Master of Technology in Intelligent Systems

Semester 2 – Pattern Recognition and Machine Learning Systems (PRML)

Group 9 Project Report

NAVCON

A Navigation & Guidance Control System for our Visually Impaired People

Project Team Members

Name	Student ID
Mehta Vidish Pranav	A0213523U
Veda Yogeesha	A0213556H
Zhang Yu	A0213498X
Anandan Natarajan	A0213514U

Table of Contents

Table of Figures

1 EXECUTIVE SUMMARY

According to the global statistics of the World Health Organization ("WHO"), 188.5 million out of 7.8 billion people have mild vision impairment, 217 million people have moderate to severe vision impairment, and 36 million people are visually impaired. Evidently, vision impairment has a significant impact on the functioning of daily activities, including the ability to navigate and recognize the environment independently. Fortunately, apart from medical advancements, technological solutions have enabled the possibility of harnessing assistive tools to improve Visually Impaired People's (VIP) quality of life and to offer them better integration into society. Based on existing literature, we recognise that assistive devices for VIP are a boon to their functional limitations. These devices help to support daily traveling and have high potential to improve VIP's social inclusion.

A market opportunity presented itself to us when we discovered that there has been a dearth of an all-in-one solution for VIP's at this juncture - one which promises to bring together all the features (such as text-to-read, object detection and navigation) in a single product to empower the VIP. Before we embark on our product/strategy creation, our team conducted an extensive background research on VIP, their families and friends, as well as caregivers and professionals in the ophthalmology field to understand some of the most desired characteristics of an assistive device. We have unravelled that family and caregivers of VIP's primarily require the product to be safe to use, have emergency warning systems to alert the owners through loaded user profile and provide timely information and warnings about the surrounding environment.

Armed with extensive market research, our team of 4 dynamic and dedicated solutionists voyaged on a journey to create an intelligent navigation and guidance control system to empower VIP and to assist in their daily mobility needs. As elaborated in the sections subsequently, we have developed a dynamic navigation system that recognizes objects and avoids collisions with path finding functionalities in both outdoor and indoor environment. Additionally, with our users in mind, we created a simple and intuitive front-end user interface with audio instructions provided throughout. This system uses a combination of the Support Vector Machine (SVM) and Principal Component Analysis (PCA) to load users profile using facial recognition to alert our VIPs family or caregiver during emergency.

Project Git Repository: Navcon.

2 BACKGROUND OF BUSINESS PROBLEM

2.1 Market Research: The Demand for Navcon

According to our research, complete visual impairment is one of the most common disabilities in the world and of all the individuals who are visually impaired, 82% are 50 years or older (refer to Figure 1 below). This shows that as an individual ages, he/she becomes more susceptible to eye diseases and/or at risk of partial/complete visual impairment. Perhaps even more alarming is the overall projection shown by WHO which estimates that the population of adults with vision impairment and age-related eye diseases would double as the world's population ages for the next three decades (refer to Figure 2 below). Clearly, our uniquely engineered solution for VIP would be highly relevant and very much sought after in the near future.

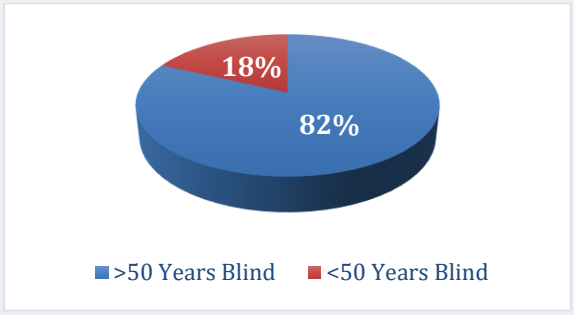
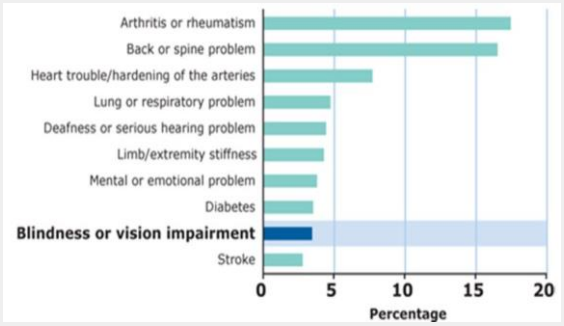
According to the World Health Organisation's Survey, **82% of individuals who suffer from mild to severe visual impairment** are **50 years old or older**.	As per Centre for Disease Control and Prevention's (CDC) survey, severe / complete visual impairment is 1 of the top 10 disabilities among adults.
	

Figure 1: Statistics on the severity of visually impairment

When we researched the needs of our VIPs, it came to our attention that many existing studies focus on increasing their quality of lives (QOL) via 2 intertwined aspects (e.g. by improving both physiological and psychological aspects). After all, QOL is associated with health, physical functioning, life satisfaction and sense of happiness.

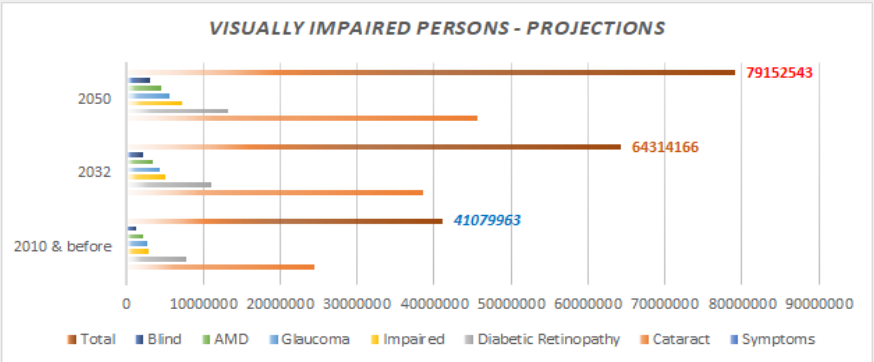


Figure 2: Future projections on the number of VIP

Interestingly, according to a research conducted by WHO in June 2015[1], the meaning of life and life satisfaction was associated with moderate physical activity. Not only that, in these groups of VIP, the effect of physical activity (PA) on the socialization process, the ability to explore own personality traits, developing creativity, and desire to overcome the difficulties associated with visual impairment were observed to be the most prominent. Based on the obtained data, regular physical activity performed by VIP could lead to incremental positive changes in their quality of lives. The research finding also shows the need for VIP to engage in both outdoor and indoor activities.

Type of PA	Group			p
	P	N	NT	
Job-related	1184.4	1692.0	1303.3	0.72
Moderate PA	432.4	728.0	153.3	0.05
Transportation	1975.3	1003.3	1297.2	0.26
Moving by bicycle	535.9 ^{B}	237.3 ^{C}	3.3 ^{B,C}	0.00
Housework	355.0 ^{B}	431.3	755.8 ^{B}	0.01
Vigorous PA in the garden or yard	112.9 ^{B}	29.3 ^{C}	380.0 $^{B, C}$	0.02
Recreation, sport, leisure time	1094.1	708.0	573.3	0.40
Vigorous PA	451.8 ^{B}	344.0	80.0 ^{B}	0.00
TEE	4608.8	3834.7	3929.7	0.87
Time spent sitting	3948.0	2674.0 ^{C}	4048.3 ^{C}	0.10

Legend:
PA – physical activity; P – properly sighted; N – visually impaired athletes; NT – sedentary visually impaired; TEE – total energy expenditures for physical activity.
p – probability, statistical significance N and NT were indicated by C and between P and NT by B.

Figure 3: Estimated physical activity and time spent sitting in hours/week

2.2 Market Competitor Analysis

To come up with an efficient and usable system for VIP, it is imperative for us to conduct market research to gather information about our competitors. This is to determine how viable, successful, and different our product could be. The market competitor analysis delves into the pros and cons of existing competitor products and evaluates the overall suitability of these products in empowering the lives of VIP.

[1] *The Assessment of The Quality of Life in Visually Impaired People With a Different Level of Physical Activity*"

Competitor Solution	The offering is…	… and, miss to support
OrCam Reader 	**Uses Text to Speech to read documents** For people with mild low vision, reading fatigue, reading difficulties including dyslexia, and for anyone who consumes large amounts of text. OrCam Read is a handheld device with a smart camera that seamlessly reads text from any printed surface or digital screen. It enables one to enjoy the morning paper, read any book, and even read all that appears on the computer or smartphone screen!	• Navigation • Object Detection • Image Recognition Costs 3500 SGD
UltraCane Mobility Aid 	**Detect obstacles in vicinity and alert user by vibrating buttons.** UltraCane is a primary electronic mobility aid for use by people who are blind or visually impaired to safely avoid obstacles and navigate around them.	• Text to Speech Costs 1000 SGD
Microsoft Seeing AI 	**Object Detection** Designed for the blind and low vision community, this research project harnesses the power of AI to describe people, text, currency, color, and objects.	• Navigation Free of Cost (under trial)
Microsoft CaptionBot	**Picture Caption** The idea is that you upload a photo to the service, and it tries to automatically generate a caption that describes what the algorithm sees.	• Navigation Free of Cost (under trial)

Figure 4: Analysis of dominant market competitors and their product offerings

In order to complete most daily tasks, a combination of services offered by our competitors (for example, shopping requires object detection and navigation) is required and all of our dominant competitors fail in providing all the features, such as text-to-read, object detection and navigation, as a single package to the user. From the above table, it is also evident that there is no dominant competitor/product offering who can provide a total navigation and guidance solution now. As such, our team believes that there is a sea of opportunities for the development of Navcon and there will be room for our product to grow in the market in near future.

3 PROJECT OBJECTIVE & SUCCESS MEASUREMENT

3.1 Project Objective & Scope

With a clear perspective on the target market and a thorough market analysis, Navcon's primary objective is to empower VIP's to carry on their daily activities independently. As proven by research, increasing VIP's physical activity will help them in attaining a higher quality of life. To carry out most of the essential daily task such as shopping or reading a newspaper, a myriad of features from object detection to indoor and outdoor navigation are required. Navcon's scope was to provide all the features in a single product for our VIP's convenience.

The aim of an object detection system in Navcon is to minimize the ambiguity and discomfort that VIP usually encounter in identifying objects in their daily life. To achieve Navcon's objective, the object detection system needs to cover extensive ground in identifying most common objects in both indoor and outdoor environments. Whilst the object detection system is helpful in an understanding of the environment for the VIP, it is certainly insufficient for navigation. Navcon needed a strong navigation and guidance control system which is flexible and adaptable to both environments. Moreover, information consumed via text also needed to be accommodated into the framework of Navcon. The simple idea is to feed the information consumed and parsed in our intelligent sensing system via audio instruction signals to VIP's.

Another key objective is to ensure the scalability of our system to support our strong user base. Volunteering, especially virtual volunteering, is a very successful avenue to get the community involved in our cause. Navcon aims to provide a usable framework for the community to come together in image tagging efforts for supporting our object detection models. In this way, Navcon's system comes to a close loop leveraging on community goodwill to build the social capital in Singapore and strive to empower our VIP's.

3.2 Success Measurements

Quantifiable measurement metrics are critical to the success of our project. We leveraged on SMART (Specific, Measurable, Attainable, Relevant and Time-based) goals for the timely delivery of our project deliverables. Navcon's deliverables were tracked via the Azure Dev Ops platform and our team regularly organised weekly sprints for the completion of action items. Epics were key deliverables that were required to meet the minimum viable product features that Navcon's project scope must cover:

- An interface to capture user image for authentication purpose.
- Object detection capabilities for visual image processing.
- Indoor navigation module for active pathfinding and collision avoidance.

- Outdoor navigation module to detect obstructions and alert users with audio instructions; and
- Optical Character Recognition (OCR) capabilities to provide users with audio prompts of the images in front of them - there are endless possibilities to this, ranging from reading menus in restaurants to reading full documents/books

Each of the above epics were sub divided into respective user stories for tracking during our biweekly meetings. Our team also set out measurable and time-based goals for the project to be carried out in 6 phases:

1. **Project Initiation:** Upon receiving the assignment, the team assessed and analysed the pros and cons of each project proposals brought forward by each team member and collectively agreed to working on Navcon to leverage the power of technology to bring good to society.

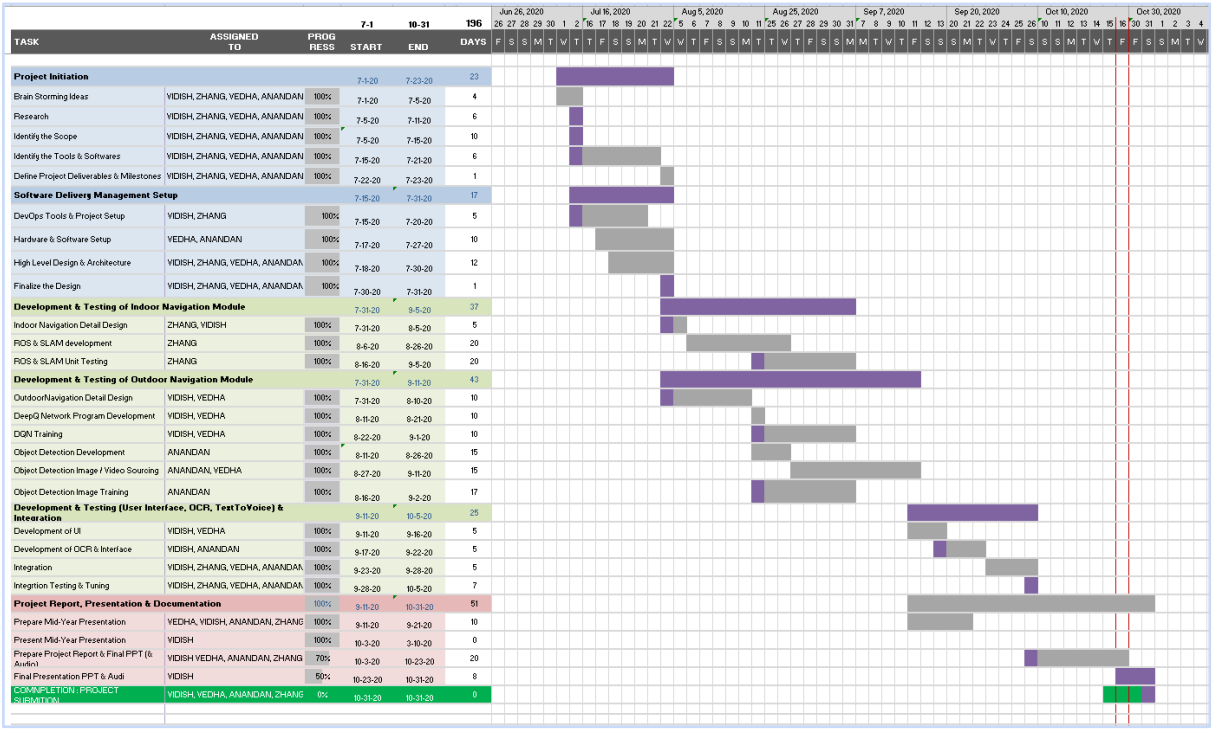
2. **Software Delivery Management & Setup**: During this period, the software and hardware development environment and set-up were made available for all team members to access remotely and contribute their part of the project. Azure DevOps was used to drive our action items and collaborate on project lifecycle. Sprint planning and the built-in reporting helped to properly communicate with team members. We were able to capture the user stories in detail by listing down the tasks required to achieve it. It also helped to boost our team productivity with easier communication, tracking and managing the complex tasks.

3. **Core development:** During this phase, the objective was to develop the entire semi-automated backend object detection pipeline as well as to develop an automated pipeline for image extraction. A key deliverable was to design and develop both the indoor and outdoor navigation module with testing to be carried out during the 2nd-3rd month.

4. **Develop UI, OCR, Common Software Components, Integration & Testing:** An important and final phase where all the common components are developed, integrated and the prototype is tested to confirm the commitment to the minimum viable product prototype. We have set out to achieve this by the 3rd month.

5. **Project Documentation & Presentation:** Finally, the team worked on the necessary documentations and final project presentation to be presented to the panel of professors.

Fig 5 shows the high-level project plan followed for Navcon. The Gantt chart helped us ensure timely and successful delivery of our project.



