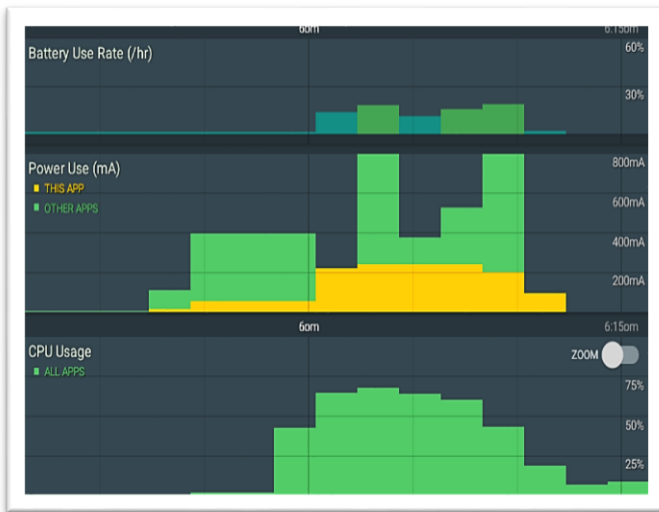*Figure 11 (b) – Phone: Battery and CPU Usage for Align AR*

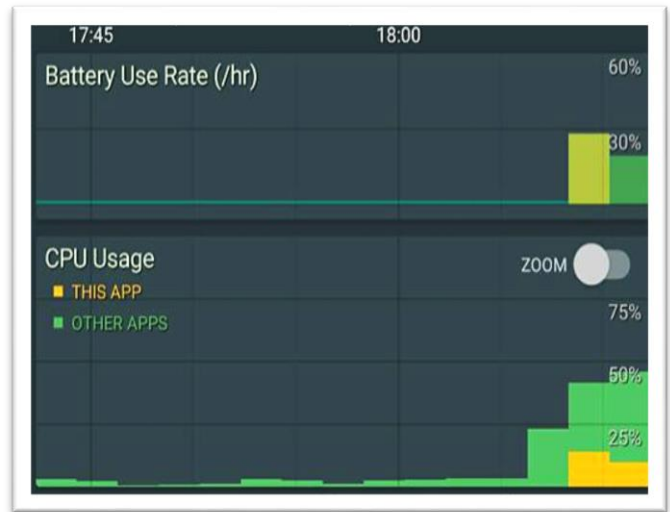*Figure 11 (c) - Tablet: battery & CPU Usage for Snapchat*

Figure 11 (d) - Phone: Battery & CPU Usage for Snapchat

11.1.2 Accuracy

In measuring the accuracy of this system, there are two primary concerns: The accuracy of the markers representing antenna locations and the accuracy of the link budget analysis used to determine the estimated received power.

To test the accuracy of the markers representing antenna locations, it is important to note whether or not the distance which the application presents to the user is accurate and also whether or not the markers are indeed, directing the user in the correct direction. To test the distance, I simply compared the distance presented in the application with the distance calculated by Google Maps. To test the direction, I inputted some fictitious antenna locations which were actually locations of easily visible landmarks around my location (The Galway Cathedral and The NUIG Quadrangle). I was then able to easily compare the direction the markers suggested with the visible landmarks. It is worth noting that whilst this test proved that the markers are accurate to a certain degree, to properly test this, I would need to use the application to align an antenna and then use a spectrum analyzer to measure the received signal.
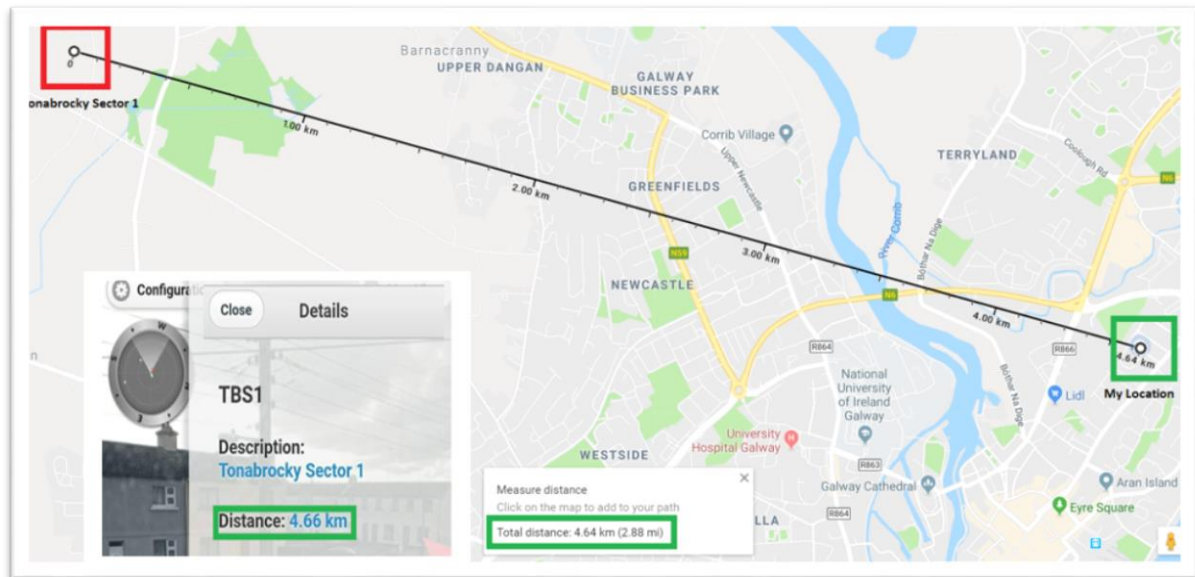
Figure 12 - Comparison of Distance Provided by Google Maps with Distance Provided by Align AR

