A Guiding Hand:

Helping the Visually Impaired Locate Close Objects on a Table

Author: Asim Aryal

ID: 2619889

Supervisor: Dr Kevin Swingler

Date: April 2020

UNIVERSITY OF STIRLING

Dissertation submitted in partial fulfilment for the degree of Bachelor of Science with Honours in Computing Science

Computing Science and Mathematics

University of Stirling

Abstract

Problem

With a growing population of individuals facing some form of visual impairment, this project tries to address the specific problems encountered in a restaurant setting, i.e. in front of a table full of objects. Without complete visual sense, it is a challenging task to find any objects on a table, especially on a restaurant table, where the expected norm is to have a conversation. This means asking for human assistance can be repetitive and might bring the experience down. Also added to the fact is that most of the objects mentioned above are usually made of glass or ceramic, which require more careful handling. This project tries to simplify and provide a solution to this challenging situation.

Objectives:

The primary goal of this project is to develop an assistive device that will allow the user to get to an object without knocking things over or having to ask for human assistance and to gain an understanding of the state-of-the-art in computer vision, specifically with object detection and distance management. As the device is meant for a noisy and conversational situation, audio feedback could create confusion and might be an annoyance. This means the device will need to communicate with the user using a different sensory organ. So, using haptics, the device will provide information about the location of the objects on the table. This addresses the issue of constant annoyance and also guides the user's hand to the object of their choosing.

Methodology:

Firstly, the part of the problem that needs solving was object detection. To do so, a camera module is used which detects objects in its frame and knows what they are. To enable that, the project uses pre-trained models that already have most of the required objects. It uses Google's Tensorflow library alongside Keras for its neural network API. Similarly, the haptic feedback will be given through a contraption around the wrist that has four vibrating motors attached to it. Following object detection, the distance between the hand and the objects are calculated and vibrating patterns give the user feedback as to where and what objects close to the hand are.

Achievements:

Object Detection: Works with accuracy, reasonable speed and a large selection of objects

Distance measurements: Works with a limited set of parameters

Vibration feedback: Successfully implemented, however, since it is closely tied with distance measurements, the limitations can translate.

Attestation

I understand the nature of plagiarism, and I am aware of the University's academic misconduct policy.

I certify that this dissertation reports original work by me during my University project except for the following:

The code discussed in Section 3.3 was created by Moses Olafenwa & John Olafenwa and was used as an open-sourced library.

The code discussed in Section 3.3.2 was partly written by my supervisor.

The project uses several python libraries which have all been cited.

Signature *Asim Aryal* **Date 28th April 2020**

Acknowledgements

I would like to first and foremost thank my supervisor, Dr Kevin Swingler, who allowed me to undertake this project, then guided me throughout it. Dr Swingler also helped enormously with giving me a perspective of what the project is and should be about and provided me with the hardware components used in this project. Secondly, I would like to thank my personal tutor Dr David Cairns who has given me invaluable advice throughout my time at the university. I would like to mention Dr Carron Shankland for running the project module smoothly and providing adequate support, along with the Computer Science and Mathematics faculty for always being available and assisting me during the undertaking of this project.

Finally, this could only be possible with massive support from my lovely partner Ashmita. Thank you for everything.

Table of Contents

List of Figures

1 Introduction

1.1 Background and Context

The most recent figures from WHO shows that the total number of individuals with some form of visual impairment is over 1.3 billion and 39 million have complete blindness. [1] These figures are also projected to treble in the next 30 years with the ageing population.[2] Similarly, the NHS report 2 million people with sight loss in the UK with 360,000 among them registered as partially sighted or blind[61]. This project focuses on addressing one of the potential problems that will befall the growing population of visually impaired. Movement, in general, is incredibly challenging without sight. As a fully sighted person, it is rather difficult to comprehend the concept when compared to individuals living without or partial visual sense. Existing assistive technologies have had a significant impact on a number of different aspects; however, some of the issues faced are yet to be addressed and is often overlooked. Assistive devices used by visually impaired individuals such as navigation technologies or even natural assistance like guide-dogs solely focus on transportation navigation; however, several other challenges in everyday life are yet to be solved. Among many such scenarios, a setting in front of a dining or a restaurant table is one of them.

Again, for a sighted individual, the thought does not cross our mind as it is a natural hand-eye coordination task. The visual senses capture the images, differentiates between objects to locate the one required, then the brain guides the hand to the object using the information of the layout of the table, usually without knocking things over or touching other objects. However, for the partially or fully sight-impaired people, the task is not only uncomfortable but is difficult as there is no information to process about the table, and thus guide the hand to the object. This requires human assistance in general in order to guide someone's hand to the object they need, for example, a glass of water, or one of the pieces of cutlery of their choosing.

Being unable to grab an object without asking for assistance or risking accidents is certainly not ideal and will occupy one's mind over enjoying the meal. It might also, at times, interrupt the flow of a conversation, or make one feel inadequately prepared to tackle such an environment which should not be the case in the technological world of today. As assistive technologies develop further, these are the finer issues that, when solved, will help the potential users enjoy things more independently, thus contributing to the general well-being as well. A nice meal without interruption or constantly having to ask for assistance at the table will enhance the experience.

The primary use-case of this project will be to alleviate this problem by allowing the user to guide their hand to an object of their choice. Having the freedom to move the hand with fewer chances of accidents and lesser need for assistance is a desirable solution to this overlooked problem. To do so, the device will require a way to replicate a sighted person. Here, to extract the information, i.e. the layout of the table, a camera will be used and instead of the user's brain processing the data, it will be a computer doing so. The principal of the project is to provide processed feedback to the user by using information that is not readily available to them in order to accomplish a task that requires their compromised senses.

1.2 Scope and Objectives

This project will incorporate an essential and widely growing field in computer science, i.e. Computer Vision(CV). This requires knowledge of topics relating to Artificial Intelligence(AI), such as machine learning, neural network algorithms, real-time video processing, object

localisation, object detection and distance measurement and management. All of the aforementioned subjects are of the growing popularity and are being used for several significant projects. Thus, Computer Vision, in itself, is a highly useful tool that can be incorporated in several fields of interests. The idea of distance management using computer vision(along with a number of other sensors) is one of the techniques utilised by the future of automobile industries in autonomous vehicles.[3] From self-driving vehicles to facial recognition, computer vision is used extensively to improve everyday life.

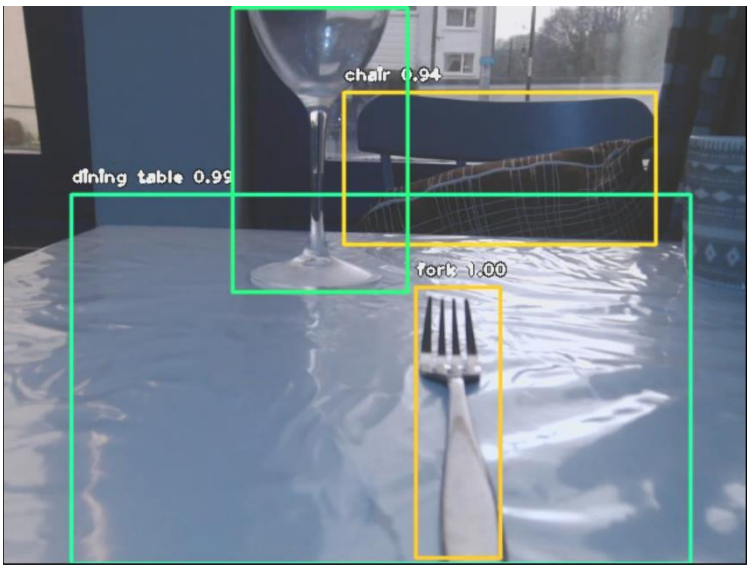


Figure 1. *Example of Object Detection with bounding boxes and confidence*

The scope of this particular project can be stretched, as the same idea can be implemented in several different scenarios. The concept of a restaurant/dining table demonstrates a limited area of operation where there are several different objects in front of the user. With a custom trained model for each scene, the same idea could be implemented with a dressing table or a nightstand as well. The application of this project also leaves room for further improvement with possibilities such as audio instructions for a quieter surrounding or using other sensors for precision. As the end goal, one could imagine having a module designed with custom trained models for each room, which would allow the user to navigate between objects much more comfortably than previous. Hence, the objective of this project is to demonstrate these possibilities using a dining table as a reference and learn about object detection and distance calculation using computer vision.

As a part of finding out what the objectives of the project should incorporate, a questionnaire has been prepared(see appendix 4) to be asked to visually impaired individuals. The questionnaires include previous experience and their views on the idea.

1.3 Achievements

1.3.1 Object Detection

Object detection works successfully, with pre-trained models with varying accuracy. Bounding boxes are created, and objects are named with a probability percentage that can be used. Furthermore, a count on every frame is also possible for each object. Limitations to accuracy and successful detection are discussed further in the following sections.

1.3.2 Vibration/Haptic Feedback

Haptic feedback using vibration modules are functional and work as intended. There are four different modules set for top, right, bottom and left that guide the motion of the hand. The entire module is in a wearable form factor.

1.3.3 Distance Measurement and Management

Successfully implemented distance measurement between the user's hand and the object using co-ordinate plot-points and geometry. There are limitations to the hardware, speed and computation, and thus extracting successful measurements are discussed further in following sections.

1.4 Overview of Dissertation

Following the current section, Section 2.0 discusses the state-of-the-art on comparable devices and projects. This section features a number of software and hardware projects, as this project revolves around both aspects. Other assistive devices are researched upon and analysed to extract features that can be implemented as well. Similarly, the sub-sections also look into the development environment, libraries and dependencies as well as the current state of developments in neural network algorithms. It also looks into other research projects undertaken that are relevant and provide ideas to the project. Section 3.0 then goes into the development cycle, starting with requirements engineering and finishing with testing. All the code discussions are done in this section under implementation with relevant analysis and figures to demonstrate the ideas behind the codes. Then the dissertation does conclusions in the section that follows with reviews and reflections on the project and the dissertation itself, evaluate the results achieved and look at potential improvements on the system. It will also discuss the limitations faced and how they can be addressed in future builds. A list of references is provided, followed by the Appendices.

2 State-of-The-Art

This section of this dissertation focuses on works that have already been done in this field that attempt to solve similar issues. There are a few devices that are commercially available, however, do not provide the same service as the project device does. Given the commercially available methods are not open-sourced, their technical analyses, thus, are based on existing technology and how they might have been implemented. This section looks at both the hardware as well as the software as it is essential to review and extract ideas for both. It also looks at other academic researches relevant to the project to understand past work done in the field.

2.1 Orcam MyEye

Orcam is one of the leading companies focusing on assistive technology for the visually impaired.[4] The company's latest iteration of the assistive device named MyEye 2.0 has the claim of being one of the most feature-rich devices in the market.[14] It provides useful features in a compact form factor that can be attached to any spectacle frames. The device itself has a camera at the top that takes the images in, processes it and provides output to the user with audio instructions. It also supports voice recognition(limited to the English language). It is battery-assisted and uses Computer Vision technologies to do an array of tasks such as text-to-speech, facial recognition and storage, object recognition and colour recognition. The form factor and the ability to distinguish between objects is a desirable feature that could be incorporated into the project as having a camera mounted on the side of the frame is ideal.

Pros	**Cons/Limitations**
Multiple features, all-in-one device	Only audio-based output
Compact form factor, versatile	Limited to the English language
Mobile phone companion available	High prices (£3000 + on amazon.co.uk)
Available in over 25 countries	
Free online training to users	

MyEye 2.0 as a hardware accessory offers a number of services; however, the problem this project tries to solve is still one that is not available on it. Without a haptic feedback plugin, it would be intrusive for the practical use cases this project is trying to tackle.

2.2 WayBand Haptic Wristband

WearWorks is a company that has developed a wristband that incorporates haptic feedback to allow notification for runners.[16] It uses GPS technology along with google maps and OpenStreetMap API, to accomplish the task[16]. Using this device, a blind person in 2017 ran a significant portion of a marathon in New York, indicating the use-case is practically viable.[15] The main takeaway from this device is the form-factor. A lot of documentation is not available for this device at the moment, however, having a compact wristband with vibration motors for feedback is the ideal most way for this project which this device can provide inspiration from.

2.3 Depth and Distance Measurement using Cameras and Sensors

There are a number of techniques of depth estimation using a single camera as seen in [22-25, 40-42] which will be reviewed in the following sections; however, all discussed methods require several other prior measurements. One of the techniques that could be employed involves moving the camera[23] to simulate the two cameras at a centre of rotation or moving specific object itself at a known speed/dimensions. Getting the speed would be rather tricky and having the dimensions of each object is also not entirely practical. So, using a single camera with this method has a lot more limitations and requires prior knowledge of several parameters.

