

Navigation Assistant: An Investigation into Obstacle Avoidance for the Visually Impaired using Raspberry Pi

Interim Report

DT282

BSc in Computer Science (International)

**Jennifer Nolan**

**C16517636**

**Bianca Schoen-Phelan**

School of Computer Science

Technological University, Dublin

**09 December 2019**

Abstract

The number of people with visual impairments or blindness is rising. On average 1 in 30 Europeans experience some form of sight loss (1). This means there is an estimated 30 million blind and partially sighted people in Europe. In 2016, there were 54,810 who are blind or visually impaired in Ireland (2). According to 2019 records, there are 2.2 billion people globally with some form of visual impairment or blindness (3). These statistics include the various eye conditions like diabetic retinopathy.

A common misconception is that the blind cannot see anything. While this may be true for some, others can see light and shadow, blurs, have tunnel vision or lack central vision (1). The goal of this project is to introduce a new technology for the visually impaired and blind, that will act as an additional aid to the traditional aids like guide dogs and white canes, to help them navigate both the inside and outside world independently. The project will be called Navigation Assistant. It will be a program run on the Raspberry Pi, as it is a lightweight and portable medium for the problem being solved. The system, which can be attached to the user using a belt or Velcro, will involve real-time processing to support the user’s real-time navigation and obstacle avoidance needs.

Existing research and projects show that there are many benefits to this type of assistive technology for visually impaired and blind people including allowing them to navigate independently (4). Navigation Assistant will be an additional aid to more traditional aids like the white cane or guide dog.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

**A close up of a logo

Description automatically generated**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Jennifer Nolan

09 December 2019

Acknowledgements

I would first like to thank my project supervisor Bianca Schoen-Phelan for her help with this project so far. Her advice and guidance during the course of this project has been greatly appreciated.

I would also like to say a big thank you to my family and friends who have supported me through this project so far. I would also like to thank them for supporting me throughout my entire time in college especially in this final year.

Table of Contents

[1. Introduction 9](#_Toc24901293)

[1.1. Project Background 9](#_Toc24901294)

[1.2. Project Description 9](#_Toc24901295)

[1.3. Project Aims and Objectives 11](#_Toc24901296)

[1.4. Project Scope 12](#_Toc24901297)

[1.5. Thesis Roadmap 12](#_Toc24901298)

[2. Literature Review 14](#_Toc24901304)

[2.1. Introduction 14](#_Toc24901305)

[2.2. Alternative Existing Solutions to Your Problem 14](#_Toc24901306)

[2.3. Technologies you’ve researched 18](#_Toc24901307)

[2.4. Other Research you’ve done 22](#_Toc24901308)

[2.5. Existing Final Year Projects 28](#_Toc24901309)

[2.6. Conclusions 29](#_Toc24901310)

[3. Prototype Design 31](#_Toc24901311)

[3.1 Introduction 31](#_Toc24901312)

[3.2. Software Methodology 31](#_Toc24901313)

[3.3. Overview of System 34](#_Toc24901314)

[3.4. Front-End 35](#_Toc24901315)

[3.5. Middle-Tier 42](#_Toc24901316)

[3.6. Back-End 43](#_Toc24901317)

[3.7. Conclusions 44](#_Toc24901318)

[4. Prototype Development 45](#_Toc24901319)

[4.1. Introduction 45](#_Toc24901320)

[4.2. Prototype Development 45](#_Toc24901321)

[4.3. Front-End 51](#_Toc24901322)

[4.4. Middle-Tier 51](#_Toc24901323)

[4.5. Back-End 52](#_Toc24901324)

[4.6. Conclusions 52](#_Toc24901325)

[5. Testing and Evaluation 53](#_Toc24901326)

[5.1. Introduction 53](#_Toc24901327)

[5.2. Plan for Testing 53](#_Toc24901328)

[5.2.1 Test Plan 53](#_Toc24901329)

[5.3. Plan for Evaluation 54](#_Toc24901330)

[5.4. Conclusions 55](#_Toc24901331)

[6. Issues and Future Work 56](#_Toc24901332)

[6.1. Introduction 56](#_Toc24901333)

[6.2. Issues and Risks 56](#_Toc24901334)

[6.3. Plans and Future Work 57](#_Toc24901335)

[6.3.1. GANTT Chart 58](#_Toc24901336)

[Bibliography 59](#_Toc24901337)

Table of Figures

[Figure 1: Example of possible wearable solution 10](#_Toc26621553)

[Figure 2: MiniGuide 15](#_Toc26621554)

[Figure 3: The Ray Electronic Mobility Guide 16](#_Toc26621555)

[Figure 4: A Raspberry Pi 3 Model B 19](#_Toc26621556)

[Figure 5: The Camera Module V2 20](#_Toc26621557)

[Figure 6: Example of instance segmentation 24](#_Toc26621558)

[Figure 7: Convolutional Neural Network (CNN) layers 25](#_Toc26621559)

[Figure 8: The Agile Methodology 31](#_Toc26621560)

[Figure 9: The Waterfall Methodology 32](#_Toc26621561)

[Figure 10: The Feature Driven Methodology 33](#_Toc26621562)

[Figure 11: System Layers 35](#_Toc26621563)

[Figure 12: Example of making Raspberry Pi wearable 36](#_Toc26621564)

[Figure 13: Use Case Iteration 1 37](#_Toc26621565)

[Figure 14: Sequence Diagram Iteration 1 38](#_Toc26621566)

[Figure 15: Use Case Iteration 2 39](#_Toc26621567)

[Figure 16: Sequence Diagram Iteration 2 40](#_Toc26621568)

[Figure 17: Use Case Iteration 3 41](#_Toc26621569)

[Figure 18: Sequence Diagram Iteration 3 42](#_Toc26621570)

[Figure 19: Example of instance segmentation generates from model on Raspberry Pi 43](#_Toc26621571)

[Figure 20: Required libraries imported 46](#_Toc26621572)

[Figure 21: Import the model 47](#_Toc26621573)

[Figure 22: Set up and configure the model 47](#_Toc26621574)

[Figure 23: Create the model object 48](#_Toc26621575)

[Figure 24: Code used to test that the model gives correct result 48](#_Toc26621576)

[Figure 25: Result from testing Mask R-CNN model 49](#_Toc26621577)

[Figure 26: Code to run Mask R-CNN model on Pi 50](#_Toc26621578)

[Figure 27: Output from prototype after object detection 51](#_Toc26621579)

[Figure 28: Gantt Chart 58](#_Toc26621580)

Table of Tables

[Table 1: Table showing features of each project 18](#_Toc25511405)

[Table 2: Table describing each R-CNN algorithm 27](#_Toc25511406)

[Table 3: Table of requirement importance 30](#_Toc25511407)

[Table 4: Table of test plan 54](#_Toc25511408)

Link to Demo Videos

<https://drive.google.com/open?id=1qKxxeY1TJAYGabZ8pWz1U8YVBPRSdtkg>

# 1. Introduction

## Project Background

The number of people with visual impairments or blindness is rising. Approximately 1 in every 30 Europeans have some form of visual impairment or sight loss (1). In total there are about 30 million Europeans who are blind or partially sighted. Records show that in 2016 approximately 54,810 people in Ireland had partial sight or were blind (2). As well as that, in 2019 it was found that an estimated 2.2 billion people in the world were either blind or visually impaired (3). These statistics include the various eye conditions in the world for example cataracts and glaucoma.

Technology has been able to aid visually impaired and blind users in their navigation of the inside and outside world. There are products, for example the MiniGuide, and projects currently available to do aid these users in their navigation. These products and projects are referred to later in the background section of this report. Using technology as a means of navigation can have many benefits for the blind and visually impaired. By simply attaching the hardware component of the intended system to their waist, the user can navigate with the help of the device in a hand free manner, allowing them to also use the more traditional aids like a guide dog or white cane. It could possibly allow for taking public transport alone, navigating around indoor and outdoor environments without assistance, travelling independently, either locally or in unknown towns or areas, and completing everyday task independently like shopping, going to work and visiting friends and family.

Although there are vast amounts of information and research products about assistive aids for the blind and visually impaired none have become a staple in the lives of the visually impaired and blind. Different technologies are being created to aid the independent living of the visually impaired and blind. It is for these reasons that Navigation Assistant is being developed.

## Project Description

Navigation Assistant is a project for the visually impaired or blind that allows them to navigate and avoid obstacles in their everyday life. Navigation Assistant should be used in conjunction with the more traditional navigational aids of the visually impaired and blind, like the white cane or guide dogs. This project will be used as an additional aid in giving a visually impaired or blind person more independence while navigating around various environments.

The complexity of Navigation Assistant comes in the accuracy of the instructions being given to the user, using audio instructions being delivered through an audio device like headphones. This project is a one-way system where the navigation instructions are provided to the user through audio, but the user cannot converse with the system, however this could be considered with regards to the future work of the project. Creating accurate navigation instructions to the user are paramount in ensuring that the user feels safe when navigating with this device. If the instruction is inaccurate it could lead to harm to the user. As well as being accurate, the instructions need to be intuitive and clear to ensure a good user experience. If the instructions are easy to understand and clear then the user will feel they can complete the instructions safely.

The approach for this project will consist of the creation of prototypes. This prototype will be a wearable device attached to the user. It will be either attached to the user using Velcro or attached to a belt, similar to the prototype shown in Figure 1. These prototypes will then be tested by various users to gain knowledge on their experiences. These user experiences and feedback will then be incorporated into changes made to the project. These steps will be repeated until a robust and thought out project is developed. This project will be a feature driven development project, once a feature is completed and tested, it will be integrated into the project.

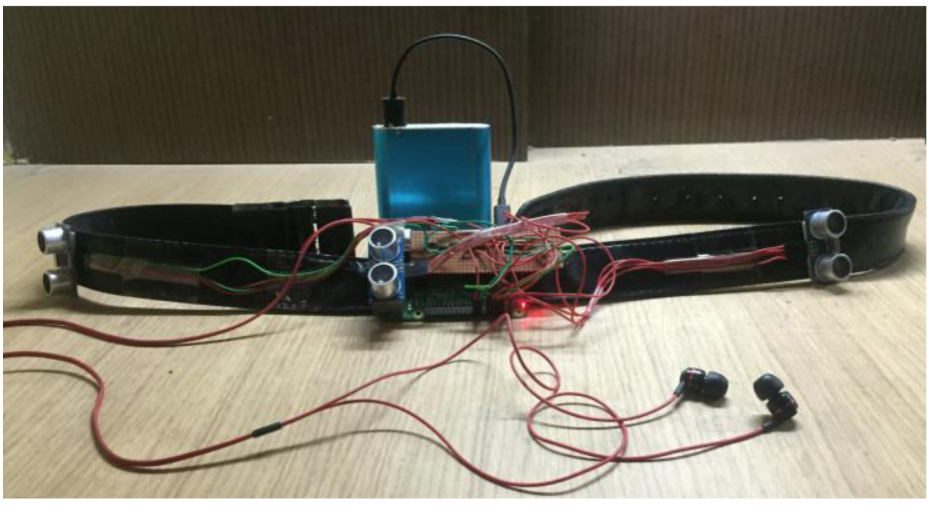


Figure 1: Example of possible wearable solution (11)

## Project Aims and Objectives

The overall aim of the project is to produce a project that allows for accurate navigational instructions, to avoid upcoming obstacles, to be given to the visually impaired or blind user.

