Indoor Navigation for People with Visual Impairment

DOI:10.18136/PE.2021.799

PhD Thesis

Submitted for the Degree of

Doctor of Philosophy in Computer Science

Mostafa Abdallah Abbas Atwa Elgendy

Supervisor:

Cecilia Sik-Lanyi

Associate Professor

Department of Electrical Engineering and Information Systems

Doctoral School of Information Science and Technology

University of Pannonia

Veszprém, Hungary

2021

INDOOR NAVIGATION FOR PEOPLE WITH VISUAL IMPAIRMENT

Thesis for obtaining a PhD degree in the Doctoral School of Information Science and Technology of the University of Pannonia.

in the branch of Information Science

Written by: Mostafa Abdallah Abbas Atwa Elgendy
Supervisor (s): Cecilia Sik-Lanyi

Propose acceptance (yes / no) .....
(supervisor/s)

As reviewer, I propose acceptance of the thesis:

Name of reviewer: ..... (yes / no)

.....
Reviewer

Name of reviewer: ..... (yes / no)

.....
Reviewer

The PhD. candidate has achieved ..... % at the public discussion.
Veszprem/Keszthely, ...

.....
Chairman of the Committee

The grade of the PhD diploma ..... (..... %)
Veszprem/Keszthely, ...

.....
Chairman of the UCDH

Acknowledgment

First and foremost, I would like to express my thanks to God for guiding and aiding me to bring this work out to light. My deep thanks and highest gratitude go to my supervisor, Dr. Cecilia Sik-Lanyi for her patience, motivation, enthusiasm, and immense knowledge. I cannot possibly express anymore of my gratitude to her, not only on the guidance she gave during my study as a PhD student but valuable life experiences as well.

I would like to thank the Director of the Doctoral School Prof. Katalin Hangos for her help and support during the doctoral school report presentation. Thanks to the staff of University of Pannonia, especially Ujvari Orsolya, Lenyi Silvia, Dulai Tibor and Gorbe Peter. They assisted me in every possible way and went through all the office works for me to have a good experience in Hungary.

I acknowledge and thank the Dean and Deputy Dean of the Faculty of Information Technology, Prof. Hartung Ferenc, and Dr. Werner Agnes, for the financial support under the project EFOP-3.6.1-16-2016-00015.

My deep thanks to the Egyptian Ministry of Higher Education and Scientific Research and to the Hungarian Ministry of Higher Education for their cooperation with Egypt to have my study in Hungary.

Last but not the least; I would like to thank my family, my parents, whose love, and guidance are with me in whatever I pursue. They are the ultimate role models. Most importantly, I wish to thank my loving and supportive wife Bosy, and my lovely daughter Salma who provide unending inspiration.

Mostafa Elgendy, 2021

Abstract

People with visual impairment face various problems in doing daily activities in comparison to people without visual impairment. Much research has been done to find smart solutions using mobile devices to help them perform tasks like navigation and shopping. One of the most challenging tasks for researchers is to create a solution that offers a good quality of life for people with visual impairment. It is also essential to develop solutions that encourage them to participate in social life. The essential steps of a typical navigation system are identifying the current location, finding the shortest path to the destination, and navigating safely to the destination using navigation feedback. In this thesis, an overview is given about the various technologies that have been developed in recent years to assist people with visual impairment navigating indoors. It introduces the latest direction in this area, which will help developers to incorporate such solutions into their research. A comparison has been made between different technologies used in developing solutions to select the best one from the available solutions. A system has been proposed to help people with visual impairment navigating indoor using markers. The system is able to detect and avoid obstacles during navigation when needed. The navigation system has been improved to detect markers from a longer distance using CNN model. The system has been improved using a deep learning model which is called Tiny-YOLOv3. Several modified versions of the original model have been implemented and compared to improve the detection accuracy. A dataset has been created by collecting marker images from recorded videos and augmenting them using some techniques such as rotation transformation, brightness, and blur processing. After training and validating this model, the performance was tested on a testing dataset and real videos.

Összefoglaló

A látássérült emberek különböző problémákkal néznek szembe a napi életvitelük során, szemben azon emberekkel, akiknek nincs látássérültségük. Számos kutatás létezik olyan okos megoldásokra, amelyek mobil eszközök segítségével segítenek nekik olyan feladatok végrehajtásában, mint a navigáció és a vásárlás. A kutatók számára az egyik legnagyobb kihívást jelentő feladat olyan megoldás létrehozása, amely jó életminőséget kínál a látássérült emberek számára. Elengedhetetlen olyan megoldások kidolgozása is, amelyek ösztönzik őket a társadalmi életben való részvételre. A tipikus navigációs rendszer lényeges lépései az aktuális hely azonosítása, a célhoz vezető legrövidebb út megtalálása és a navigáció visszajelzései útján történő biztonságos navigálás. Ebben a tézisben áttekintést nyújtok az elmúlt években a látássérült emberek beltéri navigálásának segítésére kidolgozott különféle technológiákról. Bemutatom a legújabb irányt ezen a kutatási területen, amely segíti a fejlesztőket abban, hogy az ilyen megoldásokat beépítsék kutatásaikba. Összehasonlításra kerülnek a megoldások kidolgozásához használt különféle technológiák, hogy a rendelkezésre álló megoldások közül a legjobbat válasszák ki. Javasolok a látássérült emberek számára egy beltéri markerek segítségével történő navigációs renndszert. Szükség esetén a rendszer képes felismerni és elkerülni az akadályokat a navigáció során. A CNN modell segítségével tovább fejlesztettem a navigációs rendszert, hogy nagyobb távolságból lehessen érzékelni a jelzőket. A rendszert Tiny-YOLOv3 mély tanulási modell segítségével fejlesztettem. Az eredeti modell több módosított változatát implementáltam és összehasonlítottam az észlelési pontosság javítása érdekében. Így egy adatkészlet jött létre a rögzített videókból származó marker képek összegyűjtésével és egyes technikákkal, például rotációs transzformációval, a fényerő és elmosódások feldolgozásával. E modell tanítása és validálása után a teljesítményt tesztelő adatkészleten és valódi videókon teszteltem.

Résumé

Les individus ayant une déficience visuelle sont confrontées à divers difficultés dans leurs activités quotidiennes par rapport aux personnes sans déficience visuelle. De nombreuses recherches ont été menées pour trouver des solutions intelligentes utilisant des appareils mobiles pour les aider à accomplir des tâches telles que les déplacements quotidiens et les achats. L'une des tâches les plus difficiles pour les chercheurs est de créer une solution qui offre une bonne qualité de vie aux personnes malvoyantes. Il est également primordial de développer des solutions qui les encouragent à participer à la vie sociale. Les étapes primordiales d'un système de navigation typique consistent à identifier l'emplacement actuel, à trouver le chemin le plus court vers la destination et à naviguer en toute sécurité jusqu'à la destination à l'aide des commentaires de navigation. Dans cette thèse, un aperçu est donné sur les différentes technologies qui ont été développées ces dernières années pour assister les personnes malvoyantes à naviguer à l'intérieur. Il présente les dernières orientations dans ce domaine, qui soutiendront les développeurs à intégrer de telles solutions dans leurs recherches. Une comparaison a été faite entre les différentes technologies utilisées dans le développement de solutions pour choisir la meilleure parmi les solutions disponibles. Un système a été proposé pour aider les personnes malvoyantes à naviguer à l'intérieur à l'aide de marqueurs. Le système est capable de détecter et d'éviter les obstacles pendant la navigation en cas de besoin. Le système de navigation a été amélioré pour détecter les marqueurs à plus longue distance en utilisant le modèle CNN. Le système a été amélioré à l'aide d'un modèle d'apprentissage en profondeur appelé Tiny-YOLOv3. Plusieurs versions modifiées du modèle original ont été adaptées et comparées pour améliorer la précision de détection. Un ensemble de données a été créé en collectant des images de marqueurs à partir de vidéos enregistrées et en les augmentant à l'aide de techniques telles que la transformation de la rotation, la luminosité et le traitement du flou. Après avoir formé et validé ce modèle, les performances ont été testées sur un ensemble de données de test et de véritables vidéos.

List of abbreviations

ACM	Association for Computing Machinery
AGPS	Assisted GPS
AI	Artificial Intelligence
ANN	Artificial Neural Networks
AP	Access Points
AR	Augmented Reality
AT	Assistive Technology
CNN	Convolutional Neural Networks
CV	Computer Vision
DG	Deformable Grid
DL	Deep Learning
FEU	Feature Extraction Unit
FN	False Negative
FP	False Positive
GPS	Global Positioning System
ICF	International Classification of Functioning, Disability and Health
IEEE	Institute of Electrical and Electronics Engineers
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IOU	Intersection Over Union
IR	Infrared
mAP	mean Average Precision
MAT	Mobile Assistive Technology
ML	Machine Learning
MLP	Multilayer Perceptron
NEI	Navigation Efficiency Index
NFC	Near Field Communication
P	Precision
PVI	People with Visual Impairment
QR	Quick Response
R-CNN	Region-based CNNs
R	Recall
RELU	Rectified Linear Unit
RFID	Radio Frequency Identification
RNN	Recurrent Neural Networks

SVM	Support Vector Machines
SSD	Single Shot Detector
TN	True Negative
TP	True Positive
VI	Visual impairment
YOLO	You Only Look Once

List of figures

List of tables

Table of contents

1 Introduction

People with Visual Impairment (PVI) have weaknesses in the function of their visual system. The environment lack of support causes them to depend on their relatives and prevent them from seeing and doing daily activities, such as navigation or shopping. This chapter contains the motivation and challenges behind this thesis. It highlights the challenges they face during navigating indoors and how to solve them using indoor navigation system. Then, the contributions of this thesis and the research methodology to accomplish them are introduced. Finally, the research activities and the outline of this work are presented.

1.1 Motivation and Scope

Visual Impairment (VI) is one of the world's most important health problems which reduces seeing or perceiving ability. VI results from various diseases and degenerative conditions which are hard for correction through wearing glasses or using conventional means. VI has been classified into near or distance vision impairment and results from many reasons, such as uncorrected refractive errors, age-related eye problems, glaucoma, cataracts, diabetic retinopathy, trachoma, corneal opacity, or unaddressed presbyopia [1]. The current global population is 7.6 billion. It is expected to be 9.2 billion in 2050 as shown in Figure 1-1.

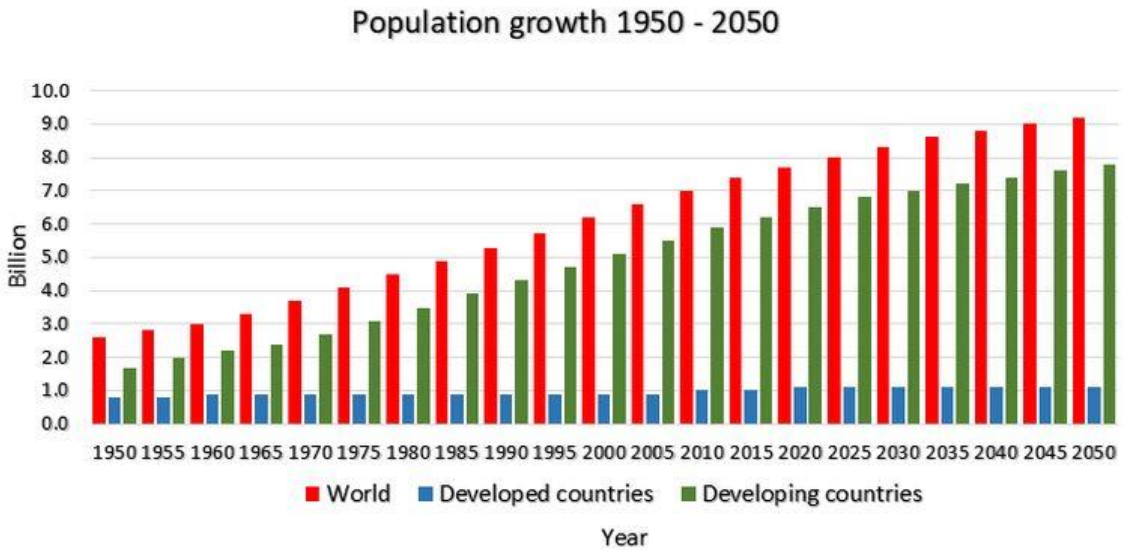


Figure 1-1. Population growth all over the world from 1950 to 2050.

Based on the World Health Organization (WHO) report, more than 200 million people worldwide are visually impaired. As shown in Figure 1-2, this number will increase in the following years. Among them, 39 million were blind, and 246 million had low vision [2] [3]. About 81% of the people who are blind or have moderate vision impairment are aged about 50 years and above. VI is one of the most sensory disabilities, which causes a deprivation of entire multi-sense perception

for an individual. About 80% of people who suffer from VI or blindness belong to middle- and low-income countries, where they cannot afford costly assistive devices.

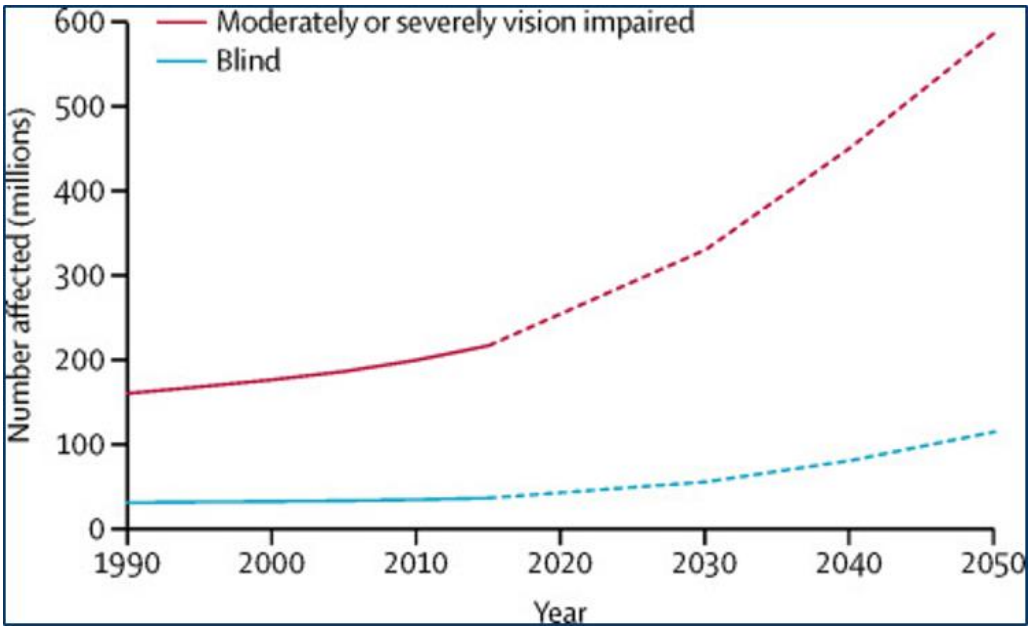


Figure 1-2. People with visual impairment numbers over years.

Modern buildings, such as airports, hospitals, and shopping malls become complex. The complex structural layout of these buildings makes it difficult to navigate easily. Not only for PVI but, also for people with normal vision who get lost easily. So, people with normal vision use landmarks and geographical layouts to find their way and navigate easily. However, PVI have limitations in their visual system so, it difficult to navigate in these places. The lack of support services in the surrounding environment makes them overly dependent on their families and prevents them from seeing and doing daily activities, such as navigation or shopping. For example, PVI have difficulties in reading product labels during shopping; they thus miss important information about the content of their food and sometimes make bad choices. During shopping, PVI face navigation troubles, which encourage them to consume takeout food. Another problem is how to walk in an environment with many barriers such as walking in unknown places or crossing a street. Last, but not least, PVI face social barriers such as the attitudes of other people and society [4][5]. Therefore, providing them with an advanced and helpful navigational tool will be necessary for the following three perspectives: First, it will reduce some of the sufferings they face, improve their mobility, and protect them from injury. Second, it will help them to live without any help from others. Third, it will encourage them to travel outside of their environments and interact socially, benefiting society by fully utilizing the talents and abilities [6].

Lately, portable devices such as smartphones, smart glasses, smartwatches, and notebooks, have become popular. These devices have various capabilities that are useful when developing new complex applications [7]. These devices can access information from any place and, at any time. So, PVI can use them in their daily activities. In this way, mobile devices are used with Assistive Technology (AT) to offer multiple solutions which are called Mobile Assistive Technology (MAT) [8]. As a result, researchers have developed several methods using Wi-Fi, Radio Frequency Identification (RFID), and Near Field Communication (NFC). These systems are useful however, they are unsuitable for daily use as they are complex to use. Furthermore, the

size or weight of these systems are hard to wear for a long time. Besides, several systems require preinstalling infrastructure, which is expensive and hard to implement in particular locations with strict requirements such as, hospital [9]. Because of these limitations, it is desirable to build an indoor navigation system with low cost and minimum preinstalled infrastructure requirement. So, this research has concentrated on Computer Vision (CV) solutions using the smartphone camera.

A typical CV navigation system for indoor navigation uses unique installed tags such as Augmented Reality (AR) markers to help in navigating indoors and recognizing objects. It consists of tags installed in place, a database to store tag information, a camera to capture real-time pictures, a processing unit to execute the used techniques, and feedback to help PVI with navigation commands. The system works as follows: A database is used to store the building map, which consists of interest points and the connection between them. When the application starts, it opens the camera to get a live stream of images and converts them to grayscale ones. Then, it sends the converted images to the processing unit to detect and identify the marker based on the used techniques. If any marker is detected, the system calculates the distance to that marker and gives a voice feedback to help PVI in reaching it. This detected marker is used as an initial point, and the destination point is given as an input using voice commands. The shortest path is calculated between these two points, and PVI start navigating to reach their destination. Detecting and identifying markers are the most crucial parts. However, the system may fail to identify markers in many real-life situations such as motion blur or distortion, lighting conditions, or the marker is too far from the camera. In recent years, Deep Learning (DL) have been used in the field of CV to improve object detection. The deep convolutional neural network increases the network level, which makes the network have stronger detection capabilities. Currently, there are several deep Convolutional Neural Networks (CNN)s however, some of them are not suitable for applications executing in real-time due to the expensive running process. The purpose of my thesis is to develop such a system for PVI. To Summarize, multiple research questions derive from the motivations presented above:

1. **Q1:** What are the main categories of MAT solutions for PVI, and What are the strengths and weaknesses of the latest MAT systems for PVI?
2. **Q2:** How to select the best technology for the navigation system, and how to build a navigation system to help PVI navigating indoors? How to help PVI to identify objects and avoid them easily during navigation?
3. **Q3:** How to improve the navigation system to detect markers from longer distances using DL techniques?
4. **Q4:** How to improve the navigation system to detect markers easily in challenging conditions using DL techniques?

1.2 Objectives

As it was stated above, the overall aim of this thesis is to build a navigation system to help PVI navigate indoors. It also discusses how the proposed solutions can help PVI in the navigation process and summarizes the challenges and drawbacks of the proposed solutions. To achieve this purpose, the following progressive objectives are defined:

- **Objective 1:** To understand, organize and systematize the existing knowledge about DL and MAT. To find the strengths and weaknesses of the latest MAT systems for PVI, and how to develop an indoor navigation system for PVI. This objective corresponds to the first research question.
- **Objective 2:** To compare different technologies and select the best one from these available solutions. Build a system to help PVI navigating indoors using the selected technology. Improve the navigation system to avoid objects during navigation. This objective corresponds to the second research question.
- **Objective 3:** To improve the navigation system to detect markers from a longer distance using CNN model. This objective corresponds to the third research question.
- **Objective 4:** To improve the navigation system to detect markers from challenging conditions using YOLOv3 CNN model. This objective corresponds to the fourth research question.