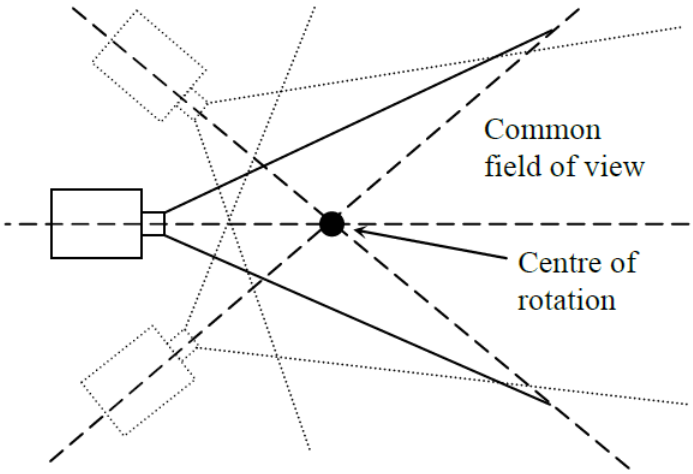


Figure 2. *Panoramic system camera*

Likewise, using the focal length of a camera, the angle of view and changing the position of the object to simulate two images, a technique known as Depth From Focus/Defocus(DFF/DFD)[41] could also be used to get a similar estimation as previous. However, neither of the discussed techniques are accurate nor efficient enough to be implemented on a live video feed. The way these problems have typically been solved is by using stereo cameras.

Advancing development in 3D technologies has contributed to the broader usage of stereoscopic cameras. These cameras have two different sensors mimicking human binocular vision that allows to capture and triangulate objects in a 3-dimensional format [27]. One of the techniques can be seen in figure 2, where two cameras are placed at a fixed, known distance (BC) pointing towards the object at point A. The angle at B will be equals $\tan\theta = \frac{AD}{BD}$ and the angle at C will be $\tan\beta = \frac{AD}{DC}$, so that would mean the total distance, which is known, will be $BC = \frac{AD}{tan\theta} + \frac{AD}{tan\beta}$. Using a similar technique on the size of a frame from both cameras, and their field of views, the angle at which the object is at from the camera can be triangulated. Once $\tan\theta$ and $\tan\beta$ have been calculated, the perpendicular distance between the cameras and the object can be derived. Then using Pythagoras, the hypotenuse, i.e. AB and AC can be calculated, which is the distance between the cameras and the object. All of this data can then be calibrated and used to understand how far each point in the scene is from both cameras, thus enabling more precise depth estimation.

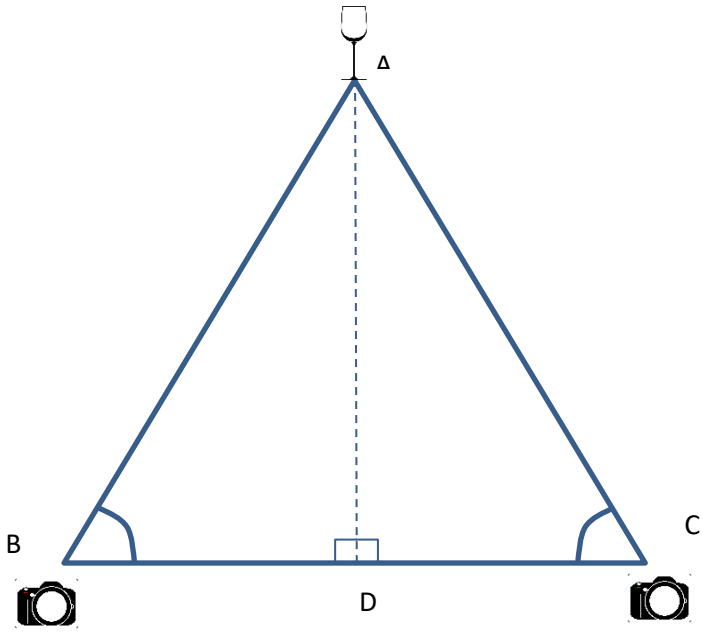


Figure 3. *Geometric representation of stereo cameras*

This technique of depth estimation, also known as homography[76], has been around for over a century with a number of other improvements made to it[36]. With the rise of neural networks, similar techniques have been implemented using image recognition as well. The idea behind this lies in measuring the pixel density and predicting adjacent pixels. The apparent pixel differences between the two images from each camera are used to create an output which is called a disparity map[37](see figure 4). The things that are closer have larger disparity than ones that are further away from the camera because depth and disparity are inversely proportional to each other[38]. Thus, the higher the disparity, the closer the object and vice versa. [38]

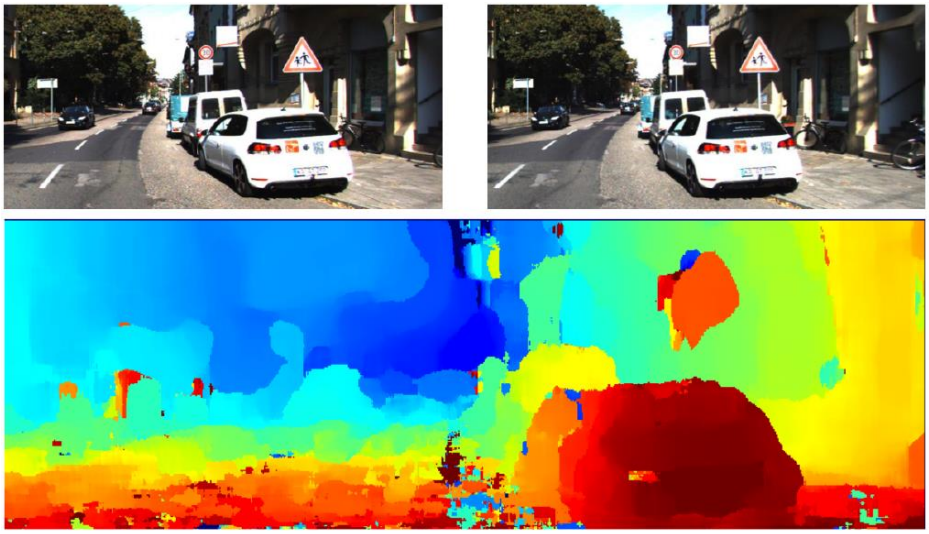


Figure 4. *Disparity Map [38]*

Other technologies such as Microsoft's Kinect V1 and Apple's 'face unlock' use structured light cameras for depth calculation using dot projectors. These technologies, with correct form factor, can provide additional information that can be used for distance calculation. These technologies mostly rely on bouncing numerous lights off the surface and timing the reflections to measure depth. With the known speed of the dots projected, distance measurement is made much more accessible [59].

2.4 Autonomous Vehicles

Autonomous Vehicles or Self-Driving cars have been a serious topic of interest in the automobile industry for over a decade and a half. Since the significant developments in machine learning and deep neural networks, autonomous vehicles have become more advanced and reliable.[39] Thus, the state-of-the-art in Computer Vision, deep learning algorithms and distance management techniques at its most sensitive, and with some of the most advanced technologies, can be found in autonomous vehicles. Given the stake at which self-driving cars function, the distance management has to be impeccable. To achieve that, different manufacturers use a varied array of sensors. In most cases, it is a combination of lidar, radar and multiple cameras. Lidars have been a topic of debate in the recent years with the CEO of Tesla Motors Elon Musk claiming they are moot when used alongside cameras as object detection and triangulation in past years has leapfrogged in accuracy[51]. However, there have been other claims that using multiple layers of measurement technologies, lidar being one of them, can be more accurate than relying on what the vision algorithms 'think' they see[52]. While the debate might be more to do with the cost of lidar technology, the impressive takeaway from this can be derived from Tesla's high reliance of cameras and deep learning algorithms for detection which shows that the technology itself can only be better moving forward. Other innovators in the field of autonomous vehicles are Alphabet's Waymo, one of the earliest successful projects in self-driving cars[53] and General Motors Cruise. Other big-name automobile companies have also invested in research of autonomous vehicles in recent years, which suggests that self-driving cars have a major market in the coming days[54]. While this project might not require as much precision as such vehicles, it certainly provides insight on what current work on this topic might constitute of.

2.5 Language, Distribution and Environment

Object detection can be done in many different languages; however, the easiest to organise and with the largest library support is with Python 3.x.[7]. Python offers an extensive range of libraries as well as documentation that are explicitly designed for object detection. As the project requires machine learning libraries in order to support the detection, anaconda distribution provides one of the more popular python distributions for data scientists, thus is used with computer vision works as well.[6] JetBrains PyCharm with Anaconda Plugin provides a feature-full integrated development environment(IDE) with anaconda distribution pre-installed. JetBrains also provide the professional package for students free of charge, which adds further plugin support, including a scientific mode to analyse results[8]. Other notable IDEs include ones such as Spyder and IDLE which only provide limited features and integrations with plugins compared to PyCharm.

2.6 Libraries and Dependencies

ImageAI is a library that is easy to use and allows developers to integrate state-of-the-art Artificial Intelligence(AI) features to projects. It is an open-source library with over 75000+ developers using it[9]. The library incorporates OpenCV library, along with other python libraries required such as *matplotlib*, *numpy*, *scipy* and *colorsys*.[10] ImageAI provides image detection and classification from images as well as object detection from video(video file and live feed) with a pre-trained model or custom trained ones. Putting all of these dependencies and libraries together, ImageAI can use three different pre-trained models: RetinaNet, YOLOV3 and TinyYOLOV3[11]. It also makes use of Google's TensorFlow platform for machine learning application used for neural network library of Keras. TensorFlow is a machine learning library/framework that makes training machine learning models easier. It is an open-source platform and has core components written and designed in C++ and CUDA.[45] While providing APIs for C++, TensorFlow also has APIs for Java, Haskell, Go, Rust as well as Python making it the most widely opted framework for machine learning.[45]

Similarly, a viable alternative to TensorFlow is Facebook's deep learning library PyTorch. PyTorch was released in 2016 and has seen several incremental updates since, making it a great alternative to TensorFlow.[43] PyTorch was developed with a python-centric design, allowing its implementation to be entirely pythonic. The primary difference between TensorFlow and PyTorch at its release was PyTorch offers dynamic graphing for machine learning models, while Tensorflow requires defining the entire computation graph before running the model. This means that PyTorch allows manipulating the graph on-the-go.[4] Tensorflow 2.0 at its release has also tried to incorporate dynamic graphing as well as extend its support to python[60]. As PyTorch is still relatively young, the support community behind TensorFlow is much larger. While PyTorch is considered to have a smoother learning curve, the relative lack of resources still makes TensorFlow the most used library.[44]

2.7 Object Detection Algorithms

Convolutional Neural Networks(CNNs) are the basis for all computer-vision based object detection algorithms[30]. These algorithms use patterns such as vertical or horizontal edges, shapes, and colour density to classify or detect objects from an image. An image, or anything in general, is simply a series of numbers(each pixel represents numerical values) that can be used to form matrices for further processing. CNNs use these matrices to be operated on with convolution filter matrices. The output matrix can then be used to recognise patterns in the pixels based on spatial distances[31]. C. Aggarwal, in his book[31] states the classification error-rate in long-standing benchmarks like ImageNet[32] have seen a reduction from 25% to 4%. Aggarwal also attributes these rapid developments in CNNs to the annual ImageNet image-classification contest: ImageNet Large Scale Visual Recognition Challenge. This contest has seen CNNs as routine winners since 2012 which has also seen raised efficiencies in CNNs.

A product of these recent developments is YOLO(You Only Look Once)[13]. YOLOv3 is widely considered one of the best model families for real-time object detection[12]. YOLO manages to consistently provide highly accurate detection for real-time detection. 'You Only Look Once' refers to the algorithm's single forward propagation pass through the neural network for detection technique.[33] YOLO uses non-max suppression to ensure each object is detected once[see figure 5]. Andrew Ng, in his Lecture, defines non-max suppression as outputting the maximal probability classifications while supressing nearby non-maximal ones[34]. Figure 2 demonstrates the technique where each object has many bounding boxes to it in the second stage, which then the algorithm only uses the ones with maximum confidence in the final detection. Similarly, YOLO also uses anchor boxes to ensure detection of multiple objects in

an image. Going back to figure 5 where this can be seen, the algorithm draws multiple anchor boxes; then with three sets of vectors, each one is associated with information for the bounding boxes, the (x, y) location and its classification. Then, the algorithm checks which bounding box has the highest overlap to non-overlap (Intersection over Union) ratio, compares to each anchor box and then decides what objects are present using that ratio [35]. YOLOv3 then applies a K-means algorithm to choose the anchor boxes more efficiently[34]. With these features, YOLO produces results that show its superiority over other existing models such as RetinaNet and the SSD family[13] as seen in figure 6. The YOLOv3 model ImageAI natively provides is pre-trained to be able to detect 80 different everyday objects including fork, spoon, knife, wine glass, bottle, cell phone, person and so on. With a higher mAP (mean Average Precision)(see Figure 3), the accuracy to speed ratio of YOLOv3 is the best model that can be used with ImageAI[13].