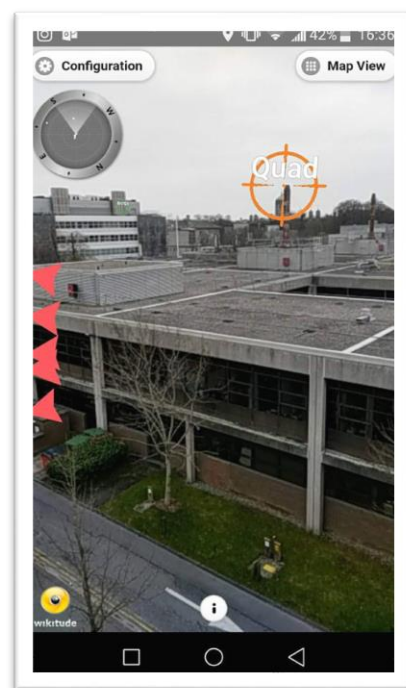*Figure 13 – Marker Indicating Location of Cathedral*

Figure 14 - Marker Indicating Location of Quadrangle

Again, without access to a spectrum analyser it is hard to determine just how accurate the link budget analysis in this application is. However, using the Excel sheet provided by Maxim integrated (Maxim, 2011) I have been able to compare the results presented in the application with those obtained by entering the same distance, frequency etc. into the Excel sheet. This has provided me with confidence in my link budget analysis as Maxim Integrated, being a Fortune 1000 company specialising in this area, can be considered a reliable source.

MAXIM
RF System (Link Budget) Calculations

System Variables	Variable	Units	Equation	Value
Frequency	f_0	MHz		5700
Speed of Light	c	m/s		299792458
Wavelength	λ	m	$λ = c/f_0$	0.052595168

Block	Variable	Units	Equation	Value
PA Power	P_{PA}	dBm		22
TX Match Loss	L_{MatchT}	dB		
TX source	P_{TX}	dBm		22
TX connector loss	L_{ConT1}	dB	(from Connector Loss sheet)	-1
TX cable loss	L_{CabT}	dB	(from Cable Loss sheet)	-1
TX connector loss (remote antenna)	L_{ConT2}	dB	(from Connector Loss sheet)	
TX power	P_T	dBm	$P_T = P_{TX}$(C&C Loss)	20
TX antenna gain	G_T	dBi		17
Effective (Isotropic) Radiated Power	EIRP	dBm	$EIRP = P_T G_T$	37
Distance	d	m		4660
Channel Medium Loss Factor	L_0	dB	(from Medium Loss sheet)	0
Free Space Loss	L_{FS}	dB	$L_{FS} = (λ/4πd)^2$	-120.9329987
Power at RX Antenna, Free Space Path	P_{ChanFS}	dB	$P_{ChanFS} = L_{FS}L_0 EIRP$	-83.93299867
Flat Earth Loss (Includes Ground Bounce)	L_{FE}	dB	(from Ground Multipath sheet)	-149.7466884
Multipath Loss	L_{MP}	dB		
Obstruction Loss	$L_{Obs-Total}$	dB		0
Power at RX Antenna, Flat Earth Path	P_{ChanFE}	dB	$P_{ChanFE} = L_{FE}L_0 L_{MP}L_{Obs} EIRP$	-112.7466884
RX antenna gain	G_R	dBi		15
RX connector loss	L_{ConR1}	dB		-1
RX cable loss	L_{CabR}	dB		
RX connector loss (remote antenna)	L_{ConR2}	dB		
RX power, Free Space Path	P_{RFS}	dBm	$P_{RFS} = P_{ChanFS}G_R$(C&C Loss)	-69.93299867
RX power, Flat Earth Path	P_{RFE}	dBm	$P_{RFE} = P_{ChanFE}G_R$(C&C Loss)	-98.74668843