The goal of this project is to provide an intuitive and easy to understand instructions to the user that will allow them to navigate an unfamiliar environment safely. The development of this project will be hugely user centred meaning that the type of user that will use this system will be kept in mind throughout the development of this project. As this project’s aim is to help the user navigate their environment it is essential that the projects research, design and development be implemented with the user in mind. The user and their experience will be the main concern throughout the whole process of this project.

To achieve the goals in this project milestones, presented in Chapter 6 of this report, have been set, and planning has been implemented. The milestones involved setting, and possibly changing, dates to complete specific features of the project. The planning, presented in chapter 6 of this report, involves planning ahead and setting out what needs to be done with the project to complete it. These both ensure that the final project version is completes as best as possible at the end.

The purpose of this project is to help the visually impaired or blind user navigate obstacles in their everyday life with a wearable device. This project will also show that technology can be used in a helpful and assistive way for those who have a disability, in this case are visually impaired or blind.

What Navigation Assistant is not about is replacing more traditional methods of navigation for the visually impaired and blind, i.e. the white cane and guide dogs. Navigation Assistant is to be used in conjunction with these more traditional methods as an additional navigational aid. Even if Navigation Assistant doesn’t work or suit every user’s needs, for example the fact that it will eventually run out of power and must be recharged, it will still be considered a success if it helps a handful of people’s lives.

## Project Scope

While Navigation Assistant is about aiding the visually impaired in their everyday navigation, it is not about replacing more traditional methods of navigation that are used like the white cane or guide dogs. Navigation Assistant is designed to be used in parallel with these more traditional methods as an additional navigational aid. While Navigation Assistant is meant to be used to aid visually impaired or blind users it may not suit all user’s lifestyles. However, even if Navigation Assistant helps a handful of visually impaired or blind users in their everyday navigation then it will still be considered a success.

## Thesis Roadmap

This section will provide a summary of each of the chapters covered in this report.

Research

This chapter explores the background research related to being visually impaired or blind, exploring some of the different visual impediments that can be diagnosed and the use of technology in helping people who are visually impaired or blind. Following on from this, an examination into products for the visually impaired or blind that are currently available on the market and other research projects that have been conducted. Finally, in this chapter any other relevant research required for this project will be discussed.

### Design

This chapter investigates the methodology chosen for this project, how the choice was made and the other methodologies that were considered for this project. After this, detailed use-cases related to the purposed system will be presented. Lastly in this chapter, the technical architecture of the system will be discussed.

### Development

This chapter examine and breaks down the entire development process of the project. This chapter focuses on the development of the technical architecture outlined in the design chapter. The challenges encountered during the development of this project will also be included in this chapter.

### Testing and Evaluation

This chapter will focus on how the testing and evaluation of the system will be executed. In this chapter, each phase will be described in detail. It will also include a full account of the user feedback received from the tests. As well as that, the system will be evaluated to see if the interaction with the user is satisfactory.

### Redevelopment

This chapter will detail the development process taken after the feedback is received from the user testing. All changes made and their importance of these changes will be discussed.

### Conclusions and Future Work

This chapter will reflect on the entire project and will discuss any conclusions gathered through the project. As well as that, any future work planned for the project will also be discussed.

# 2. Literature Review

## 2.1. Introduction

In this chapter some of the important areas of research required for this project will be discussed. These research topics include some of the navigation products available on the market for the visually impaired or blind; other research projects in the areas of assistive technology; the technology that was considered to implement this project; and previous final year projects that covered similar areas to this project.

## 2.2. Alternative Existing Solutions to Your Problem

There are existing projects on the market to help visually impaired and blind people navigate independently. As navigation for the visually impaired can be a difficult task it is important that these products are well researched and tested. If a product does not function correctly then it could potentially provide a false instruction to the user which could have a negative impact on the user. Therefore, they require in-depth testing, an example of which is discussed later in this chapter with regards to the MiniGuide product.

The following are products and projects that have been created and carried out to help visually impaired and blind users with their ability to navigate independently:

MiniGuide

MiniGuide (5) is a handheld obstacle avoidance device for the visually impaired and blind. MiniGuide uses ultrasonic sensors to detect objects and includes a single push button for controls. When the MiniGuide locates an object in the users path it vibrates to indicate and approaching object. The faster the vibration the closer the user is to the object. The MiniGuide, similarly to the plan for this project, includes and earphone socket that provides audio feedback to the user.



Figure 2: MiniGuide

Many users have found that the MiniGuide has helped them in multiple ways. Some of these include detecting overhanging objects, locating counter staff, locating the end of a queue, locating doorways and gaps and navigating around obstacles. The MiniGuide can detect large objects from four meters away and include 5 different detection ranges.

Products like the MiniGuide require in-depth user tests. An example of an in-depth test carried out on the MiniGuide came from the Department of Rehabilitation from Laval University in Quebec Canada (35). They tested the MiniGuide on four users from a deaf and blind program. These participants were trained on the product and had their experience evaluated before and after training and then again after 3 months. The researchers for this test interviewed the participants and clinicians to gather their experiences, the benefits of the product and any problems with the device.

Ray Electronic Mobility Guide

The Ray Electronic Mobility Guide (6) is a handheld, lightweight and compact navigation aid, like the MiniGuide. The Ray Electronic Mobility Guide sensitive electronic mobility aid (7) that detects obstacles and alerts the users by emitting either, or both, audio and vibrating signals.



Figure 3: The Ray Electronic Mobility Guide

The Ray Electronic Mobility Guide is battery powered, easy to use and has a short training time. It is to be used as a compliment to the more classic for of navigation for the visually impaired and blind, the white cane.

Other Projects

During the course of this research, there are multiple other projects similar to this project.

The first of these projects is by Ezhilarasi, *et al.* (8). Their work describes the implementation of an assistive aid using a Raspberry Pi. This project aimed to help visually impaired people with many areas of life including obstacle detection, which is also a feature of Navigation Assistant. In their project they completed a prototype that can be worn by a user on their waist which accommodates for the human tendency to point at the object being interacted with.

Shankir Sivan, *et al.* (9) looked into creating a computer vision-based project to aid visually impaired and blind people. They investigated numerous computer vision-based technologies available and looked into the possibility of a cheaper solution. One of the technologies they researched was called FingerReader which is a wearable text reader that aids visually impaired or blind readers with their reading. Essentially, this project investigated various effective assistive devices for the blind and visually impaired. In the end a cheaper but effective assistive aid was proposed but it was found that this conclusion needed further research to improve and add features.

Anushree Harsur, *et al.* (10) implemented a project, using a Raspberry Pi, that makes use of the surrounding sounds to provide the user with a navigation instruction, allowing the user to navigate the outside environment independently. Their project was a success and they found they had further scope to improve the abilities of the system.

Ayush Wattal, *et al*. (11) developed an electronic device, using the Raspberry Pi, for obstacle detection. Their project assists the user in obstacle avoidance by detecting obstacles from three directions and converting the data retrieved into audio instructions given through headphones or speakers. Their project was successful in warning users about upcoming obstacles.

Lastly, V.S.S.Kaushalya, *et al*. (12) created a project that allowed blind or visually impaired people to navigate around without a dependent. They found a solution to their problem, by using a Raspberry Pi, that allowed their users to move around independently and securely. In the end they were able to develop a prototype system that assisted the user and a conjoining mobile app, through which the guardian of the blind or visually impaired person can check in on them. This project, like the previous projects, is similar to the project being developed as it is looking for a solution that will allow visually impaired or blind people to navigate more independently.

All of the above-mentioned projects have elements that will be included in this project. The following table shows the elements that appear in each project in comparison to this project.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Features** | **Ezhilarasi** | **Shankir** | **Harsur** | **Wattal** | **Kaushalya** | **This Project** |
| Obstacle / Object Detection | x |  |  | x |  | x |
| Worn by user | x |  |  |  |  | x |
| Cheap assistive tech | x | x | x | x | x | x |
| Uses sound |  |  | x |  |  |  |
| Navigation instructions |  |  | x |  |  | x |
| Audio instructions |  |  |  |  |  | x |
| App for guardian |  |  |  | x |  |  |
| Assistive device | x | x |  |  | x | x |
| Allow to navigate around |  |  | x |  | x | x |

Table 1: Table showing features of each project

Conclusion

A lot of research and development in currently taking place for navigation equipment for the visually impaired and blind. An example of the success and user satisfaction of this type of research can be seen with the tests carried out on the MiniGuide. These tests showed that there was an overall satisfaction with the product from all four participants. The follow up interviews conducted for these tests provided important user experience information. Navigation Assistant aims to learn from the researched projects and products as to what features were successful and unsuccessful. Navigation Assistant will have some similar functionalities but provides its own unique user experience.

## 2.3. Technologies you’ve researched

Hardware

The hardware chosen for this project is the Raspberry Pi 3 Model B. The Raspberry Pi is a single board computer. It was developed by the Raspberry Pi Foundation (13). and boots from an SD card, the same card that also stores the files on the Raspberry Pi. It can do everything a desktop does, once it is connected to a monitor.

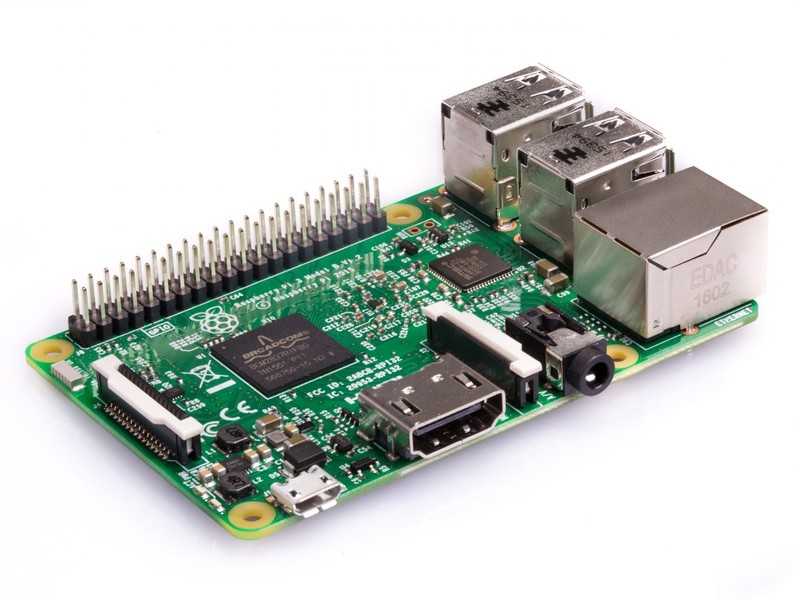


Figure 4: A Raspberry Pi 3 Model B

There are two versions of the Raspberry Pi available, model A and model B. The main difference between the two models is the amount of RAM available. Mode A has 512MB of RAM while mode B has 1GB of RAM. The Raspberry Pi 3 Model B is £32 (14). It uses a 1.2GHz quad core 64bit ARM Cortex A53 processor (14). The ARM processor is a benefit to the Raspberry Pi because it is cheaper than other processors, it consumes less power and therefore makes for a better battery life.