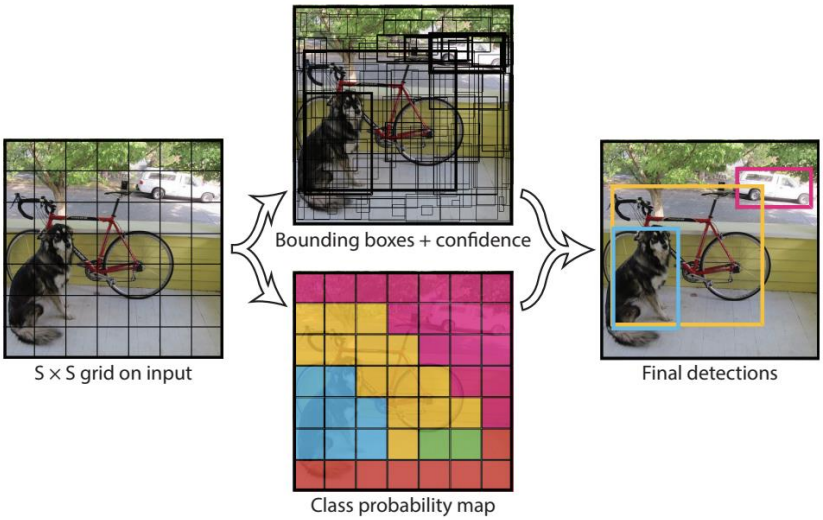


Figure 5. *Representation of YOLO detection with bounding boxes [12]*

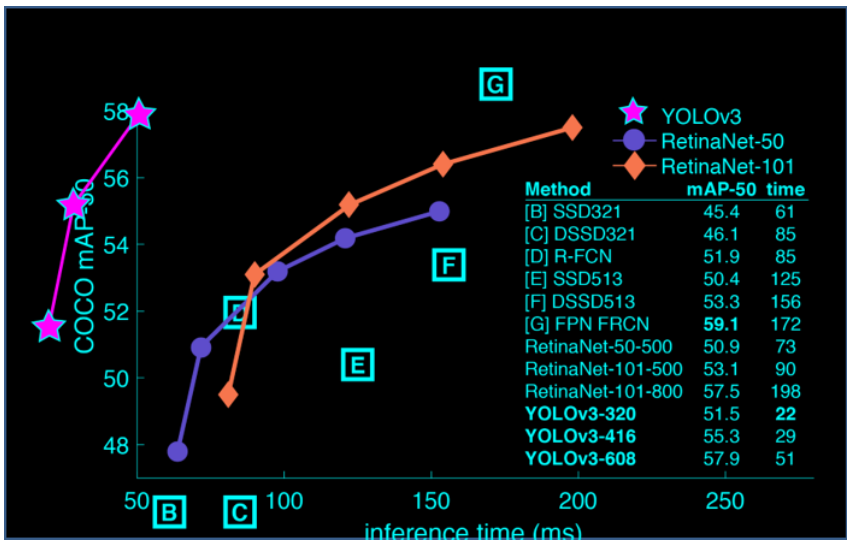


Figure 6. *Comparison between different models [13]*

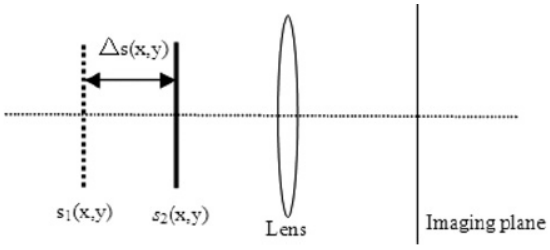
2.8 Academic Research Reviews

With its growing use-cases, Computer Vision projects have been used to develop several different assistive devices.[17-20] The project looked at [17] uses stereo cameras to enhance images in front of the user, which, as discussed in 2.3, is a technique that could be implemented to measure distances more accurately. Similarly, the critical paper in [19] looks at how each problem for assistive technology can and should be split according to their use-case and how a single device for all approach can rather limit a cross-application of Computer Vision-based assistive technologies. The idea behind the analysis in [19] revolves around device form factors and other limitations that might hinder the development of certain features, thus having to split the ideas at the end anyways.

Similar technologies such as a 'helping hand' glove paired with RFID(Radio Frequency Identification) have also been implemented with relative success[20]. The paper looks at this implementation with everyday household objects and provides an additional idea that could be implemented if Computer Vision detection by itself was not adequate for object detection. However, the usage of RFID is beyond the scope of this project.

Another aspect of this project that is extensively researched is distance measurement, especially, depth estimation. With accurate z-axis, a number of challenges posed by this project can be met. The standard method of extracting depth information is done using at least two cameras, as discussed previously. However, there have been several kinds of research done regarding distance measurement using a single camera with no additional sensors[22 – 25, 40-43]. The paper in [23] by J. Seal et al. looks into a catadioptric method to simulate two cameras by using mirrors to project virtual cameras (See Appendix 1) This research uses the focal length, size of the sensors and the mirror angles to estimate a depth resolution of 5.8cm at a working distance of 2m. This system comes with some trade-offs, especially the size. Originally proposed as a system for robotics, this technique, occupies more space than a stereoscopic camera would. At the same time, the field of view for this system will not be wide as the mirrors can never be parallel, which further introduces perspective distortion as noted by the authors.

The research article in [24] looks at optimising DFD(Distance from Defocus) techniques to provide ways to derive information for static objects with a fixed set of parameters. Most DFD algorithms use two images with different camera parameters, then measure the degree of blurriness at each point by applying a point spread function in order to estimate the depth of the image. A depth map is then produced to represent the depth at each point of the image. Y. Wei *et al.* in this paper, introduce two depth maps with a known depth change to improve upon the technique and get results with an error of 2.3% for images of 120×50 pixels. The research fails to capture if the results translate to a larger resolution of images and this method of depth mapping comes at a high computation cost. Given the test results are only



Standard theory of the new method

$s_1(x, y)$: depth map of the defocused image attained before variation
$s_2(x, y)$: depth map of the defocused image attained after variation is attained
Δs: depth change

Figure 7. *Proposed idea using defocusing [24]*

available for entirely small resolutions, the application of their proposed algorithm cannot be considered viable at this point.

Similarly, Z. Sadreddini *et al.* in a 2016 journal article devise a distance measurement method using a single camera for indoor environments [46]. This research involves making use of lines on the floor found in indoor rooms as a reference. The method used in this research revolves around converting an image into greyscale, then subtract each pixel value of the greyscale image from the coloured image. Then using Otsu's approach, the resulting image is converted to binary. Otsu's approach is a method of image segmentation and binarization that minimises the intra-class variance of black and white pixel thus maximising inter-class variance, meaning the threshold between black and white pixels are maximised [47,48]. Following this, Sardreddini *et al.* use Hough transformation to extract a line that passes through the object of interest from the greyscale image. Hough transformation[49,50] is insensitive to occlusion, as in it uses the slope-intercept formula(y=mx+c) to extract the said line. Using this line, another line is then drawn to intercept it where the object of interest is at, then the distance is calculated in pixels. This is particularly useful only where there are lines on the surface in the frame that passes through the object. Otsu's approach, as well as Hough's transformation, are well-known image processing techniques that have been used for a long time and the scope of this referenced research is limited. However, the idea behind using Hough transformation could be useful for distance measurement during the project as it is not computationally expensive and yields precise results.

In the same field, with the advancement in neural networks, L. He *et al.* published a research article where they propose a novel deep neural network in which the focal length information is embedded [55]. The study solely focuses on monocular vision, as to replicate the intrinsic human ability to perceive depth from a single image. The initial idea presented looks at developing a depth perception neural network model that is on par with state-of-the-art methods, then further the accuracy by embedding focal length information. The first fold of the research builds on pre-trained VGG models[56], then goes to global transformation layers for object localisation[57] and upscaling architectures for improved resolution/detail. Then the mid-level information is integrated in order to derive the structural information by converting the information available to the space of the depth(see Appendix 2). Finally, the focal length information is then embedded into the newly acquired depth information, which further improves upon the previous measurements. The results of this research show that it is an effective way of obtaining depth information, as the algorithm yields the lowest error-rate among the state-of-the-art reviewed in the article.

Likewise, Godard *et al.* from University College London in 2017 published a patented (yet open sourced) monocular depth estimation convolutional neural network algorithm[70]. This algorithm, unlike others, is an unsupervised learning algorithm that does not require ground truth depth data using other sensors such as lidar. As seen in previous reviews above, the usage of some form of ground truth data has been the most popular ways to implement CNNs for depth estimation. This study, however, uses image reconstruction techniques to solve the depth estimation problem. The study first takes two pictures from a calibrated stereo camera. It then tries to reconstruct the image from one of the cameras by using the image from the other. The idea behind this technique is, if the reconstruction is successful, then a function will have been modelled that has some form of 3D information about the scene. The final results conclude that the study successfully gets depth estimations, even for thin objects in the scene, suggesting the disparity map generation is extremely precise(see appendix 3).

Haptic feedbacks using vibrations have been used in many studies to develop assistive, wearable devices[61-65]. The paper in [61] provides a comparative study of a number of these devices. The devices have been implemented to provide navigation through a number of wearable forms. G. Flores *et al.* in 'Vibrotactile Guidance for wayfinding for Blind Walkers'

have used a waist belt with vibration motors to guide the visually impaired [62]. It has the same principle as this project in that it tries to eliminate intrusive instructions for the user by only providing haptic guidance for navigation. The study also looks at another older study from the late ’90s that used a vest like contraption with vibration motors placed at the back of the user. This particular study featured a matrix of vibration motors {4×4} that denoted the four cardinal directions and stop.[63] The study looked at [63] inspired several other studies [64] that adopted the same approach. Similarly, the same idea has also been used with shoes, with vibration modules and micro-controllers, an example of which can be as seen in [65]. With all of these researches having been successfully done, it suggests vibration patterns will work for the device being developed for this project.

2.9 Analysis and Portability of Reviewed Research

After reviewing a number of semantic estimation techniques, single-camera depth perception is either computationally heavy or requires a lot of known measurements/conditions. Applying any of the discussed methods to this project, however, comes with the biggest challenge of it having to be done on a real-time video feed. Depending on the frames per second, each frame will have to be first detected by the object detection module, then analysed by the algorithm to extract the depth of field information. This also comes with challenges that include having to find the real-time distance between the camera and the objects themselves as the disparity map techniques only provide disparity between the objects/pixels which would have to be converted into actual physical depth between the objects[68]. Knowing the physical distance between the camera and the object, and thus having z-axis to work with would require physical dimensions of an object. Upon further research, the pin-hole camera formula could be used to get the distance from the camera. The pinhole formula requires knowledge of the physical size of the objects, focal length, and sensor height along with the image height and the object’s pixel height. The equation itself can be written as: $Distance\ to\ object(mm) = \frac{f(mm) \times real\ height(mm) \times image\ height(pixels)}{object\ height(pixels) \times sensor\ height(mm)}$ [67]. This would give the distance of the camera to the object for an image. However, applying this would be difficult given it already requires the physical height of an object at all times, and will again have to be done for each frame of the video. This would also then have to be implemented with the depth estimation techniques to understand the distance between the different objects for it to make a difference. The two studies reviewed above in 2.8 in He *et al.* [55] and Godard *et al.* [70] are resource-heavy, thus applying them on live feed would require a significant computation power as well. Given the time limitations and lack of access to such resources, their implementation would not make a practical sense in this project.

The Haptic vibration system is one of the more used techniques to provide feedback in assistive technologies reviewed. Most researches conclude that the idea was implemented with general success, and it eliminated the intrusiveness of audio feedback. This suggests the choice of a vibration module is a great option.

3 Development Cycle

The development cycle of this project followed a hybrid waterfall model. While it was not explicitly decided that it was to be the case, the project followed that pathway. Initially, as the project began, a lot of analysis was done, followed by a design that was proposed during the preliminary reports. The implementation followed the plans as far as possible, and several tests were carried out. During the entirety of this project, the cycle had to be revisited over and over again as the hardware and time limitations came into play. Since the project maintained a natural period of analysis to implementation to accommodate the variations, it also pushed it towards resembling an agile model. The following subsections will first delve into each of the steps, then review them at the end.

3.1 Requirements Engineering and Analysis

The initial analysis of the project required an understanding of what the initial idea behind it is. After speaking to the supervisor, the idea of demonstrating usage of computer vision to develop assistive technology for such a setting was the goal. As the project involved computer vision work along with a lot of co-ordinate geometry, the mathematical aspects of it could not be overlooked. There were three significant aspects of analysing this project: Object Detection, Distance Management and Haptic Feedbacks. The following subsections will look at the three parts of the requirements engineering.