Receiver Sensitivity Calculations	Variable	Units	Equation	Value
RX Noise Figure	NF	dB		7
Operating Temperature	T_0	K		290
Effective Noise Temperature	T_e	K	$T_e = T_0(NF - 1)$	1163.442978
Boltzmann's constant	k	J/K		1.38E-23
Receive Bandwidth	BW_{RX}	MHz		20
Antenna Temperature	T_{Ant}	K		300
Noise Power (at RX)	P_n	dBm	$P_n = k(T_{Ant} + T_e)BW_{RX}$	-93.93715113
Signal to Noise Ratio	SNRRX	dB	SNRRX = PRX/Pn	23.60758241

Figure 16 – Align AR: Expected SNR

Figure 15 – Maxim Integrated: Link Budget Analysis

11.1.3 Usability

The final aspect of the application which I tested is its usability. As usability is largely subjective, I concluded that the most appropriate way to test the application's usability was to get a number of people to use the application whilst following instructions, before filling out a survey. To do this, I asked five classmates to simulate aligning an antenna using the application and gave each of them the following instructions:

(1) Open the app, and enter a gain of 15 dBi for the antenna you are aligning
(2) Locate the marker labelled TBS1, then read out its expected power
(3) Locate the marker labelled TMS1, then read out its estimated angle
(4) Locate the same marker in the map view
(5) Finally, filter out markers which are further away so that only five markers are being displayed

I then asked each user to rate, on a scale of 1-5 with 5 being completely true and 1 being completely false, how true they felt each statement was after using the application. These statements were:

(1) I entered the gain value with little, to no hassle.
(2) Locating an AR marker was simple and intuitive.
(3) Finding the signal strength of this marker was also simple and intuitive
(4) I refreshed the screen with little to no effort
(5) Locating a marker in the map view was simple and intuitive
(6) Finding and using the filtering function required minimal navigational effort
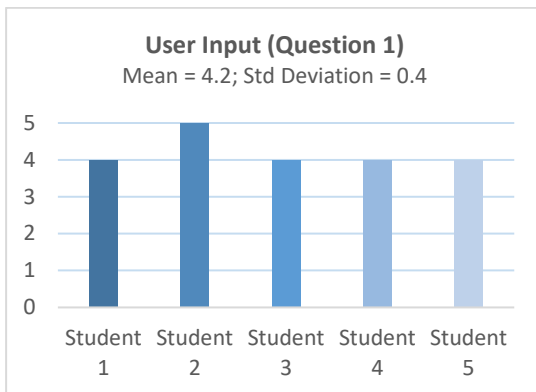
The following are the results of this survey:
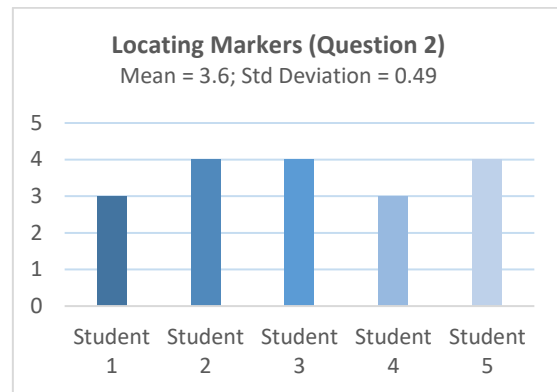