The run the Raspberry Pi an operating system is required. The recommended operating system is called Raspbian(15), a Debian based operating system for the Pi. An installation manager, called NOOBS (16), can be used to install Raspbian by inserting an SD card with the NOOBS application installed.

Also, the Raspberry Pi can have a separate camera installed. The Camera Module V2 (17) is and 8 mega pixel Sony IMX219 camera. It contains a fixed focus lens and can record video and take pictures. However, it cannot record audio.

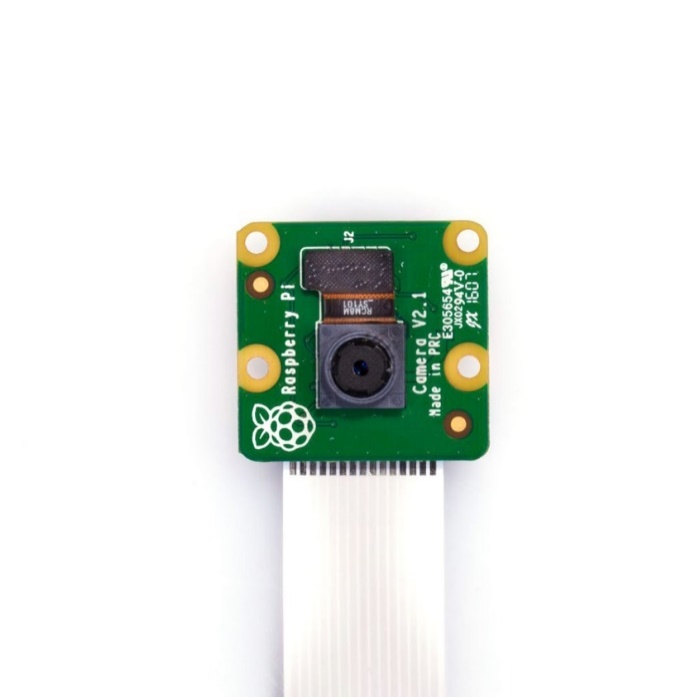


Figure 5: The Camera Module V2

The Camera Module V2 also includes a max resolution of 2592 x 1944, for a still photograph, and a max resolution of 1920 x 1080 for a video.

In conclusion, the Raspberry Pi 3 Model B with the V2 Camera Module and Raspbian operating system installed is the best choice for the development of Navigation Assistant. Not only is there a vast amount of information about object detection with the Raspberry Pi available but it has been used by programmers in the past to develop computing solutions for various disabilities. As well as that, it is a low-cost piece of hardware that is small enough to make portable, which is required for this project.

Programming Languages and Libraries

The programming language chosen to implement this project in is Python. Python is a high level, interpreted, object-oriented language. It is simple and easy to learn, found from previous experiences, an important trait for this project as I had very little previous experience with the python language. Another reason this language was chose over others is because it is easy to install required libraries in python. During object detection many libraries are required to implement the detection, especially in relation to convolutional neural networks. Python includes many pre-built libraries, like OpenCV and Keras, that can simply be imported into the python program and utilised.

One of the libraries the project will require in this project and will import into the python code is OpenCV. OpenCV, or Open Source Computer Vision, is an open source computer vision and machine learning library written in C++ and C (18). It runs on multiple platforms like Windows, Linux, MacOS etc. By using OpenCV it is possible to use the included computer vision architectures to create computer vision applications quickly and easily. OpenCV can be used with many languages, like Ruby and MATLAB, but is predominantly used with Python. OpenCV can be used to detect and identify faces and objects in videos and photos. By implementing OpenCV on a device, like the Raspberry Pi for this project, it is given the ability to see and comprehend the objects around it.

In conclusion, the python programming language is the best choice for the implementation of the Navigation Assistant project. This language is the best choice because it allows for many libraries to be imported including CNN libraries like OpenCV, Keras and TensorFlow. As well as that, all of these important libraries can be installed easily on Python.

Neural Network and Deep Learning Libraries

The two deep learning and neural network libraries researched during this project are TensorFlowand Keras.

TensorFlow is an open source platform for machine learning that can be used for deep learning (19). TensorFlow can be used within an image classification and object detection program. It would be used within machine learning to help with image classification. TensorFlow is completely supported on the Raspberry Pi and is the most popular software library for machine learning on the Raspberry Pi.

Keras is an open source neural network library written in python (20). Keras is a popular middleware for developing and evaluating deep neural networks. Keras is the recommended neural network library for beginners because it has a smooth learning curve and is easy to include in python. As well as that, Keras can run on top of TensorFlow.

For this project the Keras neural network library will be used. Not only is it the recommended library for beginners but it also includes TensorFlow. This means that not only can Keras be used but also TensorFlow.

Web Applications

To train the machine learning models required for this project the following web application were researched, Google Colab and Jupyter Notebook.

Google Colab is a free cloud service (21). It is easy to use and supports Python 3.6 and 2.7. Google Colab is based on Jupyter Notebook and can be used to develop deep learning applications using Keras, TensorFlow and OpenCV. However, it has is limited in session time and size. As well as the above mentioned, Google Colab provides its own GPU, which prevents the machine running the training model using its entire CPU.

Jupyter Notebook is an open source web application that can be used to create and share code and documents (22). Jupyter Notebook is implemented as a single document that can combine visualizations, narrative text, equations and more. It is used to iteratively and rapidly develop and present data science projects. Jupyter Notebook can be used with many different languages including Python. Jupyter Notebook is easy to use and installed with pip (23).

For this project Jupyter Notebook was used. As the model being used was a pretrained model the machine testing the model initially had enough processing power to run. However, both web application run in the same way, Jupyter Notebook just runs locally instead.

## 2.4. Other Research you’ve done

Object Detection and image segmentation

Object detection is a computer vision and image processing related technique. The aim of object detection is to find instances of object classes, specified in the training dataset, in images or videos (24). Object detection has been used in many well researched areas of computer science including facial recognition. Object detection essentially involves using machine learning and deep learning algorithms to locate objects in an image or video. The goal of object detection is to replicate, as closely as possible, how a human would recognise objects in an image but with computer technology.

Object detections can be implemented using many different approaches. One of these is image classification. Image classification assigns a label, or class label, to an image that contains an object. For example, if an image contains a dog as the main object of the image, then the entire image is labelled with the class dog. This approach is usually used in applications that process images with one object contained within it. Image classification can also include localization which allows for the location of an object to be identified using a bounding box. However, this approach cannot be used for this project as this project requires multiple objects in an image to be detected, not just a single object.

Another approach to object detection is object recognition. This approach detects what object has appeared in an image but does not mention the size or location of the object in the image. This approach simply identifies objects in an image. For this reason, this approach was also not chosen for this project.

Image segmentation is another approach that incorporates object detection in its functionality. Image segmentation is also a computer vision technique. This technique not only detects objects in an image but also detects the objects boundaries, something that other object detection approaches do not include. Image segmentation essentially creates a mask for each instance of an object in an image, essentially finding and outlining the found object by its boundaries (25). There are two types of image segmentation, semantic and instance. Semantic segmentation classifies objects of the same type as a single instance whereas instance segmentation classifies objects of the same class as individual instances, this is the reason why the instance segmentation approach to object detection was used in this project.

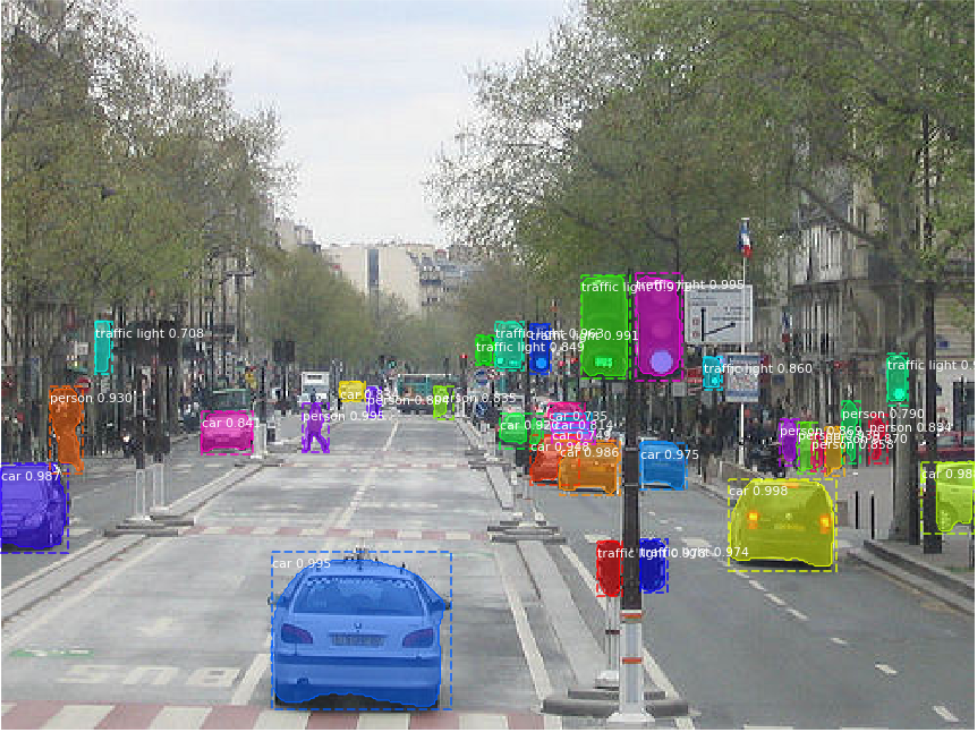


Figure 6: Example of instance segmentation (26)

Convolutional Neural Networks (CNN)

Convolutional Neural Networks are a main part of object detection and image segmentation. CNN’s are one of many deep learning techniques used to implement object detection. As there are many different forms of CNN’s available, for example R-CNN and Faster R-CNN, it was chosen as the main source of research for this project. The different forms of CNN’s available to use will be discussed in the following paragraphs.

A CNN is a neural network that can contain one or multiple convolutional layers that is used in object detection and image segmentation. A CNN takes an image in as input, processes it through the CNN layers and classifies the object with the image into specified categories. A CNN consists of three core layers, the Convolutional Layer, the Pooling Layer and the Fully Connected Layer, as shown in the figure below.