3.1.1 Object Detection

The first and foremost requirement of this project is locating where the objects laid out on the table are, and what they are. Without this portion working, the rest of the project cannot move forward. This requirement comes with a few boxes to tick. This bit, for the project to have a meaningful significance, has to be done in real-time while also leaving room for further calculations. Pointwise, the following are considered as the initial requirements for this sub-section:

- Live camera feed
- Accuracy of detection, including a human hand
- Speed of detection to accommodate calculations
- Variety of items that can be detected
- Practical viewing angle

Hardware wise, it will require a camera module with a good viewing angle to enable the video capture. Furthermore, the frame count as well as smoothness of the video should remain of concern. Real time object detection is a highly resource hungry task, which will require a system capable of handling the workload. For the most accurate and fastest operation that will be on par with the results in figure 3, it requires a certain specification.[20] The recommended configuration by makers of YOLOv3 which will be used are as follows.[20]

- GPU- 4 × EVGA GTX Titan X
- Motherboard: GIGABYTE GA-X99-UD3P
- CPU: Intel Core i7-5820K
- Power Supply: 1600W
- Memory: 32GB

[20]

3.1.2 Distance Measurement

The second part of this project, and perhaps the trickiest segment as well, is measuring the distance between the hand and the objects. Distance measurement will come with many challenges which varies according to the efficiency and accuracy of detection. The primary goal is to get the pixel distance between the hand and the object. The measurement needs to be as precise as possible and will require calculation done for every frame. It will need to keep track of how far the hand is, and what side of the hand the objects are towards. Depending on the location of the object, the reference point for hand needs to be adjusted. The requirements of distance measurements are as follows:

- Find the location of the hand
- Find the location of the object relative to hand
- Set a reference point on the hand
- Set a reference point on the object
- Keep track of the distance between hand and object

3.1.3 Haptic Feedback

Finally, with both object detection and distance measurement done, haptic feedback can be viewed as the output of the device. The system needs to guide the hand in a two-dimensional plane, meaning the table will be a grid where the hand can either move horizontally(left and right) or vertically(forwards and backwards). In order to get this requirement to work, four vibration motors that indicate the four movements will be required. In order to get this feedback to the user, this module of four vibration motors need to be housed in a wearable form factor, have a micro-controller to enable controlling the motors, and it also needs to be wireless or untethered.

Furthermore, the vibrations need to convey the distance measurements to the user, as in how far the object is from the hand and what kind of object it is. This needs to be done as the hand is moving with each changing frame from the live video feed. To sum up, the feedback requirements can be classified as follows:

- Programmable
- Wearable
- Directional
- Object type distinction
- Distance to object
- Wireless

3.1.4 Analysis

With the information acquired from the requirements engineering, a few points need to be analysed in order to understand the extent of the project. On the hardware front, especially the system requirements to achieve real-time detection through the YOLO algorithm, a few concessions will have to be made. As the system recommendation is for the top of the line, fully accurate and fastest detection, it is also rather expensive at around 800 USD(~650 GBP) on amazon.com[21] for each Graphics Processing Unit. The recommended system would also require a custom-built system with compatible casing to house the four GPUs taking the total cost of the machine to over 10,000 USD, including all other components. The project, for the purposes of demonstration of these ideas, can be programmed with a lesser machine as it does not readily require that level of speed and accuracy.

The camera module, requiring a suitable viewing angle, can be an external webcam that can be shifted around. An overhead view of the camera will provide a 2-dimensional grid, which can be used to calculate distance. The potential problems with object detection such as pose variation or in this case orientation of an object could need addressing if the viewing angle changes. Similarly, issues such as illumination variability and occlusion will also come into question while attempting object detection.[26]

To meet the requirements of the haptic feedback, making it directional, wearable, wireless and programmable have already been done by Dr Swingler while assembling a glove with vibration motors around the wrist. The distance to the object and type of object are two aspects that need to be addressed.

3.2 Design and Setup

Before delving into the specific codes, the motives behind their implementation needs to be discussed. To do so, this section begins with the design, then looks into libraries and dependencies used while justifying each choice in the project setup.

3.2.1 Design

This subsection will look at the design overview for project implementation. By considering the requirements, the design is formulated to tackle each one as close to complete as possible.

Firstly, for object detection, the camera module is chosen to be a high-definition webcam. The webcam will be placed with an overhead view of the table and the objects resting on it facing the camera. The camera will be the first point of input which will then trigger the rest of the functions.

Then, to address the haptic feedback, as mentioned earlier, the glove with the vibration motors will be used. This wearable device features a board for programming, a battery, and the motors. The details for each will be discussed in the project setup.

Figure 8. *Overview of the Input-Processing-Output Flow*

3.2.2 Project Setup

This sub-section will look into how the project is setup. Starting with hardware, several concessions have been made from the state-of-the-art specifications discussed as they are not viable for the project's timeline. Having the camera at the first-person point of view would require depth calculation from the single camera. None of the reviewed literature in 2.8 provided a depth estimation technique that could be applied to a live video feed without the cost of additional computation power. However, the setup opted for manages to run the demonstration to a reasonable result. Following the design plan, initially, the webcam is setup. The webcam of choice is Logitech C920 HD PRO, with a field of view of 78° and focal length 3.67 mm [28]. The webcam is adhered to the ceiling at the height of 152cm from the table from where the table, with its contents, is in full view. The objects on the table are fork, knife, cup, bottle, cell phone and wine glass and are laid facing towards the camera. This setup is used for the majority of the project for implementation as well as testing purposes. Figure 5 shows a representation of this setup.

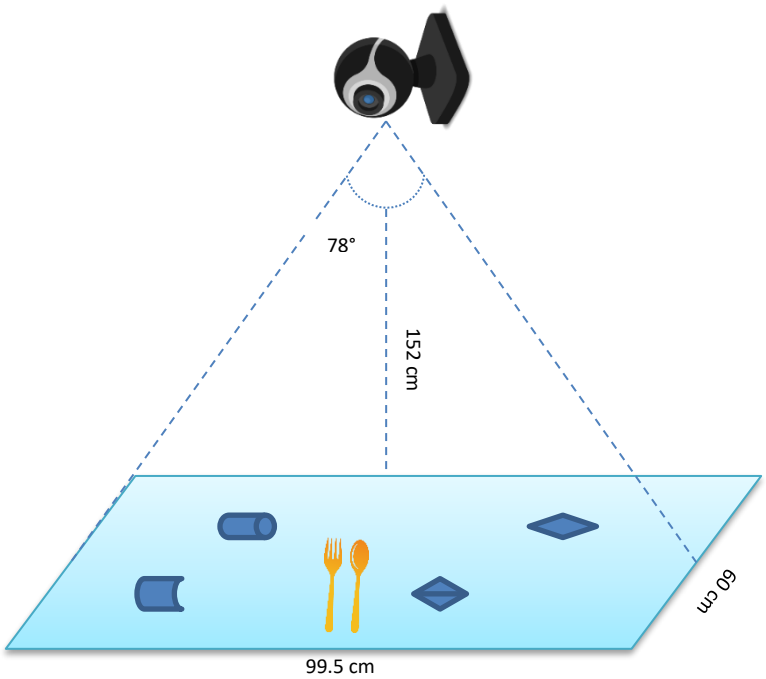


Figure 9. *Representation of the physical setup*

Similarly, for the haptic feedback to work, a bright red coloured glove has been used. The use of a bright red colour is to facilitate detection[29], if a different approach was to be used. Using the colour value of pixels, the red coloured object could be determined as the hand. At the basis of it all, everything the computer reads are numbers, hence, to keep the possibilities and scope of the project-wide, the colour selection for the glove is preferred brighter. The glove being used is for the right-hand with texture for grip on the palm side(see figure 6). An 'Adafruit ESP32 feather' board has been sewn on top of the glove with a lithium-ion battery attached for power supply attached with a Velcro. The vibration modules are connected to the board via cable and are also sewn on the inside of the wrist of the glove. A power switch has also been attached on top of the battery with indicators 1 (on) and 0 (off). The ESP 32 has a built-in Bluetooth module along with a number of other features[30] that can extend the potential of the glove itself.

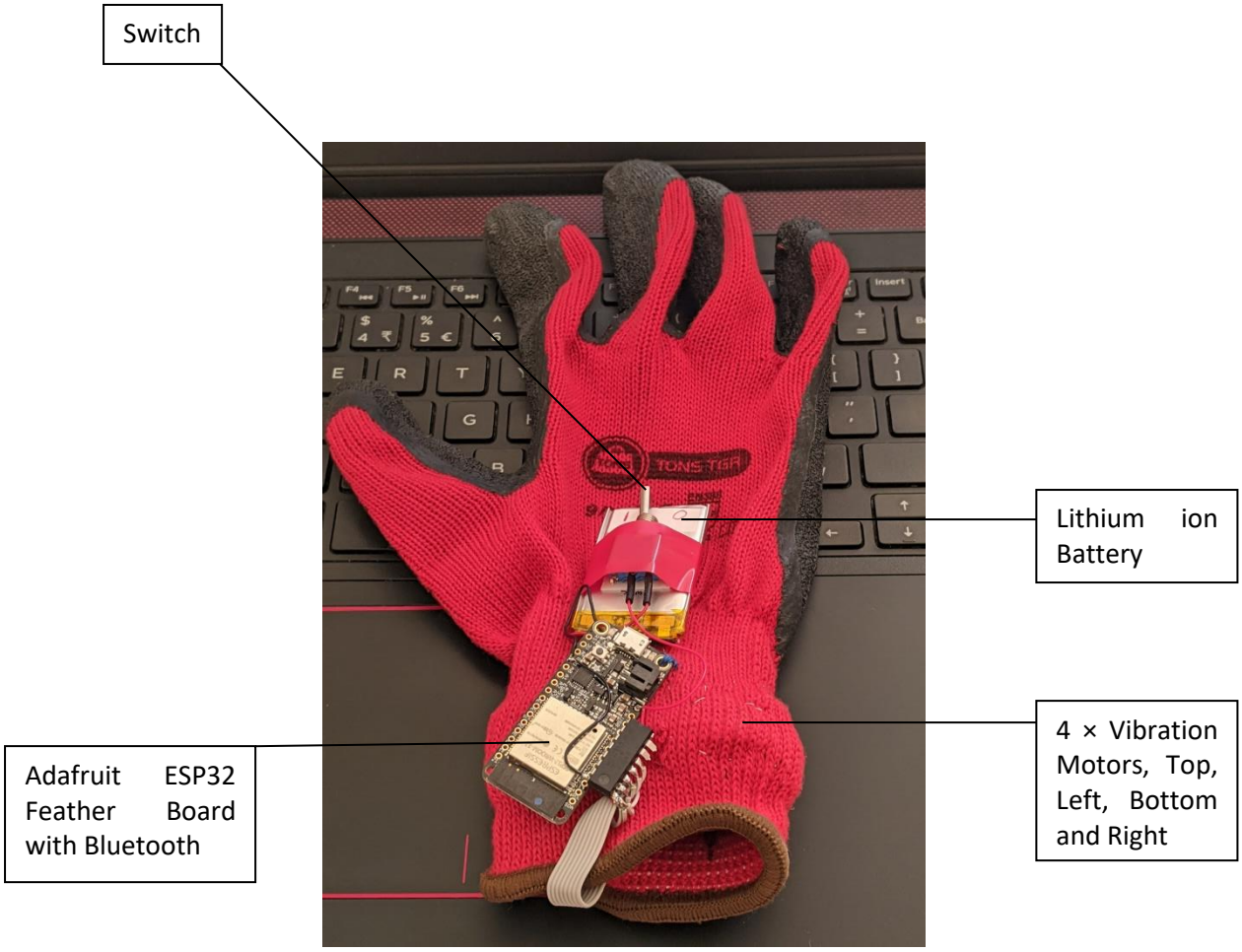


Figure 10. *Glove with micro-controller, battery, switch and vibration motors*

The software development and a big part of testing have been entirely done in a personal machine with a reasonably capable specification. It is not close to the recommended system requirements; however, for the demonstration of the idea, the machine serves the purpose. The specification of the development machine is as follows:

CPU - Intel Core i5-6300HQ Quad Core 6MB; 2.3GHz	GPU: Nvidia GeForce GTX 960M 4GB GDDR5
RAM – 16GB DDR3L 1600Mhz	Bluetooth 4.0
Dell Inspiron 15 7559	

This configuration is a far cry from the required recommendation for optimal use; however, they do run the program with reasonable accuracy and speeds. With lack to access to a more powerful machine at the time of testing, it was challenging to perform as many tests without compromising the speed. The Laptop's internal throttling also slows the speeds down significantly if/when the internal temperatures rise while running the program. (See Appendix 1 for full specification of the machine)

Following the hardware assembly, the software required for the project to move forward have been carefully selected from a number of state-of-the-art options available. The IDE of choice for development in Python 3.7 was 'JetBrains PyCharm Professional with Anaconda Plugin'. As discussed earlier, the IDE comes packaged with the Anaconda distribution pre-installed and

provides intuitive and productive features. Then, ImageAI was installed for the object detection library which requires several other libraries to be installed as well (refer to [10] for list of libraries). As ImageAI uses TensorFlow and Keras to make the detection possible, they required separate installations. ImageAI is an incredibly resourceful library that provides the project with object detection along with information extracted from the detection such as object count, bounding boxes and classification names.

The project is divided into two separate files *detection.py* and *vibrations.py. Detection.py* contains the part that involves all the object detection and distance calculations, while *vibrations.py* controls the vibrating motors. Each file is modularly written with several functions that are used throughout the project.