Figure 17 (a) – Question 1 Results
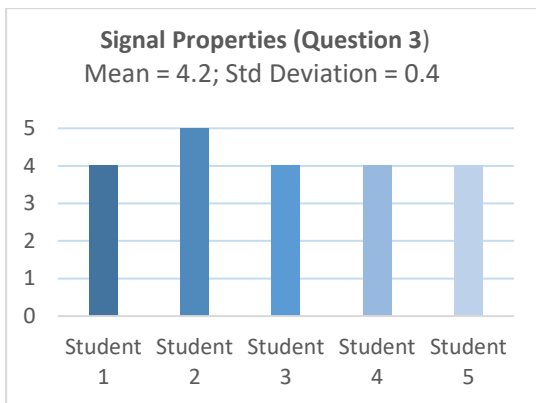

Figure 17 (b) – Question 2 Results

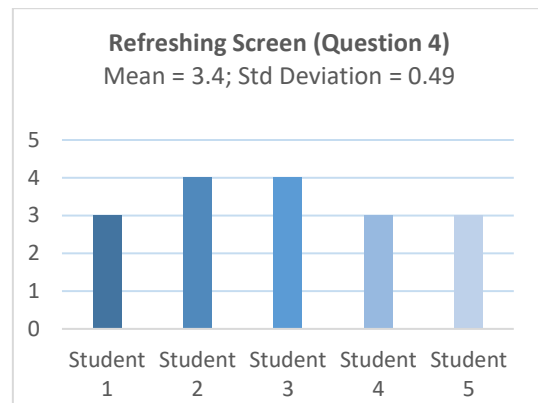*Figure 17 (c) – Question 3 Results*

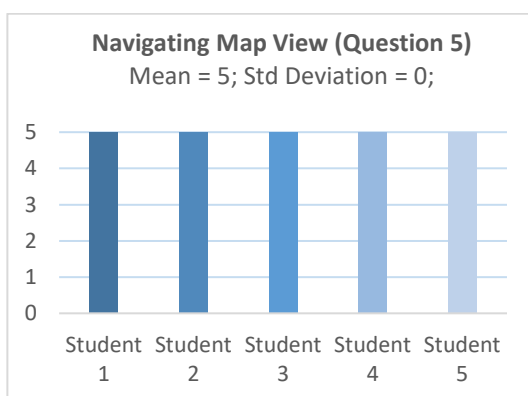*Figure 17 (d) – Question 4 Results*
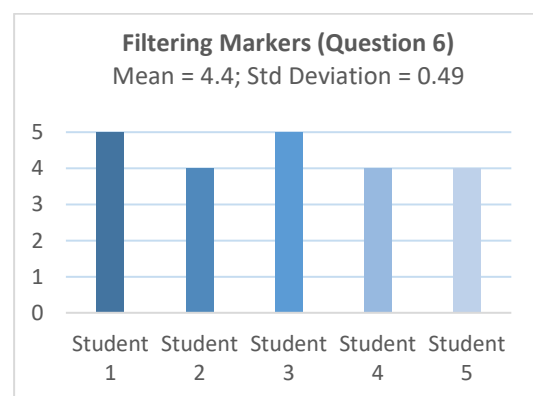

Figure 17 (e) – Question 5 Results

Figure 17 (f) – Question 6 Results

11.2 EVALUATION

The following evaluation is divided into four segments:

- Evaluation of test results
- Evaluation of functionality
- Evaluation of primary features
- Evaluation of secondary features

11.2.1 Evaluation of Test Results

11.2.1.1 Performance

In *Fig.11 (a)* and *Fig.11 (b)* we can see that the application performed relatively similar on both devices. The peak battery usage was between 25% - 40% per hour and the peak CPU usage for all running applications was between 40% and 60%. Comparing this with Snapchat's peak battery usage of between 25%-30% and peak CPU usage of between 45% - 75% (*Fig.11 (c)* and *Fig.11 (d)*), I have concluded the application performs relatively well against these metrics. Moreover, as the distribution of battery usage and CPU usage was similar to that of Snapchat's, I also consider the application to be stable in this regard.

11.2.1.2 Accuracy

In *Fig.12* we see that the distance between my current location and the chosen antenna is reported as 4.66 Km in the application and is reported as 4.64 Km in Google Maps. In this scenario, the application appears to be over 99% accurate if we take Google Maps as being reliable. Also, this discrepancy could be accounted for by a change in altitudes between the two points which is accommodated in the application, but not in Google Maps.

In *Fig.13* and *Fig.14* we see that the antenna markers appear to be accurately indicating the correct locations. Whilst testing this, I did notice that the markers may not initially be incredibly accurate, particularly if a user is moving. However, when the device was made stationary these markers calibrated within a matter of seconds.