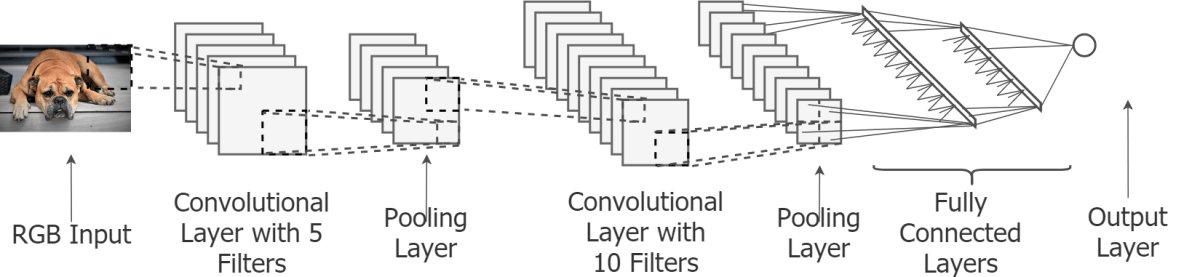


Figure 7: Convolutional Neural Network (CNN) layers (27)

The first layer is the Convolutional Layer. This layer is a mathematical operation and can be implemented multiple times. This first layer is used to extract features from an image, while preserving the relationship between pixels. This layer learns about object features using small squares of input data and different filters for edge detection, blurring or sharpening. After each convolution the output from this layer reduces in size. The convolution layer reduces the size of the output after each iteration of the layer because images tend to be very large in size. It would be inefficient to use every pixel of these large images as input. Therefore, pre-processing like the convolutional layer is required to reduce the image to a smaller size before applying the image to a neural network. The convolutional layer essentially applies convolution to small areas of an image to gather a single sample value of the pixels in that area. This process is continued for all areas of the image thus creating a new smaller image made up of the sample pixel values. This means that many convolutional layers can lead to a very small output.

The next layer in CNN implementation is the Pooling Layer. Like the Convolutional Layer, the Pooling Layer can be run multiple times, immediately after a Convolutional Layer has been implemented. The Pooling Layer has three different forms of pooling, max pooling, average pooling and sum pooling. The most commonly used being form of pooling is max pooling. The purpose of this layer is to reduce dimensionality, when an input image is too large, but retain important information about the image and its extracted features. This is done by applying a max filter to any overlapping areas of the initial image. This reduces the number of parameters available for learning and therefore reduces computational costs.

The last layer in a CNN implementation is the Fully Connected Layer. The Fully Connected layer essentially compiles all the information from the previous two layers and creates an output. This layer receives inputs from the previous layers, combines the found image features, classifies these features and outputs them.

Region Convolutional Neural Networks (R-CNN) algorithms

R-CNN makes use of CNN and its layers. R-CNN extracts regions from an image or video, using a selective search, and checks if the region contains an object. The R-CNN extracts regions and uses the CNN layers to extract specific features from the regions. These features are then used to detect objects. Although R-CNN can extract objects from an image or video it is slow due to the multiple steps involved. Due to R-CNN being slow, multiple version were created and built on top of the basic RCNN algorithm. These other versions are Fast R-CNN and Faster R-CNN.

Fast R-CNN was the first of the two versions to be built and is built from R-CNN. With Fast R-CNN instead of extracting regions from an image, the image is passed to CNN to generate a region of interest. All steps in Fast R-CNN are simultaneous making Fast R-CNN faster than R-CNN. However, Fast R-CNN is still not fast enough on a large dataset. Because of this Faster R-CNN was developed.

Faster R-CNN builds on top of Fast R-CNN to improve classification speed. Faster R-CNN fixes the issue with selective search of an image, thus reducing the processes computational needs. This is done by replacing the selective search with a Region Proposal Network (RPN). The RPN is used to decide where to look in an image for objects. Faster R-CNN extracts feature maps from the inputted image using CNN. These maps are then passed through the RPN to find and return areas with objects. These returned objects are then classified, and their bounding boxes are gathered.

Although all three R-CNN algorithms have some limitations (28), of the three R-CNN algorithms Faster R-CNN is the fastest, as shown in the table below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm** | **Elements** | **Time to make prediction** | **Constraints** |
| CNN | Image is divided into regions. Each region is classified. | NA | High computation time. Requires a lot of regions to predict accurately. |
| R-CNN | Selective search generates regions. | 50 seconds | High computation time. Uses three different models to make predictions. |
| Fast R-CNN | Image passes through CNN once. Feature maps extracted from image. Selective search used on maps. Combines three models of R-CNN. | 2 seconds | Computation time is high because of selective search. |
| Faster R-CNN | Selective search replaced with RPN. RPN makes algorithm faster | 0.2 seconds | Object detection takes time but is faster than other algorithms. |

Table 2: Table describing each R-CNN algorithm

Mask R-CNN

Mask R-CNN is an extension of the Faster R-CNN model. Unlike Faster R-CNN that only retrieves the bounding box and class label of an object, Mask R-CNN gets the class label, bounding box and object mask of an object. Mask R-CNN follows the same approach as Faster R-CNN with a minor difference. Mask R-CNN generates an objects segment mask in the region of interest.

Mask R-CNN, like the other R-CNN algorithms, is a deep neural network algorithm that solves the problem of instance segmentation in machine learning and computer vision. This algorithm can determine objects in images and videos.

Mask R-CNN consists of two stages, gather all the proposed regions where an object might be placed and get the class label, bounding box and mask of the objects found. Both stages are connected to a standard CNN backbone, usually ResNet50 or ResNet101 (29). This backbone allows for feature extraction that detects both low-level features, like edges and corners of objects, and high-level features, like people and vehicles. This backbone incorporates a Feature Pyramid Network (FPN) to improve feature extraction. The FPN allows for features to be passed between higher and lower levels to improve feature extraction.

After considering the above algorithms, the Mask R-CNN algorithm is the R-CNN algorithm that will be used in this project to accurately extract objects from the camera input from the Raspberry Pi.

## 2.5. Existing Final Year Projects

A couple of previous Final Year Project were looked at in the research phase of the project. There was an attempt to focus on projects with similarities to the problem being tackled in the creation of Navigation Assistant.

Virtual Environment Navigation: The Development and Evaluation of Virtual Rooms to Aid Visually Impaired Navigation – Mark Courtney

This project involved developing an assistive game to aid the visually impaired to improve their navigation in a particular environment. At the time of this project blind and visually impaired people had to learn to navigate by trial and error which could lead to injuries. They determined the current issues the visually impaired have when navigating new rooms and the various software that is available to improve their lives.

In this project the various technologies available were discussed and evaluated to find the best outcome for this project.

A series of 3D objects and default rooms were created to enable realism for the user. There was also the option to create custom rooms for the user. By navigating these environments, the user could gain a better understanding of the layout of a physical environment that they were familiar with. Users were placed at the entrance of the environment and were given an objective to perform. Haptic and audio queues were given to the user as indications of obstacles depending on their location.

With regards to the project being developed, this project has a similar theme to it, in that, it was created to provide a form of assistive technology for blind and visually impaired people. It is also similar in its use of audio queues with object avoidance.

This project provides another perspective on assistive technology in aiding navigation for visually impaired and blind people.

Monitoring Room Occupancy Using a Raspberry Pi – Sean Meehan

This project created a system capable of measuring a rooms occupancy. This project measured the number of people in a room. The data gathered from this project could be useful when creating timetables for the new academic year as it would give an average number of people in a class and that information could then be used to find the most optimal room.

The application runs on the Raspberry Pi and detects and tracks people entering and leaving a room using OpenCV and C++. It is run on the Raspberry Pi with a camera module attached as it is low powered and can be placed above the doorway.

Using the Raspberry Pi, the occupancy of the room is calculated and updated and sent to the database occasionally. From there data is sent to a web applications frontend, created using PHP and MySQL, and the results are displayed graphically to the user.

With regards to the project being developed, this project uses the Raspberry Pi and camera module for monitoring but for a different reason, room occupancy. This project also like the project being developed uses object detection, for identifying people instead of obstacles, and the OpenCV library.

## 2.6. Conclusions

Using the gained information about visual impairments and blindness, the development stage of this project can begin while keeping this knowledge in mind. Through this research the scope of visual impairments was discovered. All of the available products and previous research projects on this topic were also researched to further understanding of the scope of this project topic.

As a result of this research the technologies best suited for this project were chosen. This included the various different CNN versions available.

Requirements Table

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Priority** |
| Object detection | Detects objects in user’s path using camera module | HIGH |
| Audio instructions | Using the information gathered from the object detection to give audio instructions to the user | HIGH |
| Image classification | Deployed on Raspberry Pi to aid in object detection | HIGH |
| Real time processing | Ensure program runs in real time, or as close as possible, to mimic user’s real time navigation | HIGH |
| Make project portable | Ensure that the user can wear and navigate around with the Pi project easily i.e. no extra wires and portable power source | MEDIUM |
| GPS Navigation | GPS navigate to an address specified by the user | LOW |
| Input through speech | Allow the user to specify and address or location through a microphone | LOW |

Table 3: Table of requirement importance

# 3. Prototype Design

## 3.1 Introduction

Following on from the previous chapter, where the important background research was discussed, these same topics will be continued in this chapter. This is the chapter where the design of the project will be presented. The first section of this chapter will look at the different software methodologies available for use in this project. This section will describe each methodology, which methodology was chosen and why. After that some use cases examples will be presented. Lastly, the technical architecture of this project and how it will work will be discussed. This will include the front-end, back-end and middle tier aspects of the project.

## 3.2. Software Methodology

Agile Methodology

The Agile Methodology is an incremental approach that is collaboration heavy (30). This methodology is designed to accommodate change and the need to produce software faster. Agile methodology is equipped to handle complexity and variability and focuses on presenting software more than documentation.

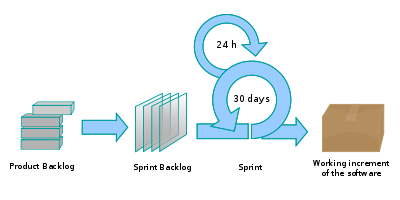


Figure 8: The Agile Methodology (31)

The Agile Methodology is implemented using short sprints or iterations (30). The project is initially broken down into small chunks and completes in sprints, which are usually 2 - 4 weeks long. At the end of a sprint the projects priorities are revaluated, and the design and development plans are changed accordingly.

This type of methodology allows a developer to be more flexible with the development of a project. However, it can be easy to stray from the original plan. Therefore, it is essential to keep checking the original plan to stay on track.

The Agile methodology is close to what is required for the Navigation Assistant project. The Navigation Assistant project requires a methodology that is more feature focused as all elements of the project need to be working accurately for the instructions to be correct. For this reason the general Agile method will not be used for the development of this project.

Waterfall Methodology

The Waterfall Methodology is a traditional but outdated sequential method (30). This methodology requires a lot of documentation and structure. The Waterfall Methodology is divided into self-enclosed stages that are rigid and follow a sequence. These stages are determining the requirements and scope, analyse requirements, design, implement, test, deploy and maintain.

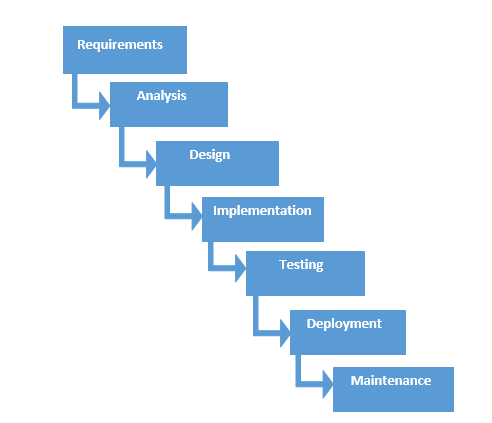


Figure 9: The Waterfall Methodology (32)

While the Waterfall Methodology is easy to use and manage and gives a clear idea of the scope of the project, it has its disadvantages. Firstly, in this methodology there is a lack of flexibility. One stage must finish before the next starts and once a stage is finished it should not be revisited. Secondly, this methodology doesn’t facilitate change. If there are any changes to the initial project scope, then the Waterfall Methodology requires a full restart.