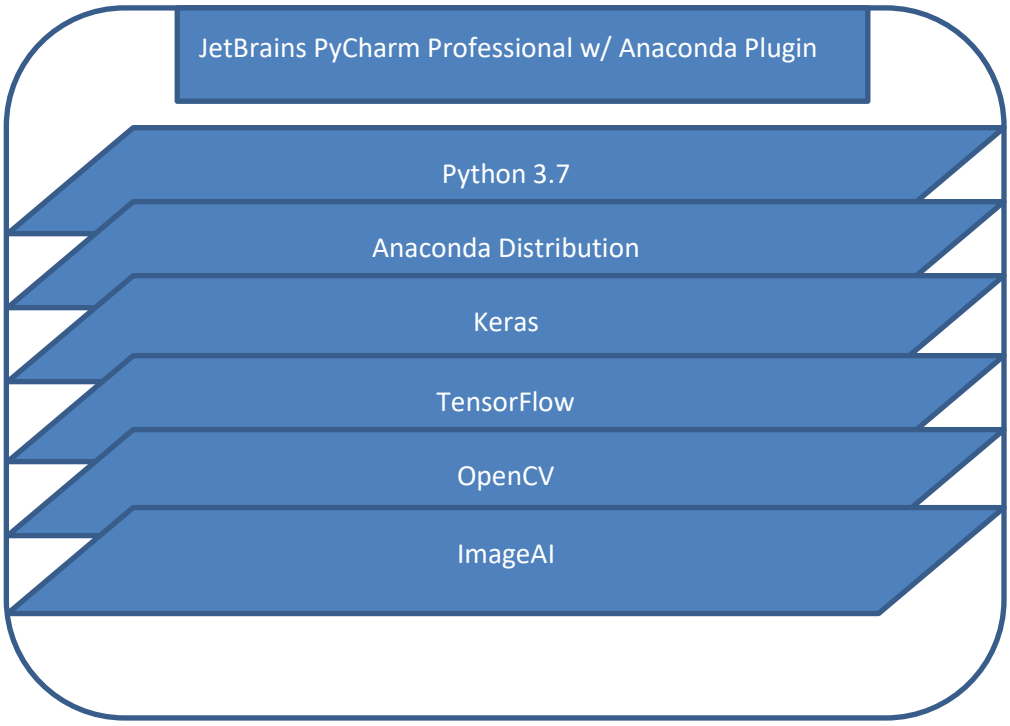


Figure 11. *Project Setup for coding*

3.3 Implementation and Code Discussion

Having considered, planned and designed the requirements, scope and objectives of the project, the next step involves coding in order to program the discussed features. This section looks at the implementation of the ideas presented, justify the choices and also discuss code snippets that require further explanation. It also outlines how the ImageAI library accomplishes object detection and the different ways of distance measurement used to calculate the distance. It also features other methods of distance calculation that could have been implemented. Finally, it looks at implementation features that can be added in the future by discussing the mathematics behind them.

3.3.1 detection.py

Firstly, in order to get the object detection working , the *VideoObjectDetection* class from ImageAI library is imported. A series of other imports have been made, including library *os* that is used to fetch the working directory in order to set the path to the model that is being used. After loading the model and initialising the camera, the program goes to call the video detection library from ImageAI. ImageAI, as discussed in the previous section, makes use of TensorFlow, Keras and OpenCV to get object detection working. With the usage of ImageAI library, the code written in *detection.py* will also demonstrate the convenience and efficacy of using Python as a programming language for object detection.

The function *detectCustomObjectsFromVideo* is used to get the live video detection feature of ImageAI. This function has a number of parameters that are being used to get the information derived from the live video feed. The code documentation/comments describe all the parameters used within the function. The most important part of the function statement is *per_frame_function* which allowed usage of the convenient feature in python- passing a custom function as the argument, which will be called for every frame. The function defined as *process_per_frame* provides the logic to be calculated for every frame. The process_per_frame has the algorithm to calculate the distance and send haptic feedback to the glove via vibrations.py. Likewise, the parameter *custom_objects* allows only enabling particular object from being detected, which has been used for items commonly found on a dining table.

The parameters of the function provide the information extracted by ImageAI, namely: the position of the frame; an array of dictionaries with names of detected objects, their probability percentage and location in the frame; a dictionary with object name as keys and number of instances as value; and a *numpy* array of the frame itself. As the detected objects are returned in an array, an enhanced for loop is implemented to go through the items of the array. For each item in the array, firstly the program checks if a person was detected. The application only starts processing the distance measurement if it detects a person as the hand detection makes use of the person class. Making everything nested under the condition of a person being detected was decided to ensure the distance measurements could be possible to the selected object. Following the detection, the (x, y) coordinates of the lower-left of the bounding box and the top-right of the bounding box are extracted to find a number of reference points. Firstly, the centre of the hand is calculated by using the midpoint formula $(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$. This midpoint is later used to find the location of the objects in regard to the position of the hand. Similarly, using the (x, y) coordinates, the midpoints for each side of the hand is also calculated so based on which side of the hand the object is at, the reference point for the hand can be set to mid-point of that side. The plot-points for each object detected according to the *box_points* extracted is represented in figure 12. Following the midpoints for each side, the algorithm implements a nested loop, this time to find if the object sought for was detected in the frame. A similar approach is used as previous with an enhanced for loop,

which then checks if the input object name matches the name of the detected object in each iteration. Once a match is found, the centre of the object is calculated similar to previous, and the midpoints for each side of the object is calculated as well. The location is checked based on the x and y coordinate values, i.e. if the x coordinate of the object's centre is less than the x co-ordinate of the hand's centre, the object is towards the left and so on, and is done using eight conditional statements for the eight directions. For instance, if the object is determined to be on the right side of the hand, the mid-point of the right side of the hand and the mid-point of the left side of the objects are used to calculate the distance. Similarly, if the object is located above the hand and towards the right, the top-right vertex is set as the reference point for the hand, and the bottom right is set as the reference point for the object. This allows the distance calculation between the object and the hand to be more precise and ensures the hand does not go through the object's bounding box.

Figure 12. *Bounding box vertices from object detection*

Following determining where the object lies relative to the hand, the guiding process begins by using *vibrations.py* function named buzz(further discussion in following sub-section). Then, the distance is calculated. There are a number of ways to calculate the pixel distance in an image[66].

Given the information available of both the reference points, two different distance measurement methods have been applied. Firstly, the Euclidean method- where the two points are used to get the diagonal length between the two points. Euclidean distance can be implemented for the project as the frame does represent a two-dimensional grid. However, for additional measurements for testing, the Manhattan distance is also implemented. Manhattan distance is the sum of vertical and horizontal distances. Manhattan distance, also known as L_1 distance, where the distance between two points (x_1, y_1) and (x_2, y_2) is calculated by : $|x_1 - x_2| + |y_1 - y_2|$. This can further be extended if there are other dimensions such

that if each point has n-dimensions such as : $x_1, x_2, \ldots, x_n$ and $y_1, y_2, \ldots, y_n$, the Manhattan distance formula can translate to be $|x_1 - y_1| + |x_2 - y_2| + \ldots |x_n - y_n|$. Manhattan Distance is considered to be more precise for image processing in general[67]. The testing subsection will further look at the differences between using Manhattan distance and Euclidean distance.

An overview of the algorithm to get the locations of the person and the object, and the distance between them for the code in *process_per_frame* can be seen below:

Algorithm 1 : Get object, location and distance

```
1. BEGIN
2.  FOR each item in objects_array
3.     IF item = "person"
           # get location
4.         person x1,y1, x2, y2 := get_box_points(item.get("box_points_person"))
5.         FOR each item in objects_array
6.            IF item = input
                 # get location
7.               object x1,y1, x2, y2 := get_box_points(item.get("box_points_person"))
                  # Calculate distance
8.               distance_to_object := manhattan_distance(object, person)
9. END
```

Similarly, following the object location and distance, an overview of the algorithm to get the location of the object relative to the hand is shown below for object at the left and above:

Algorithm 2 : Get object location relative to hand and vibrate

```
1. BEGIN
    # check for object on the left
2.  IF centre_x_object < centre_x_person
        # set reference point for hand to left midpoint
3.      ref_x_person, ref_y_person := left_midpoint_x_person, left_midpoint_y_person
        # set reference point for object to right midpoint
5.      ref_x_object, ref_y_object := right_midpoint_x_object, right_midpoint_y_object
        #buzz the left vibration motor
    # check for object above
6.   ELIF centre_y_object > centre_y_person:
       # set reference point for hand to top midpoint
7.     ref_x_person, ref_y_person := top_midpoint_x_person, top_midpoint_y_person
       # set reference point for object to bottom midpoint
8.     ref_x_object,ref_y_object:=bottom_midpoint_x_object, bottom_midpoint_y_object
       #buzz the top vibration motor
9. END
```

The algorithm mentions triggering the vibration motors at each conditional statement. This is done through the function *buzz* in *vibration.py*. The arguments passed to this function are the location of the vibration motor, the length of time of the vibration in milliseconds, and the name of the object. The initial vibration, when the relative object location is found, is shorter at 5 milliseconds and is used as a notification to the user that the object has been found and to move slowly towards the direction of the haptic vibration. Further discussion in the following sub-section.

With the distance between the hand and the object calculated, it is to be monitored in each frame. So, a couple of conditionals checking the distance at two different points will let the user know how close the hand is to the object by an incrementing length of the vibrations. It also lets the user know what kind of object it is by vibration patterns which will further be discussed in the Vibrations.py subsection. The vibration times increase at a distance between 300px and 200 px, and then more at a distance lesser than 300px. Once the hand is within 50px, all vibration motors are activated for 100 milliseconds letting the user know the hand is at the object location.

Amongst the parameters of ImageAI function *detectCustomObjectsFromVideo,* the parameter named *return_detected_frame* takes in a custom function *set_return_detected_frame* as argument. This is particularly done to ensure the program terminates when the user wants as *return_detected_frame* returns the frame of object detection from the video. Similarly, the parameter *custom_objects* takes in the defined objects to be detected. A list of all objects that can be set to *True/False* can be found in appendix 6. Also, the *frames_per_second* parameter was particularly interesting as in real-time, since the program is dealing with each frame, processing multiple frames in a second could rather be moot as the haptic feedback itself will require the time to be acted upon by the user. It is set at 2 frames per second as that is how many times the haptic feedbacks can be followed by a user in a second in an effective manner. This detection module itself is inside a while loop which looks for user confirmation to continue after an object has been found. This implementation requires a text input of “Y”/”y” or “N”/”n”. Further discussion about this can be found in the future work sub-section of conclusions. At the end, the camera is shut, the object detection is stopped, and the glove’s serial port is closed as well.

3.3.2 vibrations.py

This file, as the name suggests, provides the haptic feedback to the user. This file of the program uses Windows COM ports to communicate through the Bluetooth to ESP32 board. Detailed instructions can be found at the top of the file itself to set the device up in a windows 10 machine. It uses two different libraries, *serial* - for port connections, and *time* - for vibration patterns. Firstly, a base location code dictionary is established, with location codes being (t, b, l, r, a) or (top, bottom, left, right, all) respectively. Each location indicates the location of the vibration motors on the glove itself. Each location key in the dictionary has a string of numbers as its value. The format of the vibration signal follows the location value + ‘buzz’ + length of time as a string value and a punctuation mark (‘.’). This is then encoded into utf-8 [69] and then written into the serial which converts this digital utf-8 value into analogue through the control board via Bluetooth and ultimately the vibration motors.

The implementation of the *vibrations.py* has a function to open the glove- i.e. activate the serial port set up for the board and another to close it as well. Following the two, the main function where the vibrations are sent through is named *buzz*. This function takes three parameters, the location of the vibration motor, the length of vibration, and the name of the

object. Using these three parameters, the vibrations are sent according to the type of object that the user has input and where the hand is. There are 5 different kinds of objects programmed initially using conditional statements, namely bottle, wine glass, cell phone, knife and fork. Each object has a function to maintain the modularity of the codes. The functions use loops to set the vibration patterns, and the function *sleep* imported from the library *time* is used to create the patterns. The pattern list formulated for the five objects can be seen in the table below:

Object	Pattern
Bottle	• • •
Cell Phone	• – •
Wine Glass	• – –
Fork	• –
Knife	– •

(•) – Short vibration

(–) – Long vibration

The number of objects and patterns can be further extended as required; however, for the purpose of demonstration of the idea, these five patterns show the techniques that could be employed.

The code also has a few more functions for vibrating the top + right, top + left, bottom + right and bottom + left. They are all used while the location of the object is being checked in relation to the location of the hand in as mentioned in *detection.py.*

3.3.3 Analysis

Coding the detection and the vibrations modules have been done following standard practices. As indentation is an essential paradigm of python codes, the formatting had to be done carefully. The codes also use some pythonic ways to do tasks, such as usage of for loops in variable initialisation. Code repetition has been kept at minimal, and the only repeated where it is unavoidable. The files *detection.py* and *vibrations.py* were both regularly backed up on a personal cloud storage on top of having it version controlled using git. A *requirements.txt* file with all dependencies and libraries required to fork/run the project in a different machine has been placed in the root folder.

The coding practices try to follow the industry standards to maintain a professional setting. While the code itself could be more pythonic, the general conveniences of the language have been utilised to the best of knowledge. The code has been formatted for readability and has brief, useful comments for complex statements and instruction to setup. Each function also has documentation associated with it.