11.2.1.3 Usability

As can be seen from the results in *Fig.17 (a-f),* the entire application has scored relatively well throughout. However, two areas of concern have been highlighted from this testing. Firstly, users scored question two on locating markers with a mean score of 3.6, which indicates that there may be some more work to be done in helping users find the markers they require – one option is to add audio cues along with the visual cues already provided, another option may be to increase the size of the radar. Secondly, users scored question four on refreshing the screen with a mean score of 3.4, which is somewhat expected as this functionality is not explicitly explained within the app nor is there an icon which indicates such functionality. However, on questioning the users after filling out the survey, many of them commented that their first reaction when unable to reload the markers was to just tap the screen which may indicate that this functionality is actually intuitive but possibly slightly more time-consuming when first using the application. A potential improvement in this area would be to offer a short description of how the application works when a user first opens it. It is also worth noting, that as this survey was only taken by five people, these results may fluctuate significantly given a larger sample size.

11.2.2 Evaluation of Functionality

In this section, I provide screenshots of the finished application and backend alongside a short description of the functionality. I then evaluate the application with respect to the requirements initially set out in the project definition document. Following on from this, I present an evaluation of the project with respect to the supplementary requirements agreed upon by my supervisor and I.
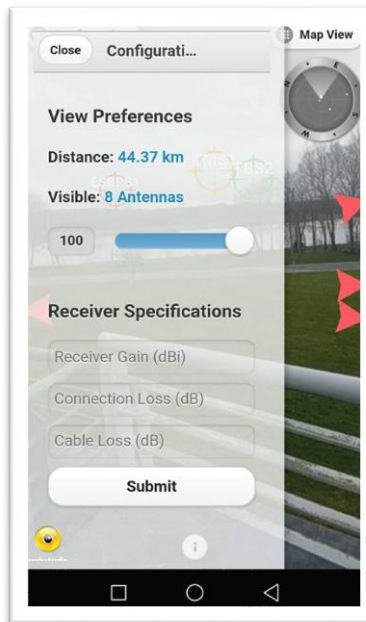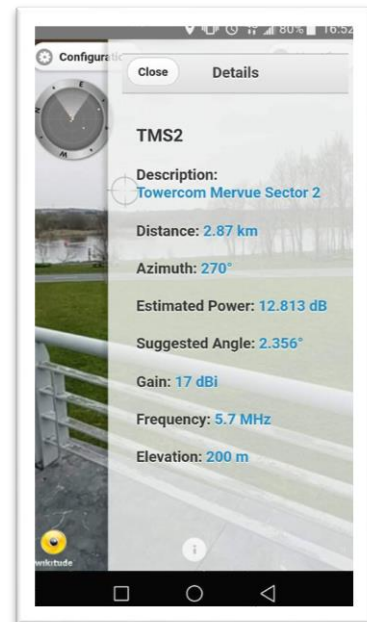

Figure 18 – Main Screen   Figure 19 – Configuration Screen   Figure 20 – Antenna Properties Screen

Fig.18 depicts the first screen a user is presented with after the splash screen (which is the Wikitude logo as I am using a student version of the SDK). Here we can see the different coloured markers representing antenna locations and strength (red = weak; orange = medium; green = strong), red pointers indicating the presence of other antennas in the vicinity, a radar in the top left corner of the screen and navigational buttons, also along the top.

The next figure (*Fig.19*) depicts the configuration overlay which, as you can see, offers the user functionality to change the number of antennas being displayed via the slider and to input receiver antenna specifications via the short form.

Fig.20 depicts the screen presented to a user after clicking on one of the antenna markers. Here, information about the chosen antenna is displayed to the user including signal frequency, expected signal to noise ratio and recommended angle.

Also, *Fig.20* only shows the selected marker on the screen and hides all unnecessary markers.
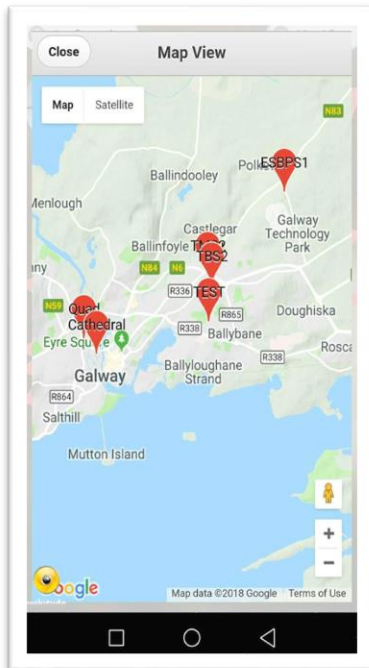