1.3 Research Methodology

The research field of this thesis is evolving fast due to technological advances and the continuous generation of new knowledge in MAT and DL. Therefore, an iterative research methodology that allows us to review the state-of-the-art regularly was followed. The main idea of this cyclical process is that the knowledge acquired in its initial phases helps us to design increasingly original contributions capable of improving the understanding and knowledge in the areas wherein this thesis is focused. This cyclical process has multiple iterations done during the three years of this Ph.D. thesis. Figure 1-3 shows the different phases of this research methodology as briefly described:

1. **Review and Analysis of the state-of-the-art:** this stage is focused on investigating the state-of-the-art related to the field MAT under consideration to identify gaps and challenges in current literature. To achieve this aim, the relevant bibliography is used, reviewing both publications from the scientific community published in journals and proceedings of worldwide conferences. The knowledge acquired in this phase led to formulate the research proposal in the first year of this Ph.D.
2. **Design and Development:** in this phase, different proposals to approach the identified challenges are designed and developed. To this end, previously acquired or updated knowledge (new literature review) was used to ensure that the solution was always up to date with the current state-of-the-art.
3. **Experimentation and Evaluation:** the goal of this phase is to test the proposals resulting from the previous step to a process of experimentation. To carry out this procedure, it is crucial to provide some criteria and evaluation methods with which the results will be compared in the subsequent phase. All these criteria and methods must be built using the knowledge acquired in the first stage of the methodology.
4. **Results Analyses and Comparison:** after carrying out experimentation, results must be analysed and contrasted with those obtained in the state-of-the-art. At this point, it is needed to check if the results obtained are enough to address the challenges identified in the first phase. In such a case, another methodological cycle begins to approach the following challenge identified or to keep working with the challenge under consideration

if it was not still solved. In this stage, conclusions must be drawn from analyses of results, and knowledge obtained must be materialized in scientific dissemination, either through journals or conferences.

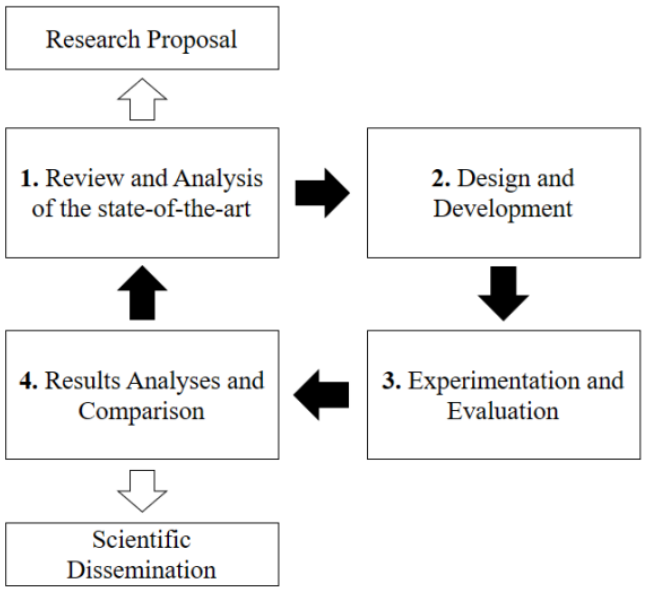


Figure 1-3. Research methodology of this thesis.

1.4 Contributions and Publications

The work presented in this dissertation focuses on building a navigation system for PVI. The main contributions of this thesis and their associated scientific production are presented below:

- A taxonomy that provides a view of the different MAT solutions that helps PVI in different problems. *This contribution is approached in Chapter 3.*

- Comparing different technologies and select Aruco markers as the best ones from the available solutions. Build a navigation system to help PVI navigating indoors using Aruco markers. Improve the navigation system to avoid objects during navigation. *This contribution is approached in Chapter 4.*

- Improve the navigation system to detect markers from longer distances using CNN model. *This contribution is approached in Chapter 5.*

- Improve the navigation system to detect markers from challenging conditions using YOLOv3 CNN model. *This contribution is approached in Chapter 6.*

1.5 Thesis Organization

The structure of the remainder of this thesis dissertation is outlined below.

- **Chapter 2** shows background about MAT and DL.

- **Chapter 3** reviews related work about using DL and MAT to build navigation systems for PVI. This chapter is therefore aligned with Specific Objective 1.
- **Chapter 4** presents the comparison criteria between different technologies, and which one is the best for the proposed system. It shows the architecture of the navigation system. Introduces the problem of avoiding objects and how to solve it using DL This chapter is therefore aligned with Specific Objective 2
- **Chapter 5** provides a thorough analysis of the drawbacks of the proposed navigation system and how to solve the problem of detecting markers from a longer distance. The work presented in this Chapter is therefore directly related to Specific Objective 3.
- **Chapters 6** introduces You Only Look Once (YOLO) models and how to use them to detect markers in challenging conditions. This solves some of the drawbacks of the methods identified in Chapter 4. This Chapter is aligned with Specific Objective 4.
- **Chapters 7**. revisits the main goal and specific objectives posed in this Ph.D. thesis and summarizes the main contributions of this research.

2 Background

There are a lot of technologies and research done in helping PVI. This thesis focuses on using AT and DL to help PVI navigating indoors. This chapter presents a brief background and theory of some of the concepts used throughout this thesis such as AT, CV, Machine Learning (ML), and DL. First, AT is introduced, the necessity for using it for building efficient systems. Afterward, the basis of CV and the main task which can be done by it are detailed. Next, what interpretability means within the ML field and why it is necessary are explained. Finally, a brief description of DL is provided.

2.1 Assistive Technology

Impairment is defined as any loss or abnormality in an anatomical structure or a physiological or psychological function. Disability is any restriction or lack of ability to perform an activity in the manner or within the range considered normal for a human being [10]. This way, VI is defined as the limitation of actions and functions of the visual system. These lacks lead to loss of visual acuity or visual field, photophobia, diplopia, visual distortion of images, visual perceptual difficulties, or any combination of the above features. Vision function has four categories: normal vision, moderate VI, severe VI and blindness. So, the last three can be combined and called VI. Vision loss is the sensor disability causes a deprivation of entire multi-sense perception for an individual. Currently, more than 285 million PVI worldwide and this number will increase in the coming decades. PVI face a lot of difficulties during their daily activities. The lack of services support in the environment makes them always need help from their relatives. It also prevents them from being economically active and socially included [11].

International Classification of Functioning, Disability and Health (ICF) provides a framework for organizing and documenting functioning and disability. Using this model provides a common language for discussing the type of disability and how to overcome it by interaction with people and the environment around them. ICF describes situations regarding human functioning and its restrictions and serves as a framework to organize this information. It structures the information in a meaningful, interrelated and easily accessible way. The ICF offers a multidimensional aspect to describe health and disability as a dynamic interaction between a person's health condition, environmental factors and personal factors [12]. ICF organizes information in two parts, as shown in Figure 2-1. The first part deals with functioning and disability, while the second part covers contextual factors. The first part is divided into body, activity, and participation components, while the second part is divided into environmental factors and personal factors. The body component has two classifications, one is related to the functions of the body systems, while the other is for the body structures. Activity component is the execution of a task or action by an individual. Finally, the participation component is related to incorporate people with a disability in a life situation. Environmental factors are the second part which has an impact on all functioning and disability components. They are organized in sequence from the individual's most immediate environment to the general environment, while personal factors are also components of contextual factors [13].

To apply ICF framework for PVI, each component in this framework should be applied on them. Let's talk first about body functions and structure: PVI have limitation of actions and functions of the visual system which prevent them from seeing and communicating with environment in a

normal way. Moreover, they have some activity limitations and participation restrictions which lead to problems with overweight and isolation from the world around them. For example, PVI face a lot of difficulties in reading product labels, as a result they fail to obtain information about the nutritional content of food. This makes it difficult to choose a good food. Furthermore, the independent navigation difficulty in a grocery may limit the frequency of shopping, making it less viable to buy healthier food and encourage them to consume more prepared food. This way, they always need help while shopping which might make them feel uncomfortable. Activity limitations in walking, environmental barriers, and the lack of accessible exercise equipment can hamper a person's ability to be physically active. To talk about environmental factors, PVI may experience environmental barriers as attitudes of individuals and the society communication services, systems, and policies. PVI face some challenges which put them in a dangerous situation, so they always need help from other people. As a result, they create and live in their own world and feel that they are isolated from the world around them [14].

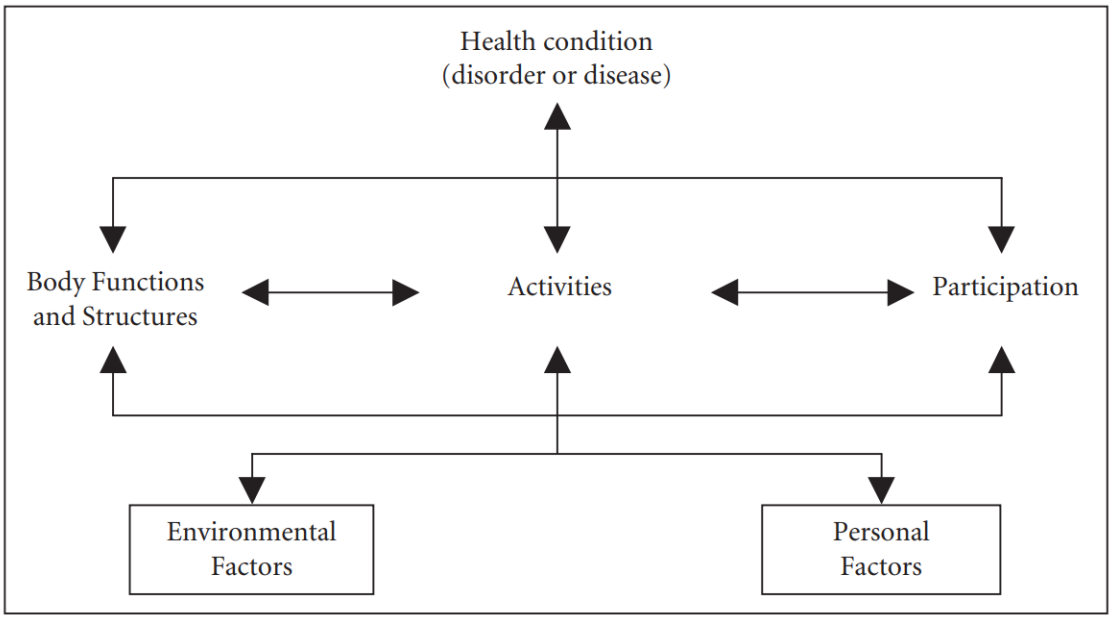


Figure 2-1. ICF architecture.

AT helps PVI to support body functions and prevent any activity limitations or participation restrictions. So, they enhance their quality of life and allow them to be included in society and act as typically developed people. There are various definitions of AT. Common in all of them is that "Assistive technology consists of devices and other solutions that assist people with deficits in physical, mental, or emotional functioning as alternative ways of performing actions, tasks, and activities". Appropriately, a growing number of PVI are using smartphones in their daily activities. The advent of mobile phones, in particular smartphones, has piloted a new era of connectivity where users can access information any time and, in any place. This way, smartphones are used with assistive technology to provide several solutions, which is called MAT. MAT helps PVI do a lot of tasks like navigation, shopping and controlling everything at home. It has the potential to enhance the quality of life for PVI via improved autonomy and safety. Furthermore, it encourages and pushes them to travel outside their environment and to interact socially [15].

2.2 Computer Vision

The core concept of any Artificial Intelligence (AI) system is to perceive the environment and act based on these perceptions. CV is a subfield of AI concerned with the visual perception part. It is the science of perceiving and understanding the world through images and videos. It constructs a physical world model to take appropriate action. For humans, vision is just one aspect of perception, and there are other perceptions like sound, smell, and other senses. Depending on the application you are building, select the sensing device that best captures the world. So, Visual perception is the act of observing patterns and objects through visual or visual input. For autonomous vehicles as an example, visual perception means understanding the surrounding objects such as pedestrians or traffic signs and understanding what they mean. So, CV is the way of building systems that can understand the environment through visual input. At the highest level, vision systems are almost the same for humans, animals, and most living organisms. They consist of a sensor that captures an image and a brain that processes and interprets an image. Then, it outputs a prediction based on the data extracted from the input image, as shown in Figure 2-2.

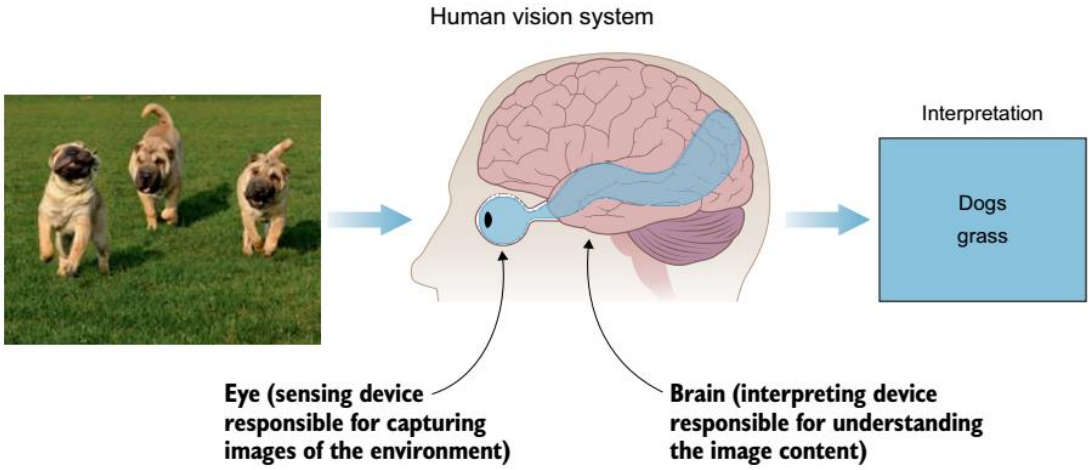


Figure 2-2. The human vision system.

Researchers have done interesting jobs in imitating visual ability with machines. To do it, there is a need to have the same two main components. The first one is a sensing device to mimic the function of human eyes. The second component is an algorithm to mimic the brain function in interpreting and classifying image content, as shown in Figure 2.3. An important design aspect is selecting the best sensing device to capture the surrounding environment, such as a camera, X-ray, or CT scan. These devices provide the environment's full scene to fulfill the task. For example, the main goal of CV in an autonomous vehicle is to understand the surrounding environment and move safely on time. So, they added a combination of cameras and sensors that can detect pedestrians, cyclists, vehicles, roadwork, and other objects. CV systems consist of a sensor and an interpreter. For sensors, cameras are most often considered the equivalent of the eyes for a computer vision system. There are other sensors, such as distance sensors, laser scanners, and radars. However, different combinations of these sensors are selected depending on the application. The interpreter, such as CV algorithms is the brain of the vision system. It takes the output image from the sensing device and learns features and patterns to identify objects. So, it is important to build an artificial brain [16][17].

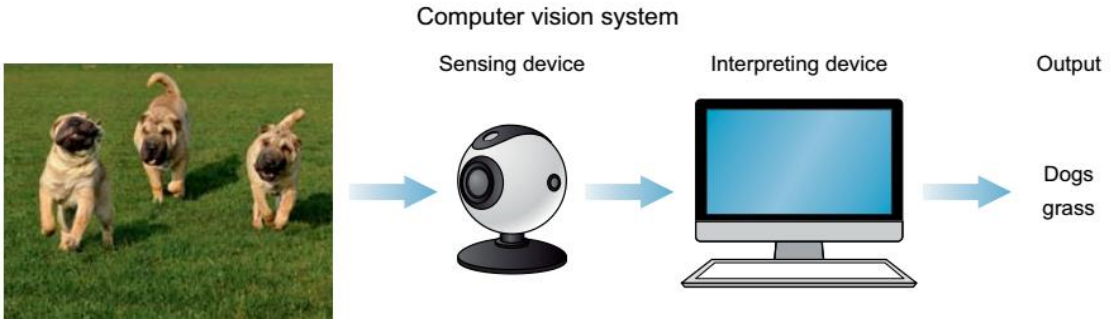


Figure 2-3. The components of the computer vision system.

CV is used for a set of tasks to achieve highly sophisticated applications:

- **Image classification:** to determine the category of a given image based on a set of predefined categories. Let's take a simple binary example: if you want to categorize the input images according to whether they contain a cat or a dog [18].
- **Localization:** is used to find the exact location of a single object in an image. For example, if you want to know the place of the dog in the input image. The standard way to perform localization is to define a bounding box enclosing the object in the input image [19].
- **Object detection:** to find and then classify several objects in an image. It is a combination of localization and classification repeated for all objects in the input image. An application of object detection is detecting people or obstacles [20].
- **Object identification:** is different from object detection, although similar techniques are used to achieve them both. Given an input image, object identification is used to find whether a specific object appears or not. If the object is found, it specifies the exact location of it. An example may be searching for images that contain the logo of a specific company [21].
- **Instance segmentation:** is the next step after object detection. It creates a mask for each detected object that is as accurate as possible [22].
- **Object tracking:** is to track the moving object over time by utilizing consecutive video frames as the input. It is useful in human tracking systems that try to understand customers behavior. Object tracking is done by applying object detection to each image in a video sequence. Then, it compares the instances of each object to determine how they moved [23].

2.2.1 Computer vision pipeline

A typical vision system uses a sequence of distinct steps to process and analyse image data which are referred to as a computer vision pipeline. Many vision applications follow the flow of acquiring images and data, processing that data, performing some analysis and recognition steps, and then finally making a prediction based on the extracted information, as shown in Figure 2-4. To apply the pipeline to an image classifier example. Suppose there is an image of a motorcycle, and there is a model to predict the probability of the object from the following classes: motorcycle, car, and dog. Let us see how the image flows through the classification pipeline:

1. **Image input:** A computer receives visual input from an imaging device like a camera. This input is captured as an image or a sequence of images forming a video. CV applications deal with images or video data. An image is represented as a function of two variables x and y,

which define a two-dimensional area. The pixel is the raw building block of an image. Every image consists of a set of pixels with values representing the intensity of light in a given place in the image. To represent a specific pixel, F is the function and x, y is the location of the pixel in x- and y-coordinates. For example, the pixel located at x = 12, and y = 13 is white; this is represented by the following function: F (12, 13) = 255.

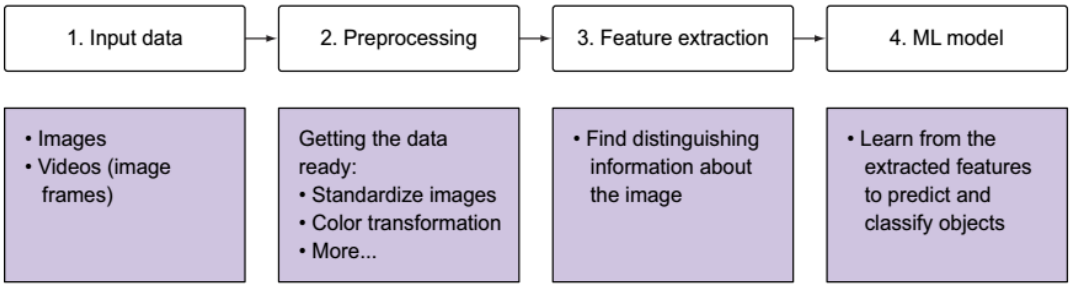


Figure 2-4. The computer vision pipeline.

2. **Image pre-processing:** The acquired data is usually messy and comes from different sources. To feed it to the ML model, it needs to be standardized and cleaned up. Based on the problem and the dataset, some image processing is required before feeding them to the ML model. Each image is sent through some pre-processing steps whose purpose is to standardize the images. Common pre-processing steps include resizing an image, blurring, rotating, changing its shape, or transforming the input image from one colour to another, such as from colour to grayscale.