3.4 Testing

The device was initially tested on two different machines during development. One of the testing machines the project had access to, had a much stronger specification with a higher configuration of GPU as well as CPU. As the project progressed, testing was not done on the bigger machine as access to it was lost due to unforeseen circumstances and an alternative machine of the same calibre was not available. So, most of the tests are run on the system specifications that was mentioned in 3.2.2.

Firstly, the object detection worked as intended, with every frame extracting names of the objects, the bounding boxes, and their locations. Figure 13 shows an example of detections captured via screenshots. The accuracy of the detection using YOLO as predicted previously was fairly high, with failing detections only with issues such as occlusion, illumination variability and low-light. As it can be seen from figure 13, despite a fairly low-light scene, the probability of detections is quite high. The tests done in daylight were more accurate than ones done with room-lights at night. There were other instances of failed detections which were attributed to *minimum_percentage_probability,* which is set at 75. Decreasing its value to 50, which is also the default value, did get more detections; however, it also meant the chances of getting the detections wrong were higher. The most commonly found error was the detection of fingers as knives, which also mostly happened in low-light situations.

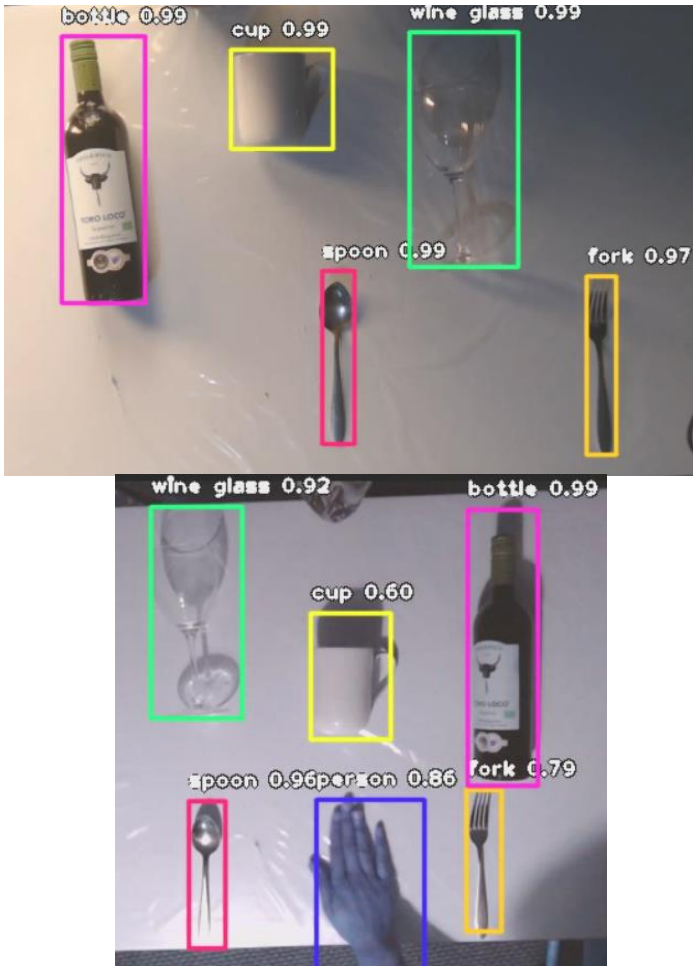


Figure 13. *Examples of Object Detection from overhead view*

Distance measurements wise, there were two different methods used. The usage of Euclidean distance measurement was used first, which was consistently reliable, however, because the Manhattan distance measurements would accurately represent the pixel widths, it was adopted instead. The ratio between Manhattan Distance to Euclidean Distance was approximately 1: 1.30, with the measurements that were used. Manhattan distances were always easier to keep track of as well, as the distance is measured in pixel units. As the hand movements are conveyed vertically or horizontally, and not diagonally, the final tests were carried out using Manhattan distance to compliment the movements.

The vibrations module has also been integrated successfully and is fully functional, wearable and wireless. The battery life proved to be enough to find all five objects during testing, which took around 2 minutes to 2 minutes 40 seconds from the time the detection began when done consecutively. Vibration patterns, however, were not easily discernible from one another but because the objects have to be selected before finding them, it did not serve much of a purpose for this build. The problem with the patterns was the time it takes to be fully output. For instance, a 100-millisecond vibration three times in a row with 100-millisecond gaps between them takes half a second. If the process is at two frames per second, this will not be able to be output as it will take longer to be carried out after taking the processing time into account and having shorter vibrations are not noticeable enough to be distinguished. However, the patterns will be useful for future builds, depending on a different kind of implementation which is further discussed in 4.2.

The tests would get a lot of insight on real-life applications if it were to be conducted on the target users, i.e. visually impaired individuals. Several attempts were made to contact the Forth Valley Sensory Centre with no avail. Set of questionnaires was prepared as well(see appendix 4) but because of lack of contact, it did not materialise. Also, at the time of testing the device, with the UK in lockdown, the tests with several users were not possible. The tests were all carried out with a blindfold to simulate visual impairment. While this is not the ideal most way to test, because of the lack of testers, two house members volunteered to test the device. The two users were consensually blindfolded and asked to comment on the device. As neither of the testers was visually impaired, the tests do not reflect a real-world use case by someone who has experienced visual impairment. However, the layout of the table and the object positions were shuffled after blindfolding the users to make the tests as close to realistic as possible. The user feedback from both the users was that the device guided the hand close to the object successfully; however, they were always wary if they would hit other things on the way. The objects were spaced in a layout that allowed the hand to move to the object without obstruction; however, the issue mentioned is one of the lacking features of the project. Because of the computation and time limitations, the implementation has not been done; however, a solution is discussed in the future work section. The two test users were asked to test the device throughout the development process, who reported the haptic feedbacks were getting easier to follow in each iteration of implementation as they were getting a feel for the vibration patterns as well as directions. While personally testing the device, the top and the bottom vibrations were difficult to discern at first, however, the more I tested it, the more comfortable I felt distinguishing between them. The tests were all carried out with the right hand, as the glove is oriented specifically for the right hand. However, the same idea can be applied to a left-handed glove, which would translate in the same way as the directions are based on the relative position of the objects to the hand.

The tests could have been carried out more extensively, but because of the circumstances and the unavailability of a more powerful device, the results are limited. Despite the shortcomings, all in all, the device functioned as expected, and the implementations were available to be experienced and tested upon.

4 Conclusion

4.1 Summary

The project has managed to achieve relative success in all different aspects of development. Starting with the design and setup that allowed a relatively smooth development environment, moving on to successfully implementing a form of solution. A lot of research has been done, with a number of literature reviews for different aspects of this project which has increased the understanding of the techniques used in the discussed areas. If depth estimation can be integrated with the project, the final form factor can be achieved with a single camera such that it can be mounted at a first-person view. However, while working under the limitations, the solution presented demonstrates the idea successfully, which, along with the expansion of my understanding of these topics, was the objective of the project.

4.2 Future Work

As seen from previous sections, several improvements have been touched upon. This sub-section will look into them in detail and propose solutions where applicable. This project itself is entirely replicable with the specifications provided within the dissertation.

One of the limitations that could be faced would be the battery life of the device. As mentioned previously in 3.4, the battery comfortably lasted the working time of around 2 minutes to 2 minutes 40 seconds to find the five objects, which also includes having to input run again confirmation as well as to the object type. However, prolonged use would require a larger battery or a more efficient vibration module. Especially if the form-factor is to be a wristband ultimately, the battery will have to be compact and optimised for the vibration module. For this implementation of the solution, as the user inputs the object they want to find, the vibration patterns only serve as a distinguishing feature; however, if the project were to be implemented in a different format, it would become incredibly useful. For instance, an implementation idea would be to make the device vibrate every time the hand is close to an object without a specific object being sought for. In this case, the vibration patterns would allow the user to distinguish between the objects, and pick which one they would want. The implementation of this would have to be re-engineered slightly. The distance measurements could still be implemented in the same way; however, the hand's location would have to be monitored such that if it gets closer than a certain distance to any object, the vibrations will fire up with patterns of the object closest and the direction. The location references of each object should be monitored, which would allow the calculation of the distance to any of the locations detected. This would require a lot of computation, but with a powerful enough system, it could be an option to be explored as most of the logic to be able to do so are already present in this version of the solution.

Likewise, training a custom model for each use-case would be one of the most desirable features for the improvement of this project. Training models for specific scenarios would also help in minimising the size, as currently the pre-trained YOLOv3 model this project uses sits at 237 megabytes. This size could potentially be minimised to a third by training custom models with only objects required for the scenes as the model used has 80 different objects pre-trained. Training a custom model would vastly improve the system as having the hand or the back of the palm isolated will make the measurements significantly more precise. Article at [77] provides a great tutorial on how to train a custom YOLOv3 model. This project did not dwell on custom training a model as the pre-trained model provided all the objects that were necessary for the demonstration of the ideas discussed. However, the addition of a custom

trained model with specific objects will likely improve the performance as well as the experience.

The biggest feature that will need to be implemented is avoiding obstruction. For this particular implementation, a potential solution could be by treating it as a spatial problem, i.e. by using the four points between the object and the hand, which will always be a quadrilateral as the detection bounding boxes are always rectangular. Then, using the four points between the object and the hand, check if any of the location references of other objects detected are within that area. There are multiple ways to check if a point lies within a polygon/quadrilateral [71,72]. An implementation of one of the ideas has been done in function *in_polygon* in *detection.py* which uses *matplotlib* and *numpy* to accomplish the task. This function will enable checking all the object location points to be checked and returns the ones that are inside the specified points(here the vertices), which can then be used to determine which objects are inside the area. The function has not been used in this version of the implementation but is left there to facilitate further work within this project. There are other methods that can be applied that require manual calculations using area of triangles that can be found in appendix 5. Finding out if there are other objects in the space is only half the problem as the hand needs to be guided around the object. Given it is implemented on a right hand, this could mean the hand would have to be guided around the right side of the objects. To accomplish this, the distance between the relevant vertex of the bounding boxes would have to be monitored, thus allowing the feedback to be able to guide the hand. All of these calculations will require high computation power, so seeing these features implemented will require an equally capable system as well.

Similarly, as discussed, using stereo cameras will vastly change the system while enabling the camera to be mounted in a first-person view. The solution implemented in this project is mostly two dimensional so the entirety of the code will have to be re-engineered when the third dimension is introduced as ImageAI does not support a z-axis, the library will have to be abandoned as well. Hence, implementing a 3-dimensional system will also need more time and computation power than what was required and available for this project. Moving to 3D could be done with TensorFlow, or it could also be worth looking into shifting the development to using PyTorch. Projects such as StereoLabs' 'ZED' SDK provides methods for 3D object detection that uses TensorFlow[74]. Meanwhile, PyTorch also in February 2020 introduced the open-source PyTorch 3D library for 3D Deep Learning[75]. Taking heed from these resources, the device can be extended to extract the 3D information and build a more advanced version of the ideas presented by this project.

Other ideas that could be implemented may include integrating this idea with knowledge representation and predicting what items generally sit at what side of the table. This is a rather large addition that would make the project more AI driven, as knowledge representation in itself is a complicated topic. If successful, the system would be able to understand the general layout of dining tables, i.e. what item goes where, then use that to further enhance the detection and thus give a more comprehensive feedback to the user as a result.

Further along, image segmentation could also be an avenue to be explored for a more granular level of distance calculation [78,79]. Multiple techniques of image segmentation have been discussed in [78], which include region-based segmentation and Mask R-CNN, both of which have been explained thoroughly with sample codes provided within. Segmentation is generally used for highly sensitive environments such as bio-medical engineering/sciences, which require precision at the highest of levels. The article in[80] compares the segmentation techniques and concludes that it is unrealistic to find an image segmentation method that can adapt to all images, suggesting further developments will be required. It also suggests that using multiple techniques of image segmentation might be necessary at present, which leads to the belief that applying it to live video feeds will be even more difficult. However, as the

research continues, it can be one of the more advanced features of the device to be considered in the future.

4.3 Final Evaluation

To reflect back on the problem that was tackled, admittedly, there were several aspects that were overlooked before the undertaking of the project. For instance, object detection issues such as illumination variability and pose variability; and implementation of monocular depth perception with distance management on the ground were incredibly difficult subjects that have been researched extensively over the years which were not in my horizon of knowledge. Ambitious as it was, having a 3-D device with a fully wearable form factor was rather unattainable in hindsight. Having had little knowledge about object detection, CNNs and depth estimation at the start of the project, it served as a motivation to understand the general field of deep learning and object detection techniques along with implementation of distance management calculations. The task of getting the objects detected from a live feed in a 3D environment with the available resources was a reaching idea with the time constraints and other university work/commitments. However, while pursuing it, the research done on the state-of-the-art has provided me with an understanding of industry standards in the field and where the innovations are focused at. It has also taught me about object detection algorithms, how custom models are trained, and the libraries and platforms that are available towards it. The techniques discussed in this dissertation such as homography, Otsu's approach and creating disparity maps have wide use cases and have been used for several innovative researches in computer vision. The field of depth estimation and deep learning object recognition is incredibly vast and is ever evolving. This project provided an unmissable opportunity to delve into these subjects and understand how they work.