The Waterfall methodology is not a good method to use for this project. This is because the testing and possible extra research will need to take place throughout the project. Changes may also need to be made, something that this methodology does not support, during this project. Therefore, for these reasons the Waterfall development method will not be used in Navigation Assistant.

Feature Driven Methodology

The Feature Driven Methodology, or FDD, is derived from the Agile Methodology (30). It is an older methodology that is iterative and incremental. Feature Driven Methodology follows 5 processes in its development cycle. These processes are as follows: develop the overall model, build a feature list, plan by feature, design by feature and build by feature.

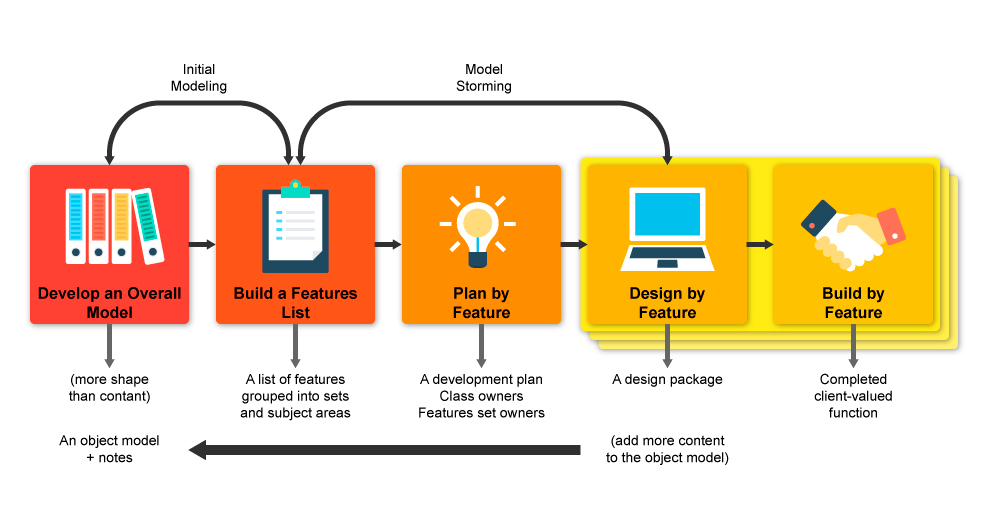


Figure 10: The Feature Driven Methodology (33)

FDD focuses on delivering elements of the software frequently. FDD is designed feature by feature and focuses on short iterations (30). FDD is a good fit for smaller development teams and for long term projects.

There are many advantages to using the FDD methodology. Some of these include continuous tracking of project progress and regularly updating and identifying errors.

The Feature Driven Development methodology will be the methodology used for the development of this project. This methodology is a good match as it allows for individual features to be fully developed, tested and implemented before moving onto the next. For this reason, this methodology will be used to develop Navigation Assistant.

## 3.3. Overview of System

A feature driven development (FDD) approach will be used for this project, as it allows for each feature of the project to be fully developed, tested and integrated into the system before moving onto the next. This development approach will allow for features of the project to be planned, implemented and tested. Once a feature is completed then the development of the next feature will begin.

The FDD approach is required for this project as the features of the project need to be working completely and correctly throughout the project. The design and the development of the code of this project will be delivered in parts. Using FDD a feature will be thoroughly designed and researched before the implementation begins, for example the RCNN models available.

The general approach to this project consists of the following:

1. Design and implement a basic version of the project, making sure to keep the most important features, shown previously in table 3, as a high priority. This basic version will include the detection of objects that will be used to return navigational instructions to the user.
2. Design a feature (for example the calculation of the instructions to be provided to the user)
3. Implement the feature
4. Test the feature
5. Repeat steps 2 to 4 for other features

The technical architecture of the system will show the number of layers there are in the system and how these layers communicate with each other. For this project a 3-tier architecture will be used, as seen in the figure below. This architecture was chosen as the layers effectively communicate with one another.

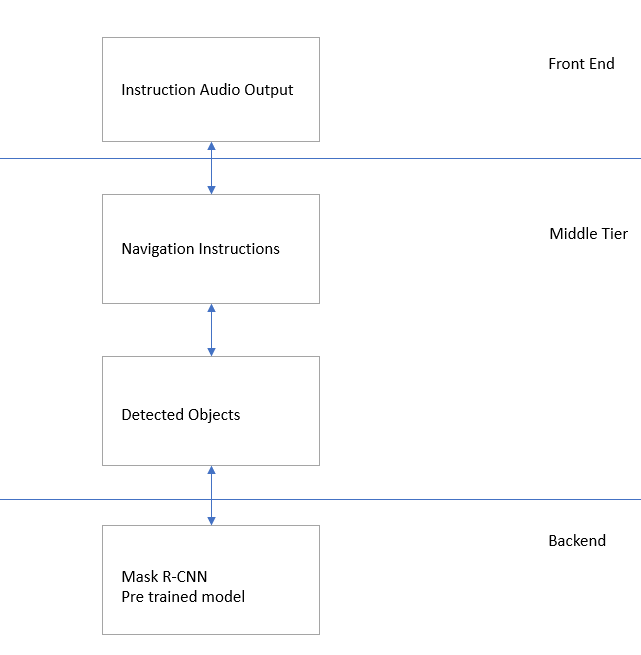


Figure 11: System Layers

## 3.4. Front-End

The presentation layer or front-end layer of the project is the layer that interacts with the user. This layer allows the user to interact with the system and receive audio navigation instruction.

Prototypes of the type of instructions the user will receive will be created. These mock-ups can be shown to potential users, allowing for feedback during the design process. The following are prototype instructions for the project.

Next, the wear ability of the device began. The device needs to be attached to the user to ensure that the users path it being examined fully. This means that the device needs to be either attached to the user’s front using Velcro or attached to a belt, as shown in the prototype diagram below.



Figure 12: Example of making Raspberry Pi wearable (34)

## Use Case Diagrams and Sequence Diagrams

The following use case diagrams are used to identify system functionality and communicate the systems behaviour. The following use case diagrams show the progression of the project’s functionality.

**1st Iteration Use Case Diagram**

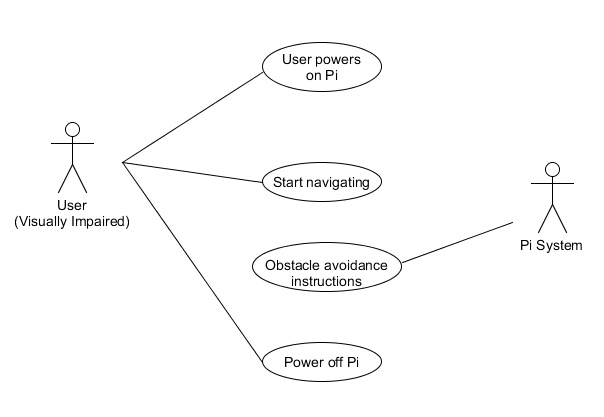


Figure 13: Use Case Iteration 1

The first step of the above process is for the visually impaired or blind user to start the system interaction by powering on the Raspberry Pi and attaching the device to themselves. Once the device is powered on the user can begin to navigate around their environment, either indoor or outdoor. As the user is navigating around their environment the Raspberry Pi system is running a program that takes in a video stream of the user’s path. This video stream is then analysed to detect any objects or obstacles that are in the user’s path. These detected objects are then used to create obstacle avoidance instructions that will be provided to the user. Once the user is finished navigating their environment they will power off the Raspberry Pi and detach it from themselves.

**1st Iteration Sequence Diagram**

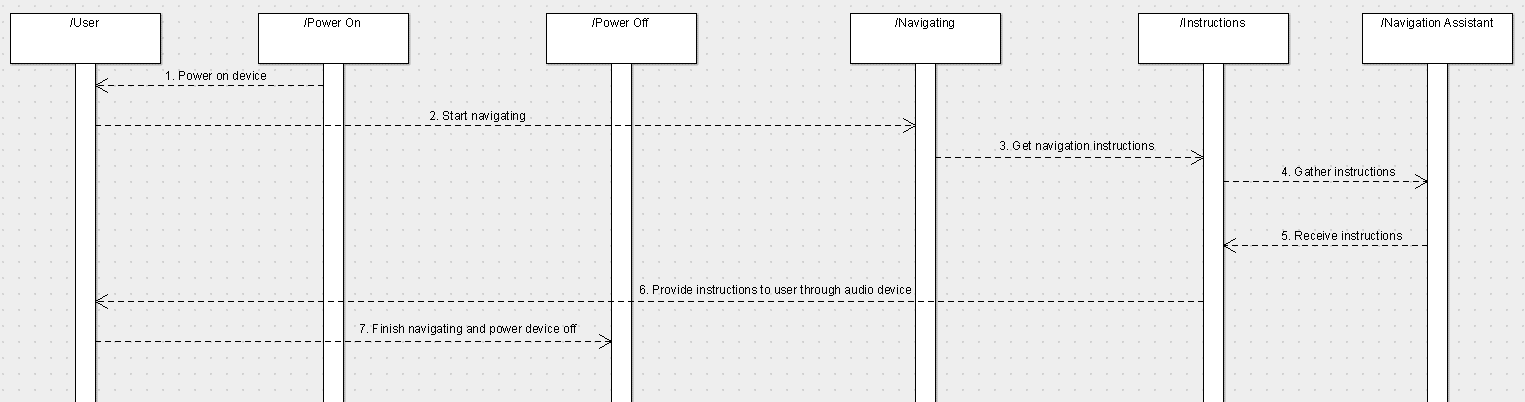


Figure 14: Sequence Diagram Iteration 1

The above figure shows the sequence of processes taken by the system to provide the user with obstacle avoidance instructions. Firstly, the user must power on the device to start the system. Once the device is powered on the user can start navigating around their environment. During this navigation obstacle avoidance instructions are being gathered. These instructions are gathered from the video input gathered from the Raspberry Pi camera. Using the video input obstacle avoidance instructions are created and gathered from the Navigation Assistant program. Once the instructions are gathered they are provided to the user using audio through their connected audio device, i.e. headphones. These processes continue until the user has completed their navigation and they have powered off the Raspberry Pi device.

**2nd Iteration Use Case Diagram**

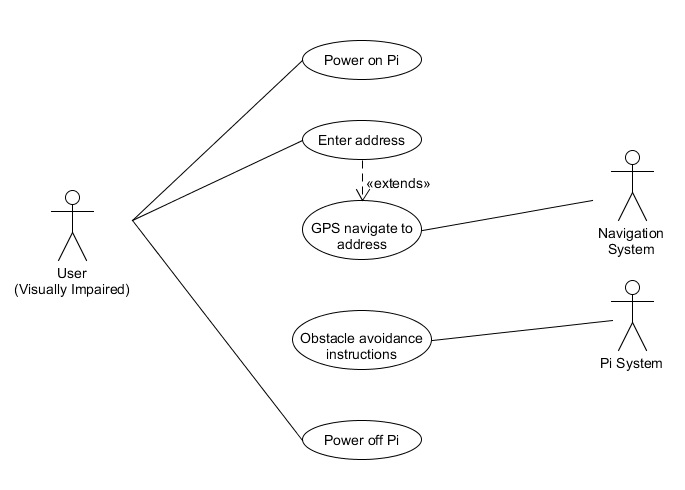


Figure 15: Use Case Iteration 2

The above figure is an extension of a user’s interaction with the system purposed in Figure 13 previously. The above figure follows the same interactions as previously discussed with some extra components. Once the user had powered on the Raspberry Pi device they have the option of entering an address they would like to navigate to, using a microphone. If an address in entered then the navigation system of the device begins to compute the required directions to get to the destination. These directions are then sent back to the user through audio.