TASK	ASSIGNED TO	PROGRESS	START	END	DAYS
			7-1	10-31	196
Project Initiation			7-1-20	7-23-20	23
Brain Storming Ideas	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-1-20	7-5-20	4
Research	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-5-20	7-11-20	6
Identify the Scope	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-5-20	7-15-20	10
Identify the Tools & Softwares	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-15-20	7-21-20	6
Define Project Deliverables & Milestones	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-22-20	7-23-20	1
Software Delivery Management Setup			7-15-20	7-31-20	17
DevOps Tools & Project Setup	VIDISH, ZHANG	100%	7-15-20	7-20-20	5
Hardware & Software Setup	VEDHA, ANANDAN	100%	7-17-20	7-27-20	10
High Level Design & Architecture	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-18-20	7-30-20	12
Finalize the Design	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-30-20	7-31-20	1
Development & Testing of Indoor Navigation Module			7-31-20	9-5-20	37
Indoor Navigation Detail Design	ZHANG, VIDISH	100%	7-31-20	8-5-20	5
ROS & SLAM development	ZHANG	100%	8-6-20	8-26-20	20
ROS & SLAM Unit Testing	ZHANG	100%	8-16-20	9-5-20	20
Development & Testing of Outdoor Navigation Module			7-31-20	9-11-20	43
OutdoorNavigation Detail Design	VIDISH, VEDHA	100%	7-31-20	8-10-20	10
DeepQ Network Program Development	VIDISH, VEDHA	100%	8-11-20	8-21-20	10
DQN Training	VIDISH, VEDHA	100%	8-22-20	9-1-20	10
Object Detection Development	ANANDAN	100%	8-11-20	8-26-20	15
Object Detection Image / Video Sourcing	ANANDAN, VEDHA	100%	8-27-20	9-11-20	15
Object Detection Image Training	ANANDAN	100%	8-16-20	9-2-20	17
Development & Testing (User Interface, OCR, TextToVoice) & Integration			9-11-20	10-5-20	25
Development of UI	VIDISH, VEDHA	100%	9-11-20	9-16-20	5
Development of OCR & Interface	VIDISH, ANANDAN	100%	9-17-20	9-22-20	5
Integration	VIDISH, ZHANG, VEDHA, ANANDAN	100%	9-23-20	9-28-20	5
Integrtion Testing & Tuning	VIDISH, ZHANG, VEDHA, ANANDAN	100%	9-28-20	10-5-20	7
Project Report, Presentation & Documentation		100%	9-11-20	10-31-20	51
Prepare Mid-Year Presentation	VEDHA, VIDISH, ANANDAN, ZHANG	100%	9-11-20	9-21-20	10
Present Mid-Year Presentation	VIDISH	100%	10-3-20	3-10-20	0
Prepare Project Report & Final PPT (& Audio)	VIDISH VEDHA, ANANDAN, ZHANG	70%	10-3-20	10-23-20	20
Final Presentation PPT & Audi	VIDISH	50%	10-23-20	10-31-20	8
COMPLETION / PROJECT SUBMITTION	VIDISH, VEDHA, ANANDAN, ZHANG	0%	10-31-20	10-31-20	0

Figure 5: High Level Project plan for Navcon

3.3 Hardware Comparison

Having the right hardware platforms and accelerators is critical to run the GPU intensive navigation and object detection modules of Navcon. Intel Movidius Neural Compute Sticks and Google Coral USB connected to raspberry pi were compared against Nvidia's Jetson Nano on performance and cost criteria. Based on research, the average inference time (number of objects detected per frame) and CPU usage for Jetson Nano were the lowest. Another important characteristic of the Jetson Nano is its high frame processing rate per second as compared to its competitors. Moreover, it's integrability with other hardware systems such as the raspberry pi camera, Wi-Fi adapters and display system made it a perfect choice for Navcon.

Figure 6: Navcon's hardware setup

Parameters	Nvidia Jetson Nano	Google Coral USB	Intel Movidius NCS
Inference time	~38 ms	~ 70 – 92.32 ms	~ 225- 227 ms
fps	~25	~ 9 – 7	~ 4.43 – 4.39
CPU usage	47- 50 %	135%	87 -90 %
Memory usage	32%	8.70%	7%
OS	Ubuntu 18.04 aarch64	Raspbian GNU/License 10 (Buster)	Raspbian GNU/License 9 (Stretch)

Figure 7: Hardware comparison for performance and resource utilization

4: PROJECT SOLUTION

Navcon is a mobile and embedded solution consisting of a 130-degree 8MP FOV camera, 9 inch small and compact display screen, headphones and Wi-Fi adapter with all components connected to the Jetson Nano core module. The solution primarily involves analysing the camera feed frame by frame and processing each frame in Jetson Nano's powerful GPU to provide audio instruction to our visually impaired users. To address the challenges faced by visually impaired in different environments, the project solution is further subdivided into 2 major sections: Indoor and Outdoor Navigation.

4.1 Project Overview

Navcon comprises two main components: outdoor and indoor navigation to cater to different environments. Both outdoor and indoor navigation rely on an object detection and recognition module for intelligent navigation. Additional features also include an optical character recognition system for actively voicing out written information and a stored user profile indicating home address and next of kin contact details for emergency.

4.1.1 Object Detection

Object detection forms an important backbone of Navcon and is actively used in both indoor and outdoor navigation. As our solution is a mobile and embedded, lighter neural network that provide fast prediction time without sacrificing accuracy were preferred. Navcon's object detection model runs on a lightweight SSD Mobilenet V2 architecture providing highly accurate predictions at less computational intensity.

4.1.2 Outdoor Navigation

An important aspect of outdoor navigation is a collision avoidance system. Navcon's collision avoidance comprises a combination of a self-architecture Deep Q network for proactive navigation and a fail-safe inference rule engine. These models operate in synergy and rely on the object detection model for perceiving the environment.

4.1.3 Indoor Navigation

Challenges and obstacles offered in outdoor and indoor navigation vastly differ and so do the solutions. For indoor navigation, Navcon employs a state-of-the-art Visual Simultaneous Localisation and Mapping (VSLAM) system for spatial position mapping and connects to the Robotic Operating System (ROS) for publishing raw data from the object detection model to access pathfinding functionalities.

4.1.4 Optical Character Recognition

Apart from fulfilling the core functionalities of navigation, Navcon's optical character recognition system is designed to voice out visual information from sign boards, letters and even the newspaper ensuring that our users are always well informed.

The information from each of these models is provided as audio instructions to the visually impaired for their appropriate actions and these instructions are logged for safety. Following section displays the system design and architecture providing a high-level overview of how each of these models interact and operate in synergy with each other.

4.2 System Design

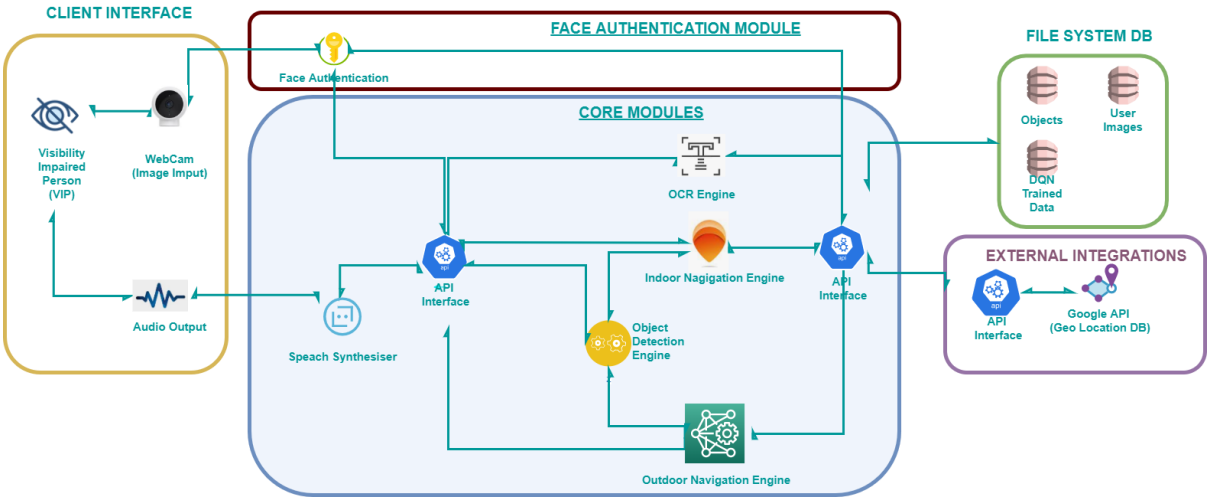


Figure 8: Navcon's system architecture diagram

Navcon's visually impaired users interact with core modules using a specially designed interface (which is compliant with the principles postulated by Stina Oloffson in her study on interfaces for visually impaired) with sufficient audio instruction for usability. The camera attached on Jetson Nano feeds live camera frames to an API interface used to call each of the functionalities. In outdoor spaces, the outdoor navigation engine and object detection engine can be invoked for active navigation whilst in indoor spaces, the indoor navigation engine connected to VSLAM and ROS is invoked for simultaneous mapping and localisation. If the visually impaired user would like to read a document, the OCR engine can be invoked for voicing out the document. Navcon is connected to several database systems including MongoDB for logging historical data as well as to external APIs such as google API for geo positional coordinate and mapping. Moreover, it features a fully integrated facial authentication system for loading users' profile and to login and access Navcon's functionality.

4.2.1 Data Flow Diagram

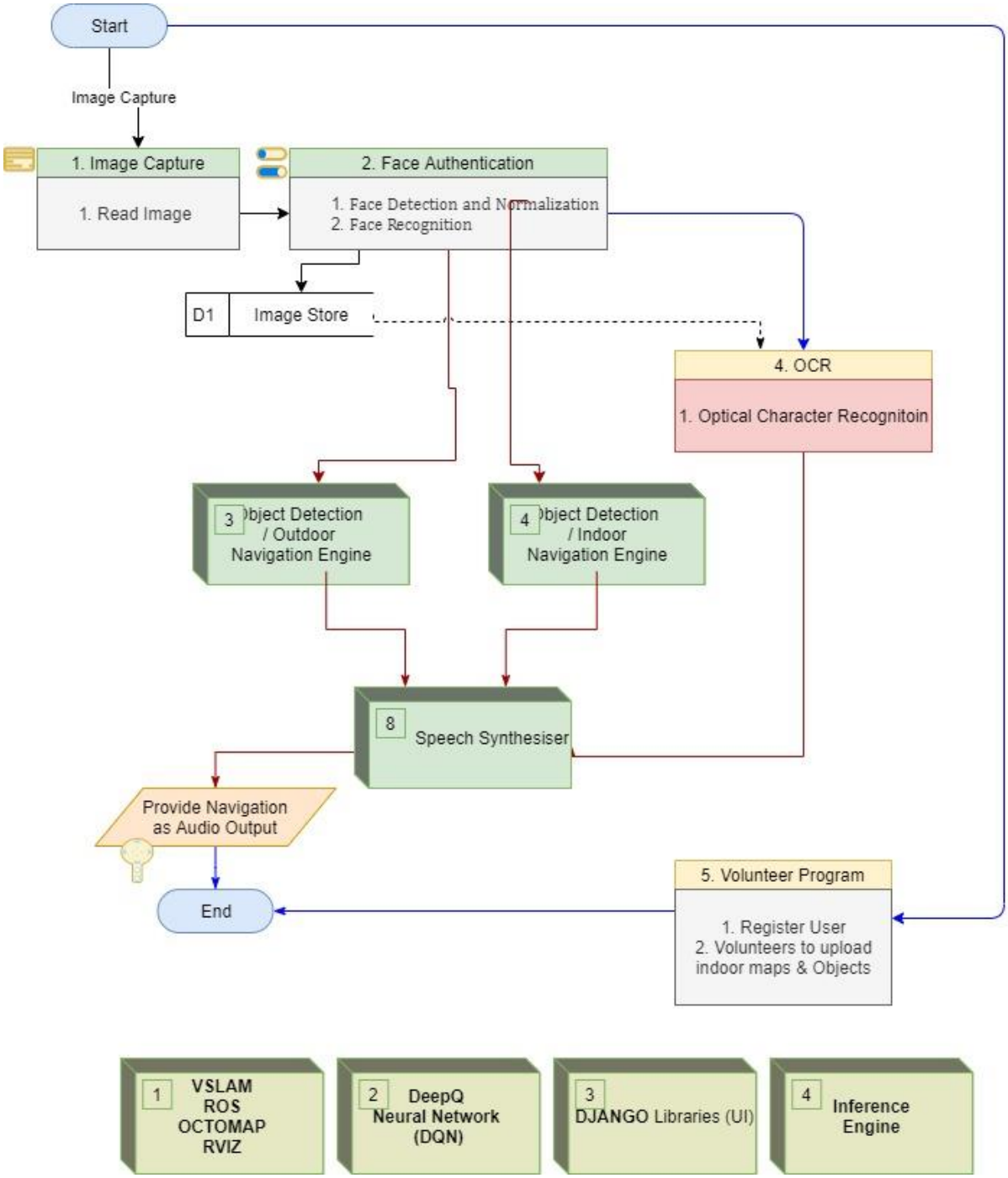


Figure 9: Navcon's data flow diagram

Fig 9 displays the flow of data in Navcon's fully integrated system. The feed to the system is raw images from the camera capture and post authentication, the raw feed data is fed to the different systems. The output from each of the systems is fed to a speech synthesizer that provides audio instructions for our user. Not connected to the core system is Navcon's front-end interface designed to generate traffic and bring awareness

of our mission by inspiring the visually abled to volunteer in the image tagging efforts thereby directly contributing to the object detection model. The following section delves into an in-depth analysis of each of the models used in Navcon and its contribution in achieving Navcon's final mission.

4.2.2 Object Detection Model

At Navcon, we aim to empower our visually impaired friends. The object detection module enables detection of common objects that can guide a user in carrying on with their daily activities. Building on from a large, customized dataset ensured that we covered sufficient ground for most common objects observed in the Singapore context.

As our system is mobile and embedded, choosing the right architecture was critical to the success of the model. We analysed several architectures ranging from faster R-CNN to Efficient Net and concluded that SSD Mobilenet provided the most optimum architecture with the right mix of prediction accuracy and computational resources required for running in our embedded systems. Prior to training the object detection model, it is important to build a comprehensive image dataset which is discussed in the following section.

4.2.2.1 Building Image Dataset

This section explains in detail the steps taken to build the image dataset. It is the initial and most crucial step in the Object Detection model development. Before delving into the details, it is preliminary to perform a study on the open source dataset which are publicly available such as Coco dataset, Kaggle and cityscape dataset. However, after going through several datasets, it was not completely satisfactory for Navcon's design. Since the product is planned to be launched initially in the Asian market primarily focusing on Singapore, the focus needed to be more upon local objects such as the SGD currency, local food delicacies to name a few. Fig 10 shows the schematic diagram to demonstrate the series of steps performed with preparation of the images.

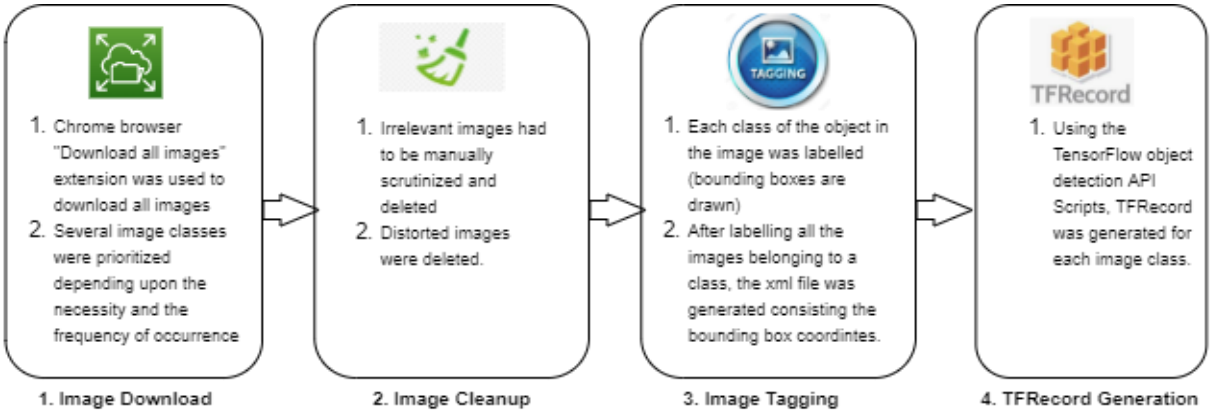


Figure 10: Schematic diagram of building an image dataset

Image Download

As mentioned previously, the available datasets from Kaggle, Coco and cityscape repositories were not enough to support the visually impaired to perform the daily activities and were missing essential items such as trees, staircase, dustbins, potholes and local food delicacies. Several primary objects for detection were listed and downloaded from Google Image through an automated data extraction pipeline. Following is the list of downloaded objects:

SGD Currencies (Notes and Coins)
Local Foods (Nasi Lemak, Biryani, Chicken rice, Bubble tea.)
Local Fruits (Dragon fruit, Durian, Longan, Rambutan.)
Other items (Staircase, dustbins, potholes, trees)

Image Clean-up

As the process of downloading the images is automated, very often downloaded images consist of unwanted or unclear images. Hence a thorough scanning of all the images was performed to ensure all the remaining pictures can be correctly labelled.

Image Tagging

Image tagging is the manual procedure of encompassing objects detected by bounding boxes containing the pixelated coordinates of the object.