3. **Feature extraction:** Features help us define objects. A feature is a measurable piece of data in your image that is unique to that specific object. It may be a distinct colour or a shape such as a line, edge, or image segment. A strong feature can distinguish objects from one another. For example, the wheel is a strong feature that clearly distinguishes between motorcycles and dogs. However, it is not strong enough to distinguish between a bicycle or a motorcycle. In CV projects, the image is transformed into a feature vector and is used by the learning algorithm to learn the characteristics of the object. As shown in Figure 2-5, the raw input image of a motorcycle is feed into a feature extraction algorithm which produces a vector that contains a list of features. This feature vector is a 1D array that makes a robust representation of the object. The output of this process is a feature vector to identify the object.

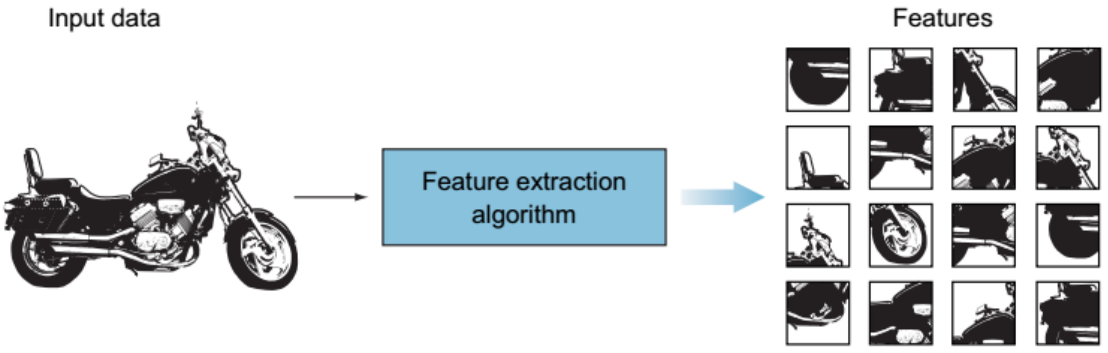


Figure 2-5. Input image is fed to a feature-extraction algorithm to create the feature vector.

4. **Learning algorithm:** The features are fed into a classification model. This step looks at the feature vector from the previous step and predicts the class of the image. The classification task is done using traditional ML algorithms like SVMs, or deep neural network algorithms like CNNs. While traditional ML algorithms might get decent results for some problems, CNNs truly shine in processing and classifying images in the most complex problems. To do it, you look at the list of features in the feature vector one by one and try to determine what is in the image: a- First you see a wheel feature; could this be a car, a motorcycle, or a dog? It is not a dog, because dogs do not have wheels (at least, normal dogs, not robots). Then this could be an image of a car or a motorcycle. b- You move on to the next feature, the headlights. There is a higher probability that this is a motorcycle than a car. c- The next feature is rear mudguards—again, there is a higher probability that it is a motorcycle. d- The object has only two wheels; this is closer to a motorcycle. e- the model keeps going through all the features, like the body shape and pedal until the best guess of the object in the image. The output of this process is the probability of each class. As shown in Figure 2-6, the model can predict the right class with the highest probability. However, there is still a little confusion about distinguishing between cars and motorcycles. To improve accuracy, more training images can be added, more processing to remove noise, extract better features, change the classifier algorithm, or allow more training time.

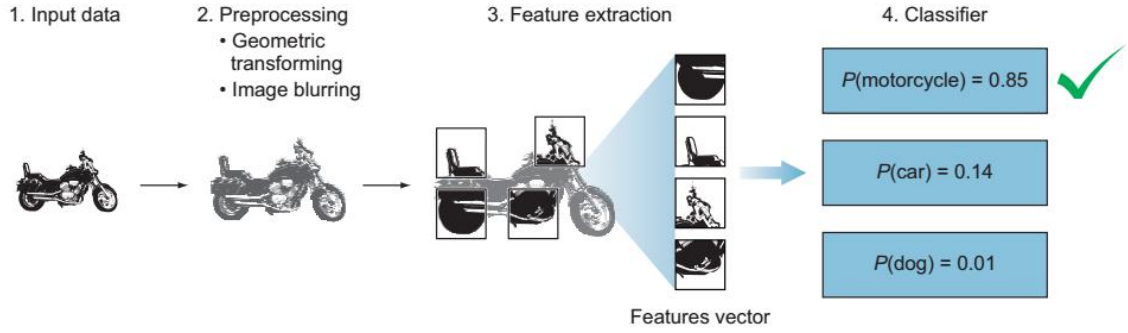


Figure 2-6. Using ML model to predict the probability of the motorcycle object.

2.3 Machine Learning

ML is the field focusing on algorithms which have the ability of modifying and adapting their behaviour employing an iterative learning process without the need of being explicitly programmed. Applications like e-mail spam and malware filtering, face recognition are built upon ML. Within the ML field, detection techniques can be classified into supervised and unsupervised learning. Techniques regarding these categories differ in the type of data they use, which may be labelled or not. The fact that the data is labelled means that each observation of the dataset has an associated label that identifies it as normal or anomalous. In contrast, if no label is available, it is not possible to know the nature of a given observation. Also, all these applications do not include only ML algorithms but also data pre-processing techniques, as it has proven key to gleaning quality data before applying ML methods [24]. The connection between data pre-processing techniques and an ML algorithm generates a pipeline whose main structure is presented in Figure 2-7.

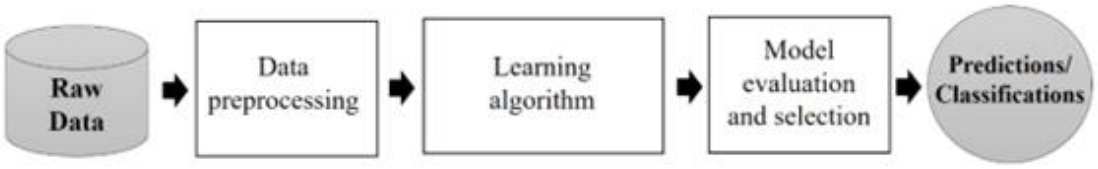


Figure 2-7. Standard structure of a machine learning pipeline.

2.3.1 Data Pre-processing

Input data must be provided in the format that suits ML algorithms. Unfortunately, real-world databases are highly influenced by the presence of noise, missing values, inconsistent data, among others. Therefore, low-quality input data can considerably affect the performance of ML methods. In this section, a general overview of data pre-processing techniques is described that improve the quality of data before fed it into ML algorithms [25]. Data preparation is usually a mandatory step in supervised learning problems. It converts prior raw and, sometimes useless, data into new data that fits the input of ML methods. If data is not prepared correctly, ML methods will not operate and will report errors during their runtimes or will generate results that do not make sense within the context wherein the data comes. Below representative approaches within the data preparation are presented.

- **Data Cleaning:** This approach includes operations related to inconsistent data corrections and reduction of redundant data. The primary purpose is the detection of discrepancies and dirty data, which means identifying fragments of the original data that do not make sense in the context under study [26].
- **Data Transformation and Data Integration:** In the data transformation process, data is converted to enable that the supervised learning process can be more efficient. Examples of possible paths to follow are feature generation, feature aggregation or data normalization, among others. For the vase of data integration, this pre-processing approach involves the merging of data that comes from multiple data sources. This process requires caution to avoid redundancies and inconsistencies in the resulting dataset [27].
- **Data Normalization:** Input data can have multiple variables with different measurement scales. Such diversity of measurement units can affect the data analysis. Therefore, all the variables should be expressed in the same measurement units and should use a standard scale or range. This process gives all variables equal or similar weight and is particularly useful in statistical learning methods [28].
- **Missing Data Imputation and Noise Identification:** Here the objective is to fill in the variables of the input data that contain missing values following a particular strategy. In most cases, adding an estimation of the missing data is quite better than leaving blank. Complementary to this approach includes smoothing processes whose purpose is to detect random errors or variances in the input data [29].

2.3.2 Learning Algorithms

Once data pre-processing is completed, the next step is selecting an ML algorithm to extract knowledge previously unseen in input data. ML algorithms can be subdivided into multiple areas, among which the best known are supervised learning and unsupervised learning. Unsupervised learning looks for patterns in data with no pre-existing labels. The approach of unsupervised

learning usually focuses on clustering groups of data points. Given a set of data points, clustering organizes *X* data points into specific groups such as is shown in Figure 2-8. Data points that are in the same group should have similar properties, while data points in different groups should have highly different features. It is important to note that these potential groups are not previously defined in the input data and is the purpose of unsupervised learning algorithms to discover them. Representative applications of unsupervised learning are marketing segmentation and anomaly detection [30].

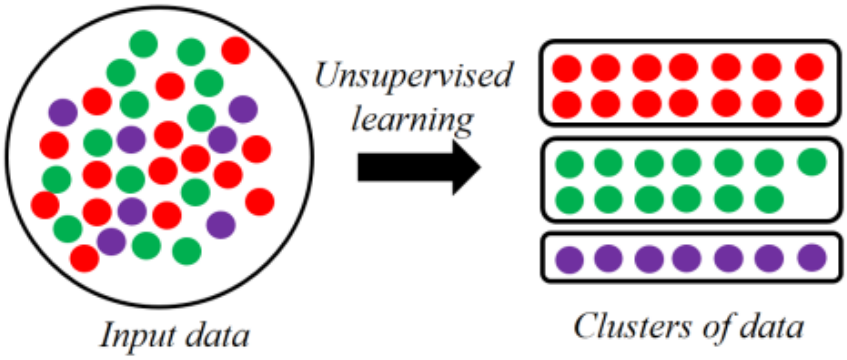


Figure 2-8. Unsupervised learning: clustering.

Supervised learning typically uses labelled data, that is, during the training process of a model, the target values are clearly defined in *Y*. It basically consists of algorithms that learn a function ($f : X \rightarrow Y$) by training with a finite number of input-output pairs, being *X* the input domain and *Y* the output codomain. Supervised learning problems can be processed by learning from a training dataset composed of instances that take the form (x, y). In this format, $x \in X$ is a vector of values in the space of input variables (features) and $y \in Y$ is a value in the target variable such as shown in Figure 2-9. Once trained, the obtained model can be used to predict the target variable on unseen instances [30].

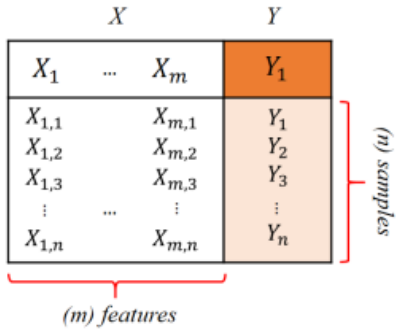
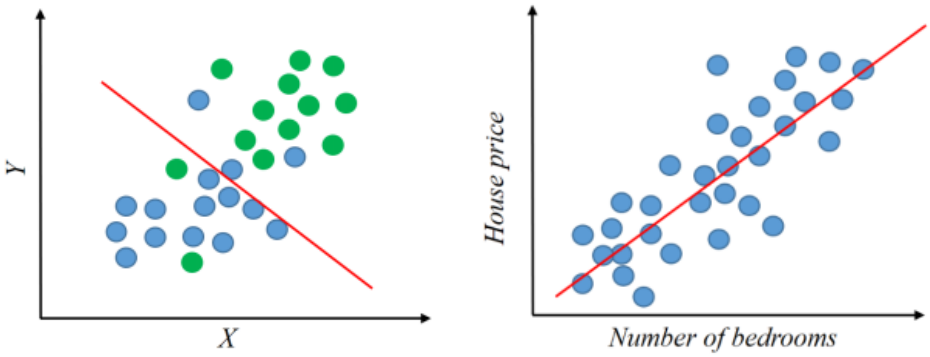


Figure 2-9. Format of a machine learning dataset.

Supervised learning problems can be usually divided into two categories: classification and regression. In both cases, the basis is an input dataset *X*, and their difference is the type of target variable, *Y*, to be predicted. On the classification case, *Y* is divided into discrete categories, while in regression, the purpose is predicting continuous values. Standard classification problems can be either binary or multi-class problems. In the former case, an instance can only be associated with one of two values: positive or negative that is equivalent to 0 or 1, such as seen in Figure 2-10 (a). Examples of this binary classification are email messages that can be categorized into spam or non-spam. Regarding multi-class problems, they involve cases wherein there are more

than two classes under consideration. That is, any given instance will belong to one of the multiple possible categories. For example, a flower image can be categorized within a wide range of plant species. Diversely, a regression problem consists of finding a function which can predict, for a given example, a real value among a continuous range. The latter is usually an interval in the set of real numbers *R*. For example, the price of a house may be calculated using multiple characteristics such as the number of bedrooms as observed in Figure 2-10 (b) [31].



(a) **(b)**

Figure 2-10. Supervised learning **(a)** binary classification, **(b)** regression.

2.3.3 Model Selection and Assessment

Model selection is the task of selecting a statistical model from a set of candidate models given input data. In ML, model selection is the process of choosing one final ML model from a set of candidate models. This task implies estimating the performance of the different models to choose the best one to address the problem at hand. The best approach to model selection requires enough data that sometimes could not be the case due to the complexity of the problem under study. In a data-rich situation, the best way to proceed is to split the input dataset into three parts randomly: a training set, a validation set, and a test set is introduced in Figure 2-11. The training set is used to fit the set of available models; the validation set is then used to estimate prediction error for model selection; and finally, the test set is used to assess the generalization error of the final chosen model. Then the best model is selected based on the validation error, and the test set should be brought out only at the end of the process when the best model has been selected. A typical data split maybe 50 % training, and 25% validation and 25% test. However, the approach mentioned above could be impractical on ML supervised problems wherein there are no sufficient data. In these cases, the most common approach is using re-sampling strategies to carry out the model selection using Cross-validation [32].

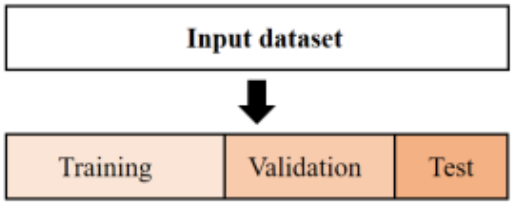


Figure 2-11. Training, validation, and test data partitions for model selection.

Cross-validation is the re-sampling strategy most used in situations where there are not enough data. In this approach presented in Figure 2-12, the training set is split into *k* smaller subsets and the next steps are followed for each of the *k* folds: an ML method is trained using *k-1* of the folds

as training data. The resulting trained model is validated on the remaining part of the data. The final performance metric is the average of the metric reported by every *k* fold. This approach can be computationally expensive, but it does not waste too much data, which is a significant advantage in some supervised learning problems [33].

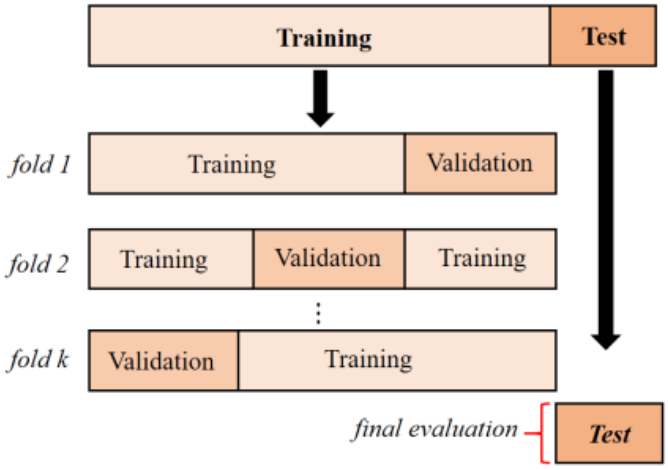


Figure 2-12. Cross validation to approach the model selection problem.

Finally, although the approaches presented above about model assessment and selection in ML bring guidelines to choose the most promising method over a set of candidates, this process is usually tedious and computationally expensive. This is generally done by ML experts who make use of their knowledge or by nonexpert users who tackle the problem using a trial and error approach that causes the success of ML comes at a high-cost [34].

2.4 Deep Learning

DL is a subfield of ML that is based on Artificial Neural Networks (ANN)s. The difference between DL and ML is shown in 2-13. Traditional ML algorithms require manual feature extraction. A deep neural network automatically extracts features by passing the input image through its layers.

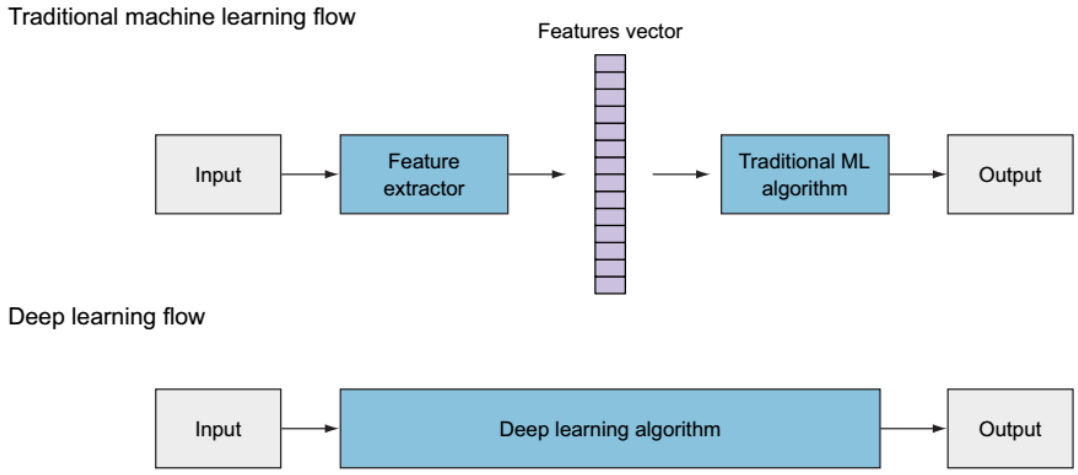


Figure 2-13. The difference between DL and ML

ANNs were designed to simulate the human brain by providing computers with the capabilities of humans to learn. A perceptron consists of a single neuron the same as a biological neuron, as shown in Figure 2-14. A biological neuron receives electrical signals from its dendrites, modulates them in various amounts, and then fires an output signal through its synapses. The output is fired only when the total strength of the input signals exceeds a certain threshold. The output is then fed to another neuron, and so forth. To simulate it, the artificial neuron performs two consecutive functions. First, it calculates the weighted sum of the inputs. Then, it applies a step function to the result to determine whether to fire the output or not. Note that, not all input features are equally important and to represent that, each input node is assigned a weight value to reflect its importance.

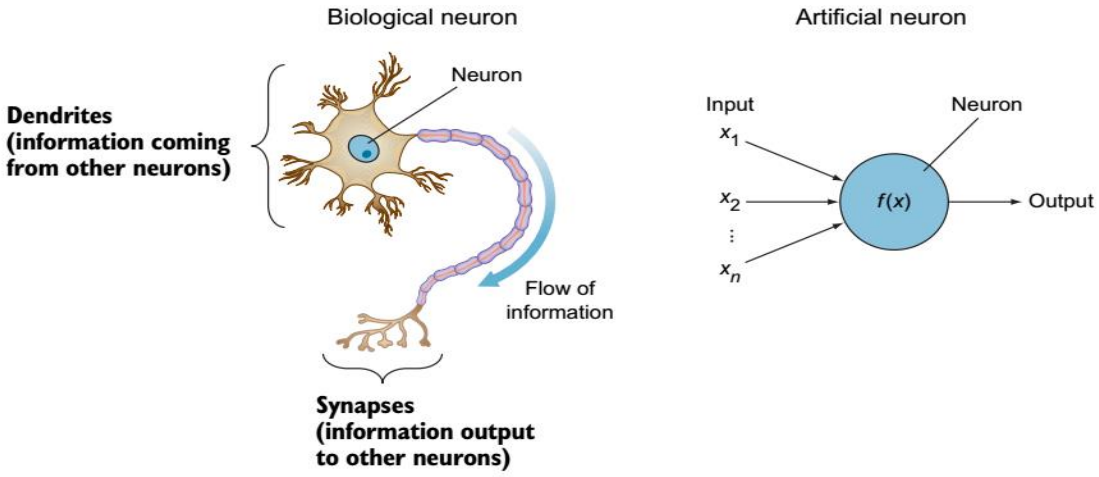


Figure 2-14. Artificial neurons were inspired by biological neurons.