**2nd Iteration Sequence Diagram**

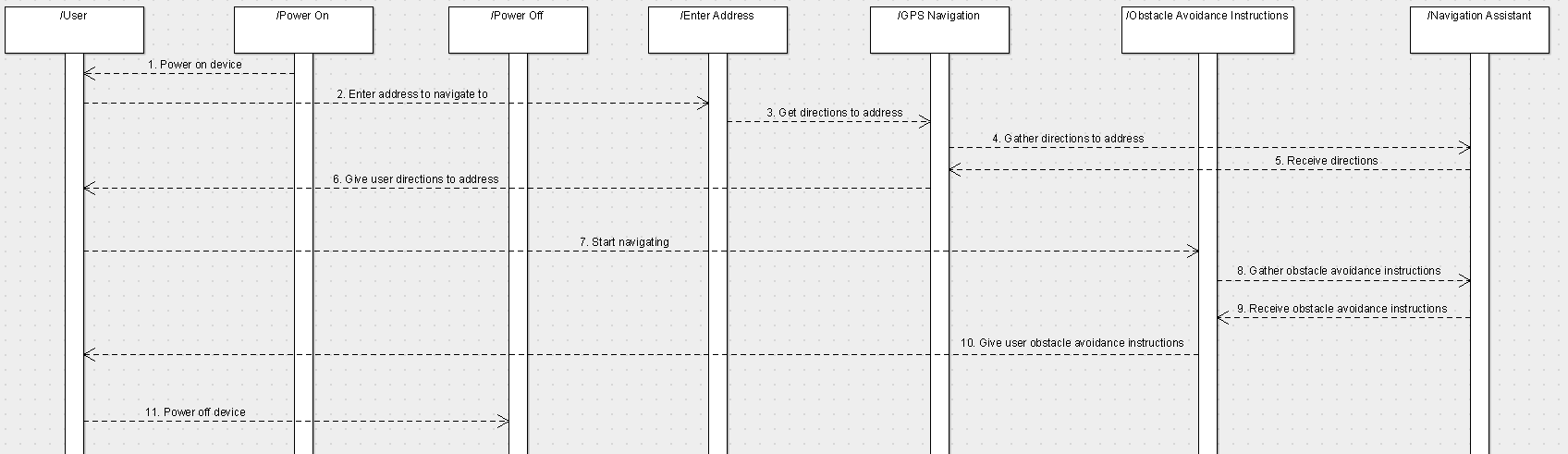


Figure 16: Sequence Diagram Iteration 2

The above figure is an extension of the sequence of processes carried out by the system purposed in Figure 14 previously. The above figure follows the same interactions as previously discussed with some additional processes. After the initial set up of the Raspberry Pi device the user enters the address they want to navigate towards, using a microphone. This address is then sent to the navigation system running on the Pi. This navigation system gathers the directions needed to get to this address. Once the directions are received they are given to the user through their connected audio device. Once the user is finished their navigation they can power off the device to end the process.

**3rd Iteration Use Case Diagram**



Figure 17: Use Case Iteration 3

The above figure is an extension of a user’s interaction with the system purposed in Figure 15 previously. Extending from the previous version of this diagram, the user can specify, using the microphone to input their answer, whether they are navigating indoor or outdoor environments. This specification allows for the system to distinguish if only object detection for obstacle avoidance is needed for indoor environments, or if both object detection and address navigation are required for outdoor environments.

**3rd Iteration Sequence Diagram**

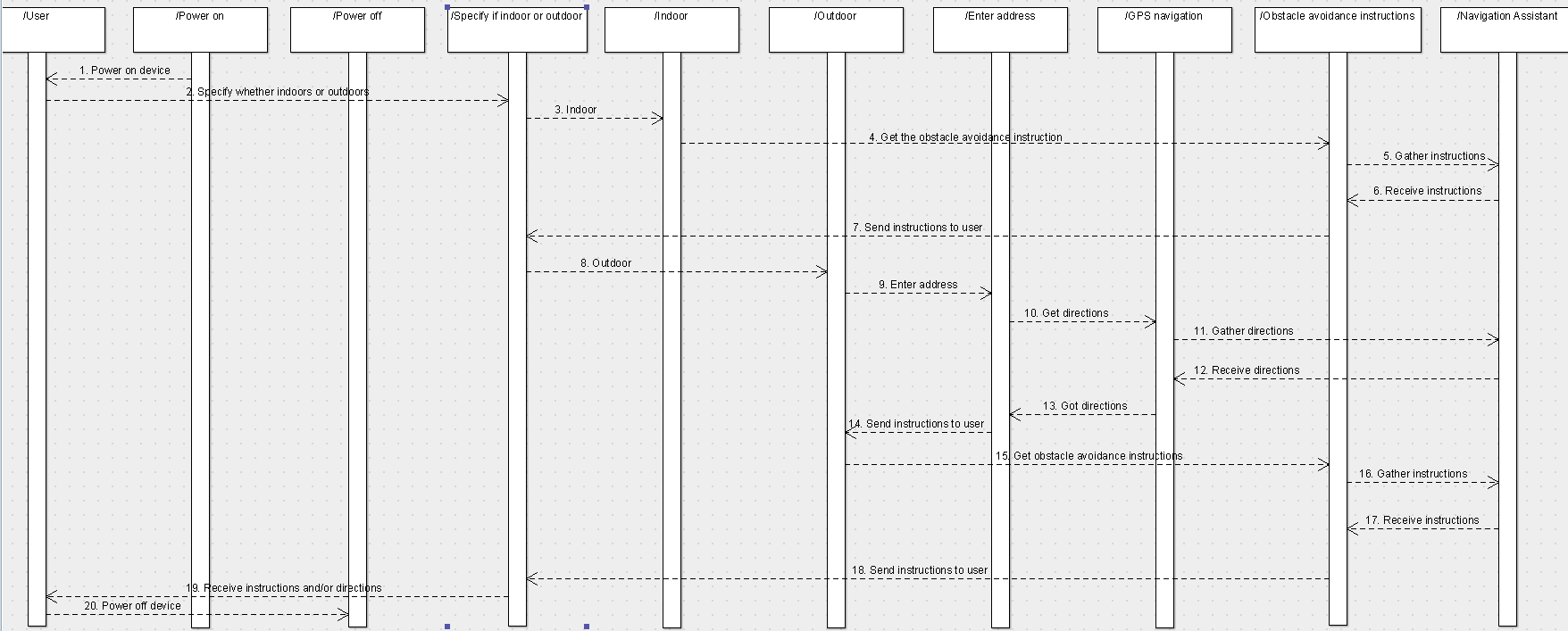


Figure 18: Sequence Diagram Iteration 3

The above figure is an extension of the sequence of processes carried out by the system purposed in Figure 16 previously. Extending from the previous version of this diagram, this sequence allows the user to input their type of navigation environment i.e. indoor or outdoor. When navigating indoor, only the object detection for obstacle avoidance is required. This means that only obstacle avoidance instructions need to be gathered and returned to the user. However, when navigating outdoors and the user has a specific address they would like to navigate towards then both the object detection for obstacle avoidance and the GPS directions will need to be gathered and provided to the user.

## 3.5. Middle-Tier

The middle tier contains all the functional logic which runs the projects main capabilities.

This layer will provide the underlying functionality for the calculation of distances from objects using the pretrained model results, converting these calculations to navigation instructions and outputting these instructions through audio.

For example, detected objects will be gathered from the pre trained Mask R-CNN model in the backend. This pre-trained Mask R-CNN model was created by data scientists from FAIR, also known as Facebook AI Research. This model, what it does and how it works, was discussed previously in the background research chapter of this report. The detected objects gathered from this model will be used by the middle tier to calculate and create navigation instructions that will be provided to the user through an audio device. These instructions will be calculated using the estimated distance from the detected object. These instructions will then be sent from the middle tier to the front end to be outputted to the user through an audio device.

## 3.6. Back-End

The Data access layer is accessed through the middle layer. This allows for the pretrained model to be accessed through the middle layer.

Mask R-CNN is the pre trained model used in this project that has been trained using the MSCOCO 2017 dataset. This model allows the project to detect objects and object edges in images or videos. By using a pre trained model, the model was able to be implemented immediately after testing its accuracy. This meant that more time could be put into understanding the R-CNN model being used and implementing it to run on the Raspberry Pi.

A picture containing indoor, accessory

Description automatically generated

Figure 19: Example of instance segmentation generates from model on Raspberry Pi

## 3.7. Conclusions

In this chapter the design of the system was presented. Firstly, the different development methodologies considered for this project were explored. Next an overview and outline of the technical architecture of the system was discussed. Lastly, the detail of the front-end, back-end and middle tier design was discussed.

Based on the topics discussed in this chapter, the next chapter will cover the prototype development process, including revisiting the topics covered in this chapter. The following development chapter will present how the designs were implemented and will also include the challenges that were encountered along the way.

# 4. Prototype Development

## 4.1. Introduction

This chapter continues to discuss the topics covered in the previous chapter and outlines the development process of the project. The chapter will discuss the crucial development processes and the challenges encountered during the development of this project.

All code mentioned in this project can be found here: <https://github.com/jennifernolan/FYP-Development-Navigation-Assistant/tree/master/Codingfiles>

## 4.2. Prototype Development

The first step of the development process of this project was to set up version control. The chosen version control software was GitHub. This software was chosen as it is a reliable web-based hosting system that can be used to manage the various versions of this project. Once the project folder was set up and linked to the GitHub repository the development of the project could begin.

The next step of the development process was to set up the Raspberry Pi and install the Raspbian operating system. The Raspbian operating system was preinstalled on a formatted SD card. This allowed for a simple installation once the instructions to set up the device were followed. Once the operating system was installed a disk image of the installed operating system was taken and stored on an external hard drive, as the file was too big to be uploaded to GitHub. This means that if the Raspberry Pi were to crash or become corrupt then the original operating system could be reinstalled.

The goal of the initial prototype was to have the instance segmentation model running on the Raspberry Pi and correctly detecting objects, giving their bounding boxes and object masks.

## Vertical Prototype

This section will discuss the process and code used to develop the vertical prototype of this project. The purpose of this vertical prototype is to prove that this project is feasible. This is done by creating an element of the project that runs correctly, gives the correct output and can be built from for the duration of the rest of the project.