Similarly, getting a chance to apply all the aspects of development learned in theory such as engineering the requirements, designing a solution and trying to follow the design, provided a sense of what real-world projects could be like. Also, maintaining a professional communication channel with the supervisor helped in emulating a work-like project environment. While it did not involve collaboration, it certainly encouraged the application of other theoretical and practical knowledge gained throughout the degree.

The other aspect of this project was programming for a wearable device, of which the results can be physically seen. Having never had done such projects previously, the experience was incredibly stimulating and the reward of getting to see the physical manifestation of the software written was gratifying. Hence, the coding process was easily the most rewarding portion of this project as it demanded research as well as problem solving, both of which a computer scientist will always have to do. The ideas that have been used in the codes have been clearly and concisely commented and the codes make use of the convenience python provides as a language.

Objectively, the project implementation itself is usable and demonstrates the idea quite clearly, however, it is not at the stage where it can be reliably used by visually impaired individuals on grounds of the discussed shortcomings. In saying so, the project ran smoothly, provided challenges and required a lot of research; and the dissertation captures the entirety of the project's ideas, their implementations and its future potential. A number of future improvements have been suggested along with providing resources to address them which can help take it further. The ultimate objective of the project was to understand the topics at hand and develop a device using the acquired knowledge, which was the case throughout the duration of this project.

References

[1] "Vision impairment and blindness", *Who.int*, 2019. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment. [Accessed: 01- Mar- 2020]

[2] T. Mazumdar, "Global blindness set to 'triple by 2050'", *BBC News*, 2017. [Online]. Available: https://www.bbc.com/news/health-40806253. [Accessed: 01- Mar- 2019]

[3] S. Crowe, "Researchers back Tesla's non-LiDAR approach to self-driving cars", The Robot Report, 2019. [Online]. Available: https://www.therobotreport.com/researchers-back-teslas-non-lidar-approach-to-self-driving-cars/. [Accessed: 01- Mar- 2020]

[4] About OrCam: Meet the purpose behind the Tech of Ziv Aviram and Amnon Shashua", OrCam, 2020. [Online]. Available: https://www.orcam.com/en/about/. [Accessed: 01- Mar- 2020]

[5] Object detection", En.wikipedia.org, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Object_detection#/media/File:Detected-with-YOLO--Schreibtisch-mit-Objekten.jpg. [Accessed: 26- Feb- 2020]

[6] Why Anaconda for Data Science?", Anaconda, 2020. [Online]. Available: https://www.anaconda.com/why-anaconda/. [Accessed: 02- Mar- 2020]

[7] A. Beklemysheva, "Why Use Python for AI and Machine Learning?", Steelkiwi.com, 2020. [Online]. Available: https://steelkiwi.com/blog/python-for-ai-and-machine-learning/. [Accessed: 03- Mar- 2020]

[8] "JetBrains PyCharm for Anaconda", JetBrains, 2020. [Online]. Available: https://www.jetbrains.com/pycharm/promo/anaconda/. [Accessed: 02- Mar- 2020]

[9] M. Olafenwa, "ImageAI", http://imageai.org/, 2020. [Online]. Available: http://imageai.org/. [Accessed: 03- Mar- 2020]

[10] M. Olafenwa, "OlafenwaMoses/ImageAI", GitHub, 2020. [Online]. Available: https://github.com/OlafenwaMoses/ImageAI/blob/master/imageai/Detection/__init__.py. [Accessed: 02- Mar- 2020]

[11] M. Olafenwa, "Video and Live-Feed Detection and Analysis — ImageAI 2.1.5 documentation", Imageai.readthedocs.io, 2020. [Online]. Available: https://imageai.readthedocs.io/en/latest/video/. [Accessed: 03- Mar- 2020]

[12] J. Brownlee, "A Gentle Introduction to Object Recognition With Deep Learning", Machine Learning Mastery, 2019. [Online]. Available: https://machinelearningmastery.com/object-recognition-with-deep-learning/. [Accessed: 03- Mar- 2020]

[13] J. Redmon, "YOLO: Real-Time Object Detection", Pjreddie.com, 2020. [Online]. Available: https://pjreddie.com/darknet/yolo/. [Accessed: 2- Mar- 2020]

[14] OrCam MyEye 2.0", OrCam, 2020. [Online]. Available: https://www.orcam.com/en/myeye2/. [Accessed: 03- Mar- 2020]

[15] J. Longman, "Blind Runner's Wearable Technology Gets Off to Complicated Start", Nytimes.com, 2017. [Online]. Available: https://www.nytimes.com/2017/11/05/sports/blind-marathoner-technology.html. [Accessed: 03- Mar- 2020]

[16] "How It Works — WearWorks", WearWorks, 2020. [Online]. Available: https://www.wayband.co/new-page. [Accessed: 03- Mar- 2020]

[17] G. Balakrishnan, G. Sainarayanan, R. Nagarajan, and S. Yaacob. A stereo image processing system for visually impaired. *International Journal of Signal Processing*, 2(3):136, 2008.

[18] P. Chippendale, V. Tomaselli, V. D'Alto, G. Urlini, C. Modena, S. Messelodi, S. Strano, G. Alce, K. Hermodsson, M. Razafimahazo, T. Michel and G. Farinella, "Personal

Shopping Assistance and Navigator System for Visually Impaired People", *Computer Vision - ECCV 2014*

[19]M. Leo, G. Medioni, M. Trivedi, T. Kanade and G. Farinella, "Computer vision for assistive technologies", *Computer Vision and Image Understanding*, vol. 154, pp. 1-15, 2017.

[20]J. Redmon, "Hardware Guide: Neural Networks on GPUs (Updated 2016-1-30)", Pjreddie.com, 2020. [Online]. Available: https://pjreddie.com/darknet/hardware-guide/. [Accessed: 05- Mar- 2020]

[21]Amazon.com EVGA GeForce GTX TITAN X, 2020. [Online]. Available: https://www.amazon.com/EVGA-GeForce-GAMING-Graphics-12G-P4-2990-KR/dp/B00UVN21RQ?th=1. [Accessed: 13- Mar- 2020]

[22]P. Kornreich and B. Farell, "Single eye or camera with depth perception", Photonics North 2012, 2012.

[23]J. Seal, D. Bailey and G. Sen Gupta, "Depth perception with single camera. ", 2005. [Online]. Available: https://www.researchgate.net/publication/228946792_Depth_perception_with_a_single_camera [Accessed: 05- Mar- 2020]

[24]Y. Wei, Z. Dong and C. Wu, "Depth measurement using single camera with fixed camera parameters", IET Computer Vision, vol. 6, no. 1, p. 29, 2012.

[25]A. Lamża, Z. Wróbel and A. Dziech, "Depth Estimation in Image Sequences in Single-Camera Video Surveillance Systems", 2020.

[26]L. Fridman, "Computer Vision for Self-Driving Cars", MIT Deep Learning, 2018. [Online]. Available: https://www.dropbox.com/s/a5iqp1df5j1xydj/computer_vision_2018.pdf?dl=0. [Accessed: 02- Mar- 2020]

[27]F. Lee, "Choosing a 3D Vision Camera", IoT For All, 2017. [Online]. Available: https://www.iotforall.com/choosing-3d-vision-camera/. [Accessed: 02- Mar- 2020]

[28]"10 Wide-Angle HD Webcams for Skype HD Video Calling » TechUserFriendly.com", TechUserFriendlycom, 2015. [Online]. Available: https://techuserfriendly.com/10-wide-angle-hd-webcams-skype-hd-video-calling-list-field-of-view-1080p-720p-logitech-c930e-cc3000e-c920-genius-f100/. [Accessed: 04- Mar- 2020]

[29]G. Finlayson, "Colour and illumination in computer vision", Interface Focus, vol. 8, no. 4, p. 20180008, 2018.

[30]P. Grover, "Evolution of Object Detection and Localization Algorithms", TowardsDataScience, 2018. [Online]. Available: https://towardsdatascience.com/evolution-of-object-detection-and-localization-algorithms-e241021d8bad. [Accessed: 06- Mar- 2020]

[31]C. Aggarwal, Neural Networks and Deep Learning, 1st ed. Springer, 2018, pp. 315-419.

[32]"ImageNet", Image-net.org, 2020. [Online]. Available: http://www.image-net.org/. [Accessed: 16- Mar- 2020]

[33]"Overview of the YOLO Object Detection Algorithm", Medium, 2020. [Online]. Available: https://medium.com/@ODSC/overview-of-the-yolo-object-detection-algorithm-7b52a745d3e0. [Accessed: 15- Mar- 2020]

[34]A. Ng, "Non-max Suppression - Object detection | Coursera", Coursera, 2020. [Online]. Available: https://www.coursera.org/lecture/convolutional-neural-networks/non-max-suppression-dvrjH. [Accessed: 04- Mar- 2020]

[35]A. Christiansen, "Anchor Boxes — The key to quality object detection", Medium, 2020. [Online]. Available: https://medium.com/@andersasac/anchor-boxes-the-key-to-quality-object-detection-ddf9d612d4f9. [Accessed: 09- Mar- 2020]

[36]S. Praveen, "Efficient Depth Estimation Using Sparse Stereo-Vision with Other Perception Techniques", Coding Theory, 2019.
[37]S. Paul, "Stereoscopic Depth Sensing – A Python Approach", Artificial Intelligence - The Key To The Next Door of Evolution, 2017. [Online]. Available: https://aikiddie.wordpress.com/2017/05/24/depth-sensing-stereo-image/. [Accessed: 07- Mar- 2020]
[38]"CSC420: Introduction to Image Understanding - Winter 2019", Cs.utoronto.ca, 2020. [Online]. Available: http://www.cs.toronto.edu/~fidler/slides/2019/CSC420/lecture12.pdf. [Accessed: 05- Mar- 2020]
[39]J. Kocić, N. Jovičić and V. Drndarević, "An End-to-End Deep Neural Network for Autonomous Driving Designed for Embedded Automotive Platforms", Sensors, vol. 19, no. 9, p. 2064, 2019.
[40]M. Diaz-Cabrera, P. Cerri and P. Medici, "Robust real-time traffic light detection and distance estimation using a single camera", Expert Systems with Applications, vol. 42, no. 8, pp. 3911-3923, 2015.
[41]Yin, C.Y.: 'Determining residual nonlinearity of a high-precision heterodyne interferometer', Opt. Eng., 1999, 38, (8), pp. 1361–1365
[42]P. Alizadeh, "Object Distance Measurement Using a Single Camera for Robotic Applications", 2015.
[43]S. Shetty, "What is PyTorch and how does it work? | Packt Hub", Packt Hub, 2018. [Online]. Available: https://hub.packtpub.com/what-is-pytorch-and-how-does-it-work/. [Accessed: 02- Mar- 2020]
[44]S. Yegulalp, "What is TensorFlow? The machine learning library explained", InfoWorld, 2019. [Online]. Available: https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html. [Accessed: 05- Mar- 2020]
[45]S. Lobo, "Deep learning wars: Facebook-backed PyTorch vs Google's TensorFlow", Packt Hub, 2018. [Online]. Available: https://hub.packtpub.com/dl-wars-pytorch-vs-tensorflow/. [Accessed: 03- Mar- 2020]
[46]Z. Sadreddini, T. Cavdar and H. Jond, "A distance measurement method using single camera for indoor environments", 2016 39th International Conference on Telecommunications and Signal Processing (TSP), 2016.
[47]A. Greensted, "Otsu Thresholding Explained", Labbookpages.co.uk, 2010. [Online]. Available: http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html. [Accessed: 07- Mar- 2020]
[48]N. Clinton, "Otsu's Method for Image Segmentation", Medium, 2020. [Online]. Available: https://medium.com/google-earth/otsus-method-for-image-segmentation-f5c48f405e. [Accessed: 16- Mar- 2020]
[49]R. Fisher, "Image Transforms - Hough Transform", Homepages.inf.ed.ac.uk, 2003. [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm. [Accessed: 04- Mar- 2020]
[50]K. Bapat, "Hough Transform using OpenCV | Learn OpenCV", Learnopencv.com, 2019. [Online]. Available: https://www.learnopencv.com/hough-transform-with-opencv-c-python/. [Accessed: 04- Mar- 2020]
[51]B. Templeton, "Elon Musk's War On LIDAR: Who Is Right And Why Do They Think That?", Forbes, 2019. [Online]. Available: https://www.forbes.com/sites/bradtempleton/2019/05/06/elon-musks-war-on-lidar-who-is-right-and-why-do-they-think-that/#7c25ab872a3b. [Accessed: 05- Mar- 2020]