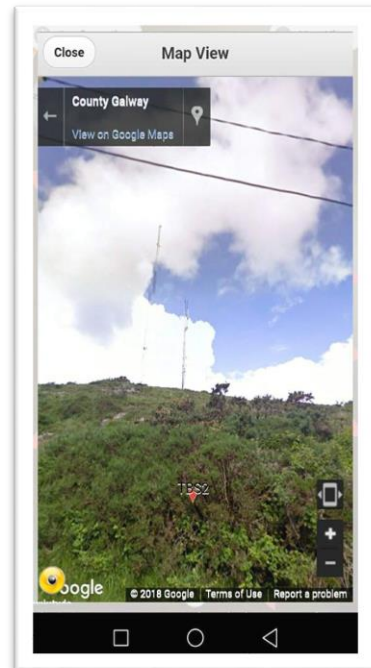Figure 21 – Map View   Figure 22 – Satellite View   Figure 23 – Street View

In *Fig.21*, we see the map view screen which shows the location of antennas. In *Figs.22* and *Fig.23* we can see how this map view can be used to examine a satellite image and street view of the desired antenna location.

Figure 24 – Backend Login

In *Fig.24* we see a simple login screen which includes a Google login option. Once logged in, a user is directed to the screen seen below (*Fig.25*) where they can edit any property value by simply clicking and typing and can add or delete antenna information by utilizing the red and green buttons.

Id	Name	Description	Latitude	Longitude	Altitude	Power	Gain	Frequency	Azimuth	Delete
84	ESBPS1	ESB Parkmore Sector 1	53.306255	-8.99122	300	21	15	5.7	90	x
85	TMS1	Towercom Mervue Sector 1	53.289853	-9.018182	300	21	15	5.7	90	x
86	BVS1	Ballyvaughen Sector 1	53.110971	-9.187289	400	24	18	6.2	120	x
87	RTES1	RTE Maghera Sector 1	52.980163	-8.719998	350	25	16	5.8	120	x
88	TBS2	Tonabrocky Sector 2	53.2915	-9.110929	250	22	17	5.7	90	x
89	TMS2	Towercom Mervue Sector 2	53.289853	-9.018182	200	22	17	5.7	270	x
90	TEST	dasfdasf	53.281	-9.018	200	22	15	5	180	x
92	Cathedral	Galway Cathedral	53.2745921	-9.0578653	100	22	15	5.7	180	x
93	Quad	Quadrangle	53.2776039	-9.0621904	100	22	15	5.7	180	x
										+

Figure 25 – Live Table of Antenna Properties

11.2.3 Evaluation of Primary Features

The application should be compatible with all android devices where a recent version of android is installed

Align AR has been built with Android SDK version 15 (Android 4.0.3, 4.0.4) as its minimum compatible version and has been extensively tested with Android SDK version 23 (Android 6.0, 6.0.1). The app has been tested on several different versions of Android, ranging from Android 5.0 up to the new Android 8.1 and yields satisfactory performance on all. The app has also been tested on a range of different devices including both new and old Android phones and tablets and has not experienced any major issues with performance or with the GUI.

The application should have read-only access to a database containing the GPS co-ordinates and other relevant information for transmitting antennas

This requirement makes up a key part of the functionality of the application. As has been described in section 9, in order to draw each marker on the screen and calculate the necessary link budget, a JSON file is read from a secure server, which in turn, gathers its information from a MySQL database. The access which the app has to this JSON file is strictly read-only and as the app only interacts with this file and not the database itself, a possible security hole is avoided. A possible improvement to this would be to store the JSON file locally and have it sync with the server whenever an internet connection becomes available. In this way, antennas could be loaded even when there is no internet connection.

The application should be capable of using this data to calculate the necessary link budget

This requirement has also been completed and somewhat extended – Additionally, the app offers the user the opportunity to input not just the receiving antenna gain value, but also the connection loss and cable loss if they so wish. This offers a higher level of accuracy for the user.

Using the device's camera, the application should display markers on the screen indicating the location of the transmitting antennas

Again, this requirement makes up one of the core functionalities of the app and is also a requirement which I have expanded on, as will be explained further on.