As shown in Figure 2-15, a perceptron takes *1* to *n* variables as input and it then performs a weighted sum. Each input variable has an associated weight that measures how relevant each variable is. In this way, some variables will have more influence than others in the final decision. Finally, a linear activation function is applied over the resulting weighted sum to get a binary output [35].

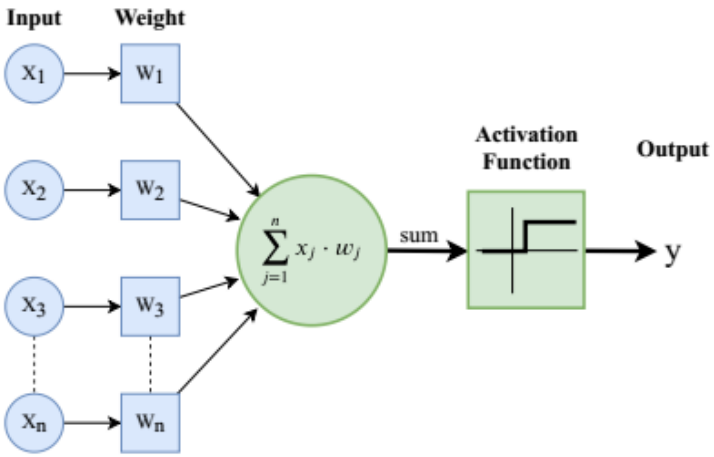


Figure 2-15. Structure of a perceptron.

Multilayer Perceptron (MLP) is a supervised feed-forward neural network that is composed of stacked layers of the perceptron. Thus, it simulates a human brain where multiple layers of neurons are connected to each other. The MLP is composed of at least three layers: input, hidden, and output as shown in Figure 2-16. In this network, the data flows through the network from the first to the last layer. Here, the neurons represent a computational node, whereas the edges represent the corresponding weights. In the MLP, the input layer refers to the layer responsible for feeding input data into the network. This layer contains as many neurons as input variables are. Hidden layers are considered those all layers located between the input and the output layer, and there can be one to *n* layers. Opposite to the original perceptron, neurons within these layers apply non-linear activation functions because an output (y) that varies non-linearly with its explanatory variables is required. Finally, the output layer is responsible for giving the result (y). For classification tasks, this layer typically contains as many neurons as target classes [36].

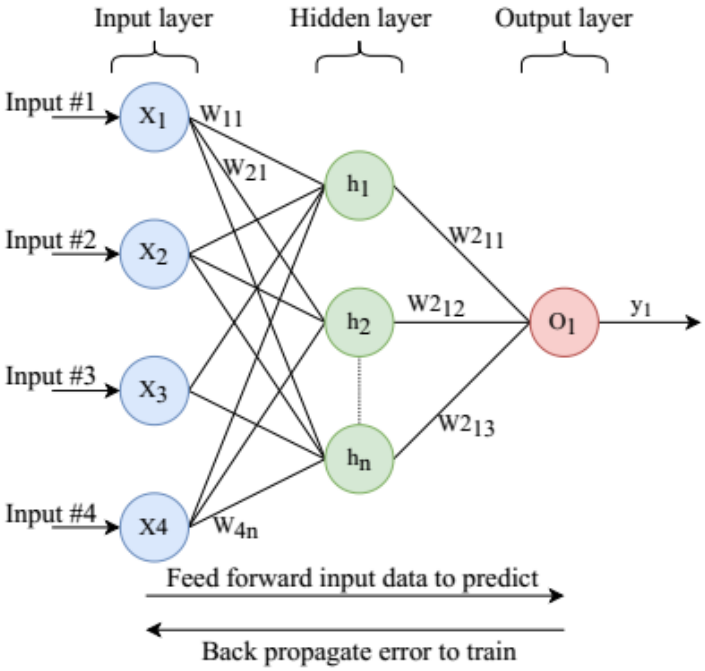


Figure 2-16. Multi-Layer Perceptron.

At the training phase, the input data is feed-forwarded into the network until obtaining the final prediction. At this stage, all the neurons of each layer process the corresponding received data, which may come from the input data or from other neurons within the previous layer. As this is a supervised method, the output values are compared against the ground truth to compute the error between them. Next, the error is propagated from the last to the first layers of the network using the backpropagation method. Hence, all the weights are adjusted consequently to approximate the target function. As mentioned, the number of hidden layers can be increased as much as required, where a more abstract representation of the input data can be obtained as this number increases, thus allowing to solve more complex problems. Since the number of hidden layers determines the depth of ANNs, DL can also be defined as MLP with a larger number of hidden layers. However, there is not a concrete number of layers at which a network is considered deep. Therefore, DL algorithms are able to build hierarchical concepts by stacking lower to higher feature representations to learn complex functions, mapping the input to the output directly from raw data, without requiring human-based features. As a downside, increasing the number of hidden layers will also increase the required computational power to train the network, since

computing the calculations of each neuron and then adjusting their weights is a computationally expensive process. So, new DL architectures such as CNNs and Recurrent Neural Networks (RNN) were designed not to connect all the neurons to each other, thus reducing the required number of operations. As a consequence, large neural networks can currently be trained with large volumes of data to solve more complex problems, outperforming more traditional algorithms in a wide range of domains [37].

A CNN is a type of DL algorithm that was originally designed for processing images and CVs such as object detection and face recognition. Its success lies in its ability to extract the most relevant features from input data, which is then used to make the final decision [38]. MLPs are composed of dense layers that are fully connected. Fully connected means that every node in one layer is connected to all nodes from the previous and the next layers. So, each neuron has weights to train, which is not big for small images. However, what happens with images with 1,000 × 1,000 dimensions. It will yield 1 million parameters for each node in the first hidden layer. So, if the first hidden layer has 1,000 neurons, this will yield 1 billion parameters even in such a small network. Imagine the computational complexity of optimizing 1 billion parameters after only the first layer. This number will increase drastically and get out of control fast when you have tens or hundreds of layers. On the other hand, CNN is a locally connected layers, as shown in Figure 2.17. Nodes are connected to only a small subset of the previous layers' nodes. Locally connected layers use far fewer parameters than densely connected layers.

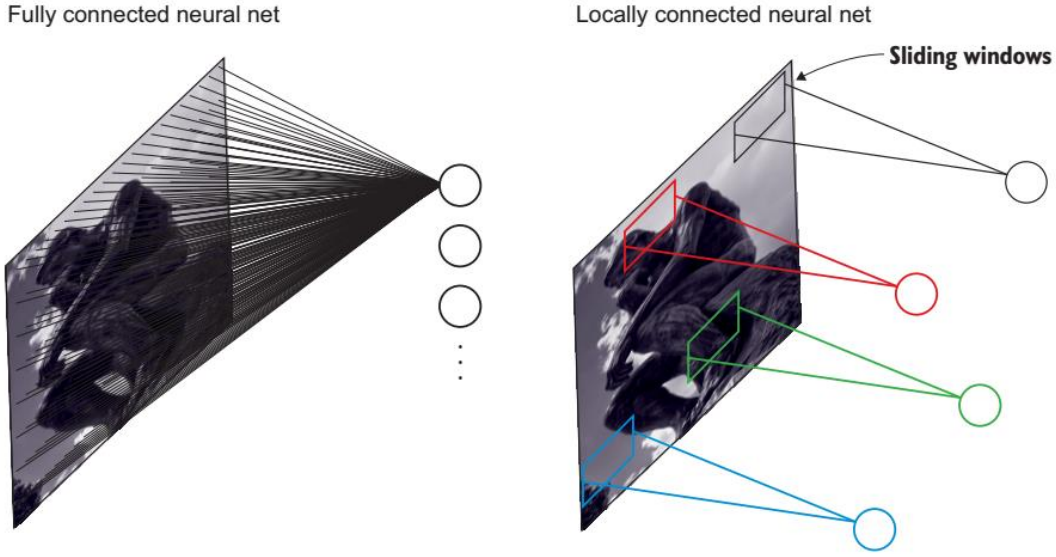


Figure 2-17 (Left) Fully connected neural network, (Right) Locally connected network.

This local connectivity is called filter or kernel and is slid over the input data to extract features from it. A filter is a set of weights that are learned during the training phase. In this context, a convolution is defined as the dot product between the filter and the segment of the input data where the filter is applied. As a result of the convolution, a feature map containing the features of a given segment is obtained. Thus, filters act as feature detectors. At the time of applying a convolution, multiple filters can be used so that different features of the same data can be detected. Furthermore, convolutional layers are composable, meaning that the output of a convolutional layer can be fed into another. Consequently, several convolutional layers can be stacked to form a deeper CNN so that the network can detect higher-level, more abstract features. Figure 2-18. illustrates typical CNN architecture for a toy image classification task. An image is an input directly to the network, and this is followed by several stages of convolution and pooling. Thereafter, representations from these operations feed one or more fully connected layers.

Finally, the last fully connected layer outputs the class label. Despite this is being the most popular base architecture found in the literature, several architecture changes have been proposed in recent years with the objective of improving image classification accuracy or reducing computation costs[39].

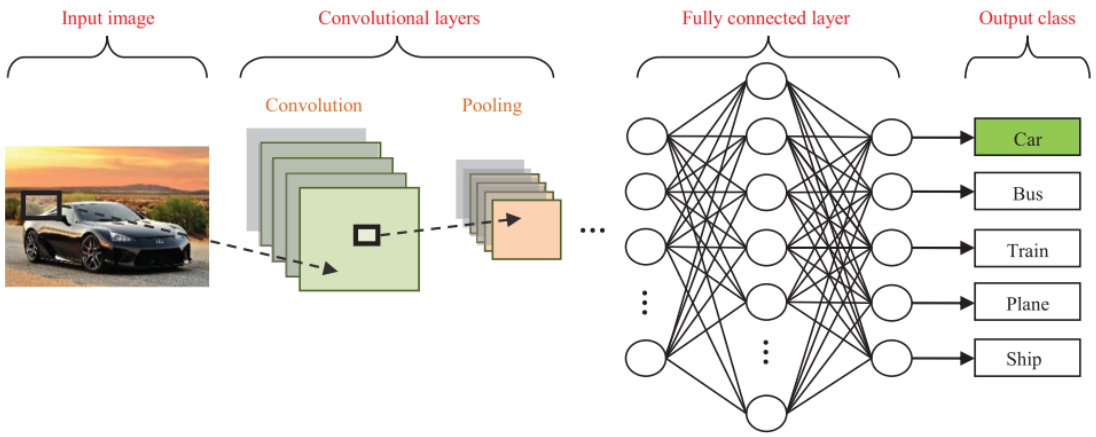


Figure 2-18. CNN image classification pipeline.

- **Convolutional Layers.** The convolutional layers serve as feature extractors, and thus they learn the feature representations of their input images. The neurons in the convolutional layers are arranged into feature maps. Each neuron in a feature map has a receptive field, which is connected to a neighborhood of neurons in the previous layer via a set of trainable weights. Inputs are convolved with the learned weights to compute a new feature map, and the convolved results are sent through a nonlinear activation function. All neurons within a feature map have weights that are constrained to be equal; however, different feature maps within the same convolutional layer have different weights so that several features can be extracted at each location. More formally, the k_{th} output feature map Y_k can be computed as $f\ (W_k * x)$ where the input image is denoted by x; the convolutional filter related to the k_{th} feature map is denoted by W_k; the multiplication sign in this context refers to the 2D convolutional operator, which is used to calculate the inner product of the filter model at each location of the input image; and f (·) represents the nonlinear activation function. Nonlinear activation functions allow for the extraction of nonlinear features. Traditionally, the sigmoid and hyperbolic tangent functions were used; recently, rectified linear units have become popular.
- **Pooling Layers.** The purpose of the pooling layers is to reduce the spatial resolution of the feature maps and thus achieve spatial invariance to input distortions and translations. Initially, it was common practice to use average pooling aggregation layers to propagate the average of all the input values, of a small neighborhood of an image to the next layer. However, in more recent models, max-pooling aggregation layers propagate the maximum value within a receptive field to the next layer. Formally, max-pooling selects the largest element within each receptive field such that. Figure 2-19 illustrates the difference between max-pooling and average pooling. Given an input image of size 4 × 4, if a 2 × 2 filter and stride of two is applied, max-pooling outputs the maximum value of each 2 × 2 regions, while average pooling outputs the average rounded integer value of each subsampled region.
- **Fully Connected Layers.** Several convolutional and pooling layers are usually stacked on top of each other to extract more abstract feature representations in moving through the network. The fully connected layers that follow these layers interpret these feature representations and perform the function of high-level reasoning.

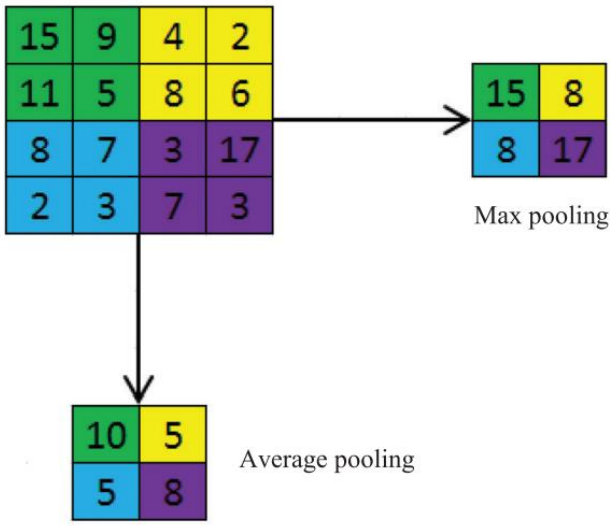


Figure 2-19. Average versus max pooling.

2.5 Execution Environments

This section details the resources used in the experimentations conducted in this thesis. This includes both hardware and software resources. In this thesis, two types of experimentations can be differentiated: the ones involving the implementation of DL architectures to detect markers and objects, and the other ones referring to the mobile application for the indoor navigator system. Hence, both types use different execution environments, which are detailed next.

Experimentations involving DL to detect markers and objects, three models have been trained:

1. The first one is used to detect markers from long distance with the following hardware characteristics (explained in Chapter 5):
   - Laptop: DELL INSPIRON N5110 computer.
   - Processor: Intel Core i7-2630 QM 2.00 GHz CPU, 6 MB cache, quad-core.
   - RAM: 8 GB.
   - Graphics: 1 GB NVIDIA GeForce GT 525 M.
   - Model implementation: Keras and Tensorflow.
   - Operating System: Ubuntu 18.04 LTS.
2. The second and third models are used to detect markers in challenging conditions and to detect objects respectively (explained in Chapters 4, 6):
   - Cloud: Google Colab.
   - CUDA Version: 10.1
   - Processor: Tesla P100-PCIE
   - Model implementation: Keras and Tensorflow.

Experimentations involving the indoor navigation system using a smartphone. Regarding the hardware and software resources used in this type of tests, the hardware resources are detailed:

- Processor: octa-core (4 x 1.7 GHz Cortex-A53 and 4 x 1.0 GHz Cortex-A53).

- GPU: Adreno 405 GPU.
- Smartphone model: HTC Desire 826.
- RAM: 2 GB

The software used:

- IDE: Android Studio.
- Anaconda.
- Spyder.

2.6 Summary

In this chapter, PVI and AT have been presented. The main problems they face in doing daily tasks have been discussed. These problems make them isolated from the world around them. It showed an introduction about CV and how it can be used to detect objects and objects. This chapter has also highlighted the importance of the ML field, describing its bases and the impact it can have in building navigation systems. Moreover, the term DL has been presented in addition to explaining how it can be used to improve ANN models. Finally, the hardware and software settings used to conduct the experimentations throughout this thesis have been described. In the following chapter, the literature review involving the main contributions of this work will be presented.

3 Literature Review

This chapter presents a brief literature review regarding the main contributions of this thesis. First, the main taxonomy is shown in Section 3.1. Secondly, the state of the art involving the outdoor navigation is presented in Section 3.2. Thirdly, research related to obstacle detection is shown in Section 3.3. Finally, the research and technological trends for indoor navigation are exposed in Section 3.4.

3.1 Taxonomy

To talk about PVI, people should be aware of the obstacles they face while navigating alone when the support is limited. In order to identify most of the available MAT solutions for PVI, the following databases were used: Springer, Science Direct, Web of Science, Institute of Electrical and Electronics Engineers (IEEE) Xplore, Google Scholar, Association for Computing Machinery (ACM) Digital Library and Microsoft Academic. We used the following keywords to search for peer reviewed journal articles: ("Assistive technology" OR "Assistive technology devices" OR "Mobile assistive technology devices" OR "navigation solution") AND, ("visual impairment" OR "blind *"), ("avoiding obstacles" OR "write * notes" OR "text to speech") AND ("visual impairment" OR "blind *"). We set the search period to articles published between January 2010 and December 2020. Figure 3-1 shows the flowchart of choosing methodology based on PRISMA flowchart

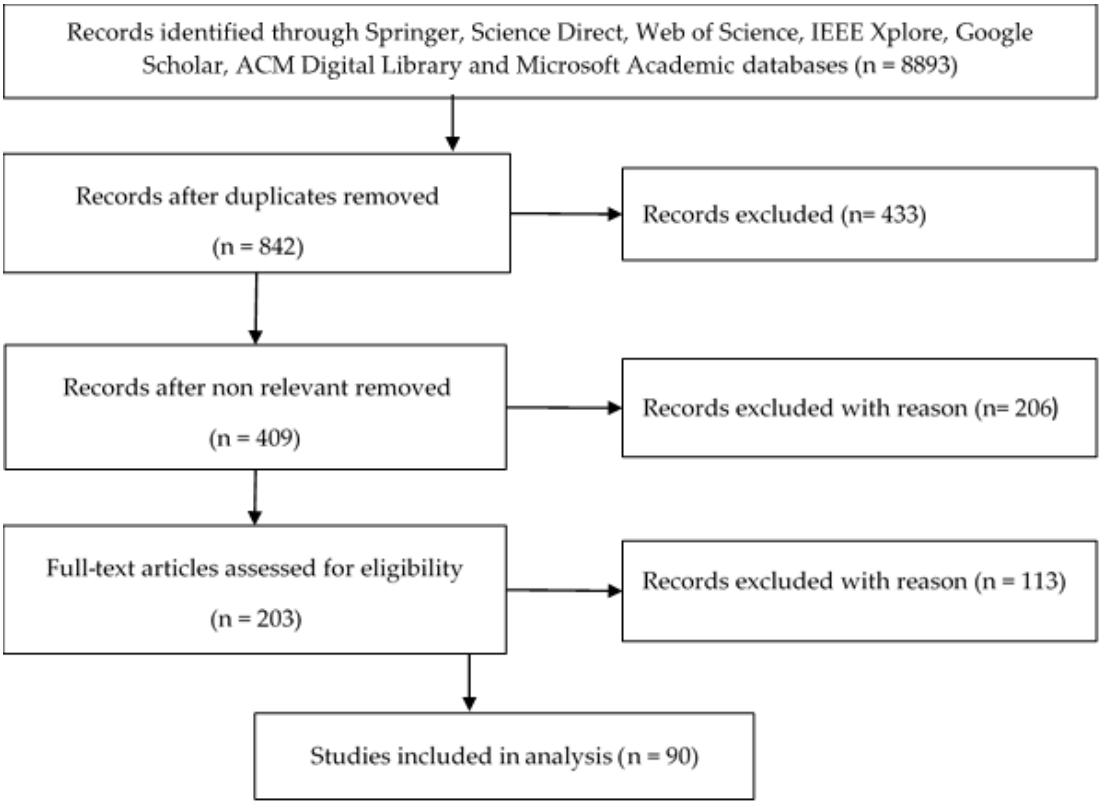


Figure 3-1. The flowchart of choosing methodology based on PRISMA flowchart.