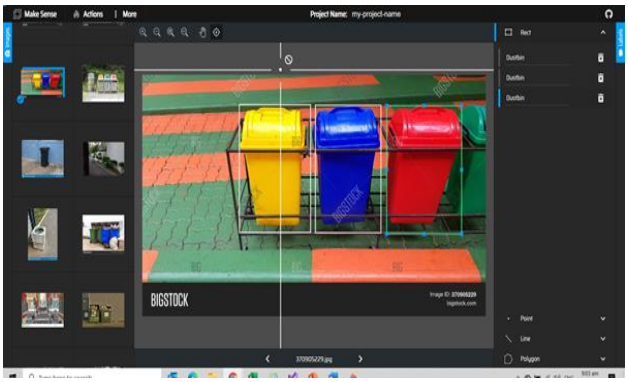


Figure 10: Image tagging using make sense.ai

A combination of make sense AI and LabelImg were used to manually tag the images and through the duration of the entire project more than 3000 images were manually tagged. Navcon looks towards extending the image tagging efforts to the community through virtual volunteering for ensuring sufficient objects are detected for visually impaired individuals to carry on their daily activities independently. The outcome of the image tagging effort is an xml file per image, which needs to be combined for both training and testing datasets and converted to a csv format prior to TF record loading.

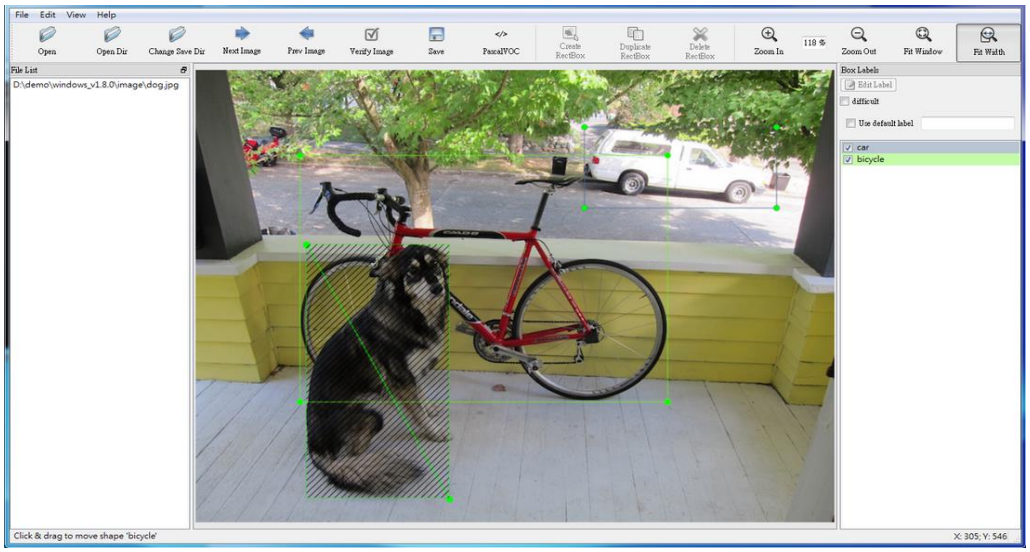


Figure 11: Image Tagging with Labelling

TFRecord Generation

These annotated images that have split into train and test images and the tagged image csv files containing all the bounding box information are converted into a binary format, called TF Record, for model training purposes. These TF Record files are used by the model classifier while training and validating the dataset. The next step involves deciding on the classifier for Navcon.

4.2.2.2 SSD Mobilenet V2

SSD Mobilenet V2 is a light-weight neural net architecture that leverages on the innovations done in its predecessor SSD Mobilenet V1. SSD Mobilenet V1's depth-wise separable convolutions revolutionized model implementation on mobile devices by dramatically reducing the complexity cost and model size of the network. In the SSD Mobilenet V2, a new module allowing for inverted residual structure was introduced to remove non linearities in narrow layers. This module takes an input in low-dimensional compressed representation which is first expanded to a higher dimension and filtered with a lightweight depth wise convolution. Features extracted are subsequently projected back to a lower dimensional representation with linear convolutions. The next

section describes the above features in detail and the integration with the overall architecture.

4.2.2.2.1 Depth Wise Separable Convolutions

In a standard CNN model, convolutions are applied to filter inputs to a new set of output in a single step. Depth wise separable convolutions split convolutions into two steps: first step performing feature extraction on each channel whilst the second step performs feature extraction to understand the relationship across channels. Factorizing the convolutions into dual steps significantly reduces model computations and network size. Fig 12 displays the splitting of the convolutions into feature extraction on each channel and across channels. The time and space complexity and the mathematics behind the depth wise separable convolutions are further discussed in detail in Appendix 7.4.

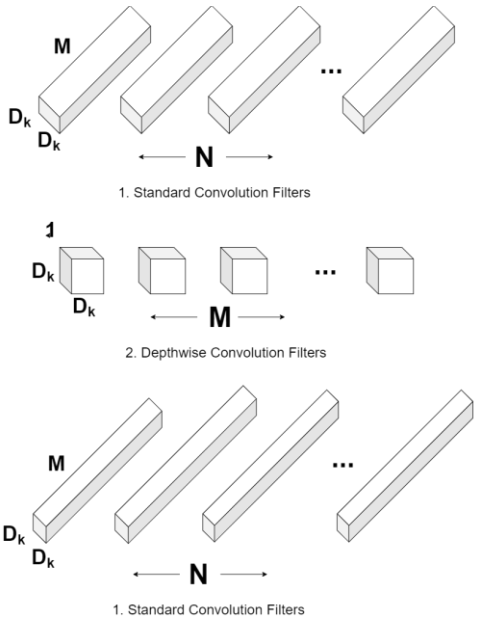


Figure 12: Depth Wise Separable Convolutional Filters

4.2.2.2.2 Bottlenecks and Inverted Residual Model

The inverted residual block is an innovative solution for boosting the gradient backpropagation ability across multiple layers. The use of an inverted design increases efficiency considerably for the error backpropagation. Fig 13 displays the entire architecture of the SSD Mobilenet V2 with depth wise separable convolutions and an inverted residual model to capture valuable information stored in the bottleneck.

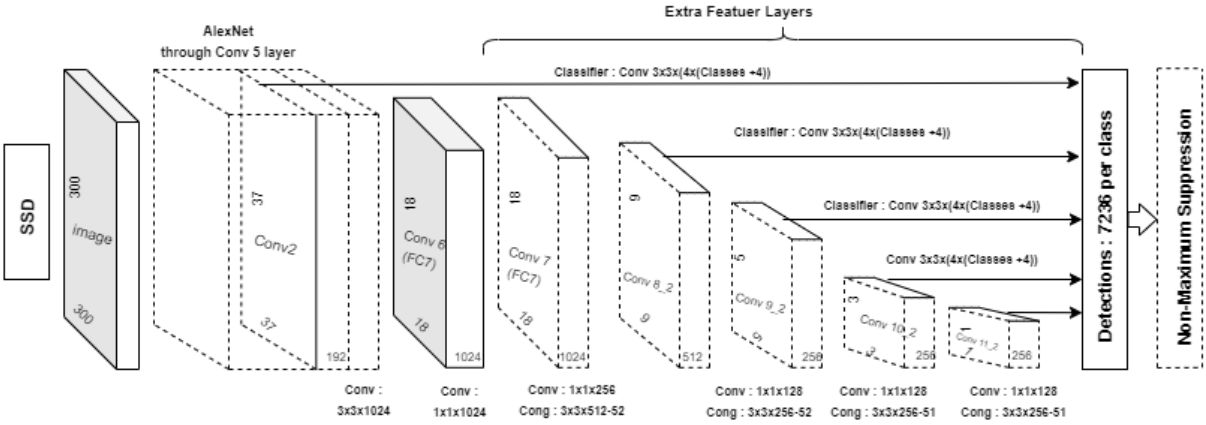


Figure 13: Bottlenecks and Inverted Residual Model

There are several other alternatives to SSD Mobilenet V2 such as the YOLO and faster RCNN models however YOLO is a much lighter neural net often leading to lower accuracy of the detected objects and faster RCNN models take higher average prediction time. SSD Mobilenet V2 offers the advantages of faster average prediction without sacrificing the accuracy of the object detection.

4.2.2.2.3 Model Training

After aligning on the SSD Mobilenet V2 architecture for our object detection model, hand tagged images in tensor flow record format are fed to the classifier for training the model. Prior to starting the classification, the model configuration file is created containing the location of training as well as validation images. Additionally, several objects for identification forming the final output for the neural net and batch sizes are also specified. A sample image of the model configuration file used during one instance of training is shown in Appendix 7.4.

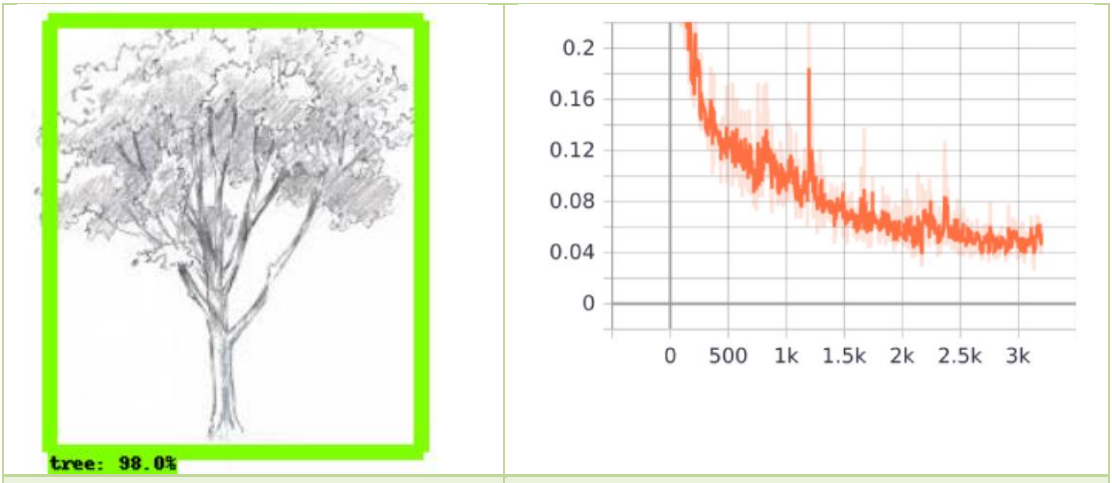
The model configuration file also contains specifications for the use of depth wise separable convolutions, l2 regularizer hyperparameters, batch normalization decay and epsilon rate as well as learning rate for training the neural net. The values used for all these hyperparameters are tabulated in Fig 14 below.

	Hyperparameter	Value
Regularization – L2	Weight (alpha)	0.00004
Model Training Parameters	Kernel Size	3
	Learning Rate	0.004
	Learning Rate Decay Factor	0.95
	Learning Rate Decay Step	800720
	Activation Function	RELU_6
	Optimizer	RMS Optimizer
Batch Normalization	Decay	0.9997
	Epsilon	0.001

Figure 14: Constant hyperparameters used for training the SSD Mobilenet

Apart from the configuration file, the object detection labelling text file is created for one hot encoding of the output by the model. A sample format from one instance of object detection training is shown in the Appendix 7.4.

The model was trained for an average of 80,000 steps using GPU resources from google co-laboratory. Fig 15 shows the detection of custom objects including trees as well as the total classification loss over 3000 epochs. Appendix 7.4 contains a table displaying detection of common indoor and outdoor objects as well as fruits and vegetables required for carrying out essential activities. These images together comprise less than 10% of the total objects that are detected by Navcon models.



Batch Normalization allows recognition of even black and white sketch of a tree to a high accuracy

Classification loss over 3000 epochs

Figure 15: Navcon's object detection model for classifying custom objects and classification results over several epochs

4.2.3 Indoor Navigation

As the challenges and impeding obstacles involved in indoor navigation vastly differ from those in outdoor navigation, a separate system for indoor navigation was imperative. Furthermore, the aim was to build a robust system which can provide autonomous navigation for indoor locations where the GPS cannot function. Based on preliminary research, VSLAM (Visual Simultaneous Localisation and Mapping) provided a great interface for real time positioning and mapping of indoor spaces. Moreover, its integrability with ROS (Robotic Operating System) allowed for leveraging on existing ROS framework of nodes and topics to actively subscribe to front-end raw camera feed of the object detection model and simultaneously publish detected objects in the ROS maps (accessed via RVIZ tool). This assisted in the development of active pathfinding function incorporated into the ROS path planner system. Prior to mapping the indoor spaces, it is important for the feed frames to be calibrated and processed.

4.2.3.1 System Design and Overview

Fig 16 provides a simple system overview for Navcon's indoor navigation. Appendix 7.6 provides a more comprehensive system architecture diagram showing all the precise linkages of all systems.

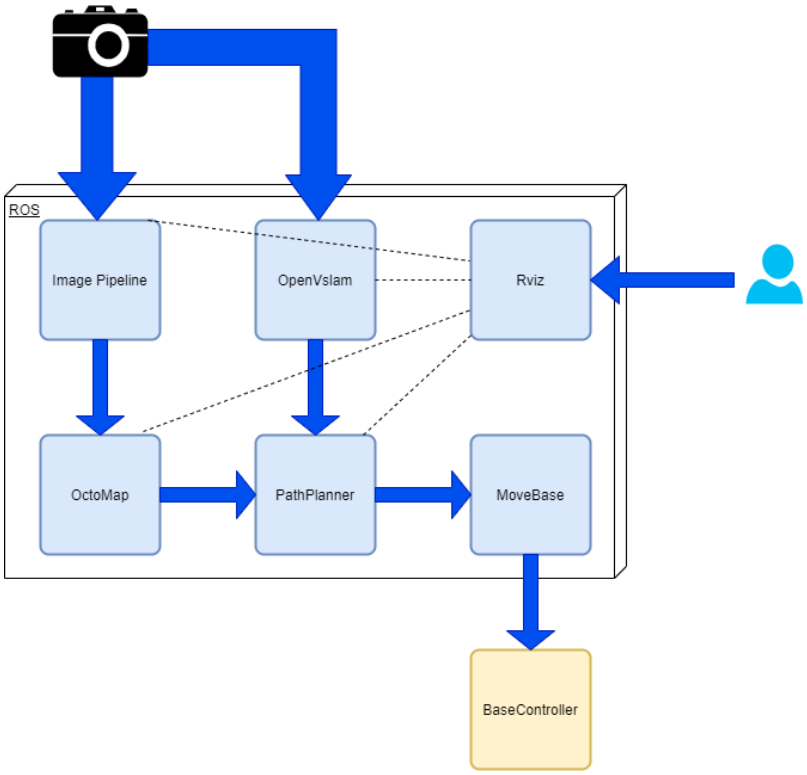


Figure 15: Navcon's system design and overview for indoor navigation model

Images captured from the raw object detection active camera feed are fed to both the image pipeline and open Vslam for publishing the objects detected to the Octomap server (3-Dimensional map interface) as well as for simultaneous localization and mapping respectively. The objects detected and published to the Octomap can be referenced by the path planning functionality for assigning a target object (object that users want to go). Open Vslam provides the visually impaired user's positioning coordinates in real time and both these data sources are required by path finder to optimise the shortest path to the target object. These systems operate simultaneously and are connected to a visualisation tool called Rviz which is a front-facing interface for our visually impaired users.

4.2.3.2 Image Calibration and Processing

Indoor environments occupy a 3-Dimensional representation of our surroundings. Conventional cameras only provide a 2-Dimensional view of the surrounding. Either very specialised and integrated RGB-D (four channel - 3 colour channels and 1 depth channel) camera or stereo images can be employed to formulate an understanding of the depth of the images. Depth Z at any point is observed with a given disparity d, focal f, baseline T

$$Z = \frac{fT}{d}$$

Navcon employs stereo images based on two cameras focusing on the same object (analogous to how human eyes function - each eye a camera) allowing for depth abstraction from the image. The working mathematics behind depth abstraction from stereo images is provided in Appendix 7.6. Below figure represents a stereo image captured during the testing of Navcon's indoor navigation functionality.