[52]T. Lee, "Elon Musk: “Anyone relying on lidar is doomed.” Experts: Maybe not", Ars Technica, 2019. [Online]. Available: https://arstechnica.com/cars/2019/08/elon-musk-says-driverless-cars-dont-need-lidar-experts-arent-so-sure/. [Accessed: 19- Mar- 2020]
[53]M. Bayern, "The top 3 companies in autonomous vehicles and self-driving cars | ZDNet", ZDNet, 2019. [Online]. Available: https://www.zdnet.com/article/the-top-3-companies-in-autonomous-vehicles-and-self-driving-cars/. [Accessed: 19- Mar- 2020]
[54]"Autonomous Vehicles & Car Companies l CB Insights", CB Insights Research, 2020. [Online]. Available: https://www.cbinsights.com/research/autonomous-driverless-vehicles-corporations-list/. [Accessed: 19- Mar- 2020]
[55]L. He, G. Wang and Z. Hu, "Learning Depth From Single Images With Deep Neural Network Embedding Focal Length", IEEE Transactions on Image Processing, vol. 27, no. 9, pp. 4676-4689, 2018.
[56]"VGG in TensorFlow · Davi Frossard", Cs.toronto.edu, 2016. [Online]. Available: https://www.cs.toronto.edu/~frossard/post/vgg16/. [Accessed: 19- Mar- 2020]
[57]A. Cook, "Global Average Pooling Layers for Object Localization", Alexisbcook.github.io, 2017. [Online]. Available: https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/. [Accessed: 19- Mar- 2020]
[58]I. Salian, "NVIDIA Blogs: AI-enhanced Upscaling on the NVIDIA Shield TV", The Official NVIDIA Blog, 2020. [Online]. Available: https://blogs.nvidia.com/blog/2020/02/03/what-is-ai-upscaling/. [Accessed: 19- Mar- 2020]
[59]D. Shankar, "How Does Face Unlock Work in the Smartphones and is it Safe?", Geeksmate.io, 2018. [Online]. Available: https://www.geeksmate.io/how-does-face-unlock-work-on-android-smartphones-69707. [Accessed: 05- Mar- 2020]
[60]"TensorFlow 2.0 is now available!", Blog.tensorflow.org, 2019. [Online]. Available: https://blog.tensorflow.org/2019/09/tensorflow-20-is-now-available.html. [Accessed: 08- Mar- 2020]
[61]D. Pawluk, N. Bourbakis, N. Giudice, V. Hayward and M. Heller, "Guest Editorial: Haptic assistive technology for individuals who are visually impaired", IEEE Transactions on Haptics, vol. 8, no. 3, pp. 245-247, 2015.
[62]G. Flores, S. Kurniawan, R. Manduchi, E. Martinson, L. Morales and E. Sisbot, "Vibrotactile Guidance for Wayfinding of Blind Walkers", IEEE Transactions on Haptics, vol. 8, no. 3, pp. 306-317, 2015.
[63]S. Ertan, C. Lee, A. Willets, H. Tan and A. Pentland, "A wearable haptic navigation guidance system", Digest of Papers. Second International Symposium on Wearable Computers (Cat. No.98EX215), 1998.
[64]D. Dakopoulos, S. Boddhu and N. Bourbakis, "A 2D Vibration Array as an Assistive Device for Visually Impaired", 2007 IEEE 7th International Symposium on BioInformatics and BioEngineering, 2007.
[65]M. Khder, M. AlZaqebah and A. Abazeed, "Smart Shoes for Visually Impaired/Blind People", 2018.
[66]"METHODS FOR MEASURING DISTANCE IN IMAGES", Shodhganga.inflibnet.ac.in, 2020. [Online]. Available: https://shodhganga.inflibnet.ac.in/bitstream/10603/33597/12/12_chapter4.pdf. [Accessed: 19- Mar- 2020]
[67]"Pinhole camera model", En.wikipedia.org, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Pinhole_camera_model. [Accessed: 09- Mar- 2020]

[68]"Depth Map from Stereo Images — OpenCV 3.0.0-dev documentation", Docs.opencv.org, 2020. [Online]. Available: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_calib3d/py_depthmap/py_depthmap.html. [Accessed: 22- Mar- 2020]
[69]"Python String encode()", Programiz.com. [Online]. Available: https://www.programiz.com/python-programming/methods/string/encode. [Accessed: 23- Mar- 2020]
[70]C. Godard, O. Aodha and G. Brostow, "Unsupervised Monocular Depth Estimation with Left-Right Consistency", CVPR, 2017 [Online]. Available: http://visual.cs.ucl.ac.uk/pubs/monoDepth/
[71]"How to check if a given point lies inside or outside a polygon? - GeeksforGeeks", GeeksforGeeks, 2020. [Online]. Available: https://www.geeksforgeeks.org/how-to-check-if-a-given-point-lies-inside-a-polygon/. [Accessed: 25- Mar- 2020]
[72]"Point in Polygon & Intersect — Intro to Python GIS documentation", Automating-gis-processes.github.io, 2020. [Online]. Available: https://automating-gis-processes.github.io/CSC18/lessons/L4/point-in-polygon.html. [Accessed: 25- Mar- 2020]
[73]"How to check if a point is inside a rectangle?", Mathematics Stack Exchange, 2017. [Online]. Available: https://math.stackexchange.com/questions/190111/how-to-check-if-a-point-is-inside-a-rectangle. [Accessed: 25- Apr- 2020]
[74]"How to Use TensorFlow with ZED | Stereolabs", Stereolabs.com, 2020. [Online]. Available: https://www.stereolabs.com/docs/tensorflow/. [Accessed: 10- Apr- 2020]
[75]N. Ravi, "Introducing PyTorch3D: An open-source library for 3D deep learning", Ai.facebook.com, 2020. [Online]. Available: https://ai.facebook.com/blog/-introducing-pytorch3d-an-open-source-library-for-3d-deep-learning/. [Accessed: 10- Mar- 2020]
[76]"Introducing PyTorch3D: An open-source library for 3D deep learning", Ai.facebook.com, 2020. [Online]. Available: https://ai.facebook.com/blog/-introducing-pytorch3d-an-open-source-library-for-3d-deep-learning/. [Accessed: 10- Apr- 2020]
[77]J. Nelson, "Training a YOLOv3 Object Detection Model with a Custom Dataset", Medium, 2020. [Online]. Available: https://towardsdatascience.com/training-a-yolov3-object-detection-model-with-a-custom-dataset-4981fa480af0. [Accessed: 12- Apr- 2020]
[78]P. Sharm, "Step-by-Step Tutorial on Image Segmentation Techniques in Python", Analytics Vidhya, 2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2019/04/introduction-image-segmentation-techniques-python/. [Accessed: 20- Apr- 2020]
[79]D. Vasani, "Image segmentation with fastai", Medium, 2019. [Online]. Available: https://towardsdatascience.com/image-segmentation-with-fastai-9f8883cc5b53. [Accessed: 18- Apr- 2020]
[80]S. Yuheng and Y. Hao, Arxiv.org, 2017. [Online]. Available: https://arxiv.org/pdf/1707.02051.pdf. [Accessed: 15- Apr- 2020]

Appendix 1

Full specifications of the implementation and testing machine that was used most

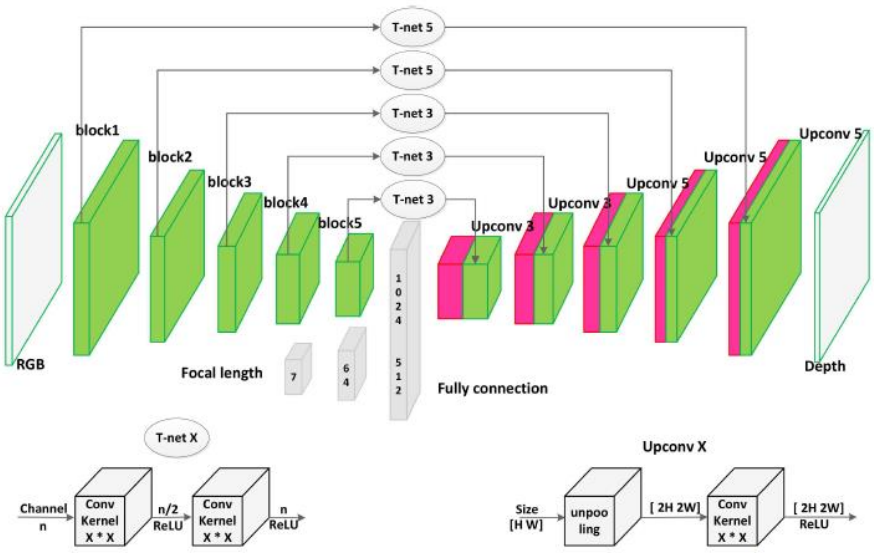
DELL INSPIRON 15

Operating System: Windows 10

Display:	15.6-inch (1920x1080)
Processor:	Intel Core i5-6300HQ Quad Core 6MB; 2.3GHz (3.2GHz Turbo)
Graphics:	Nvidia GeForce GTX 960M 4GB GDDR5
Memory:	16GB DDR3L 1600MHz (8GBx2)
Storage:	1TB 5400 rpm Hybrid Hard Drive with 8GB Embedded Flash Cache
	512GB SSD M.2
Networking:	802.11ac, Bluetooth 4.0
Battery:	74WHr 6-Cell Integrated Battery

Appendix 2 – Academic Review- L.He *et al*

Proposed new neural network in [55].



The proposed network architecture. neural network is built upon the pre-trained model (VGG), followed by a fully connection layer and upsampling architectures to obtain high-resolution depth, by effectively integrating the middle-level information. In addition, focal length is embedded in the network by the encoding mode.

Appendix 3 – Academic Review – Godard *et al*

Qualitative results on Cityscapes, CamVid, and our own urban dataset captured on foot. For more results please see our video.

Video found at https://www.youtube.com/watch?v=go3H2gU-Zck

Appendix -4 Questionnaire

A Guiding Hand: Helping the Visually Impaired Locate Close Objects on a Table Questionnaire

Participant Name
Date
Preferred Method of Contact
Contact details
Location
Time Started – Time finished

QUESTION	**ANSWERS**
1) What devices do you use, if any, to assist you in everyday life?	
2) How often do you visit restaurants?	☐Once/more every week ☐Once every two weeks ☐Once a month ☐Less frequently than once a month
3) What challenges, if any, do you face in a restaurant table?	
4) Would you consider haptic feedback to be better over audio instructions based on your experience?	
5) Have you tried anything similar to the device?	
6) If yes, was it effective? What did you like and not like about the device?	
7) What feature do you believe would be helpful to have from the project description?	
8) What were your initial impressions about this project?	
9) Any other comments?	

Appendix 5

Alternate techniques to find if a point exists in a polygon.[73]

Let $P(x, y)$, and rectangle $A(x_1, y_1), B(x_2, y_2), C(x_3, y_3), D(x_4, y_4)$

Calculate the sum of areas of $\triangle APD, \triangle DPC, \triangle CPB, \triangle PBA$.

1. If this sum is greater than the area of the rectangle, then $P(x, y)$ is outside the rectangle.
2. Else if this sum is equal to the area of the rectangle (observe that this sum cannot be less than the latter),
    1. if area of any of the triangles is 0, then $P(x, y)$ is on the rectangle (in fact on that line corresponding to the triangle of area$= 0$). Observe that the equality of the sum is necessary; it is not sufficient that area$= 0$),
    2. else $P(x, y)$ is is inside the rectangle.

Acceptably this approach needs substantial amount of computation. This approach can be employed to any irregular polygon, too.

[73]

Another way is to calculate the perpendicular distances of $P(x, y)$ from all the 4 lines AB, CD, AD, BC

To be inside the rectangle, the perpendicular distances from AB, P_{AB}(say) and from CD, P_{CD} (say) must be less than $|AD| = |BC|$ and the perpendicular distances from AD, P_{AD}(say) and from BC, P_{BC}(say) must be less than $|CD| = |AB|$. Here , the areas of each of the four triangles $<$ $\frac{1}{2}$ the area of the rectangle.

1. If one of the perpendicular distances is greater than the respective length, then $P(x, y)$ is outside the rectangle.

    This essentially implies and is implied by the statement : the area of the respective triangle $> \frac{1}{2}$ the area of the rectangle (as commented by Ben Voigt) as $\triangle APD = \frac{1}{2} AD \cdot P_{AD}$.
2. Else if $P_{AB} = 0$ and $P_{CD} = |AD|$, then $P(x, y)$ is on AB . So, $\triangle PBA = 0$ and $\triangle PCD = \frac{1}{2}$ the area of the rectangle.

    Observe that in this case, the remaining two perpendicular distances P_{AD}, P_{BC} must be $\leq$ $|AB| = |CD|$, $P_{BC} = |AB| \implies P(x, y)$ is lies on AD i.e, P coincides with A as it is already on AB .

[73]

Appendix 6

All objects that can be detected/set to be detected by the YOLOv3 model used during the project.

person, bicycle, car, motorcycle, airplane, bus, train, truck, boat, traffic_light, fire_hydrant, stop_sign, parking_meter, bench, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe, backpack, umbrella, handbag, tie, suitcase, frisbee, skis, snowboard, sports_ball, kite, baseball_bat, baseball_glove, skateboard, surfboard, tennis_racket, bottle, wine_glass, cup, fork, knife, spoon, bowl, banana, apple, sandwich, orange, broccoli, carrot, hot_dog, pizza, donut, cake, chair, couch, potted_plant, bed, dining_table, toilet, tv, laptop, mouse, remote, keyboard, cell_phone, microwave, oven, toaster, sink, refrigerator, book, clock, vase, scissors, teddy_bear, hair_dryer, toothbrush