The search query returned 8893 records. Duplicates were removed, reducing the search results to 842 articles. Then, we eliminated 433 results by restricting to articles in English, articles that

describe research intervention for PVI based on their titles, and articles that are free and downloadable. Next, all keywords were screened, which eliminated 206 articles, because they were not technical papers, or they were literature reviews or surveys. Then, abstracts of the resulting 203 papers were screened for relevance to our research goals. One hundred and thirteen of the articles were deemed inappropriate because they did not study visual impaired or blind populations, or they were not related to MAT. The remaining 90 articles met all inclusion criteria and were evaluated in this study. Based on this search, the available MAT solutions to help PVI are divided into three parts, as shown in Figure 3-2.

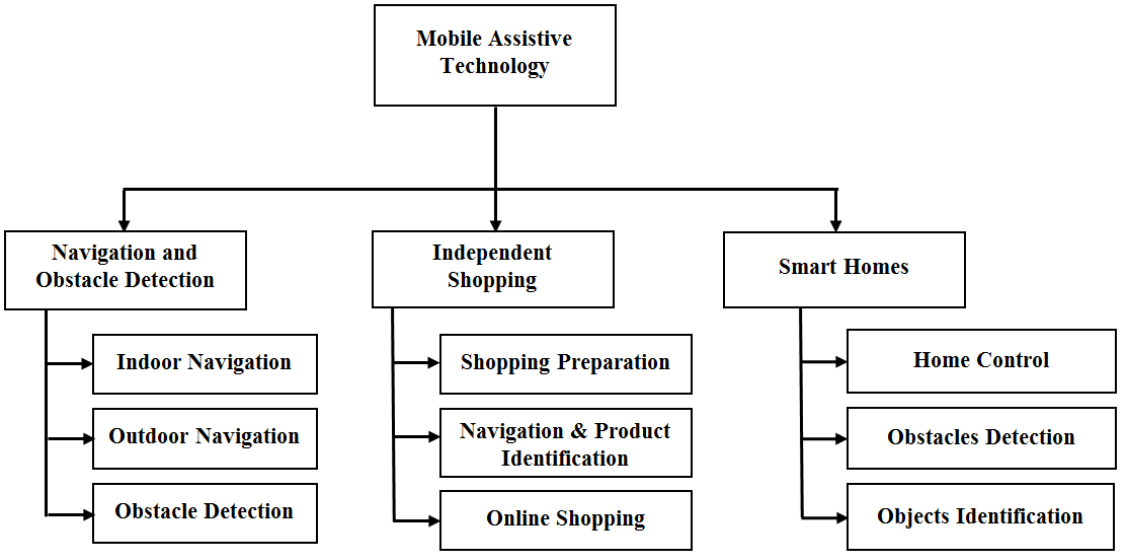


Figure 3-2. Mobile Assistive Technology

Independent shopping: PVI face several difficulties during shopping, especially when there is a limited support from a shop assistant. Store employees are always busy so, PVI should stay a long time waiting for help. This waiting time makes shopping a time-consuming and disturbing task. Thus, researchers found that PVI faces different challenges like preparing a shopping list, navigating within the store aisles, and identifying items on the shelves. Figure 3-3. shows a scenario of the shopping solutions for PVI [40][41][42].

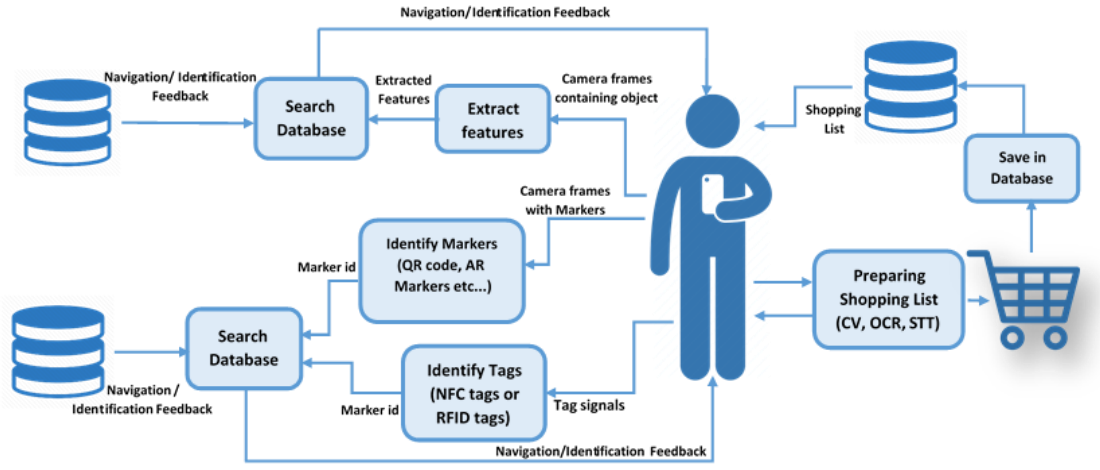


Figure 3-3. The scenario of the shopping solutions for PVI.

Smart Homes: Recently, MAT is for controlling and using home equipment which is called Smart home. The smart home concept is a recent research topic in the AT field which connects all home systems, including home devices and amusement pieces of equipment. Thanks to the interconnectivity between the environment's devices, which helps PVI to control all the devices in the home and can get feedback messages. With a smart home, PVI can navigate, avoid obstacles, identify objects, and recognize people. Figure 3-4. shows smart homes solutions for PVI [43][44][45][46].

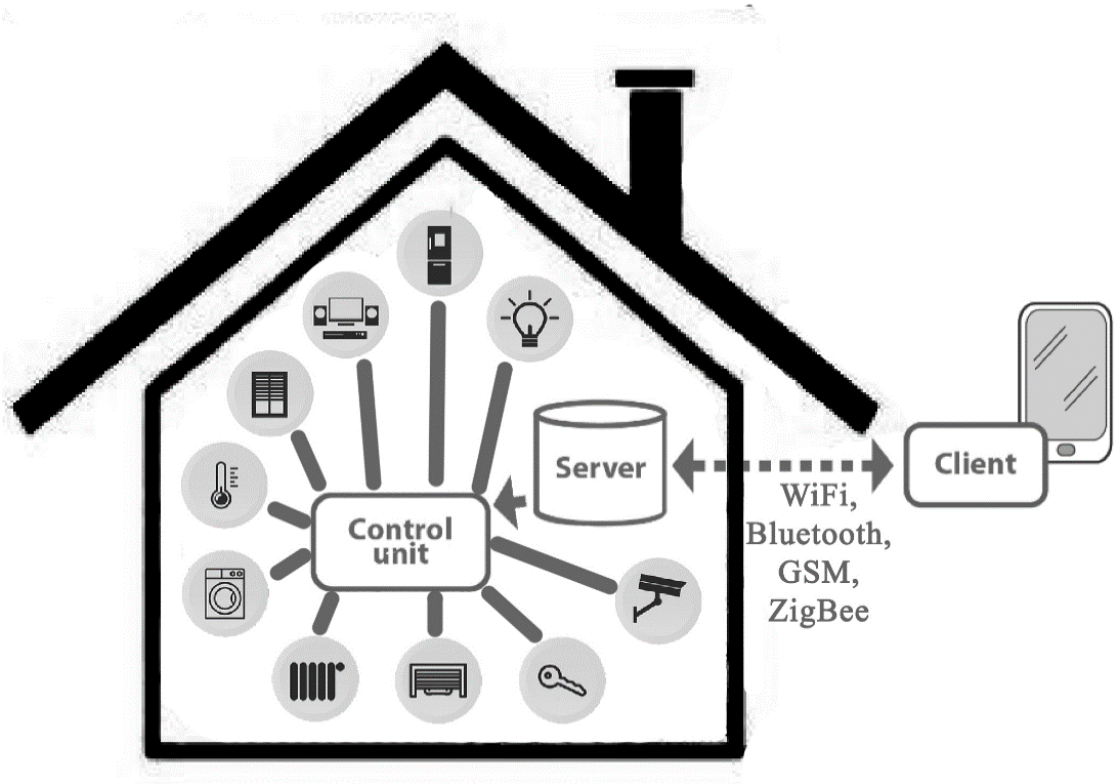


Figure 3-4. Smart homes solutions for PVI.

Navigation and Obstacle detection: Going from one place to another is a challenge for PVI. Navigation is divided into four tasks: orienting oneself in the environment, choosing the route, keeping on track, detect any obstacle during navigation and reaching the destination successfully. Many trials to combine these tasks and allow PVI to navigate safely. So, research related to navigation have been divided into indoor navigation, outdoor navigation, and obstacle detection [47][48][49]. Figure 3-5. shows a navigation scenario for PVI.

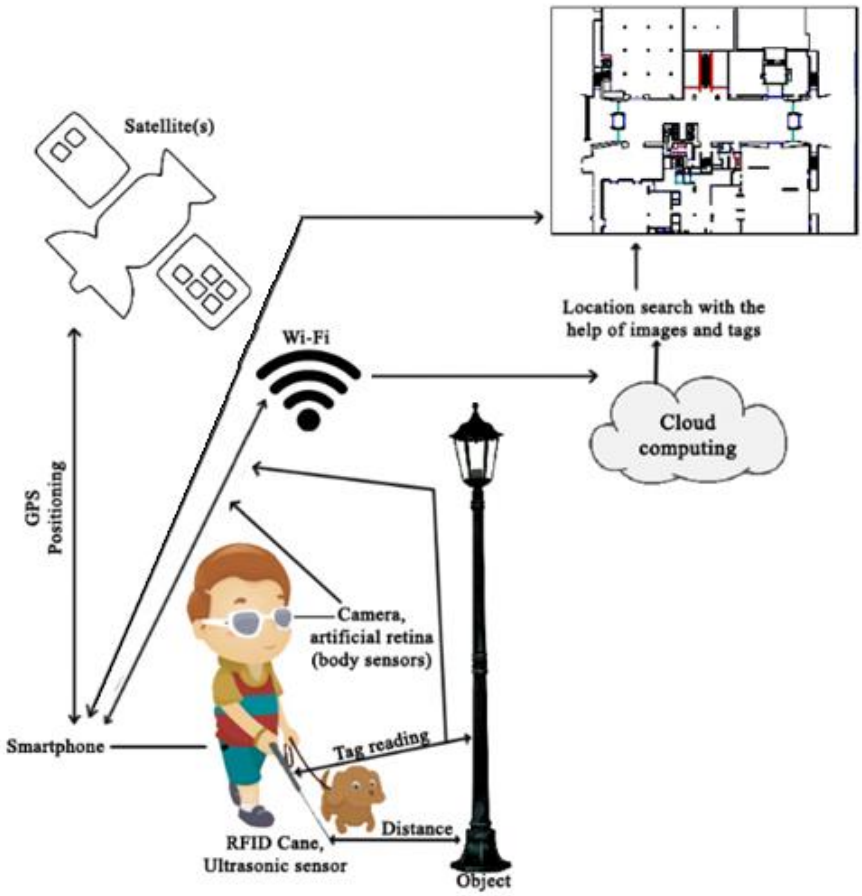


Figure 3-5. A navigation scenario for PVI.

3.2 Outdoor Navigation

PVI face difficulties while navigating outdoors. Traditional solutions tried to solve this problem. The well-known white cane tried to increase independence and social involvement through the mobility of blind people. Guided dogs can also help to avoid dangerous situations but cannot support mobility issues. While PVI depends on memorizing routes and the support from their fellow humans, going to unknown locations is still a big problem. During the last years, researchers searched for solutions for navigating outdoor.

Kammoun et. al provided a system that helped PVI improve their daily autonomy and provide their ability to represent their environment mentally. The system used a vision module to extract relevant visual information from the scene around them. The system used landmarks like signs to estimate PVI location. When detecting any of these landmarks, the system refines the current Global Positioning System (GPS) position of the user in a more accurate way. However, it is better to add tactile vibration patterns to guide the user. The PVI need to carry a bag contains a notebook for a long time. Finally, the creation of models of the current system was performed manually from recorded videos of the evaluation test site [50].

Walking Straight proposed a system to help PVI reduce their veering and walking in the right path. It used a smartphone's sensors to assist walking on a straight line by receiving updated values from the gyroscope. Furthermore, these values were used to compute and compare the

average heading for each walking period with the desired one. Then, send an alarm signal to the user if there is any deviation. However, they need to automate the process of selection of the correct heading. Sensor reliability should also be improved using sensor fusion techniques. Moreover, the smartphone orientation has to set to a specific direction which prevents the user from using the phone for some other tasks while walking [51].

Wayfinding combined several sensors such as GPS, Assisted GPS, compass and accelerometer to guide PVI among different points of interest and avoid obstacles by providing suitable auditory and tactile feedback according to the PVI's orientation. Moreover, it can avoid obstacles without a network connection by using information stored in the PVI smartphone, which is available anytime and anywhere. When the network connection is available, the device can update the information. However, using both auditory and tactile feedback is better and allows operation in a noisy environment. Using GPS and AGPS give better accuracy [52].

Cheng et. al used GPS and an Inertial Navigation System (INS) to provide continuous outdoor navigation while using WIFI positioning of Access Points (AP) instead of GPS for moving indoors which adds more accuracy. This system stores the coordinates of the APs, MAC addresses and some RSSI measurements in a DB to be used for estimating PVI location. It integrates INS with GPS to overcome the short outage of it in outdoor environments while using WIFI to improve navigation indoor. The system uses step length detection integrated with the INS for both environments to correct its output. However, feedback needs some improvements using sound commands or some haptic feedback to be suitable for PVI. PVI should carry INS/GPS devices and a notebook which is heavy, so using tablets or smartphones will be a better choice [53].

IMAGO focuses on using images to navigate outdoors using route model preparation, localization, and navigation. A person walked through the route at least once to collect images and some additional data, like the weather. These data are embedded in the images and stored on the smartphone. After that, these images are transmitted to the server to create a 3D model representing the route. The application collected images and sensory data and sent them to the server to calculate the current position of PVI. Then, it found the direction and provided it to them via a Bluetooth to the cane device as a steering command to the embedded tactile arrow. The system solved the accuracy problem compared to GPS and provided the output using tactile arrows on the white cane which is better especially in a noisy environment. However, this system is not suitable for indoor navigation as additional landmarks are needed to improve the model creation which requires a massive investment. A lot of time used for communication between the mobile device and the server. If the computation is done on smartphone it needs a lot of resources which drains the mobile battery very fast [54].

3.3 Obstacle Detection

Identify and avoid obstacles is one of the most important problems that PVI face. This problem can lead to dangerous situations. Researchers tried to develop solutions to solve this problem using sensors like ultrasonic, radar, stereo vision and laser sensors, or computer vision solutions.

Dias et. al provided a system to keep PVI on a straight path and avoid obstacles using a white cane with an ultrasonic sensor and Inertial Measurement Unit (IMU). To detect the presence of the obstacle, an ultrasonic sensor was used. An IMU sensor was used to detect the cane orientation. They used an algorithm to detect the size of obstacles and the distance between the

user and them. The system gives PVI the required direction as audio feedback using earphones. The system uses lightweight components and works independently from the smartphone orientation. Moreover, it provides a simple model to get the width of the obstacles. However, it needs more accuracy as the reliability of the model decreased when the distance to the obstacle increased. The system was tested on a single obstacle centred on the path of movement. It also uses earphones which is hard to work with in a noisy environment [55].

Smart walker helped PVI navigating to their destinations and providing them with information to avoid obstacles. It used laser scanners for perceiving obstacles during navigation and used a notebook for data-processing. It attached a vibration motor to each handle for tactile feedback to avoid obstacles. The system is connected to a vibrotactile belt around the waist to give haptic feedback. Furthermore, it incorporated a Bluetooth-enabled microcontroller to receive data from PVI and controls the vibration motors. Smart walker informs PVI if any obstacle is detected about head high. Smart walker gives haptic feedback instead of auditory signals and can easily be carried for a long time. However, the system needs more training to make PVI familiar with the vibration signals from the belt. The system has a problem to estimate the exact position and dimension of obstacles [56].

Mohamed proposed a system for obstacles avoidance while navigating indoors and a recognition system using CV technologies. The system includes a speech recognition module which receives instructions and gives voice feedback to the user. It used a laser sensor to provide information about the distance from obstacles and a set of markers to determine user location. IMU sensor to find user orientation within the indoor space and a path planning module to calculate a safe path for the user to walk through. A dataset including a bunch of images used by the recognition module to identify objects. The system performs a dead reckoning based on the previous positioning and current IMU measurements if no markers are detected. It provides a safe path for PVI from their current position to the desired destination. The recognition system allows PVI to get full awareness of objects present within their current location. However, it is better to give haptic feedback than voice commands, especially in noisy environments. The size and the weight of the processing unit is unsuitable to wear for a long time [57].

Aladren et. al developed a navigation system for PVI using visual information to improve human abilities in interaction with the environment. The system used a range camera to get range information. This information was to detect and classify the main structure of the scene. It joined the colour information with the range information to extend the floor segmentation to the entire scene. This system contains a user interface that sends navigation commands via voice commands and sound map. This sound map is created using stereo beeps with frequency depends on the distance to obstacles, while the voice commands provide high-level guidance along the obstacle-free paths. The proposed system detects near and far obstacles, so the user would have enough time to avoid them. However, it is too hard to carry a laptop for a long time, and the frame rate could be improved to work at higher frame rates. Finally, it is better to give haptic feedback than voice especially in noisy environments [58].

Caldini et. al proposed a novel system using smartphone's sensors like inertial gyroscope and camera. These sensors were used to compute the depth of the scene and detect near obstacles. The system processed images with a modified Structure from Motion algorithm, that took input also information from the built-in gyroscope. Then, it estimated the ground plane and labelled obstacles and all the structures above it. This system depends only on smartphones, so there is no need to buy new hardware which may be expensive. PVI used their hands to do other tasks as the smartphone was in their chest. The system detects obstacles and their distance from PVI.

However, using the tactile interface to provide feedback is better. It is better to use cloud computing to complete the processing [59].

Mun-Cheon et. al proposed a novel method to detect obstacles based on a new structure called Deformable Grid (DG). They used DG with a regular grid-shape which was deformed using the movement of the object in the scene. It sent the captured video sequence to a laptop, which deformed the DG using the object's motion from the scene. The unstable vertices depicted as the dots are detected and used to obtain the connected components. Finally, they provided the collision warnings based on the resultant components by using circles and the size of the circle represents the risk level. The object is considered as a dangerous one when the time to contact is smaller than 2 seconds. The proposed method used the motion information to make the obstacles detection results more accurate and robust to the motion tracking error. However, using circles to provide users with warning information is not suitable for PVI so, it is better to give haptic feedback or some voice commands [60].

Rabia et. al used a Tango tablet to assist PVI in detecting obstacles during navigating indoors. It processed the depth and motion tracking data to create a 3D reconstruction of the real-world environment as a mesh while walking. If the box collides with any solid surface in the 3D reconstruction, an audio warning is relayed to the user via bone conduction headphones. This system exploited Tango tablet for performing computationally expensive operations in real-time without any connection to an external server. It also provided a real-time detection of obstacles and gave feedback to PVI in time, which added more accuracy and safety. However, the user should hold the tablet in his hands roughly at waist level with the screen facing towards him for a long time. Moreover, the feedback mechanism should be enhanced to provide more details about the distance to obstacles [61].