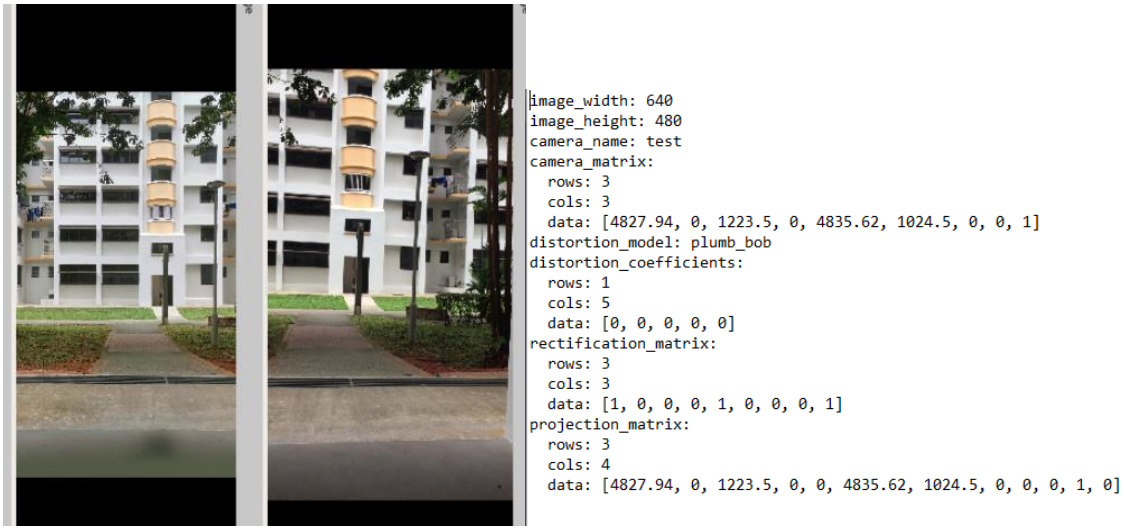


Figure 16: Stereo images from two cameras

Stereo images need to be calibrated for publishing of the point cloud topic, which is used to subscribe and publish the object detected in the live camera feed to the 3-Dimensional Octo map server. Ideal situation for best calibration requires the images from the two cameras to be fed in at the exact same timestamp. The checkerboard test is computed for the camera based on certain camera parameters such as distortion coefficient, rectification matrix and projection matrix. This allows for the calibration and projection of the image in a depth map thereby allowing for depth abstraction from the image. Other frame pre-processing including certain brightness control and frame adjustments are computed internally by the system. The depth information abstracted from the stereo images are published to the Octomap server (3-Dimensional mapping interface). Octomap server stores the coordinate information of all objects detected and is integrated with the path planner for target object tagging.

4.2.3.3 Simultaneous Localisation and Mapping

While the target objects detected in the active environment are tagged by a combination of Octomap server and path finder, the user's real time positional coordinates are mapped by the Open VSLAM architecture. SLAM (simultaneous localization and mapping) is a technique for creating a map of the environment and determining a user's position at the same time. Traditionally, sensor data including sonar and laser were used to determine the positional coordinates by calculating the distance from the time taken for an echo to return. Navcon uses a slam variant called VSLAM which relies on a complex edge detection neural network to map the surrounding environment of the user. ORB-slam is chosen to perform camera relocalisation in real time, keyframe selection and fast insertion while maintaining robust tracking. The movement of edges with reference to the user is directly related to the movement of the user allowing for simultaneous real time positioning and mapping of the indoor space. The outcome of the open vslam is fed to the path planner, thereby providing real time information of the user's positional coordinates.

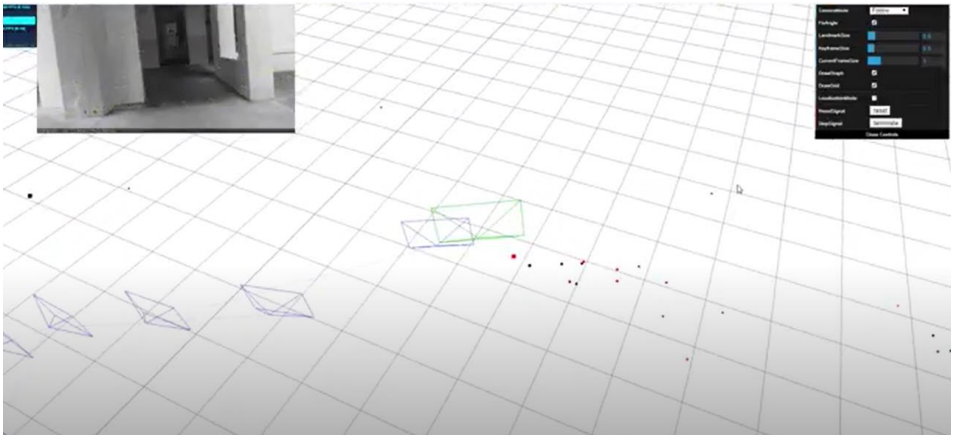


Figure 17: Open VSLAM showing simultaneous localisation and mapping

4.2.3.4 Path Planning & Collision Avoidance

The task of the path planning is to determine the sequence of manoeuvres to be taken by a user to move from starting point to destination avoiding collision with obstacles along the way. The obstacles mapped on to the surface via the stereo image camera are occupied on grid surfaces in the Octomap server and the users coordinate position is mapped using open VSLAM. Once the user decides a target (an object detected by the stereo camera), the path finding optimises the trajectory by finding the shortest line that does not cross any of occupied cells. Visualisation of the shortest path is done on the R-Viz tool and all systems operate within the Robotic Operating System leveraging on the various ROS nodes and topics for subscription and publishing data. Below figure shows the R-Viz tool in action displaying the raw data obtained from open VSLAM as well as the Octomap planner.

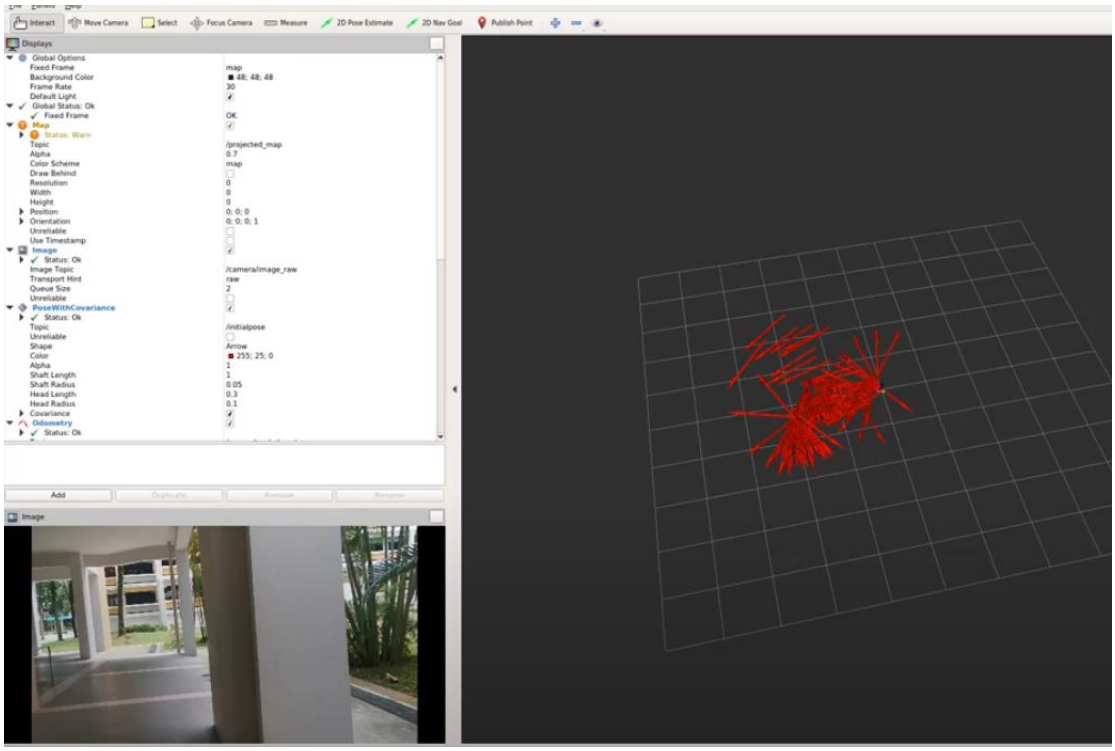


Figure 18: RVIZ visualisation tool displaying user localisation and mapping

4.2.4 Outdoor Navigation

Whilst a combination of Robotic Operating System (ROS) and Visual Simultaneous and Localisation Mapping (VSLAM) assists the visually impaired for indoor navigation and pathfinding functionality, integration of self-developed Deep Q Learning network together with an inference rule engine and GPS system allowed for navigation and collision avoidance in the outdoor environment. Analogous to the indoor navigation, the object detection model is used to detect and recognise objects for collision avoidance. Safety of our users is fundamental and hence, a strong inference rule engine is employed that intelligently alerts our users if they are in close proximity to any detected object for collision avoidance. The following section delves into the development of the outdoor navigation system of Navcon.

4.2.4.1 Collision Avoidance

The collision avoidance system is designed to prevent a collision or reduce the severity of the impact if collision cannot be prevented. As the safety of visually impaired individuals is at stake, it is pertinent that a very strong emphasis is given to the development of a fail-safe collision avoidance system. This is done via a combination of deep Q learning agent and an inference rule engine which are discussed in more detail in the following section.

4.2.4.1.1 Deep Q Learning

Reinforcement learning is a type of machine learning technique that enables an agent to learn in an interactive environment with the aim of maximising the cumulative reward received from the same environment. Deep Q Learning is a specialised form of reinforcement learning combined with a neural network model acting as an agent to predict q-values representing deterministic actions taken by an agent based on the states of the environment.

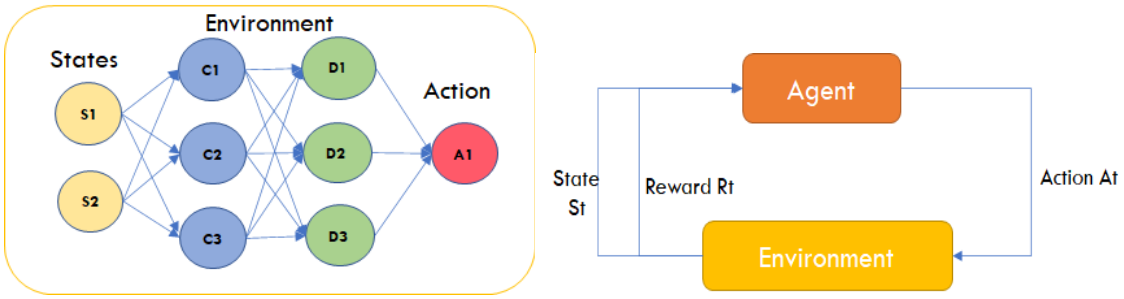


Figure 19: Deep Q Learning Overview

The inspiration of employing deep reinforcement learning for proactive collision avoidance came from the amazing achievements of a trained agent playing Atari games.

The concept is similar whereby the agent playing super mario avoided cataquacks and obstructions in its path to complete the game. The agent arrives at different states as a direct consequence of the actions it takes, which are further fed to the agent to take new actions that aim to improve the cumulative reward. Mathematically, the concept of the deep q learning derives from the bellman's optimality equation which aims to recursively optimise the cumulative reward for each state. Below is a single frame representation of the bellman's equation.

$$Q(s,a) = r(s,a) + \gamma \text{maxQ}(s', a) - 1 \text{ frame}$$

The Q value representing the action taken at state s is a summation of the reward at state s given action a and maximum Q of all past states (s'). Gamma is a discount factor and is normally set to a high value of 0.99.

$$Q(s,a) = r(s,a) + \gamma \text{maxQ}(s', a) + \gamma^2 \text{maxQ}(s', a) + ..$$

Applying the principle of recursion, each frame is cumulatively summed up by applying the single frame bellman's equation. The discount factor reaches multiple order for the older states and thus less importance is given to very old states in the prediction of new q values.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta[R_{t+1} + \gamma \text{maxQ}(s_{t+1}, a) - Q(s_t, a_t)]$$

The above equation is the final equation for the bellman's optimality equation obtained by summation of all the recursive frames. η is the learning rate for the training model. In Navcon, the reinforced agent is the user's frame of reference while the object detection is its active environment.

4.2.4.1.1.1 Environment

In reinforcement learning, the environment represents key tasks that an agent needs to accomplish to maximise its cumulative reward. The environment actively interacts with the agent through actions taken by the agent. In Navcon, the environment is an object detection active environment whereby the agent interacts with it to actively avoid potential collisions with detected objects. The object detection active environment provides the agent with new states for every action taken by the agent and penalises the agent if its movement encroaches the bounding boxes of any detected objects to avoid collisions.

Figure 20: Object detection active environment for the agent to interact

4.2.4.1.1.1 Agent

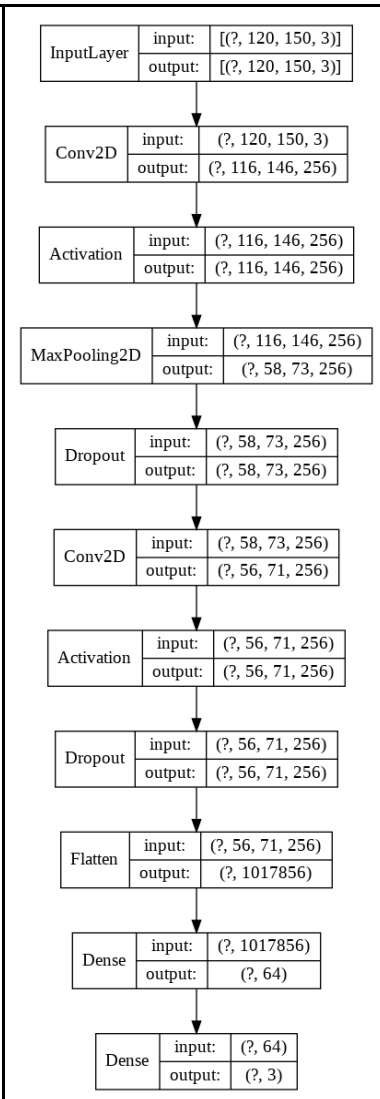
While the object detection model provides an active environment, the modelled agent in deep q learning forms the visually impaired user's frame of reference that interacts with this environment. It is represented as a big block encompassing the human skeleton as shown in figure 21 below. The agent is allowed three deterministic actions which are hot encoded to different q values: 0 is to stay in the current position, 1 is to move right and 2 is to move left. The agent's precise movements at each state were determined through trial and error during testing. As the agent moves, the user is directed to move in the direction until the agent occupies the centre of the frame. In other words, the movement of the agent corresponds to the moment of the visually impaired person. Negative penalty is given to the agent each time it encroaches more than 50% into the bounding box of any detected objects indicative of a potential collision. The agent is also penalised for unnecessary movements so as to prevent giving our users too many mixed instructions.

Figure 21: Agent (User's frame of reference) proactively navigating objects along the way

The agent learns via a self-developed neural network using keras sequential model. The neural net scheme developed for the deep q learning is shown below:

Layer (type)	Output Shape	Param #
conv2d_2 (Conv2D)	(None, 116, 146, 256)	19456
activation_2 (Activation)	(None, 116, 146, 256)	0
max_pooling2d_1 (MaxPooling2	(None, 58, 73, 256)	0
dropout_2 (Dropout)	(None, 58, 73, 256)	0
conv2d_3 (Conv2D)	(None, 56, 71, 256)	590080
activation_3 (Activation)	(None, 56, 71, 256)	0
dropout_3 (Dropout)	(None, 56, 71, 256)	0
flatten_1 (Flatten)	(None, 1017856)	0
dense_1 (Dense)	(None, 64)	65142848
dense_2 (Dense)	(None, 3)	195

Total params: 65,752,579
Trainable params: 65,752,579
Non-trainable params: 0



DQN Agent Model Summary - 65 million trainable parameters

DQN Agent Model plot - single Max Pooling to reduce positional data loss

Figure 22: Deep neural networks for DQN Agent

4.2.4.1.1.1 Challenges and Training

Deep q learning solution houses two powerful neural network models: SSD Mobilenet V2 for the object detection and Deep Q Agent for proactive navigation. As such, the training becomes highly complex and computationally intensive. Despite leveraging on very powerful colab resources, training a deep q network became extremely hard with each frame taking approximately 3 seconds to complete. Moreover, for better model generalisation, it becomes pertinent to train the agent over several videos. Hence, a novel and an innovative solution was employed to train a deep q network by slicing videos into frequency interval frames. Each frame was independently analysed with the position of agent fixed from the previous frame to observe the agent's action. Depending on its action

and the interaction with the environment, the agent was either penalised or let go. Appendix 7.5.1 provides the pseudo code for training a deep q agent for reference.