The first step of the development of the vertical prototype was to decide on an appropriate pre trained model to use. The Mask R-CNN pre trained model was chosen as it detects multiple objects and gives both their bounding boxes and object masks. The object masks allow for the edges of the detected objects to be detected as well. Once the model was selected the dataset used by this model had to be changed from the 2014 MSCOCO dataset to the more recent 2017 MSCOCO dataset. The following code was used to test if the model was working correctly before the model was deployed on the Raspberry Pi.

As shown in the figure below, all of the relevant libraries required to implement the model and display the results were imported first. Some of these libraries included skimage and matplotlib. The skimage library was required to implement image processing on the inputted image. The matplotlib library was required to create a plotting area of the image and to plot the detected object masks onto the image. Next, the root directory of the project was gathered using the os library. This library allowed for the use of os functionality that was required to gather and manipulate various file paths required for this project.



Figure 20: Required libraries imported

The next step, as shown below in Figure 21, was to gather the path of the current project and to append this to the root directory gathered earlier. This created a direct path to the project file. This will be required later when gathering specific files for the model. After this, the files required to run the pre trained Mask R-CNN model were imported along with the configured MSCOCO dataset required for the model.



Figure 21: Import the model

Once all the imports were completed the pre trained weights file and the test image paths were gathered for later use by the model, shown below in Figure 22. After all the required files were gathered the model configurations were set. These configurations ensure that the model runs inference one image at a time. These configurations were then displayed.



Figure 22: Set up and configure the model

Next the last part of the model was set up, as shown in Figure 23. In this section the object of the model was created, and the training weights trained using the MSCOCO dataset are loaded into the model. Essentially the model is now completely prepared to implement object detection and instance segmentation on the inputted image. Next, the classes that MSCOCO can distinguished are specified. Essentially these are the classes that the model has been trained to detect. A test image is then loaded from the directory and is displayed using the skimage library.



Figure 23: Create the model object

Lastly, the object detection and instance segmentation is implemented. Once the model has completed its detection the results are outputted and the image and its contained detected objects are outputted.

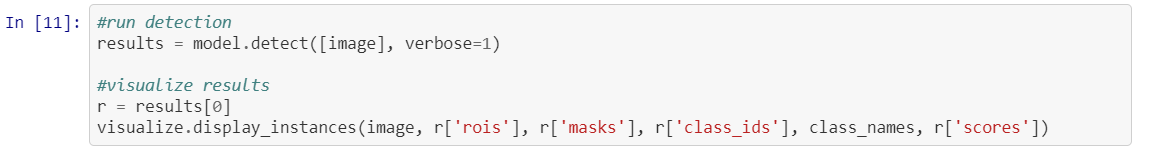


Figure 24: Code used to test that the model gives correct result

When the above code was implemented the model returned the following correctly detected objects shown in the figure below.



Figure 25: Result from testing Mask R-CNN model

Once it was determined that the model was working correctly and detecting the correct objects it was deployed on the Raspberry Pi.

The following code segment, shown in Figure 26, shows how the Mask R-CNN model was deployed on the Raspberry Pi model. The first part of this code sets up the Raspberry Pi camera to take picture. The camera object is created and then previews was the camera can see. This allows for the image to be lined up correctly. After this the picture is taken and stored in a specified path of the project folder. This image taken by the Pi camera will be used later for object detection and instance segmentation.

The rest of the code used to deploy the model on the Pi is the same as mentioned above when testing the model. The only difference is that the image is not shown, using the skimage library, before object detection has taken place. Once the image has been taken and the model has been configured and set up, the object detection takes place straight away.

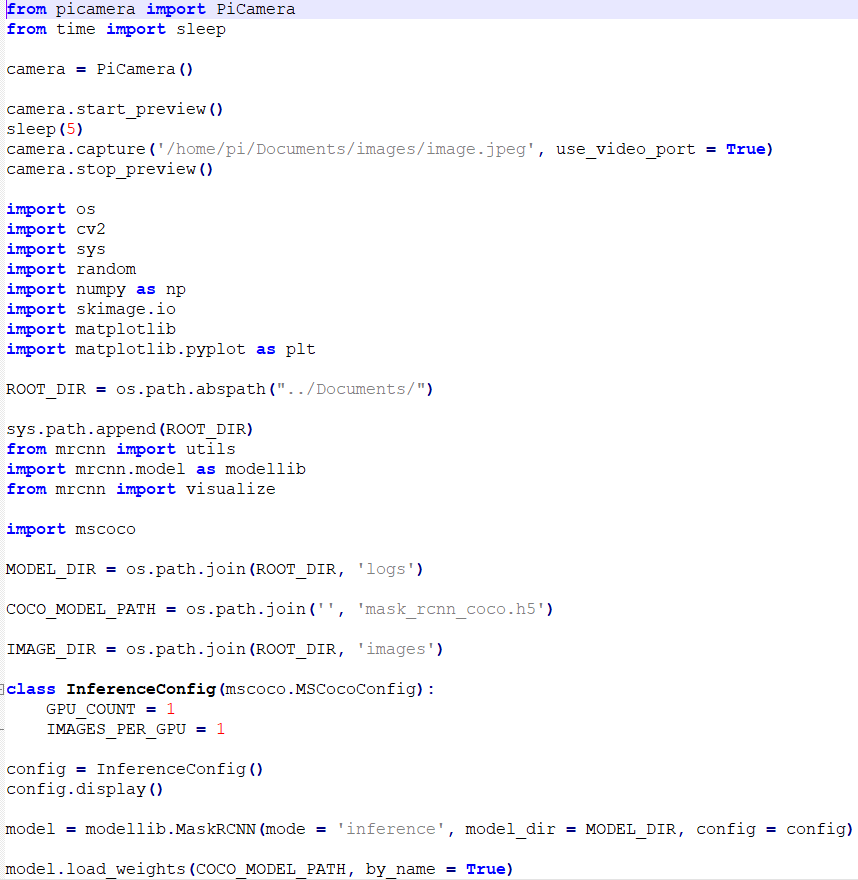
 

Figure 26: Code to run Mask R-CNN model on Pi

Although the model runs on the Raspberry Pi and can detect different objects and their masks, specified in the dataset, the speed to retrieve the result is poor. This is where the complexity of the project lays. It is important to speed up the processing of this instance segmentation result to ensure that the navigation instructions can be provided to the user in a real time manner. Also, the prototype currently runs using an image taken by the Pi’s camera. This will have to be changed at a later stage in the development process to work on the video input from the Pi’s camera.

## 4.3. Front-End

The current prototype front end shows the bounding boxes and object masks of detected objects from an image. These detected objects are then displayed along with their assigned class names. The below image shows the frontend output from the prototype.

A picture containing indoor, accessory

Description automatically generated

Figure 27: Output from prototype after object detection

When this project is fully developed and completed it will use these detected objects to calculate the distance the user is from these objects. Using these distances, calculated by the middle tier, instructions will be created and converted to an audio output that will be presented to the user through their inserted audio device like headphones. The audio instructions outputted to the user will be the front end for this project.

## 4.4. Middle-Tier

The middle tier will make use of the pretrained model, kept in the back end, to detect objects from a video stream. This tier uses the detected objects to calculate the distance the user is from these objects. These distances are then used to create navigation instructions. From there the navigation instructions will then be sent to the front end to be outputted to the user.

## 4.5. Back-End

The backend of this project comprises of the pretrained model Mask R-CNN. This model is pretrained using the MSCOCO 2017 dataset. This model gives the project the ability to detect objects in both images and videos. The current prototype of this project detects objects from an inputted image, but the end project will detect objects from a video input. This model is accessed by the middle tier to calculate the user’s current distance from objects and to create and calculate navigational instructions.

## 4.6. Conclusions

In this chapter the development process that has taken place so far was presented. This included the building of a vertical prototype. This chapter outlined what aspects of the project have been developed so far and what has yet to be developed for the duration of the project. The vertical prototype developed in this chapter, detecting objects and their instances in an image, forms the basis of this project.

# 5. Testing and Evaluation

## 5.1. Introduction

This chapter discusses the testing processes that will be used during the development of this project. In this section the forms of testing that will be used during this project and the outline of the test plan will be presented. This chapter also covers how the evaluation of this project will be conducted.

## 5.2. Plan for Testing

Backing up and committing the project to GitHub throughout the process of the project will allow for version control. It will also allow for any changes to be rolled back to a previous version in case of errors. The project will also be backed up to a personal external hard drive in case of any accidental deletions.

The project will need to be continuously tested throughout its development cycle once the model, deployed on the Raspberry Pi, runs on a video input and not an image input . Unit testing will be used to ensure that individual aspects of the system are working correctly. Unit tests are where the developed tests will interact directly with the system to ensure all aspects of the system are working correctly. Unit testing will be essential with the use of a feature driven approach as after each feature of the project is developed it will be tested.

System testing, a method of black box testing, will also be used to evaluate the complete system. This form of testing will ensure that the system meets the requirements set and that the software functions from beginning to end correctly.

Testing of this project will be done by the developer and potential users.

## 5.2.1 Test Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Test No.** | **Test Description** | **Expected Outcome** | **Pass?** |
| 1 | Does the program run automatically when Pi is powered on? | The program will run automatically when the Pi is powered on. |  |
| 2 | Does the program close correctly when the Pi is powered off? | The program will shut down correctly when the Pi is powered off. |  |
| 3 | Does the audio come through the audio device when attached to the Pi? | Audio instructions should be heard through the audio device. |  |
| 4 | Are the audio navigation instructions accurate? | The instructions should be as accurate as possible i.e. the instructions should not cause the user to bump into another object in another position. |  |
| 5 | Are the instructions in time with the user’s movements? | The instructions should be in real time, or as close as possible, to match the user’s movements. |  |
| 6 | Are the instructions accurate indoors? | The device should detect indoor objects like chairs and tables. |  |
| 7 | Are the instructions accurate outdoors? | The device should detect outdoor obstacles like people and cars. |  |
| 8 | Are the navigations instructions accurate in a busy area? | The device should still give good navigation instruction when used in busy areas like town centres. |  |

Table 4: Table of test plan

## 5.3. Plan for Evaluation

Evaluating the system will be as important as testing. One of the reasons for this is that the user’s experience while using the project is the main priority. Navigation Assistant needs to be as easy as possible to use, even though the code to make the project is quite complex. The system needs to be intuitive. The usability of the system being easy, intuitive and accurate, with regards to the instructions provided to the user, will be of high priority throughout the development process of the project. Also, having the system be evaluated and tested by potential user will ensure that the system is usable to a higher level.

It is planned that approximately 5 to 6 users of varying degree of visual impairment will test and evaluate this project. Each participant will go around the same controlled obstacle course with the developed device attached to them. Once each user has completed the course they will be provided with a survey to evaluate the device. As well as this, all the tests will be witnessed and recorded to allow for reflection later. Each session, from the beginning of the course to the completion of the survey, will take approximately 15 minutes, depending on the length of time it takes the participant to complete the course.

This section of the project details how the system will be evaluated. Evaluating the system will be done by potential users, including visually impaired people.

## 5.4. Conclusions

This chapter discussed the testing and evaluation aspects of the project. The testing consisted of implementing Unit Testing and System Testing while developing the features of this project and on its completion. The evaluation included evaluations from different potential users.