Hoang et. al proposed a system to determine the presence of obstacles in the scene in front of PVI and sends this information to the users. It took the colour image, depth image, and accelerometer information from a mobile Kinect and transferring them to the laptop to detect obstacles. It used voice commands to warn PVI about obstacles. This system provided good results in detecting and avoiding obstacles. However, it is quite heavy and should be mounted on the body. So, using a small and wearable device like Google Glass is better [62].

Rizzo et. al provided a system to detect and avoid obstacles using sensor fusion. The system used a regular camera to capture RGB images with colour value and generate a depth map of the scene. It used an Infrared (IR) sensor to complement the camera operation of accurate detection of obstacles at night, or during rain and snow. The IR sensor emitted pulses of infrared light to measures the depth. It used a photodetector to capture the reflected pulses and process them. Based on this processing, it found the presence and position of objects in its field of view. After collecting all needed data, it used a convolutional neural network for object detection in the RGB domain. However, it is better to give haptic feedback and integrate it into a wearable system so that hands and ears can remain free [63].

3.4 Indoor Navigation

Personal navigation systems are designed to provide users with spatial information and directions when travelling to new places. While outdoor navigation is already solved by using GPS, indoor navigation for PVI is still problematic. Researchers developed various technologies for indoor navigation that is useful for PVI and anyone who needs directional information. There is

increasing interest in technology that help people to locate a shop in a mall or a room in a building. Several research groups have started building mobile assistive applications to help navigating indoors.

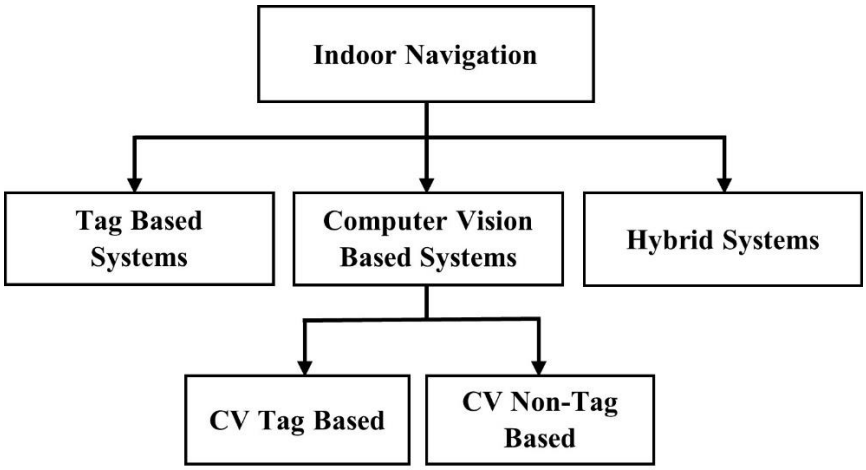


Figure 3-6. MAT solutions for the parts of the shopping process for PVI.

As shown in Figure 3-6., indoor navigation solutions are classified into three categories, based on the technology that was used: tag based, CV based, and hybrid systems.

(1) **Tag Based Systems:** such as RFID and NFC, which use wireless components to transfer data from a tag attached to an object for the purposes of automatic identification and tracking.

(2) **Computer Vision Based Systems:** some of these systems require unique visual tags such as Quick Response (QR) codes, Barcodes, or AR markers to be placed on products. These tags are used for detecting and giving PVI all available details about the products. Other systems do not require tags to be placed on products. Instead, they utilize information about the objects' physical features to identify them.

(3) **Hybrid Systems:** these take the strengths of two or more systems and combine them into a new system to deliver better accuracy and performance.

3.4.1 Tag Based Systems

Tag systems use wireless communication technology to transfer data from the tag attached to an object to a tag reader for automatic identification and tracking. Developers are using several types of tags, but I will concentrate on RFID and NFC tags. RFID uniquely identifies items by using radio waves. Each RFID system has three components: a tag, a reader, and an antenna. The reader sends a signal to the tag via the antenna, and the tag responds to this signal by sending its unique identification information. There are two types of RFID tags: active and passive. Active tags broadcast a signal up to a range of 100 m. Passive tags use electromagnetic energy from the reader, but they do not have their own power source. They can broadcast a signal up to a range of 25 m. Passive tags are typically smaller and cheaper than active tags. NFC technology is a version of RFID that operates at a maximum range of about 10 cm. The NFC system consists of three main components: a tag, a reader, and a database. The tag is a place to store information, while the reader is an NFC mobile device that reads the content of the tag. The database stores additional information to the NFC tags. NFC technology is used for active or passive communication. In active mode, the NFC tag and NFC tag reader produce a radio field during communication. In passive mode, only the NFC tag reader generates a radio field and begins the

communication session, so NFC tags do not process any information sent from other sources and cannot connect to other passive components. The main differences are the communication distance and the use of mobile devices by NFC systems or RFID readers by the RFID systems [64][65].

RFID Tags have some benefits as follow: its signals can penetrate walls and obstacles which allows to cover a wide area and the use of existing infrastructures resulting in a relative cost reduction. Also, tags are unidirectional; thus, the PVI do need to be in a certain direction to receive messages from the tagged item. Moreover, the tag does not need to be within line of sight of the RFID reader which allows being embedded in the object. However, there are some drawbacks: setup the environment, hundreds of RFID tags need to be tagged which are costly. Information overload is another major problem as it will be overwhelming to receive information about all the items in the store at the same time and attempt to use it to identify various objects. Another significant limitation is that these systems are used only in a restricted environment in which objects have been tagged and these tags need regular maintenance. Sometimes, RFID tags are attached to items such as liquid in metal cans which reflect the radio waves during communication. Furthermore, using RFID technology with glass cause reflection of radio waves which affects the system outcome. Another issue is that RFID does not meet users' demands for trust and privacy since the readers are accessible to anyone [66]. NFC tags have the benefits as follow: Users simply touch the NFC tag with a device enabled with NFC to begin the required service. An NFC device is able to read information stored in the tags and to write data into the memory of the tag, enabling users to get product information by touching the tag. Moreover, by using NFC tags, I can build low-cost indoor navigation systems using cheap passive tags. NFC technology also minimizes response time. Importantly, NFC provides accurate position and orientation information. NFC provides exclusive control over user location data which guarantee location privacy of the user. Moreover, NFC tags work well in dirty environments and do not demand a direct line of sight between the reader and tags. Finally, PVI do not have to carry large or bulky technological devices. However, there are some drawbacks: it is not as effective and efficient as Bluetooth or Wi-Fi Direct when it comes to data transfer rates. NFC can only send and receive very small packets of data. So, real time positioning cannot be provided in NFC Internal system. Moreover, users should be inside the reading area in order to identify it and must have an NFC-equipped mobile [67].

Fernandes et al. provided a system to allow PVI navigating indoors using RFID technology. It used a smartphone application to interact with the hardware components. At first, a GIS module to provide all the geographical data according to the current location; Location module to receive PVI current location by using a white can with RFID reader which reads the RFID tags and sends location information through Bluetooth to mobile device; Navigation module is used to calculate the optimal route to a specific destination and guides PVI through it; CV module which extracts information about the surrounding environment to enhance the navigation of PVI. It used a haptic actuator which is in the electronic white cane and text-to-speech technology to give navigation feedback. The software was able to run offline on a mobile device using a local representation of locations data which were stored in a remote server. However, technologies such as Wi-Fi or RFID tags must be in place and needs a lot of maintenance [68].

Shopping by blind is a system that used RFID to help PVI search for any product on the shelf while shopping. It used a database on a server where storekeepers store maps about the place and store details about products. PVI can easily scan any product and get details about that product like name, price, and special offers. Moreover, the system receives navigation message from PVI

to help reaching the place of the searched item. It allows to make a shopping list and to edit it, so at the end of shopping, PVI receive the total bill through voice message. However, in noisy environments, the use of the audio channel could require wearing headphones. Finally, technologies such as servers, Wi-Fi, or RFID tags must be in place [69].

Another solution was developed to help PVI navigate inside a grocery shop using voice messages. It combines an RFID reader on the tip of a white cane with mobile technology to allow navigation inside the shop. It allows user to identify products by using codes placed on product shelves. Moreover, it provides a web-based management component to easily configure the system, generating and binding barcode tags for product shelves and RFID tag markers attached to the supermarket floor. Finally, it gives navigation voice feedback to visually impaired people using their smartphone. However, servers, Wi-Fi, or RFID tags must be in place [70].

VirtualEyez is a system which used android phones enabled with NFC technology with an SQLite database and NFC tags. PVI can record their selected items using a voice recognition service then, the application checks the selected item availability and provides an audio message to inform them. After that, they will receive a map to guide them through the store, accompanied by an audio message that provides direction commands. Thus, the navigation system suggests the shortest route for PVI to reach their products. The system also offers a product recognition service that uses an NFC tag posted on each top shelf and an NFC reader on the smartphone to read product information and give audio feedback to PVI about products details. However, NFC is not as effective and efficient as RFID, Bluetooth or Wi-Fi Direct. PVI should be inside the reading area in order to identify NFC tags. PVI must have an NFC-equipped mobile [71].

Ozdenizci et. al proposed NFC-based indoor navigation system which enables users to navigate through a building by touching NFC tags which are spread around and orient PVI to the destination. The system orients the user by receiving descriptions of the destination locations and then it uses mobile device to collect the instant position of the user from the NFC tags which are spread all over the navigation area. Then the user gets instructions from NFC Internal application to reorient the route to the destination or create a new optimal route. However, NFC is not as effective and efficient as RFID, Bluetooth or Wi-Fi Direct. PVI should be inside the reading area in order to identify NFC tags. PVI must have an NFC-equipped mobile [72].

3.4.2 Computer Vision Based Systems

CV based systems accept visual inputs from the camera and use CV techniques to extract valuable information and recognize objects in the surrounding environment. Finally, they provide information to the PVI through tactile or auditory channels. Researchers classified CV based systems into tag based and non-tag based. In tag-based systems, unique visual tags such as QR code, Barcode, and AR markers are used to identify places and recognize object. Recognition is accomplished by capturing an image of the tag and analysing this image to determine the identity of the object based on its tag information. Finally, tactile or voice commands are used for warnings and providing direction commands to the PVI. With non-tag-based systems, developers do not attach tags to objects. They use CV techniques to analyse the images and identify objects. Non-tag systems require extensive computational power to analyse images and give accurate results.

3.4.2.1 Non-Tag-based techniques

Third eye helps PVI to select the desired item from a grocery store shelf and identifies obstacles using smart glasses and glove. Third Eye used a smart glass connected to a back-end server which support real-time video analytics to locate and identify objects properly using CV. PVI also wear a glove with a camera which guides hand movements to point and grasp things. The system provided audio commands or tactile vibration patterns that guide PVI steps and hands toward the desired item. This system has some limitations: first, navigation between the aisles is not yet completely automated and uses the navigation skills in PVIs. Feedback latency needs to be reduced so the system can be more effective. Finally, it is streaming raw video over the wireless channel to the server, which drained the battery and clogged the wireless bandwidth [73].

Kumar et. al proposed a solution to help PVI improve their safety and quality life by recognizing objects and identifying their colours. The system used two modules. Object recognition module which used a CNN, recursive neural network, and a classifier to recognize objects like doors or chairs and generate audio feedback to the user. A colour recognition module used to recognize the colour of objects in front of the camera like clothes and fruits colours [74].

Jafri et. al developed an application using the Tango tablet to assist PVI in detecting obstacles during navigation indoors. It processes the depth and motion tracking data obtained via the various sensors of the tablet to create and update a 3D reconstruction of the real-world environment in the form of a mesh and a box around the PVI. If the box collides with any solid surface, an audio warning message is given to the user via headphones. This system exploited Tango tablet for performing computationally expensive operations in real-time without the need to connect to an external server or rapidly draining the battery [61].

Hoang et. al utilized colour images, depth images, and accelerometer information from a mobile Kinect and transferred them to laptop for processing and detecting obstacles. Concerning the obstacle warning module, a tactile–visual substitution system uses voice commands to warn the PVI to avoid obstacles [75].

A navigation system was developed to improve PVI abilities in interacting with the environment and in detecting far obstacles using colour information and range camera hangs from the neck. It captured RGB images and got a wide range of information to detect and classify the main structural elements of the scene. Due to the limitations of the range sensor, the colour information is used in addition to extend the floor segmentation to the entire scene. Also, it sends voice commands to provide guidance along the obstacle-free paths and stereo beeps with a frequency depending on the distance to the obstacle [57].

3.4.2.2 QR Codes

Blind shopping offers a better shopping experience for PVI with features including product search and navigation inside the store using voice messages. The system combined an RFID reader on the tip of a white cane with mobile technology to identify RFID tags and navigate inside the shop. The system provided a web-based management part for configuration, generating QR codes for product shelves and RFID tag markers attached to the supermarket floor. It gives navigation feedback to PVI using voice commands via their smartphone. However, a Wi-Fi connection is required to retrieve the data from an online database. RFID tags and QR codes cannot be detected from a long distance [47].

Ebsar provides indoor navigation for PVI by preparing the building and then guiding them using navigation commands. At first, this system constructed a graph where each node represents a place for which a QR code is generated for each. Each edge is labelled with the number of steps and the direction between nodes connected to it. To start navigating, it seeks the nearest node to the PVI's location then, it searches for the shortest path from that node to the destination node. During navigation, it provides Arabic voice feedback to the PVI using Google Glass. However, this system requires an internet connection to download the building graphs from the server and adding haptic feedback to enable operating in a noisy environment [76].

AssisT-In used QR codes to help navigating inside new and complex environments. One of the QR codes is used as a starting point by scanning it then the system calculates the shortest path to reach the desired destination. A navigation guide is given from the start node to the subsequent nodes until reaching the destination node using text messages in the voice of a virtual pet such as a cartoon dog. However, it is difficult to capture good quality photos with their smartphone camera as most of the photos may be blurry. If more than one QR code is detected at the same time, it is better to select one based on the distance, and not select it randomly [77].

Zhang et al. proposed a navigation approach using a mobile robot in an indoor environment. QR codes are placed in a distribution such as a grid pattern at the ceiling and the system constructs a map for them. Then, an industrial camera is added to the robot to identify rapidly these QR codes. With this configuration, the camera can detect at least one QR code in its field of view and can estimate the position of the robot. The proposed recognition algorithm can localize the robot accurately and it is suitable for real-time tasks. However, the robot failed to recognize QR codes in a completely dark environment [78].

A system was developed to help PVI navigate in unknown indoor places using QR codes. It starts by determining the type of the current position, then by fetching the environmental information from colour QR codes using a simple CV algorithm. During motion, the change in location is computed continuously using two inertial sensors and routes are recorded to guide them during the return route. During navigation, feedbacks are given by using beeping or Text-to-Speech to provide productive feedback which leads to better performance and reduces navigation errors. Coloured QR codes facilitate separating and identifying them from the background. However, objects only within 2.5 m were detected, which needed improvement. In adverse conditions, such as the blurring effect of motion, the system has difficulties identifying QR codes [79].

An android navigation application was introduced for PVI using QR codes that utilizes the smartphone's camera. QR codes intended to be used by PVI are installed on the floor. Initially, the current location is defined then, it finds the shortest path to the PVI's destination. During navigation, any deviation from the predefined path is detected and corrected. All the instructions are given in an audio form to the PVI. This application provides automatic navigation on pre-defined paths for PVI and does not require any additional hardware. This application is capable of scanning QR codes of different sizes and in different challenging environments. However, instructions in audio and in haptic form should be added to increase performance and reduce navigation errors [80].

3.4.2.3 Markers

Square markers are square shaped tags, as shown in Figure 3-7. They have a thick black border, and the inner region contains images or binary codes represented in the form of grids of black

and white regions. The reason for using thick black border is to ensure quick detection on any surface.

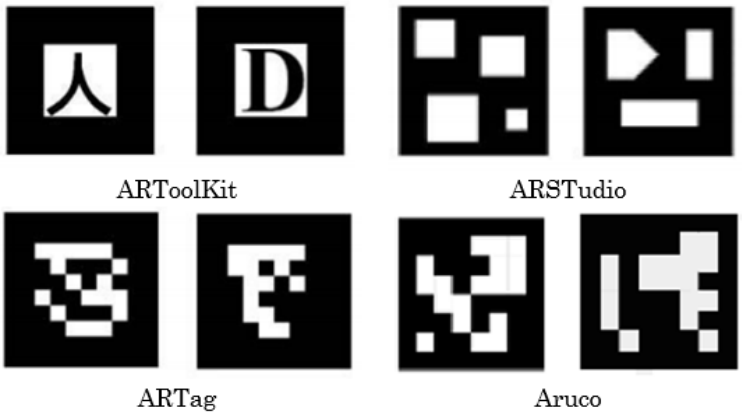


Figure 3-7. Examples of square markers.

Dash et al. proposed an AR system to be used in kindergarten to learn the alphabet by detecting the markers that may be present in a scene using CNN. Markers have been printed on papers within a rectangular box. Then, children can show them in front of the attached camera to automatically render the virtual object over the marker with appropriate position and orientation. This system achieved high accuracy in marker identification and augmentation of the virtual objects making the system resistant to environmental noises and position variation. However, it may fail to detect markers from a long distance [81].

Delfa et al. proposed an approach for indoor localization and navigation using Bluetooth and an embedded camera on the smartphone. It operates in two modes: background and foreground mode. The background mode gives a low-level accuracy position estimation using Bluetooth. The foreground mode provides high accuracy by using the smartphone's camera to detect visual tags deployed onto the floor at known points. The system can detect tags in real-time to estimate the PVI position with a high level of accuracy and navigate towards the target. The marker's colours are chosen to be different from the colour of the floor to enhance the speed and efficiency. However, the system failed to detect more than one tag at the same time [82].

Bacik et al. presented an autonomous flying quadrocopter using a single onboard camera and augmented reality markers for localization and mapping. The system can estimate the position of the quadrocopter using a coordinate system defined by the first detected marker. To improve the robustness of marker-based navigation, it used a fuzzy control to achieve a fully autonomous flight. However, the precision of the mapping approach and the response time requires improvement. The system also fails to detect markers from a long distance [83].

Kayukawa et al. proposed a collision-avoidance system for PVI using a camera that is integrated into the suitcase. It used depth images to determine the risk of collision with a blind person using a CNN model for detecting objects while YOLOv2 is used to detect pedestrians using the RGB streams. This system detects individuals efficiently. However, the execution time needed to be minimized for real-time usage [84].

Liu et al. proposed a system to develop a detection method for small objects based on YOLOv3. The darknet CNN structure was modified by increasing the convolution operation in the beginning to improve performance. The proposed method improved the performance of detecting small objects. However, it is not suitable for real-time usage by smartphones [85].

Tapu et al. proposed a navigational assistant prototype to increase the mobility and safety of PVI. It used CV algorithms and deep CNN to detect, track, and recognize objects in real-time. It modified the YOLO algorithm by adding an object tracking procedure to fulfil the missing information when YOLO is failing. It also introduced an occlusion detection and handling strategy to handle object occlusions and object movement or camera drift. The proposed system can process information from the environment and give feedback to PVI to avoid possible collisions. However, it is hard for PVI to carry this system on the back for a long time [86].

Tian et al. proposed an improved YOLOv3 model for detecting apples during growth stages. To improve the performance of the YOLOv3 while detecting apples, DenseNet was used to optimize feature layers with a low resolution by enhancing feature propagation, promoting feature reuse, and by improving network performance. The results showed that the proposed model provided real-time detection under overlapping and occlusion conditions. However, the performance needed to be improved for real-time usage by smartphones [87].