Training a Deep Q agent is an artistically beautiful process. Firstly, the discount rate in order to give less importance to older states for current actions was set at 0.99. The model was trained for over 200 episodes of the same training environment and monitored for the cumulative rewards at each episode. There are two key phases of the Deep Q Agent training process: Exploratory and Greedy. Initially the agent begins the training process by exploring the training environment. As the training progresses, the agent becomes more and more greedy in its approach to maximise the cumulative reward. The parameter controlling the agent's transition from explorative analysis to a greedy search is epsilon and over the entire training process the epsilon is decayed by a constant factor. The concept of replay memory and experience replay is critical to the training of a dqn network. With experience replay, the agent's experiences in the environment are stored at different time steps called replay memory. The replay memory structures the complete experience of the agent at each time step and consists of a tuple of the agent's current state and action at time step t and its future reward and state at time step t+1 as a direct consequence of the current action.

$$Replay\ Memory = (S_t, A_t, R_{t+1}, S_{t+1})$$

At each step, the agent is trained on random samples from the replay memory and not trained on sequential experiences to break the collinearity across consecutive state action pairs. The training procedure for the agent follow recursively the following procedure (Pseudo code is provided in Appendix 7.5.1):

1) Initialise replay memory capacity
2) Initialise the network with random weights
3) For each episode of training:
    a) Initialise the starting state
    b) For each time step
        i) Select an action: random (exploration), predict (exploitation or greedy search)
        ii) Execute the action
        iii) Observe the reward and state
        iv) Store the experience in the replay memory to be used for subsequent sampling and training.

4.2.4.2 Inference Rule Engine

Navcon uses a complex neural net solution to collision avoidance but it is pertinent to remember that the safety of the visually impaired individuals is paramount. Therefore, a fail-safe solution is essential to avoid any safety incidents that may occur due to an incorrect prediction by the DQN. The fail-safe solution is a strong and powerful inference rule engine with the active environment for the DQN and inference rules being the same. The bounding boxes for the objects detected provide pixel coordinates at each corner. With the size of the image in pixel coordinates known, relative coordinates in x and y axis are obtained. Assuming the jetson nano camera is pointing in the direction of travel and occupying the centre of the frame, these relative coordinates are compared with the mid-x and mid-y coordinates (representing the centre of the frame) and once the intersection of the two coordinates exceed a certain predefined threshold, user is provided with certain remedial actions to prevent collision. Inference rules worked perfectly well with 100% test accuracy and it has the ability to include more complex scenarios such as waiting at a junction that is beyond the comprehension of the DQN. Following are some of the examples where Navcon's complex outdoor navigation system employs inference rules.

Night Walk at Orchard road

Active instructions provided during potentially dangerous situation

Pseudo Code: Person detection

```
For frames in Video:
  For object detected in each frame:
    if object is a person:
      if score of person > 50%:
        if distance < 0.4:
          if mid_x > 0.3 and mid_x < 0.5:
              sound out person ahead!! Please stop
```

Pseudo Code: Car Detection

```
For frames in Video:
  For object detected in each frame:
    if object is a car:
      if score of car > 50%:
        if distance < 0.4:
          if mid_x > 0.7:
              sound out car to the right! Please stay back
```

Figure 23: Inference rule engine for fail safe mechanism against potential collisions

These represent some of the scenarios whereby inference rules were employed to guide our visually impaired friends. Other situations include junction crossing, detection of bicycles, potted plants in the footpath as well as any pothole or cautionary warning signs.

Inference rule engines also have the added advantage of scalability. With more objects initialised in the pipeline, rules can be expanded to cover more scenarios. Furthermore, through weeks of Navcon model training, inference rules were observed to provide best remedial actions in close call situations. The final model used for outdoor navigation is a combination of the DQN agent and inference rules with final say provided by the inference rule engine. DQN agents take extremely high computational resources and the process of training a DQN network is significantly long with model training required on multiple videos for better model generalisation. Whilst the DQN agent is continuously trained over the usage of Navcon device with the model being reinforced with every correct prediction, the inference rule engine provides a highly reliable method to avoid potential collisions.

While both these models have proven to be very effective in outdoor navigation and collision avoidance systems, it is important to understand their limitations. Firstly, these models are restricted by the object detection pipeline. Expanding the object detection pipeline to incorporate more common objects could increase the efficiency of the DQN and inference rule models. To combat this challenge, a front-end interface is developed to source for individuals and corporations for volunteering to hand-tag new objects for incorporation in the final model. This novel idea is designed to provide an avenue for individuals and organisations to virtually contribute towards building the social capital of Singapore. Another limitation to the accuracy of these models is the requirement for multiprocessing of audio instructions. As these audio instructions are provided by speech synthesis word by word, there is a substantial drop in the frame read rates while the instructions are provided. These are potential black spots whereby the collision avoidance models stop working and wait for the audio instructions to be completed. Although this time span is very small, it is important to include asynchronous processing to prevent the audio instructions from breaking down the active navigation models. The final limitation is the hardware limitation whereby the frame rates are restricted to the strength of the hardware. Although we have used the best in class Jetson Nano (128 core Maxwell GPU and 4GB RAM), there is still a limit to its performance and more powerful hardware could only improve the performance of the model. Nevertheless, based on our multiple testing, these models perform extremely well even in the complex environments and can definitely be used as a guidance and control system for outdoor navigation.

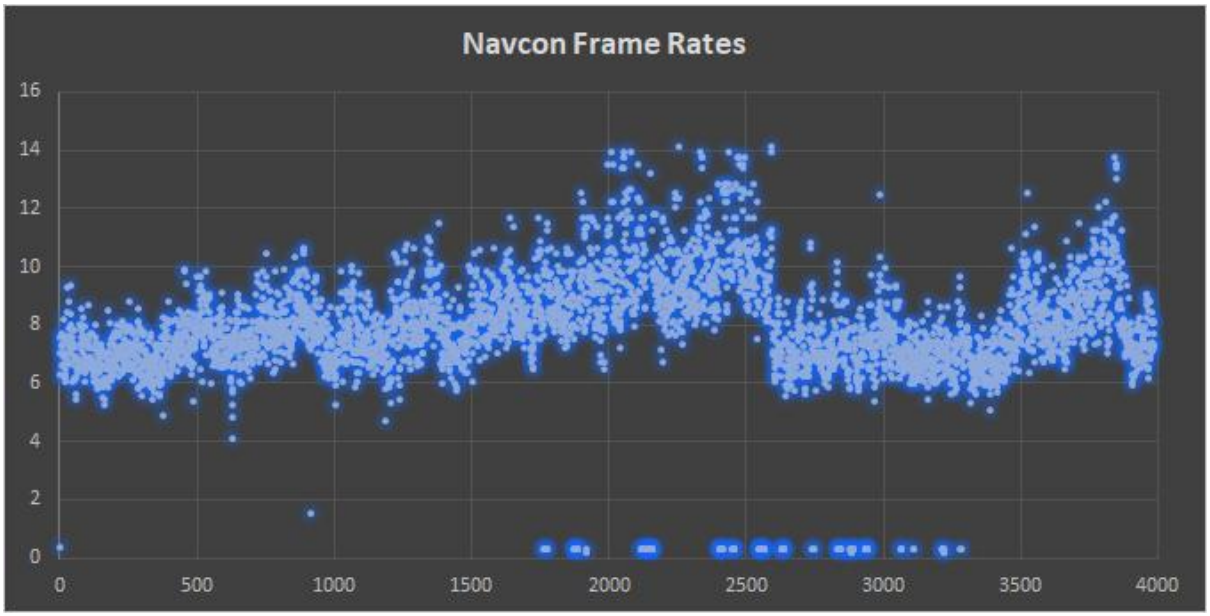


Figure 24: Navcon's consistently high frame rates

4.2.5 Additional Features

4.2.6.1 Text to Speech Generation

This is a very important feature which provides the capability for the device to listen to the text. The ability to see and perceive the world comes naturally to humans. It's second nature to gather information from surroundings through the gift of vision and perception. Navcon's intention is to allow the visually impaired person to think and make independent decisions, hence using this feature it allows him to process his thoughts and make decisions. Say perhaps at a restaurant by being able to listen to the menu independently and place food orders, to pay the bills by ensuring the breakdown and total is mentioned accurately.

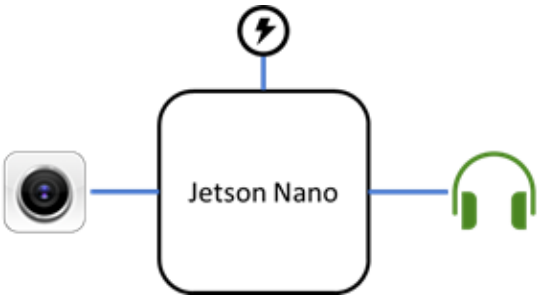


Figure 25: Components Architecture

The device consists of the Jetson Nano connected to a power supply, external camera and Bose headphones. Image captured by the camera is processed by the Jetson Nano to deliver the speech.

Below is the representation of the process to read the text and generate the audio output.

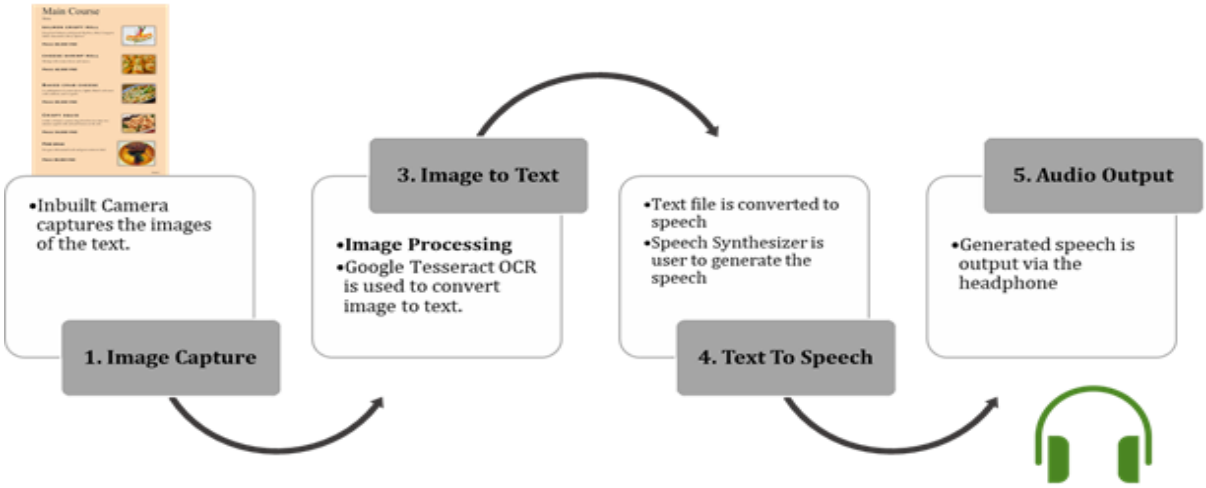


Figure 26: Process Flowchart

1. Image Capture: Inbuilt camera of the device helps to capture the image of the view and this information is passed for processing.

2. Image to Text Conversion: OCR is a field of research in pattern recognition, artificial intelligence and computer vision. It is the conversion of the images consisting of text into a digital text or computer format text. Tesseract 4.0 employs a Long Short Term Memory (LSTM) neural network to determine and classify grouping of words and is used to identify the alphabets in the word for the conversion to digital text. Prior to detecting the words in an image, it runs through an image rectification algorithm which accounts for any image rotation or alignment requirements. Once the OCR process is complete, it returns a string of text.

3. Text to Speech: The process of converting text to speech by a computer is called speech synthesis. A text to speech system (TTS) is used to perform speech synthesis. A TTS is composed of two parts: front end and back end. The front end converts the text to a symbol, for example, a number. Each symbol generated is assigned a phonetic. The back end then converts the phonetic into sound. pyttsx3, a text to speech conversion library from Python was used to perform this task.

4. Audio Output: Generated speech is fed to the Headphone of the visually impaired person to receive the clear audio instructions.

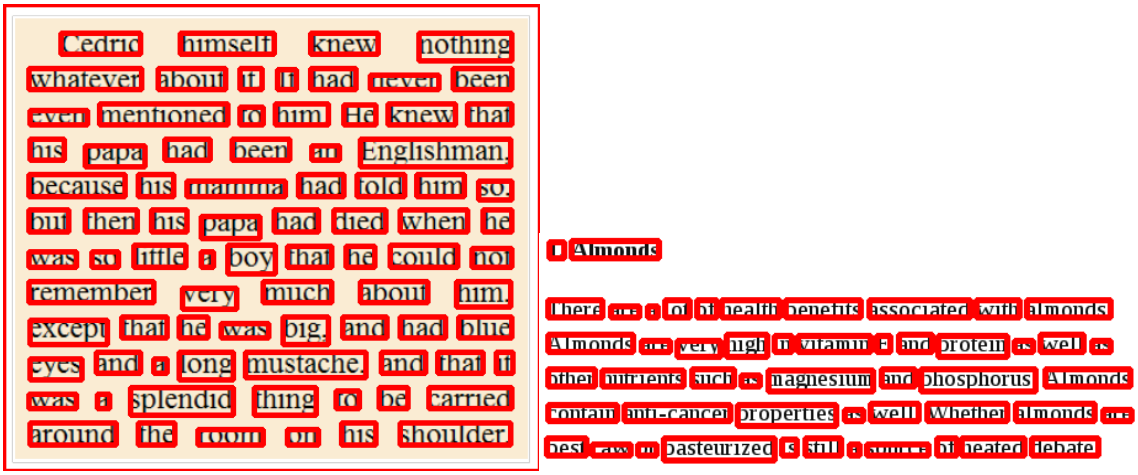


Figure 27: OCR text generation

A major challenge and limitation with the OCR text generation is the image focus. As Navcon's users are visually impaired, it may be a challenge to focus the camera on the text and sometimes could potentially lead to blurry images or cut images preventing the OCR engine from detecting the text. This challenge and limitation can be combated easily through Navcon's smart design considerations. A special location for placing paper of different sizes can be provided with the camera to be placed at the top focus zone. The image is then snapped allowing for a good focus on the textual information for the OCR engine to work its magic.

4.2.6.2 Face Recognition

A real time facial recognition and authentication functionality is provided for the visually impaired individual to initiate the system. There are numerous ways to perform this task, however we narrowed upon a combination of PCA (dimensionality reduction) and SVM (optimal classification).

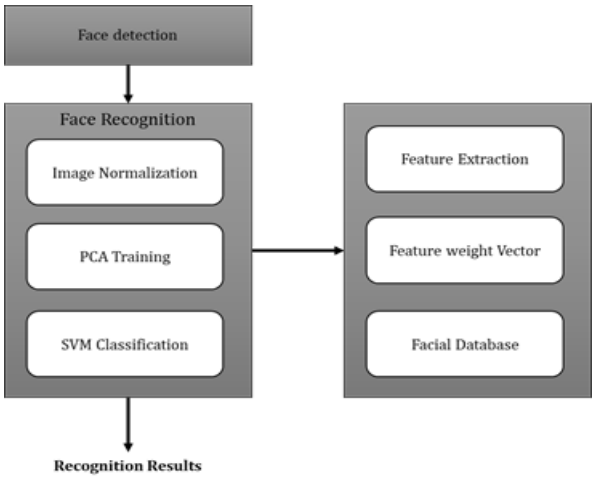


Figure 28: Haar Classifier – Face Detection

Face Detection

A Haar Cascade Classifier can identify a face of a person from a digital image or a video frame from a video source. It uses the Ada-boost learning algorithm which selects a small number of important features from a large set to give an efficient result of classifiers then uses cascading techniques to detect the face in an image. The basic principle of the cascading techniques is to scan the detector many times through the same image and each time with a new size.

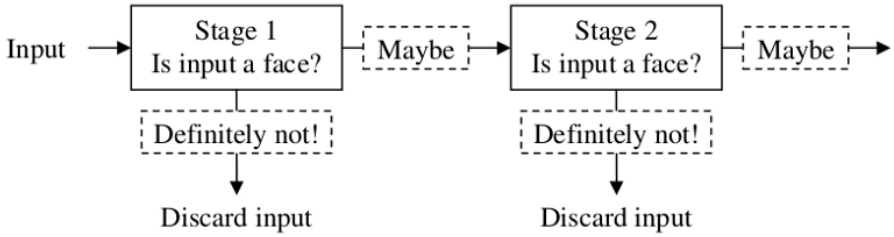


Figure 29: Process flow of face detection

Image Normalization

Dimensions of the facial image is normalized for every face. For a face recognition algorithm to perform well, variances in the face image dimensions needs to be normalized.

Principal Component Analysis (PCA):

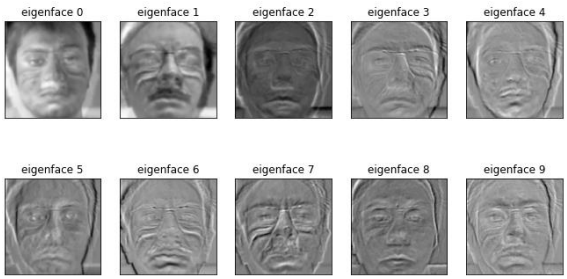
> *"As the number of features or dimensions grows, the amount of data we need to generalize accurately grows exponentially."— Charles Isbell, Professor, School of Interactive Computation, Georgia Tech*

It is used to minimize the feature distribution deviations between different datasets by mapping the high-dimensional features to a low-dimensional subspace. Also, the total computational time is considerably reduced. Hence only suitable features are captured to express the information in lower dimensions. It is a dimensionality reduction that identifies important relationships in our data, transforms the existing data based on these relationships, and then quantifies the importance of these relationships so we can keep the most important relationships and drop the others.