To help with the alignment of the antenna, the application should provide visual and/or audio cues to the user indicating the best vertical and horizontal alignment and the expected signal strength when aligned

The markers on the screen indicate the position of each transmitting antenna in both the horizontal and vertical axis. This enables the user to align the antenna visually before viewing the antenna details overlay which tells the user more about the transmitting antenna's exact position and suggests a suitable elevation angle for the receiving antenna. Again, this feature could be expanded upon. By using the gyroscope in the device, the angle at which the device is being tilted could be calculated and visual cues indicating the correct elevation angle could be displayed to the user.

11.2.4 Evaluation of Secondary Features

During the course or this project, my supervisor and I agreed upon a number of supplementary requirements which are not set out in the project definition document. My evaluation of the project deliverables implementation of these requirements is as follows:

A backend should be provided so that in a real-world scenario, ISPs can easily add, edit and delete antenna details on the fly.

Originally, the location and properties of each antenna were stored locally on the user's device. This was convenient for development purposes but did not make the app very user friendly as to edit the details of any antenna involved manually editing a JSON file on a user's device. To combat this, I have developed a backend (described in section 9) which authenticates a user and allows them to easily manage their database of antenna locations and properties. As mentioned above, this architecture could be improved to offer a synced local version of the file to a user. Another improvement to this would be to offer a map view on the backend, where a user could simply click to add an antenna location, rather than entering the co-ordinates manually.

The application should be able to filter out sectors of a given mast so that only the appropriate antennas are displayed, taking into consideration the azimuth of each antenna and the user's location.

This feature is important when taking into account how the app may be used in a real-world scenario. Typically, an ISP will have a number of antennas located on the one mast. Because of this, my supervisor and I decided to include a sector filter feature. Whilst I am content with the performance of this feature, I acknowledge that the algorithm used to filter out unwanted antennas (See section 9.3.3) could be improved. Currently, if the centre of the beam area of an antenna is closer to the user than the centre of the beam area of any other antenna on the same mast, then only that antenna's marker will be displayed. This is sufficient for filtering out any unwanted antennas, however, an improvement on this would be to model the transmitting signal's azimuth and elevation pattern and thus determine whether or not it is possible for the user to receive a signal from the chosen antenna.

Functionality should be provided to filter the number of markers displayed on the screen based on the distance between the user and these markers.

As previously mentioned, a common problem with geo-location AR applications is the overlapping of markers on the screen. To avoid this problem, a common solution is to filter out unwanted markers using some property. In this application, I have provided a simple slider to the GUI (Fig.20) which allows a user to select the number of markers they would like displayed on the screen. As the user moves the slider to the left, markers which are the greatest distance away are removed, thus leaving it easier for a user to view and select appropriate markers. As an ISP may have several antennas all over the country, I believe this is a very useful feature which, whilst relatively simple to implement, greatly improves a user's experience within the application.

A "map view" should be added so that the location of each antenna can be easily viewed by a user.

This feature is more of a "nice-to-have" rather than a necessity, however, it does allow the user to view both a map and satellite view of their surrounding areas which, in turn, allows them to verify the location of each antenna to ensure that the markers, possibly inputted to the system by someone else, represent the true locations of the antennas.

12 CONCLUSION

Having developed, tested and evaluated this application, I feel confident in its ability to act as a convenient tool for any ISP technician wishing to install an antenna. I am content with its accuracy and performance and feel that because of the easy form of data input the backend GUI provides along with the link budget analysis also provided, it could indeed be used in a commercial setting. Looking forward, I would like to add some features to this product as mentioned above. Additionally, I endeavour to carry out more thorough testing across a number of devices. Prior to this, I aim to make the application cross-platform and will work to ensure that performance, accuracy and usability remain high across each of these platforms.

To conclude, through the completion of this project I have accumulated a great deal of knowledge in the field of AR Android application development and highly directional microwave antenna alignment. My critical thinking and problem solving skills have been enhanced and my skill set expanded.

13 BIBLIOGRAPHY

[1] AR-media. (2005). Products. Available: http://www.armedia.it/products.php. [Last accessed 8th April 2018]

[2] Azuma, R. et al., (2001). Recent advances in augmented reality. Computer Graphics and Applications, IEEE, 21(6), 34-47.