Mekhalfi et al. proposed a navigation system using computer-vision technologies. It included a speech recognition module to receive instructions and give voice feedback to PVI. A laser sensor was used to calculate the distance from obstacles. A set of markers and an IMU sensor were used to determine PVI location and, a path planning module was used to calculate a safe path for the user to walk through. They used a portable camera to capture the scene and forward the shots to the navigation or the recognition units. However, the size and weight of the processing unit is a big problem as PVI cannot wear it for a long time so using a smartphone is better. The average processing time for recognition should be minimized. Obstacle detection sensor is expensive and not available for common people [57].

Bazi et al. proposed a navigation system to help PVI recognize multiple objects in images using a multi-label convolutional SVM. It used a portable camera mounted on a lightweight shield worn by the user to capture images and send it via a USB wire to a laptop processing unit. To identify objects, a set of linear SVMs were used as a filter in each convolution layer to generate a new set of feature maps. Finally, the outputs are fed again to a linear SVM classifier for carrying out the classification task. However, the size and weight of the processing unit is a big problem as PVI cannot wear it for a long time. Also, it failed to detect makers from longer distances [88].

3.4.3 Hybrid Systems

In the previous two sections, several solutions to assist PVI in navigating and identifying products were discussed. These solutions use tags such as RFID and NFC, visuals tags such as QR codes and AR Markers, or CV techniques. However, these solutions are not suitable in all situations, because each environment has specific features. For example, CV techniques cannot be used in areas with considerable light, because the quality of the taken image will be poor. In this case, it is better to use a different technology, like RFID or NFC, to improve system accuracy. In shops, items already have barcodes or QR codes which store all the needed information. Developers can use these tags for product identification and use CV techniques or non-visual tags (RFID, NFC) for navigation. Two or more such technologies can be combined, which would lead to the development of hybrid systems.

For example, McDaniel et al. proposed a system that integrates CV techniques with RFID Systems. This system identifies information about relevant objects in the surroundings and sends them to PVI [89].

López-de-Ipiña et al. integrate RFID with QR codes, to allow PVI to navigate inside a grocery store. The system used the RFID reader to identify the RFID tags to navigate inside the store. It adopted the smartphone camera to identify QR codes placed on product shelves [70].

Fernandes et al. developed a solution to help PVI identify objects and navigate in indoor locations using RFID and CV technologies. This system used the RFID reader to receive the current location of PVI and CV techniques to identify objects [90].

Table 3-1 showed the analysis of the related work and Table 3-2 showed a comparison of the proposed system with the others in the related work.

Table 3-1. Analysis of the related work.

Type	Ref	Hardware and technology	Problems
Outdoor Navigation	**[50]**	• Notebook. • Camera. • CV.	**1.** Better to add tactile vibration patterns to guide the user. **2.** The size and weight of the processing unit are cumbersome for PVI to carry for a long time. **3.** Models were created manually from recorded videos of the evaluation test site.
	[51]	• Smartphone. • Gyroscope.	**1.** Needs to automate the process of selection of the correct heading. **2.** Sensor reliability should also be improved using sensor fusion techniques. **3.** Smartphone orientation must be set to a specific direction which prevents the user from using the phone for some other tasks while walking.
	[52]	• Smartphone. • GPS, Compass • Accelerometer.	**1.** Auditory and tactile feedback is better and allows operation in a noisy environment. **2.** Using GPS and AGPS give better accuracy
	[53]	• GPS, INS. • WIFI, AP.	**1.** Using sound commands or some haptic feedback to be suitable for PVI. **2.** PVI should carry INS/GPS devices and a notebook which is heavy, so using tablets or smartphones will be a better choice.
	[54]	• Smartphone. • cane device. • CV. • GPS.	**1.** Not suitable for indoor navigation as additional landmarks are needed to improve the model creation which requires a massive investment. **2.** A lot of time used for communication between the mobile device and the server. **3.** If the computation is done on smartphone, it needs a lot of resources which drains the mobile battery very fast.
Obstacle Detection	**[55]**	• White Cane. • Ultrasonic sensor. • IMU.	**1.** Needs more accuracy as the reliability of the model decreased when the distance to the obstacle increased. **2.** The system was tested on a single obstacle centered on the path of movement. **3.** Using earphones is hard to work with in a noisy environment.
	[56]	• Laser scanners • Notebook. • Micro controller.	**1.** Needs additional training to make PVI familiar with the vibration signals from the belt. **2.** Has a problem to estimate the exact position and dimension of obstacles
	[57]	• Notebook. • Camera, CV, IMU. • laser sensor.	**1.** Giving haptic feedback is better than voice commands, especially in noisy environments. **2.** The size and the weight of the processing unit is unsuitable to wear for a long time.
	[58]	• Camera. • CV. • Laptop.	**1.** Hard to carry a laptop for a long time. **2.** The frame rate could be improved to work at higher frame rates. **3.** Giving haptic feedback is better than voice commands, especially in noisy environments.

		[59]	• Smartphone. • Gyroscope. • CV.	**1.** Using the tactile interface to provide feedback is better. **2.** Using cloud computing is better to complete the processing.
		[60]	• Camera • WIFI module • Laptop • CV	**1.** Using circles to provide users with warning information is not suitable for PVI. **2.** it is better to give haptic feedback or some voice commands. **3.** Can be extended to collision detection in the region where the accurate motion vector is not easily computed such as walls.
		[61]	• Tango tablet. • CV.	**1.** User should hold the tablet in his hands roughly at waist level with the screen facing towards him for a long time. **2.** The feedback mechanism should be enhanced to provide more details about the distance to obstacles.
		[62]	• Mobile Kinect, CV, Laptop. • Accelerometer • Tactile-visual system	**1.** Heavy and everything must be mounted on the user's body, this way it is better to use a small and wearable device like Google Glass. **2.** Using tactile–visual substitution system needs a lot of training by the user to navigate correctly without errors
		[63]	• CV, IR sensor. • ML • Notebook	**1.** Giving haptic feedback is better and leaving the ear canal open for air conduction. **2.** Better to integrate it into a wearable system, so that hands and ears remain free to manipulate objects and acoustically perceive their surroundings.
Indoor Navigation	**Tag Based Systems**	[68]	• RFID. • GIS module. • CV. • Smartphone.	**1.** Technologies RFID tags must be in place and needs a lot of maintenance. **2.** It cannot be integrated easily with other systems and the coverage area is small. **3.** Hundreds of RFID tags are needed which are costly. **4.** Receiving information from all the items at the same time to identify various objects make a lot of overhead. **5.** Another issue is that RFID does not meet users' demands for trust and privacy since the readers are accessible to anyone
		[69]	• RFID • Server Database • Smartphone	**1.** The database for the product is not always comprehensive enough to cover all the items of interest, and the update frequency is slow. **2.** In a noisy environment, the use of the audio channel could require wearing headphones. **3.** Technology such as servers, Wi-Fi, or RFID tags must be in place. **1.** A lot of RFID tags are needed in the environment which are costly. **2.** Receiving information from all the items at the same time make a lot of overhead. **3.** RFID does not meet users' demands for trust and privacy since the readers are accessible to anyone
		[70]	• RFID. • White cane. • Smartphone • Web application.	**1.** In a noisy environment, the use of the audio channel could require wearing headphones. In many instances this would be not recommended. **2.** Technology such as servers, Wi-Fi, or RFID tags must be in place. **3.** A lot of RFID tags are needed in the environment which are costly. **4.** Receiving information from all the items at the same time to identify various objects make a lot of overhead.
		[71]	• NFC • Smartphone	**1.** NFC is not as effective and efficient as RFID or Bluetooth. **2.** PVI should be inside the reading area to identify NFC tags. **3.** PVI must have an NFC-equipped mobile.
		[72]	• NFC • Smartphone	**1.** NFC is not as effective and efficient as RFID or Bluetooth. **2.** PVI should be inside the reading area to identify NFC tags. **3.** PVI must have an NFC-equipped mobile.

	Computer Vision Based Systems	**Non-Tag-based techniques**	**[73]**	• Smart glasses • Backend server system • CV. • Glove with a camera.	**1.** Navigation between the aisles is not yet completely automated. **2.** Feedback latency needs to be reduced to make it more effective. **3.** Streaming raw video over the wireless channel uses extensive computation power which drained the battery. **4.** Performance could be uncontrolled in real-world environments because of some factors such as motion blur. **5.** PVI need to take a lot of photos and it is hard for them to take them in good quality.
			[74]	• CNN. • CV. • Laptop.	**1.** These systems use extensive computation power. **2.** PVI need to take a lot of photos and it is hard for them to take them in good quality. **3.** Feedback latency must be reduced which make the systems be more effective.
			[61]	• Tango tablet. • CV	**1.** PVI should hold the tablet in his hands roughly at waist level with a way which is difficult for a long time and prevents them to use their hands for other tasks. **2.** For small obstacles, the system failed for the far positions. **3.** The feedback mechanism should be enhanced to provide more details about the distance of the obstacle from the user. **4.** Direct sunlight caused severe problems in the mesh generation with either no mesh or an incorrect mesh being produced.
			[75]	• CV. • Mobile Kinect. • Laptop.	**1.** Is heavy and everything must be mounted on the user's body, this way it is better to use a small and wearable device like Google Glass. **2.** Using tactile–visual substitution system needs a lot of training by the user to navigate correctly without errors. **3.** Performance could be uncontrolled in real-world environments because of some factors such as motion blur. **4.** These systems use extensive computation power.
			[57]	• Notebook • Camera • CV • IMU • laser sensor	**1.** Giving haptic feedback is better than voice commands, especially in noisy environments. **2.** The size and the weight of the processing unit is unsuitable to wear for a long time. **3.** Performance could be uncontrolled in real-world environments because of some factors such as motion blur. **4.** These systems use extensive computation power. **5.** PVI need to take a lot of photos and it is hard for them to take them in good quality
		QR Codes	**[47]**	• RFID, QR codes • CV • Web-based management	**1.** Wi-Fi connection is required to retrieve the data from an online database. **2.** QR codes cannot be detected from a long distance. **3.** Difficult to detect tags if the PVI is moving fast and the recognition rate decreases if the distance between the reader and tags increases
			[76]	• QR. • CV. • Google Glass. • Smartphone.	**1.** Requires an internet connection to download the building graphs from the server. **2.** QR codes cannot be detected from a long distance. **3.** Difficult to detect tags if the PVI is moving fast and the recognition rate falls if the distance between the reader and tags increases
			[77]	• QR. • CV. • Smartphone.	**1.** Capturing good quality photos with smartphone camera is difficult as most of the photos may be blurry. **2.** If more than one QR code is detected at the same time, it is better to select one based on the distance, and not select it randomly **3.** QR codes cannot be detected from a long distance. **4.** Difficult to detect tags if the PVI is moving fast and the recognition rate falls if the distance between the reader and tags increases

			[79]	• QR. • CV. • Smartphone.	**1.** Objects only within 2.5 m were detected, which needed improvement. **2.** In adverse conditions, such as the blurring effect of motion, the system has difficulties identifying QR codes. **3.** Difficult to detect tags if the PVI is moving fast and the recognition rate falls if the distance between the reader and tags increases.
			[80]	• QR. • CV. • Smartphone.	**1.** Instructions in audio and in haptic form should be added to increase performance and reduce navigation errors. **2.** In adverse conditions, such as the blurring effect of motion, the system has difficulties identifying QR codes. **3.** Difficult to detect tags if the PVI is moving fast and the recognition rate falls if the distance between the reader and tags increases.
		Markers and deep learning	[81]	• CNN. • CV. • Laptop.	**1.** Fail to detect markers from a long distance. **2.** The size and weight of the processing unit is a big problem as PVI cannot wear it for a long time. **3.** Not suitable for real-time usage by smartphones
			[82]	• Bluetooth module.CV • Smartphone.	**1.** Failed to detect more than one tag at the same time. **2.** Fail to detect markers from a long distance.
			[83]	• Quadrocopter • Onboard camera. CV. • Fuzzy control.	**1.** Precision of the mapping approach and the response time requires improvement. **2.** The system also fails to detect markers from a long distance.
			[85]	• CV. • YOLO. • Laptop.	**1.** The size and weight of the processing unit is a big problem as PVI cannot wear it for a long time. **2.** Not suitable for real-time usage by smartphones. **3.** It failed to detect makers from longer distances.
			[86]	• CV. • YOLO. • Laptop.	**1.** The size and weight of the processing unit is a big problem as PVI cannot wear it for a long time. **2.** Not suitable for real-time usage by smartphones. **3.** It failed to detect makers from longer distances.
			[88]	• SVM. • Portable camera. • Laptop.	**1.** The size and weight of the processing unit is a big problem as PVI cannot wear it for a long time. **2.** It failed to detect makers from longer distances.
	Hybrid Systems		[89]	• CV, RFID, Laptop. • wearable camera. • haptic glove.	**1.** Hybrid systems increase in infrastructure usage due to the combination of technologies which add time consumption. **2.** Hybrid systems combine more than one technology at the same time which results in increasing complexity and cost.
			[70]	• RFID, QR, CV • White cane. • Web management. • Smartphone.	**1.** Hybrid systems increase in infrastructure usage due to the combination of technologies which add time consumption. **2.** Hybrid systems combine more than one technology at the same time which results in increasing complexity and cost.
			[90]	• RFID. • GIS module. • CV. • Smartphone.	**1.** Hybrid systems increase in infrastructure usage due to the combination of technologies which add time consumption. **2.** Hybrid systems combine more than one technology at the same time which results in increasing complexity and cost.

Table 3-2. A comparison of our system with the others in the related work.

AI		Hardware	Tags	Map Usage	challenging conditions	Function	Indented users	Accuracy	Problems
No	[76]	Google Glass. Smartphone	QR Code	Automatic	No	Indoor navigation	Sighted PVI	-	It needs an Internet connection to download the graph of the building from the server. Using markers would better than QR codes which can be detected from a long distance. The use of Google Glasses is an additional burden for the user and not available for common people.
	[91]	Smartphone	Marker	Automatic	No	Indoor navigation	Sighted PVI	-	It installed markers on the ceiling of the building which is difficult for PVI to detect. Installing markers this way lowers the aesthetic value of the building. The system was tested only by blind-folded people.
	[92]	Smartphone	Marker	Manually	No	Indoor navigation	Sighted	-	Markers are installed on the floor which cannot be detected in a crowded environment. It fails to detect markers from long distances. Building maps automatically would be better to enable updates when needed.
	[93]	Smartphone IMU	Marker	Manually	No	Indoor navigation	PVI	-	Implemented as logging system which is not suitable for real time usage. The accuracy of markers' recognition needed to be improved as they are only visible and recognizable in a small fraction of video frames, and they cannot be detected in motion blur or in rapid walking speed.
	[57]	Laptop IMU Laser sensor Portable camera	Marker	Manually	No	Indoor navigation Objects - Obstacles detection	PVI	-	The size and weight of the processing unit are cumbersome for PVI to carry on the back for a long time. Obstacle detection sensor is expensive and not available for common people. The processing time for the recognition system should be minimized. Objects and obstacle detection should be improved by using deep learning techniques. IMU sensors have an acceptable positioning accuracy only for a short distance since it suffers from drift error estimation over time.
Yes	[79]	Smartphone	QR Code	Yes	No	Indoor navigation Object detection.	PVI	97%	Using markers is better than QR codes which can be detected from a long distance.
	[94]	Smartphone	Marker	Manually	Yes	Indoor navigation	PVI	97%	Used CV techniques to select the candidate markers from images and sends them to classification models which takes processing time that should be minimized. Building maps automatically would be better which enables updates when needed. Using markers would better than QR codes which can be detected from a long distance.

	[88]	Shield laptop	Marker	No	No	Objects and obstacles detection	- PVI	92.90	The size and weight of the processing unit are cumbersome for PVI to carry for a long time. It fails to detect markers from long distances and in challenging conditions. Using maps would be more accurate and better to help PVI navigation easily.
	[81]	Laptop	Marker	No	Yes	Learning alphabets using AR shapes	Kids	95%.	It fails to detect markers from a long distance and was designed for kids. Implemented as a desktop application. Used CV techniques to select the candidate markers from images and sends them to classification models which takes processing time that should be minimized.
	[86]				No	Objects and obstacles detection	PVI	91 %	The size and weight of the processing unit are cumbersome for PVI to carry for a long time.
	Our	Smartphone	Marker	Automatic	Yes	Indoor navigation	Sighted PVI	99.31 %.	Integrating orientation sensors to quickly warn PVI if they turn in the wrong direction would improve accuracy. Adding support to detect and avoid obstacles would be better.

3.5 Conclusions

This chapter has presented the current trends and technologies in building navigation systems for PVI. Research studies published in the literature from 2010 to 2019 were selected to prove the robustness of the taxonomy. As a result, the taxonomy analysis allowed to categorise families of AT solutions to help PVI in navigation, shopping, and at home as well. In the literature review, there is a concentration on navigating indoors. Concretely, the taxonomy is built based on three types of systems: Tag-based systems, CV systems, and hybrid systems. An analysis of the strengths and weaknesses of each one has been presented. From these analyses, a decision has been taken to use CV tag-based systems. Thus, in the following chapter, different CV tag-based systems have been compared and Aruco markers have been selected as the best. Moreover, an indoor navigation system using a CV tag-based will be presented.

4 Navigation System for PVI

PVI face a lot of problems in their daily activities. As mentioned in the previous chapter, there are a lot of research in building navigation systems to help PVI navigate safely. The essential steps of a typical navigation system are identifying the current location, finding the shortest path to the destination, and navigating safely to the destination using navigation feedback. This chapter focuses on using assistive technology to help people with visual impairment in indoor navigation using markers. At first, it focuses on comparing different technologies and selecting the best one for the navigation system. It shows the architecture of the proposed navigation system and how to use it by PVI to navigate indoor easily and avoid objects.

4.1 Comparing different Technologies

CV tag-based systems offer several advantages: they only need to identify tags to get product details, so they need low computational power and small storage space. Many of these approaches do not need tags to be explicitly placed, as products already have unique visual tags, such as barcodes and QR codes. Such tags can be generated and printed at a very low cost compared to non-visual tags, such as RFID, and can be easily modified. CV tag-based systems are ideal for tasks that require differentiating among groups of objects. They are vital for PVI when the contents of the items are different, such as a tube of glue versus a tube of eye drops, as they have the same shape, and it may be dangerous if they choose the wrong one. However, tag-based CV techniques require a prior selection of items and the correct placement of tags on those items. Moreover, if there are many tagged items in a small area, PVI would be confused by receiving information about them all at the same time. Visual tags must also be in line-of-sight of the camera, otherwise, they will not be detected. Furthermore, visual tags can also be damaged during movement through the supply chain or by weather. Also, it is difficult for a smartphone camera to detect CV tags if the PVI is moving fast, and the recognition rate decreases as the distance between the reader and the tags increases. CV non-tag-based systems have several advantages. These systems are cost-effective, as they need little or no infrastructure and most of them can be easily installed on smartphones. However, they have several limitations. Their performance may be unreliable in real-world environments because of factors like motion blur and image resolution, as well as changes in conditions, such as illumination, orientation, and scale. These systems use extensive computational power, and PVI need to take many photos. However, taking good quality photos is difficult for PVI. Finally, feedback latency must be reduced to make these systems more effective [49].

Finally, hybrid systems take the strengths of two or more systems and combining them. Numerous attempts have been made in this area to balance the trade-offs of the combined technologies. As a result, there is a significant improvement in accuracy, robustness, performance, and usability. However, the major drawback of these systems is that they use significant infrastructure due to the combination of technologies, which results in increased complexity and cost. Table 4-1 summarizes the criteria for the most effective solutions.