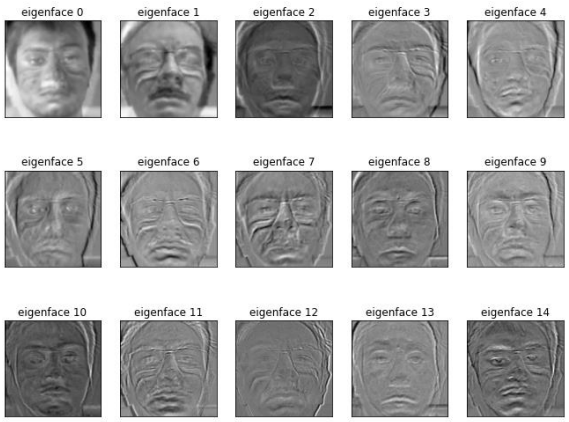
The advantages of PCA in face recognition can be classified as below:

a. Retain the largest information of the projection data in the linear projection
b. Quickly and easily determine the result.
c. It uses the complete face to do feature extraction that could overcome the presence of glasses and the changes of the facial expression

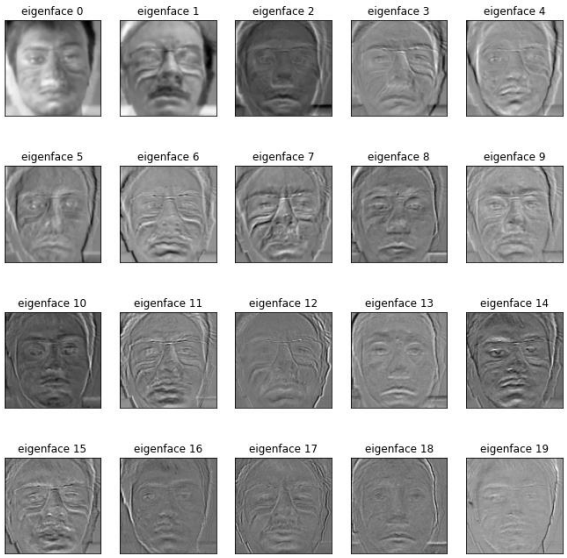
However, the dataset is not huge (webcam is able to capture only 35 images per individual at a time, 3 test users were considered), there was very negligible difference observed in the performance by performing the feature reduction.



PCA with 10 components



PCA with 15 components



PCA with 20 components

Results - PCA with 15 components

Confusion Matrix:

10	*0*	*0*
0	*9*	*0*
0	*0*	*8*

Classification Report:

User ID	*Precision*	*Recall*	*F1-Score*	*Support*
1	*1.00*	*1.00*	*1.00*	*10*
2	*1.00*	*1.00*	*1.00*	*9*
3	*1.00*	*1.00*	*1.00*	*8*

Figure 30: Principal Component Analysis

Support Vector Machine (SVM):

SVM is used to determine if the given facial image belongs to the same person. It is a supervised machine learning algorithm. The goal of the SVM is to train a model that assigns new unseen objects into a particular category. It achieves this by creating a linear partition of the feature space into two categories. Based on the features in the new unseen objects, it places an object above or below the separation plane, leading to a categorisation. It finds the hyperplane that separates the largest possible fraction of points of the same class on the same side, while maximizing the distance from either class to the hyperplane. To train a support vector classifier, we find the maximal margin hyperplane, or optimal separating hyperplane, which optimally separates the two classes in order to generalize to new data and make accurate classification predictions.

Grid search is commonly used as an approach to hyper-parameter tuning that will methodically build and evaluate a model for each combination of algorithm parameters specified in a grid. Grid-

search, set up a parameter grid (using multiples of 10's is a good place to start) and then pass the algorithm, parameter grid and number of cross validations to the Grid-SearchCV method. The parameter grid will also include the kernel details such as the Radial Basis Function (RBF) kernel.

Grid Search SVC classifier with Radial Basis Function (RBF) kernel, was able to achieve global optimization and improve the accuracy of face verification. The faces of the same individual may be very differently impacted from the variance pose, illumination, expression, and occlusion. Therefore, it is crucial to reduce the intraclass variations while enlarging the inter-class differences for face verification.

In addition, to improve the model accuracy, several parameters need to be tuned which include:

1. Kernels: The main function of the kernel is to take low dimensional input space and transform it into a higher-dimensional space. It is mostly useful in non-linear separation problems.

Figure 31:SVM Kernels

2. Gamma: It defines how far influences the calculation of plausible line of separation. If gamma is higher, nearby points will have high influence; low gamma means far away points also be considered to get the decision boundary.

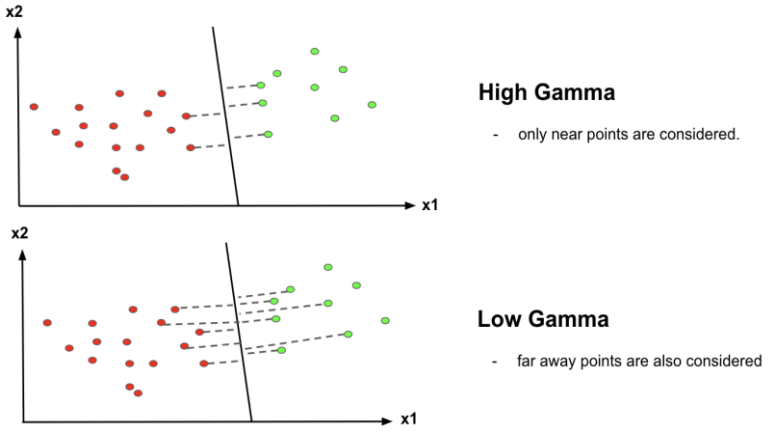


Figure 32: SVM Gamma

3. C (Regularisation): C is the penalty parameter, which represents misclassification or error term. The misclassification or error term tells the SVM optimisation how much error is bearable. This is how you can control the trade-off between decision boundary and misclassification term. If C is high it will classify all the data points correctly, also there is a chance to overfit.

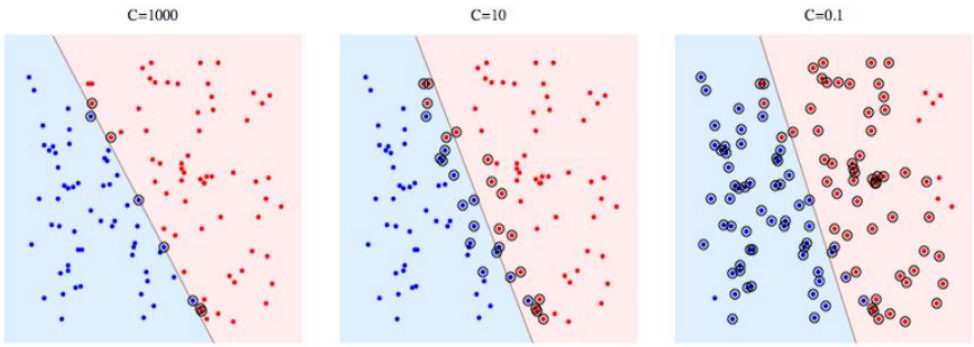


Figure 33: SVM Regularisation

4.2.6.2 Frontend Framework

Marketing of any product is fundamental to its success. The objective behind the Navcon's front-end framework is to expedite the object detection pipeline by providing an avenue for individuals and corporations to do virtual volunteering by hand tagging new objects. Navcon's frontend framework is designed to engage with potential volunteers and to drive the process of image tagging, most manual and tedious process in object detection pipeline. Furthermore, the frontend framework is also developed to generate greater awareness of the global health crisis surrounding visual impairment. As it is encouraging for volunteers to observe their work getting translated, a live demonstration of Navcon is provided within the web framework.

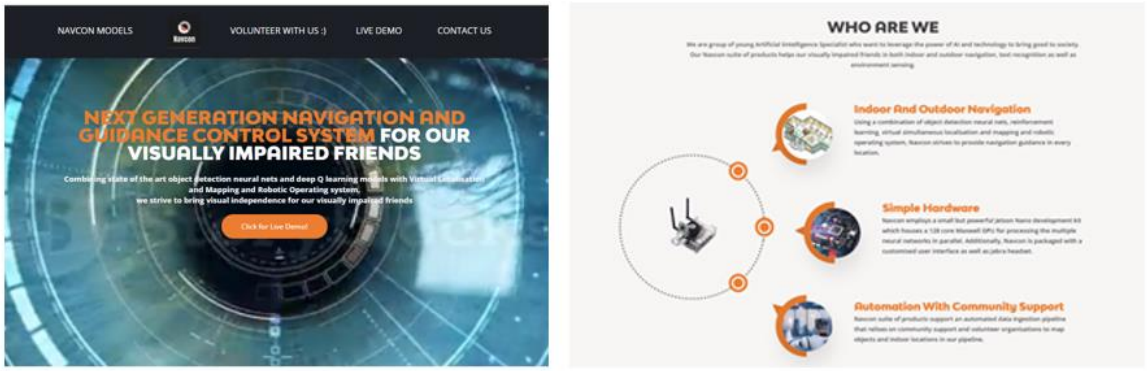


Figure 34: Home page of Navcon

The home page mainly contains information of the three different Navcon subsystems: namely outdoor and indoor navigation and object detection. It's filled with interactive videos displaying the strength of these models as navigation and guidance control systems.

4.2.6.2.1 Volunteer with Us:

As the object detection pipeline is actively used for both indoor and outdoor navigation, having more objects identified can tremendously improve model performance. In current day and age, virtual volunteering has become very popular with individuals and corporations as it provides the flexibility of volunteering at one's convenience. Virtual volunteering was introduced to combat the challenges observed in manual tagging of images. After painstakingly hand-tagging over 3000+ images covering various datasets for several hours, the importance of support from the community to scale Navcon became apparent. With the support provided from volunteering organisations, time to training for image datasets can be significantly improved and Navcon's model can be scaled. Figure 35 shows the volunteer sign-up and login page in our frontend systems. These link directly to a SQLite database where all information regarding our volunteers are stored.

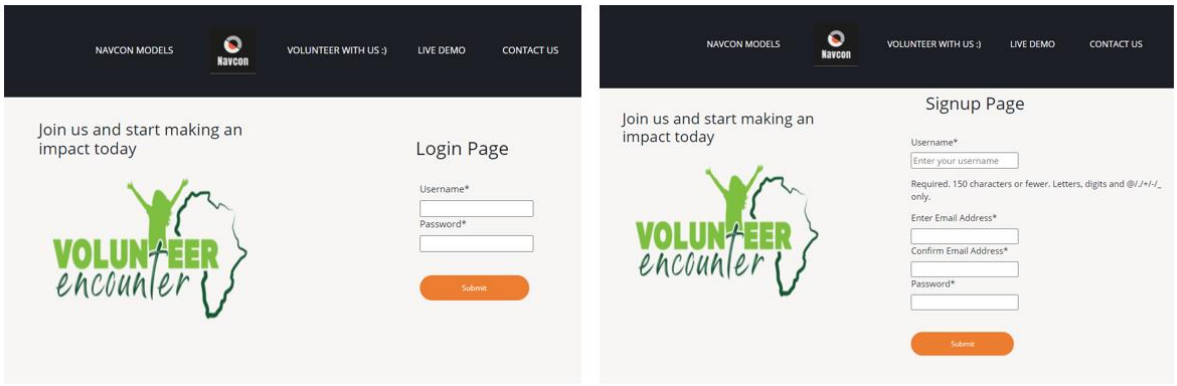


Figure 35: Login and Signup page for volunteers at Navcon

Upon logging in, volunteers can see instructions and a list of objects that are pending image tagging. They may use Navcon's inbuilt auto image downloader to get 200 images stored to their local directory or download plugins for automated image installation. LabelImg is the integrated tagging tool for our volunteers to tag images. Once completed, the generated xml file along with the images are to be stored in google drive shared folders linking them with our automated object detection pipeline. Moreover, there are mass volunteering session links provided allowing for interactive volunteering over zoom calls.

4.2.6.2.1 Live Demo

The interactive live demo allows volunteers to directly feel the impact of their contribution. The demo page displays Navcon from the perspective of its visually impaired users. The audio instruction panel displays audio instructions provided to the users whilst the admin panel provides an interface for the visually impaired to access the different functionalities of Navcon. Volunteers are free to play around with the facial authentication systems and can see the different models live in action. Figure 36 shows the demo page displaying all Navcon's model live in action

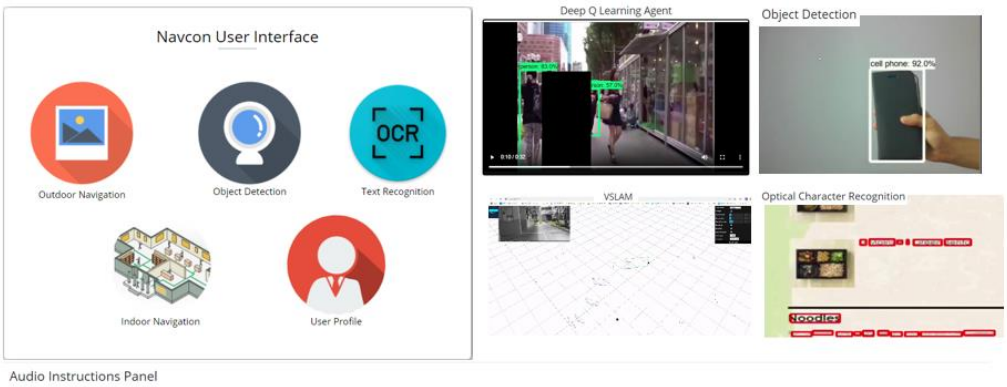


Figure 36: Navcon front-end interface and audio instruction panel

5. FUTURE WORK

From its inception, Navcon's model were designed especially for the VIP's to carry out their daily activities independently. There are very strong opportunities to further develop our system by incorporating Logitech Dual camera capable of both depth abstraction and stability. This will directly correlate to an improvement in performance of our simultaneous localisation and mapping system in indoor navigation. On the outdoor navigation front, asynchronous speech synthesis could prevent potential blind spots during navigation.

Future work for Navcon also involves continuous outreach programme for community and corporates to become part of our mission. This will ensure a strong object detection pipeline with models capable of identifying all objects in both indoor and outdoor environment. Another potential enhancement for Navcon will be incorporation of caption bot which provides a descriptive analysis of the environment VIP's are on a real time basis. Upon completion of these enhancements, Navcon device can be provided to a VIP through collaboration with Singapore Association of Visually Handicapped (SAVH). The feedback from an active user will further help Navcon in its pursuit of continual development.

6. CONCLUSION

In conclusion, Navcon addresses a key global health issue and leverages the power of advancing technology to provide a solution for visually impaired individuals to independently carry on their daily activities. Navcon's strong object detection pipeline and collaboration with community and corporates ensure that VIP's will not miss a single object. Navcon's strongly designed architecture in both outdoor and indoor navigation prioritise on VIP's safety and ensure fail safe collision avoidance while they carry on with their daily activities. Although Navcon has achieved all its deliverable, there are plethora of opportunities for further optimising the system in terms of speed and accuracy. With International Sight Day celebrated on 8th October, please join us in our mission to make their vision count.

7. APPENDIX

The appendix section of the project document contains the initial project proposal submitted and discussed with Dr Zhu, system and functionality mapping to course notes of pattern recognition using machine learning, individual reflection on the project by all team members, images (such as training logs, xml, csv tagged datasets) referred for further technical understanding including pseudo codes and references for the completion of the project. Miscellaneous items including meeting minutes for our semi-weekly meetings are omitted and provided separately with the project.

7.1 Initial Project Proposal

During the initial phases of project initiation, our team collectively brainstormed many ideas and one thought struck us: the power of technology to enrich life by using our knowledge learnt in graduate certificate of pattern recognition to design a system that could allow visually impaired individuals to be aware of their surroundings and thereby independently complete their daily activities. After a comprehensive market research on the available products, our team was convinced that competitor products were only good in a niche segment (for e.g. object detection or optical character recognition) but none offered a combination of services very often required for the completion of several daily activities (for e.g, navigation and object detection for shopping). Navcon aimed to fill this market vacuum empowering the visually impaired to be independent.

There were several design and hardware considerations collectively brainstormed for Navcon. It was important that the system be portable and lightweight, cost-effective to be cost competitive against our competitors as well as sufficiently powerful to support complex neural network models. After giving much thought and consideration, we concluded to procure NVIDIA's Jetson Nano as the core engine for Navcon. We sourced other accessories such as 130 FOV 8 MP camera, an 8-inch display screen and Wi-Fi dongle for our machine. Similar to the final product, the feed to the system was the raw input images from the camera system and the output was the audio instructions provided to our end users. Initially the scope of the project included only a real time object detection system for outdoor navigation with a text to speech converter for providing the audio instructions. The core idea for indoor navigation had just started to take shape but was not formalised.