[3] Bloomberg. (2018). Company Overview of Maxim Integrated Products, Inc.. Available: https://www.bloomberg.com/research/stocks/private/snapshot.asp?privcapId=95673. [Last accessed 13 April 2018]

[4] Bruce Sterling. (2011). Augmented Reality: Wikitude ARchitect.Available: https://www.wired.com/2011/05/augmented-reality-wikitude-architect/. [Last accessed 10th April 2018]

[5] Electronics Notes. (2017). What is a Spectrum Analyzer: RF spectrum analyzer. Available: http://www.radio-electronics.com/info/t_and_m/spectrum_analyser/spectrum_analyzer.php. [Last accessed 8th April 2018]

[6] Exalt (2011). "Technical White Paper Microwave Fundamentals Series Antenna Alignment for Terrestrial Microwave Systems".

[7] Google. (2017). Simple, battery-efficient location API for Android. Available: https://developers.google.com/location-context/fused-location-provider/. [Last accessed 8th April 2018]

[8] Grand View Research. (2018). Wireless Infrastructure Market Size, Share & Trends Analysis Report By Technology (Macrocell RAN, Small Cells, RRH, DAS, Cloud RAN, Carrier Wi-Fi, Mobile Core, Backhaul), And Segment Forecasts, 2018 - 2025. Available: https://www.grandviewresearch.com/industry-analysis/wireless-infrastructure-market. [Last accessed 8th April 2018]

[9] Hassan et al., (2011). Automated Micro-Wave(MW) Antenna Alignment of Base Transceiver Stations Time Optimal Link Alignment.

[10] jQuery. (2013). What is jQuery?. Available: http://jquery.com/. [Last accessed 8th April 2018]

[11] Maxim Integrated. (2011). Radio Link-Budget Calculations for ISM-RF Products. Available: https://www.maximintegrated.com/en/app-notes/index.mvp/id/5142. [Last accessed 6th April 2018]

[12] MDN Web Docs. (2018). JavaScript basics. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction. [Last accessed 8th April 2018]

[13] Mixare. (2010). mixare – Open Source Augmented Reality Engine.Available: http://www.mixare.org/. [Last accessed 14th April 2018]

[14] Oracle.(2012). Java™ Programming Language. Available: https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html. [Last accessed 5th April 2018]

[15] PHP. (2009). What is PHP?. Available: http://php.net/manual/en/intro-whatis.php. [Last accessed 7th April 2018]

[16] Pryss et al.. (2016). Advanced Algorithms for Location-Based Smart Mobile Augmented Reality Applications. Procedia Computer Science 94 . 94 (1), 97-104

[17] Pryss et al., (2017). The AREA Framework for Location-Based Smart Mobile Augmented Reality Applications. Journal of Ubiquitous Systems & Pervasive Networks. 9 (1), 13-21.

[18] Bernelind, S. (2015). AN EXPERIMENTAL STUDY COMPARING NAVIGATION IN AUGMENTED REALITY AGAINST ONLINE STANDARDIZED MAPS. Navigation in Augmented Reality. 1-29.

[19] SatFinder Smart Mobile Android Store Application. (2016). Available: https://play.google.com/store/apps/details?id=com.esys.satfinder&hl=en_GB. [Last accessed 6th April 2018]

[20] SatFinder 3D Smart Mobile Android Store Application. (2015). Available: https://play.google.com/store/apps/details?id=us.kvasha.satfinder3dlite&hl=en. [Last accessed 6th April 2018]

[21]Think Gaming. (2018). Pokémon Go. Available: https://thinkgaming.com/app-sales-data/130634/pokemon-go/. [Last accessed 7th April 2018]

[22] Wikipedia.(2006). Maxim Integrated. Available: https://en.wikipedia.org/wiki/Maxim_Integrated. [Last accessed 8th April 2018]

[23] Wikitude. (2012). Awards . Available: https://www.wikitude.com/about/awards/. [Last accessed 10th April 2018]

[24] Wikitude. (2012). GeoObject Class. Available: http://www.wikitude.com/external/doc/documentation/5.1/Reference/JavaScript%20API/classes/GeoObject.html. [Last accessed 13 April 2018]

[25] Wikitude. (2013). Setup Guide Android. Available: https://www.wikitude.com/external/doc/documentation/latest/android/setupguideandroid.html. [Last accessed 5th April 2018]

[26] Wikitude. (2013). Browsing POIs. Available: https://www.wikitude.com/external/doc/documentation/5.1/android/browsingpois.html. [Last accessed 7th April 2018]

[27] Wikitude. (2015). SDK Full Features. Available: https://www.wikitude.com/products/wikitude-sdk-features/. [Last accessed 10th April 2018]

[28] Wikitude. (2017). Wikitude Augmented Reality SDK. Available: https://www.wikitude.com/. [Last accessed 5th April 2018]