# 6. Issues and Future Work

## 6.1. Introduction

This final chapter will present the future work planned for this project until its final submission. Included in this chapter are the possible risks and challenges that could be encountered for the duration of this project and their possible solutions. Lastly in this chapter, a graphical representation or Gantt chart will be presented to show the expected timescale for the different aspects of this project.

## 6.2. Issues and Risks

The challenges that are currently unresolved in the project are as follows:

* Lack of knowledge and familiarity with real time instance segmentation with Mask R-CNN on the Raspberry Pi.
* Finding and getting in contact with individuals who can evaluate and give feedback on developed system.
* Finding the balance between the user’s real time activity and the accuracy of the navigation instructions i.e. giving as accurate as possible instructions to the user at the right time.

How the author will approach these challenges are as follows (respectively) :

* Continue to research, in depth, how to implement real time instance segmentation on the Raspberry Pi using the Mask R-CNN model.
* Get in contact with the Students Learning with Communities department about the possibility of getting in contact with individuals who can give feedback on the system.
* Research how to increase the Raspberry Pi’s processing power to allow for more accurate and real time navigation instructions to be provided to the user.

The risks associated with this project are as follows:

* Raspberry Pi is slow to run instance segmentation in real time as the processing power on the Pi is low.
* Consider how to implement Raspberry Pi’s power source to make the device portable.

How the author plans to approach these risks are as follows (respectively):

* Look into devices that could possibly increase the Raspberry Pi’s processing power for example Google Coral US Accelerator.
* Decide whether to use a rechargeable power source or a disposable power source to make the Raspberry Pi portable.

Some of the above-mentioned risks have already been mitigated since the development of the vertical prototype. Through the development of the prototype knowledge and familiarity with the Mask R-CNN model has improved. As well as that, the risk of the Mask R-CNN model not running on the Raspberry Pi has been diminished through the development of the vertical prototype. This prototype proved that it is possible for the model to run on the Raspberry Pi. Even though the running of this model is slow on the Pi, it is a step in the right direction for this to implementing object detection obstacle avoidance on the Raspberry Pi required for this project.

## 6.3. Plans and Future Work

The future for this project can be seen in the Gantt chart below. Another Gantt chart will be filled out towards the end of the project and both charts will be compared to see the differences between the planned approach versus the reality.

This project will continue to be completed for the duration of the project timescale. Navigation Assistant will be implemented to work on a video feed instead of images. The project will also calculate navigation instructions to provide to the user through audio.

### 6.3.1. GANTT Chart

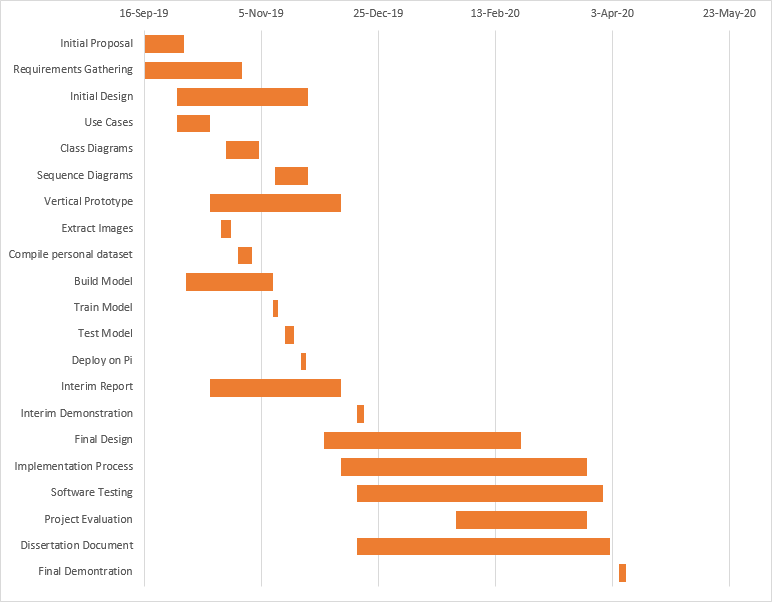


Figure 28: Gantt Chart

# Bibliography

1. Facts and figures | European Blind Union [WWW Document], 2019. URL http://www.euroblind.org/about-blindness-and-partial-sight/facts-and-figures (accessed 10.19.19).
2. Facts about sight loss [WWW Document], 2016. NCBI. URL https://www.ncbi.ie/facts-about-sightloss/ (accessed 10.19.19).
3. Vision impairment and blindness [WWW Document], 2019. URL https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment (accessed 10.19.19).
4. Smartphone GPS Navigation | American Foundation for the Blind [WWW Document], 2019. URL https://www.afb.org/blindness-and-low-vision/using-technology/smartphone-gps-navigation-people-visual-impairments (accessed 10.19.19).
5. The Miniguide, an ultrasonic mobility aid, electronic travel aid (ETA) [WWW Document], 2019. URL http://www.gdp-research.com.au/minig\_1.htm (accessed 11.01.19).
6. MaxiAids | Ray Electronic Mobility Aid for the Blind [WWW Document], 2019. URL https://www.maxiaids.com/ray-electronic-mobility-aid-for-the-blind (accessed 11.01.19).
7. CareTec - Products for the Blind and Visually Impaired: Ray® - the handy mobility aid! [WWW Document], 2019. URL http://www.caretec.at/Mobility.148.0.html?&cHash=a82f48fd87&detail=3131 (accessed 11.01.19).
8. C. Ezhilarasi, R. Jeyameenachi, Mr. A.R. Aravind, 2017. A Raspberry Pi Based Assistive Aid for Visually Impaired Users. Volume 3 Issue 2.
9. Shankar Sivan, Gopu Darson, 2016. Computer Vision based Assistive Technology For Blind and Visually Impaired People.
10. Anushree Harsur, Chitra.M, 2015. Voice Based Navigation System for Blind People Using Ultrasonic Sensor. International Journal on Recent and Innovation Trends in Computing and Communication. Volume 3 Issue 6.
11. Ayush Wattal, Ashutosh Ojha, Manoj Kumar, 2016. Obstacle Detection for Visually Impaired Using Raspberry Pi and Ultrasonic Sensors. National Conference on Product Design.
12. V.S.S.Kaushalya, K.D.D.P.Premarathne, H.M.Shadir, P.Krithika, S.G.S.Fernando, 2016. Automated Help aid for Visually Impaired People using Obstacle Detection and GPS Technology. International Journal of Scientific and Research Publications. Volume 6 Issue 11.
13. What is a Raspberry Pi? [WWW Document], 2019. Raspberry Pi. URL https://www.raspberrypi.org/help/what-is-a-raspberry-pi/ (accessed 11.03.19).
14. Raspberry Pi 3 Model B [WWW Document], 2019. Pi Hut. URL https://thepihut.com/products/raspberry-pi-3-model-b (accessed 11.03.19).
15. Installing operating system images - Raspberry Pi Documentation [WWW Document], 2019. URL https://www.raspberrypi.org/documentation/installation/installing-images/ (accessed 11.03.19).
16. NOOBS - Raspberry Pi Documentation [WWW Document], 2019. URL https://www.raspberrypi.org/documentation/installation/noobs.md (accessed 11.03.19).
17. Getting started with the Camera Module - Introduction | Raspberry Pi Projects [WWW Document], 2019. URL https://projects.raspberrypi.org/en/projects/getting-started-with-picamera (accessed 11.03.19).
18. About – OpenCV [WWW Document], 2019. URL https://opencv.org/about/ (accessed 11.03.19).
19. Image Recognition With TensorFlow on Raspberry Pi, 2019. Image Recognition With TensorFlow on Raspberry Pi [WWW Document]. Instructables. URL https://www.instructables.com/id/Image-Recognition-With-TensorFlow-on-Raspberry-Pi/ (accessed 11.03.19).
20. What is Keras? The deep neural network API explained | InfoWorld [WWW Document], 2019. URL https://www.infoworld.com/article/3336192/what-is-keras-the-deep-neural-network-api-explained.html (accessed 11.03.19).
21. Deep Learning Development with Google Colab, TensorFlow, Keras & PyTorch [WWW Document], 2018. KDnuggets. URL https://www.kdnuggets.com/deep-learning-development-with-google-colab-tensorflow-keras-pytorch.html/ (accessed 11.03.19).
22. Jupyter Notebook for Beginners Tutorial — Dataquest [WWW Document], 2019. URL https://www.dataquest.io/blog/jupyter-notebook-tutorial/ (accessed 11.03.19).
23. Project Jupyter [WWW Document], 2019. URL https://www.jupyter.org (accessed 11.03.19).
24. What Is Object Detection? [WWW Document], 2019. URL <https://uk.mathworks.com/discovery/object-detection.html> (accessed 18.11.2019)
25. Step-by-Step Implementation of Mask R-CNN for Image Segmentation [WWW Document], 2019. URL <https://www.analyticsvidhya.com/blog/2019/07/computer-vision-implementing-mask-r-cnn-image-segmentation/> (accessed 18.11.2019)
26. matterport/Mask\_RCNN: Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow, 2017. URL <https://github.com/matterport/Mask_RCNN> (accessed 06.12.2019)
27. Thushan Ganegedara, 2019. Intuitive Guide to Convolutional Neural Networks, 2019. URL <https://towardsdatascience.com/light-on-math-machine-learning-intuitive-guide-to-convolution-neural-networks-e3f054dd5daa> (accessed 18.11.2019)
28. Implementing Faster R-CNN in Python for Object Detection [WWW Document], 2018. URL <https://www.analyticsvidhya.com/blog/2018/11/implementation-faster-r-cnn-python-object-detection/> (accessed 18.11.2019)
29. Waleed Abdulla, 2018. Splash of Color: Instance Segmentation with Mask R-CNN and TensorFlow. URL <https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46> (accessed 18.11.2019)
30. LeanKit, 2019. Top 6 Software Development Methodologies [WWW Document]. LeanKit. URL https://leankit.com/blog/2019/03/top-6-software-development-methodologies/ (accessed 10.19.19).
31. Scrum - the diagram [WWW Document], 2019. Emergn. URL https://www.emergn.com/blog/scrum-the-diagram/ (accessed 11.03.19).
32. What is Waterfall Model in software testing and what are advantages and disadvantages of Waterfall Model [WWW Document], 2015. Testingfreak. URL http://testingfreak.com/waterfall-model-software-testing-advantages-disadvantages-waterfall-model/ (accessed 11.03.19).
33. Feature Driven Development Methodology [WWW Document], 2018. URL https://newline.tech/blog/feature-driven-development-methodology/ (accessed 11.03.19).
34. Pibow Belt Base for Raspberry Pi by pburgess [WWW Document], 2019. URL <https://www.thingiverse.com/thing:56100> (accessed 06.12.19)
35. Claude Vincent, François Routhier, Valérie Martel, Marie-Ève Mottard, Frédéric Dumont, Lise Côté, Danielle Cloutier, 2014. Field testing of two electronic mobility aid devices for persons who are deaf-blind. Disability and Rehabilitation. Assistive Technology. Volume 9 Issue 5. URL <https://www.ncbi.nlm.nih.gov/pubmed/24266810> (accessed 07.12.19)