During the third week of semester, we pitched Navcon's idea to Dr Zhu and it was extremely well received. He also encouraged our team to pursue the project and provided great guidance during our subsequent meetings. It was extremely gratifying to receive an outpouring of positive reviews for our mid project presentation from our fellow classmates and many contacted us after class wanting to know more about our models. We hope our final idea and project serves as an inspiration to fellow AI specialists to leverage the power of technology to bring good to the world.

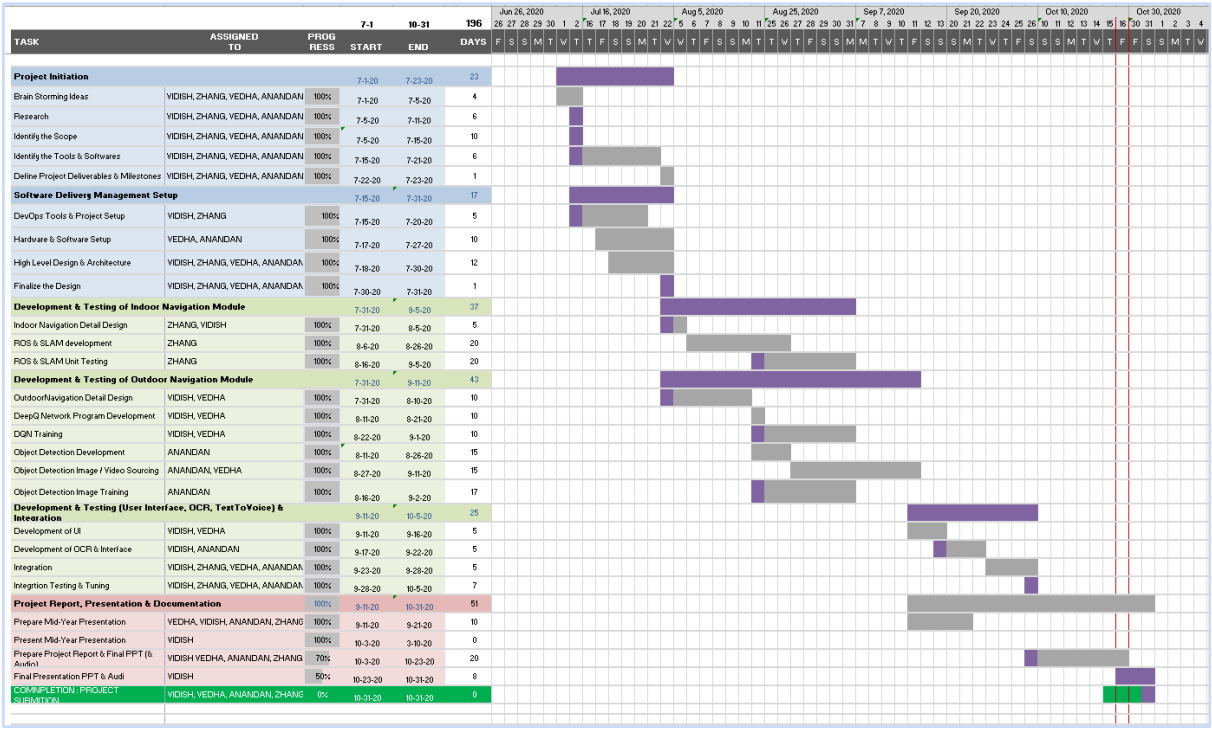
7.2 System and Functionality Mapping

Navcon employs materials taught from all the modules of Graduate Certification for Pattern recognition systems.

Feature	Mapped Functionalities
Object detection model	Supervised learning Convolutional Neural Network (CNN) framework for object recognition that maps to **Pattern Recognition and Machine Learning Systems - Neural network models and designs.**
Outdoor Navigation	DQN agent is another supervised convolutional neural network created from self-designed architecture that maps well to **Pattern Recognition and Machine Learning Systems** The interpretability of the inference rule engine maps well to the certain rule based machine learning systems taught in **Problem Solving using Pattern Recognition** Moreover, the framework used for outdoor navigation also matches the intelligent sense making pipeline published by MIT as taught in the **Intelligent Sensing and Sense Making** Frame read analysis for active outdoor navigation maps well to **Intelligent Sensing and Sense Making**
Indoor Navigation	VSLAM (Visually Simultaneous localisation and mapping) employs a strong edge detection neural network based of an existing architecture that maps well to **Pattern Recognition and Machine Learning Systems** Although Robotic Operating Systems (ROS), including ROS nodes, ROS topics and rviz do not map well to current graduate certificate, knowledge learnt in ROS directly translates to the Graduate Certificate in Intelligent Robotics system.
Facial Authentication	Facial authentication system employs a combination of Principal Component Analysis (PCA) and Support Vector Machine to create a linear classifier for higher dimensional data and maps extremely well to techniques taught in **Problem Solving using Pattern Recognition as well as Intelligent Sensing and Sense Making**
Optical Character Recognition	Optical character recognition employs a Long Short-Term Memory (LSTM) model to accurately detect words/numbers in a sentence mapping the OCR model extremely well to **Problem Solving using Pattern Recognition**

Figure 37: Functionality mapping of each of Navcon's models

7.3 High Level Project Plan



TASK	ASSIGNED TO	PROGRESS	START	END	DAYS
			7-1	10-31	196
Project Initiation			7-1-20	7-23-20	23
Brain Storming Ideas	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-1-20	7-5-20	4
Research	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-5-20	7-11-20	6
Identify the Scope	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-5-20	7-15-20	10
Identify the Tools & Softwares	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-15-20	7-21-20	6
Define Project Deliverables & Milestones	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-22-20	7-23-20	1
Software Delivery Management Setup			7-15-20	7-31-20	17
DevOps Tools & Project Setup	VIDISH, ZHANG	100%	7-15-20	7-20-20	5
Hardware & Software Setup	VEDHA, ANANDAN	100%	7-17-20	7-27-20	10
High Level Design & Architecture	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-18-20	7-30-20	12
Finalize the Design	VIDISH, ZHANG, VEDHA, ANANDAN	100%	7-30-20	7-31-20	1
Development & Testing of Indoor Navigation Module			7-31-20	9-5-20	37
Indoor Navigation Detail Design	ZHANG, VIDISH	100%	7-31-20	8-5-20	5
ROS & SLAM development	ZHANG	100%	8-6-20	8-26-20	20
ROS & SLAM Unit Testing	ZHANG	100%	8-16-20	9-5-20	20
Development & Testing of Outdoor Navigation Module			7-31-20	9-11-20	43
OutdoorNavigation Detail Design	VIDISH, VEDHA	100%	7-31-20	8-10-20	10
DeepQ Network Program Development	VIDISH, VEDHA	100%	8-11-20	8-21-20	10
DQN Training	VIDISH, VEDHA	100%	8-22-20	9-1-20	10
Object Detection Development	ANANDAN	100%	8-11-20	8-26-20	15
Object Detection Image / Video Sourcing	ANANDAN, VEDHA	100%	8-27-20	9-11-20	15
Object Detection Image Training	ANANDAN	100%	8-16-20	9-2-20	17
Development & Testing (User Interface, OCR, TextToVoice) & Integration			9-11-20	10-5-20	25
Development of UI	VIDISH, VEDHA	100%	9-11-20	9-16-20	5
Development of OCR & Interface	VIDISH, ANANDAN	100%	9-17-20	9-22-20	5
Integration	VIDISH, ZHANG, VEDHA, ANANDAN	100%	9-23-20	9-28-20	5
Integrtion Testing & Tuning	VIDISH, ZHANG, VEDHA, ANANDAN	100%	9-29-20	10-5-20	7
Project Report, Presentation & Documentation		100%	9-11-20	10-31-20	51
Prepare Mid-Year Presentation	VEDHA, VIDISH, ANANDAN, ZHANG	100%	9-11-20	9-21-20	10
Present Mid-Year Presentation	VIDISH	100%	10-3-20	10-3-20	0
Prepare Project Report & Final PPT (& Audio)	VIDISH VEDHA, ANANDAN, ZHANG	70%	10-3-20	10-23-20	20
Final Presentation PPT & Audi	VIDISH	50%	10-23-20	10-31-20	8
COMPLETION : PROJECT SUBMITTION	VIDISH, VEDHA, ANANDAN, ZHANG	0%	10-31-20	10-31-20	0

Figure 38: High Level Project plan

7.4 Object Detection

The mathematics behind the lower computational cost of the depthwise separable convolutions is provided below.

As a comparison, in a standard convolutional filter with stride one and padding, the filter size is given by where is the image dimension in x-axis, is the image dimension in y axis and is the number of channels (3 for RGB input frames). Mathematically, the output from the convolution can be represented as:

$$G_{k,l,n} = \sum_{i=0}^{i} \sum_{j=0}^{j} \sum_{m=0}^{m} K_{i,j,m,n} \circ F_{k+i-1,l+j,m}$$

Hence, the total computational cost for standard convolutional network is $F_x \times F_y \times N \times M \times O_x \times O_y$ (1); where $O_x \times O_y$ are the output dimensions and M is the output channel.
In contrast, the use of depthwise convolutions simplifies the output from the first convolutions to:

$$G_{k,l,m} = \sum_{i=0}^{i} \sum_{j=0}^{j} K_{i,j,m} \circ F_{k+i-1,l+j,m}$$

With this the total cost of the depthwise convolutions is $F_x \times F_y \times M \times O_x \times O_y$. Notice that the N, representing the number of input channel is missing from total depthwise cost. The next step is the pointwise convolutions which computes a linear combination of depthwise convolutions via 1X1 kernel filter. The cost of pointwise convolutions can then be represented as $N \times M \times O_x \times O_y$. Together, these convolutions are called the depthwise separable convolutions with a total cost of:

$$F_x \times F_y \times M \times O_x \times O_y + N \times M \times O_x \times O_y$$
$$M \times O_x \times O_y \times (F_x \times F_y + N) - (2)$$

Evidently comparing equations (1) and (2), the total cost of depthwise separable convolutions is significantly lower (8 to 9 times) making it perfectly applicable for our mobile embedded solution.
The depth wise separable convolution is a novel concept that was first introduced in the SSD mobilenet V1 and has been adopted in the SSD mobilenet V2.

This section consists of training logs, xml, csv tagged datasets referred for further technical understanding on the object detection model. As mentioned in the image data processing, post tagging of the images with the bounding box, the images are first saved as xml and are later converted to csv format for final conversion to tensor record binary encoding. Below images are an xml and csv data for training images for the tree object.

```
<annotation>
        <folder>my-project-name</folder>
        <filename>_108667012_affricifte-nb-184762.jpg</filename>
        <path>/my-project-name/_108667012_affricifte-nb-184762.jpg</path>
        <source>
                <database>Unspecified</database>
        </source>
        <size>
                <width>976</width>
                <height>849</height>
                <depth>3</depth>
        </size>
        <object>
                <name>tree</name>
                <pose>Unspecified</pose>
                <truncated>Unspecified</truncated>
                <difficult>Unspecified</difficult>
                <bndbox>
                        <xmin>85</xmin>
                        <ymin>109</ymin>
                        <xmax>807</xmax>
                        <ymax>849</ymax>
                </bndbox>
        </object>
</annotation>
```

Figure 39: Single tagged tree image xml

```
filename,width,height,class,xmin,ymin,xmax,ymax
images268.jpg,275,183,tree,97,24,170,157
images84.jpg,100,100,tree,4,3,94,100
images64.jpg,275,183,tree,15,4,128,178
images176.jpg,309,163,tree,35,3,145,162
image26.jpeg,260,194,tree,6,50,46,156
image26.jpeg,260,194,tree,168,37,259,171
image1.png,100,100,tree,7,12,100,95
images341.jpg,225,225,tree,41,13,148,224
images309.jpg,300,168,tree,83,6,219,140
images111.jpg,300,168,tree,113,7,222,164
images325.jpg,300,168,tree,4,3,214,159
images327.jpg,301,167,tree,131,5,297,166
images246.jpg,211,239,tree,13,8,203,192
107667228_beech-tree-NEWS-xlarge_trans_NvBQzQNjv4BqplGOf-dgG3z4gg9owgQTXEmhb5tXCQRHAvHRWfzHzHk.jpg,1280,799,tree,477,73,1197,684
images63.jpg,300,168,tree,73,4,217,168
images31.jpg,252,200,tree,40,12,215,200
images291.jpg,275,183,tree,85,8,200,181
images62.jpg,300,168,tree,27,4,279,168
images230.jpg,275,183,tree,1,9,79,104
```

Figure 40: Training data for tagged images in the CSV format.

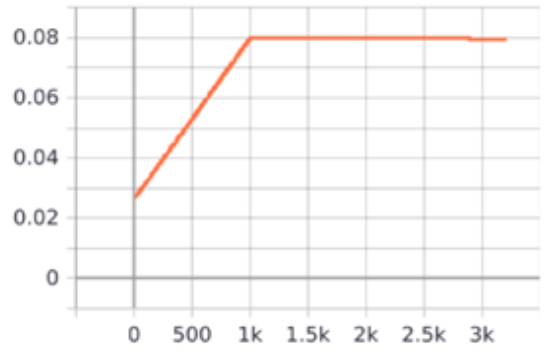
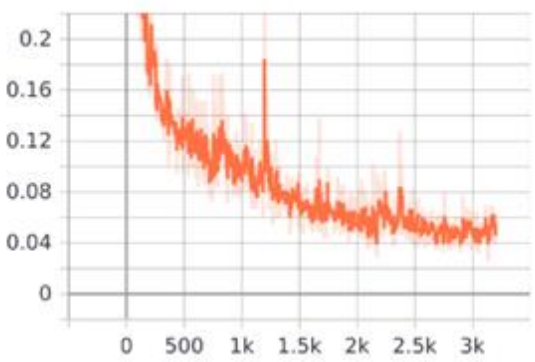
The object detection file contains the list of items for a single training and the configuration file links the object detection, training and testing TensorFlow records as well as the SSD Mobilenet model together and provides configurable parameters such as batch size, learning rate coefficient and even batch normalisation options.

```
item {
  id: 1
  name: 'tree'
}
```

```
# SSD with Mobilenet v2 FPN-lite (go/fpn-lite) feature extractor, shared box
# predictor and focal loss (a mobile version of Retinanet).
# Retinanet: see Lin et al, https://arxiv.org/abs/1708.02002
# Trained on COCO, initialized from Imagenet classification checkpoint
# Train on TPU-8
#
# Achieves 22.2 mAP on COCO17 Val

model {
  ssd {
    inplace_batchnorm_update: true
    freeze_batchnorm: false
    num_classes: 1
    box_coder {
      faster_rcnn_box_coder {
        y_scale: 10.0
        x_scale: 10.0
        height_scale: 5.0
        width_scale: 5.0
      }
    }
    matcher {
      argmax_matcher {
        matched_threshold: 0.5
        unmatched_threshold: 0.5
        ignore_thresholds: false
        negatives_lower_than_unmatched: true
        force_match_for_each_row: true
        use_matmul_gather: true
```

The process of training can itself be recorded using a tensor board. Below is an example of an instance of a tensor board instance while training the tree dataset.

Learning Rate vs Epochs	Classification Loss vs Epochs
	

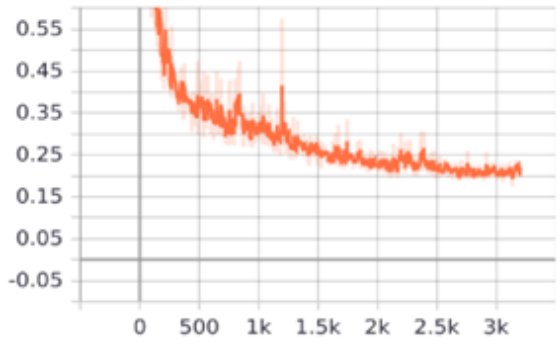
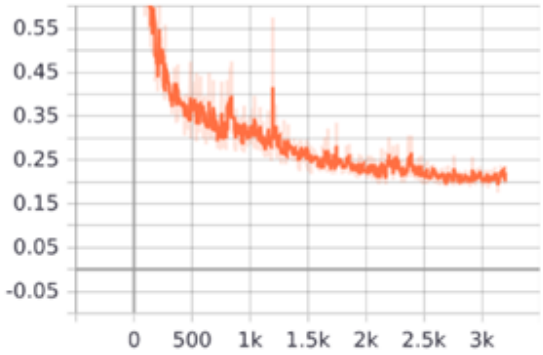
Normalised Loss vs Epochs	Total Loss vs Epochs
	

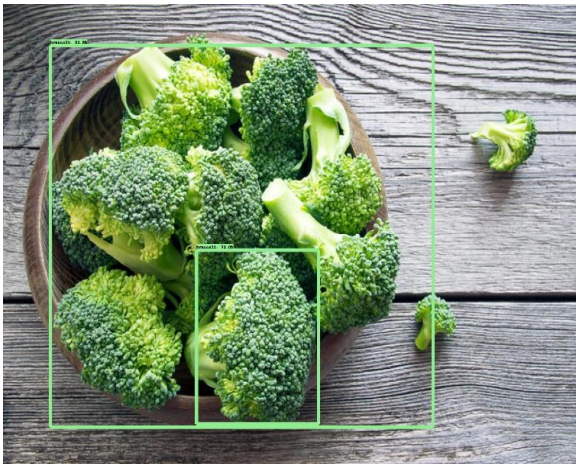
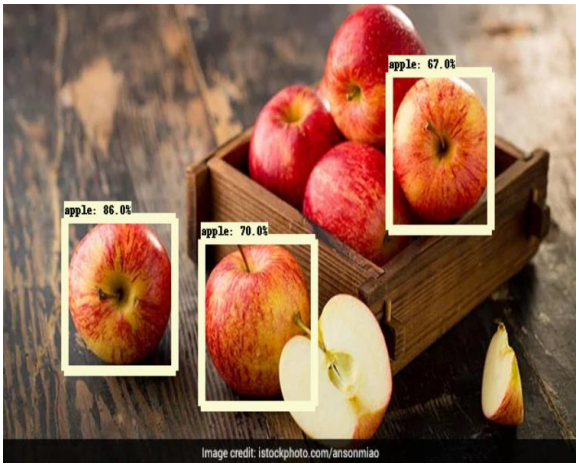
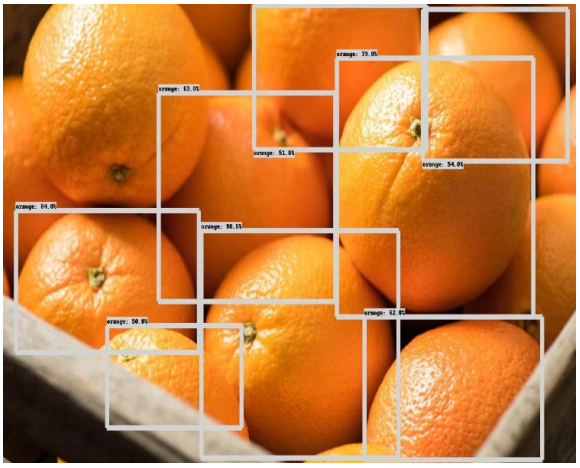
Figure 41: Comparison of Loss vs Epoch

Below table displays the power of Navcon's SSD Mobilenet V2 is recognising many common objects found in both indoor and outdoor environments.

	
Detection of Microwave & Pizza	Detection of Laptop & Dining table
	
Detection of Bed	Detection of Refrigerator

	
Detection of TV	Detection of Couch & Potted Plant

Not only does Navcon's object detection model detect many common indoor objects, it also detects shopping items including fruits & vegetables.

	
Detection of Broccoli	Detection of Bananas
	
Detection of Apple	Detection of Orange

To ensure the safety of Navcon's users, the scope of coverage for objects in the outdoor environment is especially vast covering all modes of transportation to even traffic lights for detecting junctions.

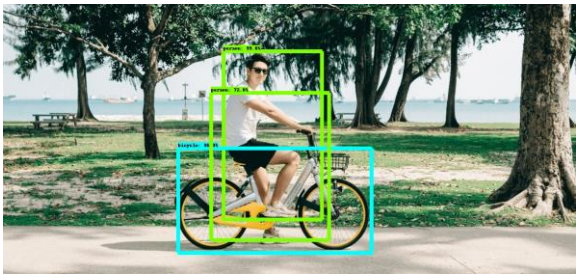
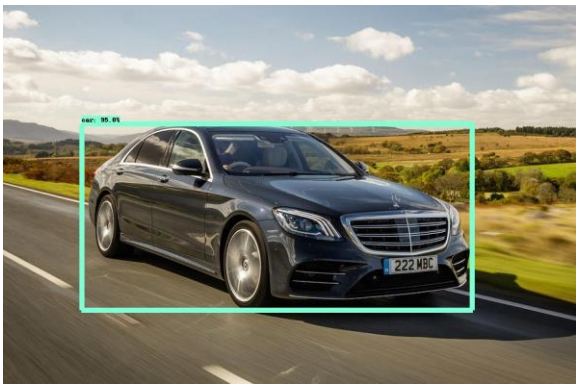
<table>
<tr><td>  </td><td>  </td></tr>
<tr><td>Detection of Bicycle & Person on it</td><td>Detection of Bus</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>Detection of Traffic Light</td><td>Detection of Car</td></tr>
<tr><td>  </td><td></td></tr>
<tr><td>Detection of Stop Sign</td><td></td></tr>
</table>

Figure 42: Object Detection

7.5 Outdoor Navigation

The outdoor navigation section provides example pseudo code used in the development of both deep Q network and inference rule engine as well as provide the training log for the deep Q network.

7.5.1 Deep Q Network

The following pseudo code was used for the development of the deep Q network with state replay memory and recursive bellman optimality equation implementation.

Algorithm 1: deep Q-learning with experience replay.
Initialize replay memory D to capacity N
Initialize action-value function Q with random weights θ
Initialize target action-value function $\hat{Q}$ with weights $\theta^{-} = \theta$
For episode = 1, M **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  For $t = 1$,T **do**
    With probability ε select a random action a_t
    otherwise select $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
    Set $y_j = \begin{cases} r_j & \text{if episode terminates at step j+1} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^{-}) & \text{otherwise} \end{cases}$
    Perform a gradient descent step on $\left(y_j - Q(\phi_j, a_j; \theta)\right)^2$ with respect to the network parameters θ
    Every C steps reset $\hat{Q} = Q$
  End For
End For

```
local/download_and_untar.sh: Successfully downloaded and un-tarred /Users/vidish/Desktop/asr-data/train-other-500.tar.gz
utils/validate_data_dir.sh: Successfully validated data-directory data/train_other_500
local/data_prep.sh: successfully prepared data in data/train_other_500
steps/make_mfcc.sh --cmd run.pl --nj 40 data/train_other_500 exp/make_mfcc/train_other_500 mfcc
utils/validate_data_dir.sh: Successfully validated data-directory data/train_other_500
steps/make_mfcc.sh: [info]: no segments file exists: assuming wav.scp indexed by utterance.
steps/make_mfcc.sh: Succeeded creating MFCC features for train_other_500
steps/compute_cmvn_stats.sh data/train_other_500 exp/make_mfcc/train_other_500 mfcc
Succeeded creating CMVN stats for train_other_500
utils/combine_data.sh data/train_960 data/train_clean_460 data/train_other_500
utils/combine_data.sh [info]: not combining utt2uniq as it does not exist
utils/combine_data.sh [info]: not combining segments as it does not exist
utils/combine_data.sh: combined utt2spk
utils/combine_data.sh [info]: not combining utt2lang as it does not exist
utils/combine_data.sh: combined utt2dur
utils/combine_data.sh: combined utt2num_frames
utils/combine_data.sh [info]: not combining reco2dur as it does not exist
utils/combine_data.sh: combined feats.scp
utils/combine_data.sh: combined text
utils/combine_data.sh: combined cmvn.scp
utils/combine_data.sh [info]: not combining vad.scp as it does not exist
utils/combine_data.sh [info]: not combining reco2file_and_channel as it does not exist
utils/combine_data.sh: combined wav.scp
utils/combine_data.sh: combined spk2gender
fix_data_dir.sh: kept all   253992 utterances.
fix_data_dir.sh: old files are kept in data/train_960/.backup
steps/align_fmllr.sh --nj 40 --cmd run.pl data/train_960 data/lang exp/tri5b exp/tri5b_ali_960
steps/align_fmllr.sh: feature type is lda
steps/align_fmllr.sh: compiling training graphs
steps/align_fmllr.sh: aligning data in data/train_960 using exp/tri5b/final.alimdl and speaker-independent features.
steps/align_fmllr.sh: computing fMLLR transforms
```

The training log is available in the full submission along with the best model file.

7.5.2 Inference Rule Engine

As mentioned earlier, Inference Rules are reasoning techniques applied on every frame allowing for a fail-safe mechanism to collision avoidance. Provided below are some of the pseudo code implemented in Navcon for identification as well as reaction mechanism on some of the objects.

Identification and reaction mechanism for car:

```
Initialise video camera
Initialise object detection model
For frame in live feed
        if class in objects detected is a car
                if confidence score of the object detected > 50%
                        Set mid x and mid y for object detected using bbox coordinates
                        if mid x > 0.7
                                voice out "Car to your right! Please stay back"
                        elif mid x < 0.3
                                voice out "Car to your left! Please stay back"
                        else
                                voice out "Car in front, please move to the left"
```

Identification and reaction mechanism for junction:

```
Initialise video camera
Initialise object detection model
For frame in live feed
        if class in objects detected is a Traffic light
                if confidence score of the object detected > 50%
                        if moving man green:
                                voice out "You may cross the junction!"
                        else
                                voice out "Please stop"
```

7.6 Indoor Navigation

The indoor navigation architecture is as shown below,

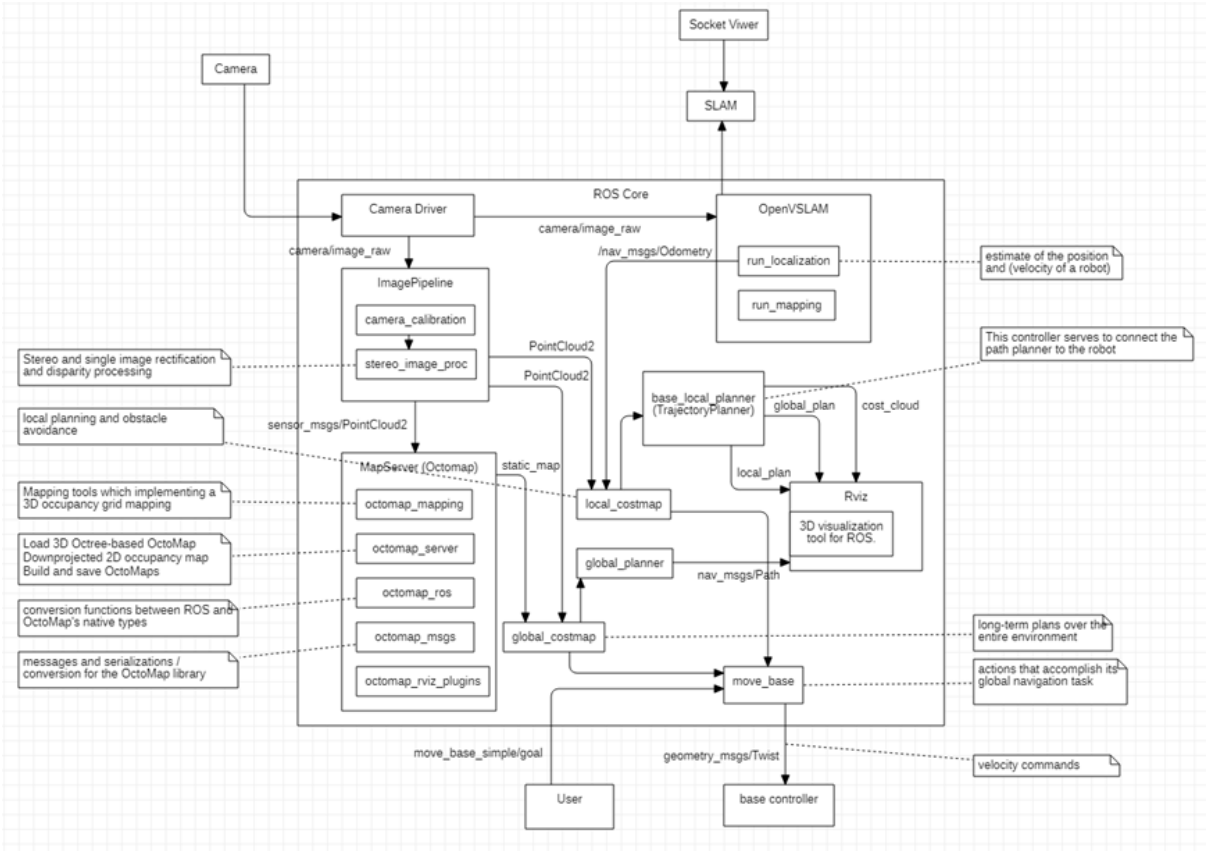


Figure 43: Indoor Navigation Architecture

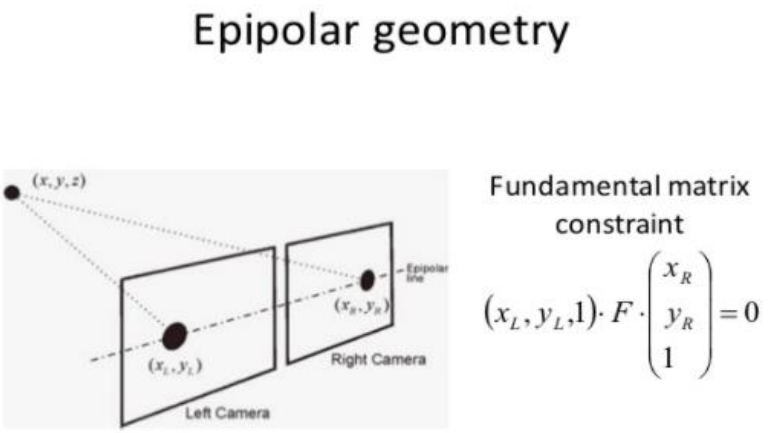


Figure 44: Math behind Indoor Navigation

Fundamental Matrix F contains the translation and rotation matrix, which is the location of the second camera relative to the first in global coordinates, it has the information about two cameras in pixel coordinates, together with these 2, we can map a point in one image to a line (epiline) in the other image.

References:

1. *The assessment of the quality of life in visually impaired people with a different level of physical activity. (n.d.). ResearchGate.* https://www.researchgate.net/publication/277944100_The_Assessment_of_The_Quality_of_Life_in_Visually_Impaired_People_With_a_Different_Level_of_Physical_Activity
2. *Bakharia, A. (2020, October 10). SVM parameter tuning in Scikit learn using GridSearchCV. Medium.* https://medium.com/@aneesha/svm-parameter-tuning-in-scikit-learn-using-gridsearchcv-2413c02125a0
3. *Bangash, I. (2020, July 5). NVIDIA Jetson Nano vs Google coral vs Intel NCS. A comparison. Medium.* https://towardsdatascience.com/nvidia-jetson-nano-vs-google-coral-vs-intel-ncs-a-comparison-9f950ee88f0d
4. *Installation — OpenVSLAM documentation. (n.d.). Contents — OpenVSLAM documentation.* https://openvslam.readthedocs.io/en/master/installation.html
5. *Models. (n.d.). Coral.* https://coral.ai/models/
6. *NVIDIA-AI-IOT/tf_trt_models. (n.d.). GitHub.* https://github.com/NVIDIA-AI-IOT/tf_trt_models
7. *OpenCV: Epipolar geometry. (n.d.). OpenCV documentation index.* https://docs.opencv.org/3.4/da/de9/tutorial_py_epipolar_geometry.html
8. *OpenCV: Epipolar geometry. (n.d.). OpenCV documentation index.* https://docs.opencv.org/3.4/da/de9/tutorial_py_epipolar_geometry.html
9. *Wiki. (n.d.). Documentation - ROS Wiki. Retrieved October 31, 2020, from* https://wiki.ros.org/ROS/Concepts
10. *Wiki. (n.d.). Documentation - ROS Wiki. Retrieved October 31, 2020, from* https://wiki.ros.org/costmap_2d
11. *Wiki. (n.d.). Documentation - ROS Wiki. Retrieved October 31, 2020, from* https://wiki.ros.org/navigation/Tutorials/RobotSetup/Odom
12. *Wiki. (n.d.). Documentation - ROS Wiki. Retrieved October 31, 2020, from* https://wiki.ros.org/navigation/Tutorials/RobotSetup
13. *Wiki. (n.d.). Documentation - ROS Wiki. Retrieved October 31, 2020, from* https://wiki.ros.org/navigation/Tutorials/Using%20rviz%20with%20the%20Navigation%20Stack
14. *Wiki. (n.d.). Documentation - ROS Wiki. Retrieved October 31, 2020, from* https://wiki.ros.org/octomap_server?distro=noetic
15. *Wiki. (n.d.). Documentation - ROS Wiki. Retrieved October 31, 2020, from* https://wiki.ros.org/stereo_image_proc
16. *Wurm, K. M., & Hornung, A. (n.d.). OctoMap.* https://octomap.github.io/