Table 4-1 Comparison of identification technologies for PVI.

Technology	Cost	Equipment	Number of Scanned Items	Requires Line Of-Sight	Range	Capacity
NFC	Low	NFC reader	1	No	Up to 10 cm	Maximum 1.6MB
RFID	Low	RFID reader	Multiple	No	Up to 3 m	Maximum 8000 bytes
QR code	Free	Camera	1	Yes	Depends on code size.	Maximum 2953 Bytes
Barcode	Free	Camera	1	Yes	Depends on code size.	N/A
Markers	Free	Camera	Multiple	Yes	Depends on marker size.	N/A
CV techniques	High	Camera	Multiple	Yes	Depends on camera	-

The first criterion is the cost of applying the technology to any solution. It is shown that CV tag techniques can be used without any cost except for printing the QR codes or AR markers and putting them in the correct place. When using a barcode, there is no need to print them, as they are already placed on each product. Tag based techniques can be used at a low cost, as shops only need some RFID or NFC tags to be installed on the correct places. If CV techniques are used, high-quality equipment, such as cameras, are needed for good results. The second criterion is the equipment needed to detect and identify products or places. For CV tag-based solutions, PVI need only their smartphone cameras to detect and identify items. In CV techniques, some solutions only need smartphone cameras, while others need high-quality cameras to take high resolutions images and machines with powerful processors for computations. In tag-based techniques, PVI need RFID reader or smartphones supporting NFC technology. The third criterion is the number of items able to be scanned at the same time. Only RFID readers, AR markers, and CV techniques can scan more than one item at the same time, which is useful in some situations, such as if PVI want to identify and count the items in their shopping cart. The fourth criterion is whether the PVI must be in the line of sight with the identified products. For tag-based solutions, PVI do not have to be in the line of sight of items, and the PVI can identify them in any direction. In tag-based solutions, tags must be within 3 m for RFID tags and within 10 cm for NFC tags, while CV solutions depend on some other parameters, like the tag size in the QR codes or barcodes, and the marker or camera parameters for CV techniques. The last criterion is the storage capacity of each solution. Only some tags, such as RFID, NFC and QR codes, have a storage capacity, while others, such as AR markers and barcodes, do not have any storage capacity. Researchers can select and design new technology solutions based on specific requirements and which criteria to focus on, and how to evaluate trade off. Based on the evaluation of the available technologies, this thesis concentrated on CV tag-based techniques. The goal is to build an indoor navigation system for PVI using CV tag-based techniques.

4.1.1 Comparing QR code with Aruco markers

Based on the previous evaluation, QR codes and markers are the most suitable markers to select. So, QR codes were compared with markers to select which give better accuracy. Two applications have been developed to identify the location of PVI using the architecture shown in Figure 4-1.

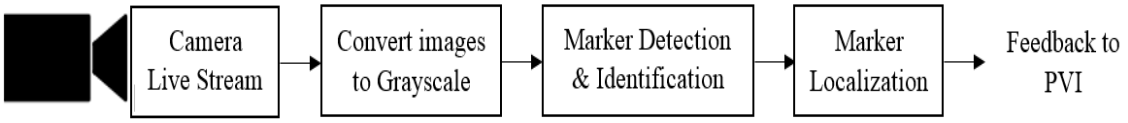


Figure 4-1. Main components of the comparison application.

These applications work as follow: at first, the application opens the camera to get a live stream of images. Then, it converts the image to grayscale and sends it to the desired library to detect and identify the marker. If it detects any markers, it gives feedback to the PVI using voice commands. QR codes were used as markers in the first application, while Aruco markers were used instead of QR codes in the second one. In the first application, QR codes with dimension of 10 × 10 cm are printed and installed in the environment at regular intervals. Each QR code stores information about the current location. For pre-processing, an open-source CV library called OpenCV was used that implement many algorithms for image processing and CV. An open-source library called Zxing was used for detecting and identifying QR codes. When PVI use this application, it activates the camera to capture photos until a QR code is detected. The position details are stored in the detected QR Code, and the distance are given to the PVI as voice commands. The second application is the same as the first one, but it uses Aruco markers with the same dimension instead of QR codes. An open-source library called Aruco library was used for detecting and identifying markers. After testing the two applications in different situations, a comparison was done to select which one is the best, as shown in Table 4-2. Based on this comparison and the geometrical differences between the QR code and the Aruco marker shown in Figure 3-7, Aruco markers can be detected from distances up to 4 m while QR codes were limited to only 2 m. To detect Aruco markers from long and short distances, there is no need for the camera to be in their line of sight. For QR codes, they cannot be detected from farther than 2 m and whether the camera was in their line of sight was irrelevant. From these results, Aruco markers are better than QR codes and are selected as markers for the indoor navigation system.

Table 4-2. Evolution of Aruco markers and QR codes in different conditions.

Aruco Markers	1 Meter	2 Meters	3 Meters	4 Meters
Scanned items	Multiple	Multiple	Multiple	Multiple
30 degrees	√	√	√	√
60 degrees	√	√	√	√
80 degrees	√	√	√	√
Moving fast	√	X	X	X
Line of sight	X	X	X	X
Blur	X	X	X	X
Camera out of focus	X	X	X	X
Occlusion	X	X	X	X

QR Codes	1 Meter	2 Meters	3 Meters	4 Meters
Scanned items	Multiple	Multiple	X	X
30 degrees	√	√	X	X
60 degrees	√	√	X	X
80 degrees	X	X	X	X
Moving fast	X	X	X	X
Line of sight	X	X	X	X
Blur	X	X	X	X
Camera out of focus	X	X	X	X
Occlusion	X	X	X	X

4.2 Navigation System Architecture

In this thesis, a system to help PVI navigate indoors using markers is proposed. At first, tags are printed on multiple pieces of paper and placed in the specified location of interest. Then, a graph is created where nodes represent the markers' position. Edges are labelled with the number of steps and the direction between nodes are connected to it. This graph is stored in a database to be used during navigation. The system starts by requesting the PVI to select their starting point based on the surrounding tags. When a marker is detected by the smartphone camera, the system will use this marker as a starting position. To start navigation, the system asks the PVI to choose their destination using voice commands. Then, it searches the database for the shortest path from this point to the destination. This path is a list of checkpoints that the PVI should pass to arrive at their destination. During navigation, continuous guidance is given to them when moving from one point to the next. The system uses a voice recognition API to covert PVI commands to text. As it is difficult for machines to understand the natural speech patterns of human beings and without proper structuring, the meaning behind the words may be interpreted wrong by machines. It uses test to speech to provide audio feedback to the blind person. Figure 4-2 shows a diagram to illustrate these steps.

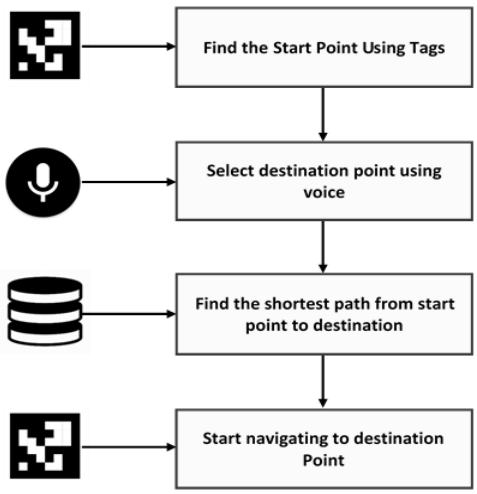


Figure 4-2. Components of the proposed system.

Based on this figure, the essential components of the navigation system are: 1. How to find PVI accurately location at any time based on the installed tags so, they can continue navigate safely. 2. How to navigate from the starting point to the destination based on these tags. The proposed system functions through two major phases: (a) Preparing the building for PVI by installing markers at the specified locations and building a map for them. (b) Guiding PVI during navigation using text to speech. The following sections describe these phases in detail.

4.2.1 Building a Map

Before using the navigation system, a map should be constructed for each floor in the building by sighted people. They should move inside the building to find interest points such as laboratories and lecture rooms. Then, markers are printed and installed on the wall at the chosen places. Later, these markers help in guiding PVI to navigate inside the building. After that, an admin application is used to scan each marker and store details about it in a Firebase Database. This information includes marker id, the floor number, and the name of the place. This process is repeated for each floor in the building. Figure 4-3 shows a blueprint of the fourth floor of the

faculty of information technology at the University of Pannonia with interest points marked in red circles.

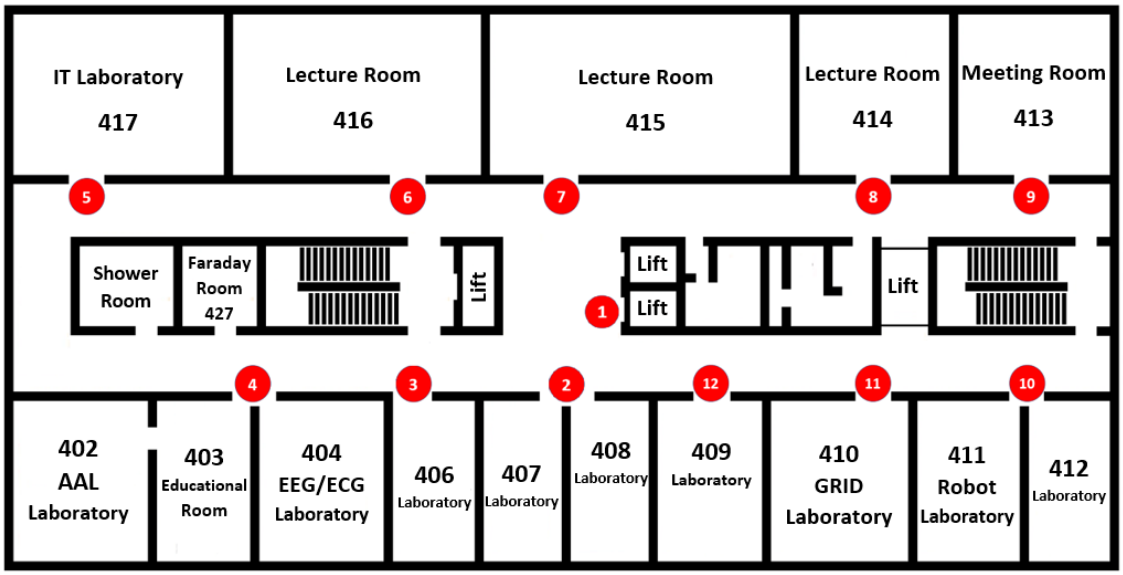


Figure 4-3. The plan of the fourth floor's corridor.

After that, a sighted person should explore all the available paths from each interest point to the points around it and measure the number of steps between them. For example, starting at node 7 in Figure 4-3, a sighted person should explore and count the number of steps from it to its neighbour nodes (1, 2, 6 and 8). The number of steps is counted at different distances to simulate different persons having different step lengths based on their impairment. Then, the average numbers are calculated for them. After that, a virtual map is constructed using a graph to store these points and the relations between them. In this graph, nodes represent interest points, and edges represent the connection between them. The average number of steps was calculated between markers as the graph edges. In the above example, four edges are created to connect point 7 with the four other points around it. The first one is between points 7 and 1, where the number of steps were stored on it. The second edge is between points 7 and 2, while the third edge connects points 7 with 6. The last edge connects points 7 with 8. This process is repeated for all interest points. Figure 4-4 shows the constructed graph for the blueprint of the fourth floor.

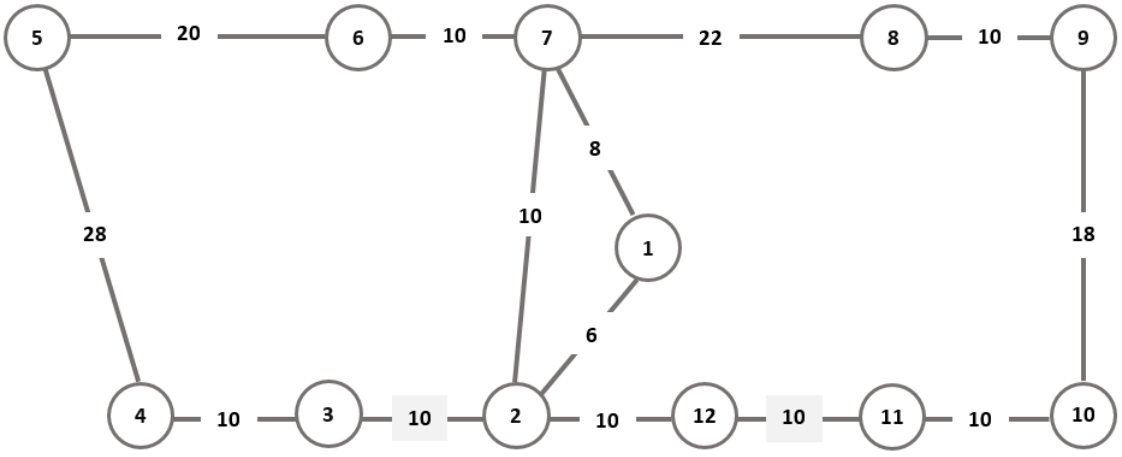


Figure 4-4. A graph of the fourth floor's corridor at the same faculty.

Circles represent interest points while edges represent the available paths between them. In addition to adding graph nodes to the Firebase Database, the admin application is used to store edges in it. To add this data to the database, a sighted person uses the admin application to scan two markers connected by an edge to declare it then adds the number of steps in text format. This process is repeated for all edges in the graph until the Firebase Database contains all nodes, edges, and number of steps for all edges that will be used by the navigation commands for PVI. The admin application allows removing existing markers and adding new markers. When PVI install the navigation system for the first time, it downloads the building's graph from the Firebase Database and stores it in the smartphone's local database to allow to use it without Internet connection.

4.2.2 Navigation

The navigation system was designed for ease of use by PVI using an audio interface. With a single tap on any part of the screen, the prototype application opens the camera to get a stream of frames and converts them into grayscale images. After opening the camera, an audio message asks PVI to move the smartphone left and right to search for any marker using a Text to Speech (TTS) module. This module is used to give an audio message when it is needed to provide audio feedback to PVI. There are two fundamental components for a typical TTS model: text analysis and speech synthesis. These components convert symbols like numbers and abbreviations to written words. Then, a speech synthesis converts them into sound to be understood by humans [95]. If any marker is detected, the system uses it as a starting position then it asks PVI to select the destination point using a voice command. Aruco library was used to detect markers during navigation and calculate the distance from them to the camera. It depends on the size of the markers as seen on the captured images so, camera calibration is needed at first [96]. Figure 4-5 shows the architecture that PVI should follow to reach the destination point.

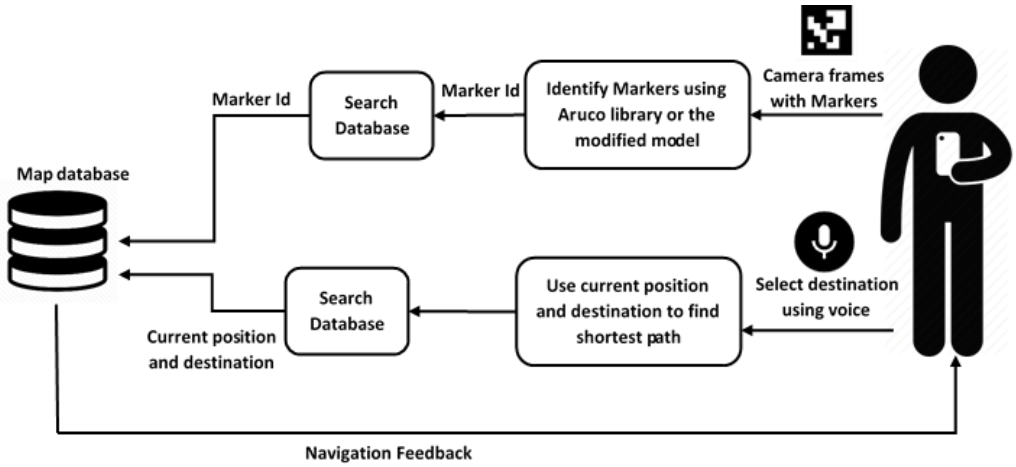


Figure 4-5. System architecture that PVI should follow to reach the destination point.

When PVI communicate with the navigation system using voice command, a speech recognizer API is used to covert these commands to text using Natural Language Processing (NLP). Using NLP algorithms provide a way to convert voice commands correctly to text. The NLP algorithm found in the Google API is used. Then, audio feedback is given to PVI confirming whether their command was recognized or not. If it is unrecognized, the system asks PVI to input the destination again. Once the starting point and destination are identified, the prototype calculates the shortest path from the initial point to the target destination using the Dijkstra algorithm and

instructs the PVI to start walking in the appropriate direction. This returned path is a list of marked points that PVI should go through to reach the destination [97]. The PVI should follow the navigation commands to move from one point to the next until arriving to their destination. When the PVI reach any point by detecting the marker placed on the wall, the prototype gives navigation commands guiding them to the next point on the graph. Figure 4-6 shows an example to illustrate this process.

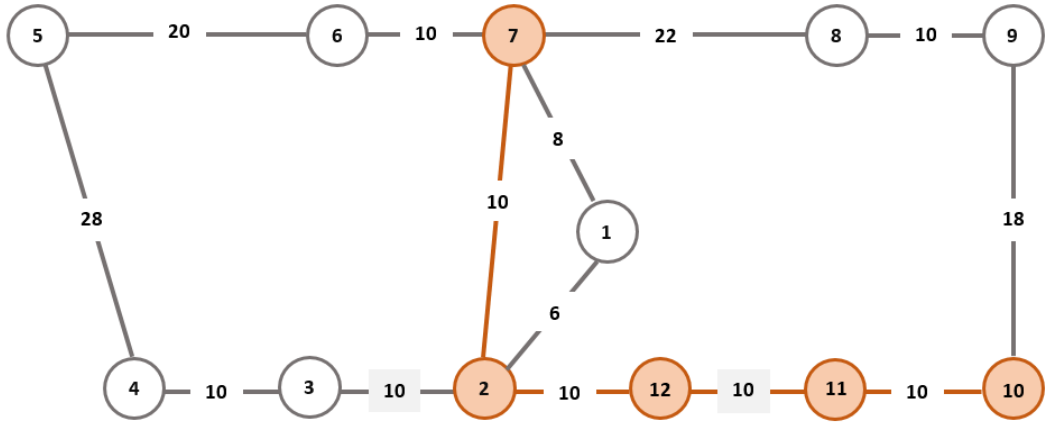


Figure 4-6. The shortest path to destination example.

Suppose PVI stands in front of point 7 and wants to go to point 10. PVI taps the screen that opens the camera and moves the phone around as instructed by the application's voice command. The system detects and identifies point 7 as starting point and gives voice feedback that the initial point was selected. If more than one marker is detected simultaneously, the nearest one is selected as starting point based on distance between the phone and the marker. In this example, PVI selects point 10 by saying "lab 411" to be their destination using voice commands. Then, the shortest path is calculated from point 7 to point 10 and returned as a list. PVI is instructed to follow this list of points and go from point 7 through point 2, 12, 11 and 10 to reach their destination. From point 7, the proposed system gives PVI navigation feedback to reach the next point which is point 2. To reach it successfully, PVI should follow the instruction to detect the marker installed on the wall. When the marker for point 2 is detected, the application asks PVI to go towards it and gives a notification about the distance when needed. To make the system more accurate, it counts the number of steps that PVI take by walking from one marker to another using smartphone sensors then it compares it with the number of steps stored in the database. From point 2, the same process is repeated to guide PVI to go to the next point which is point 12. Then, go to point 11 and finally reach point 10 which is the final point. When they reach the final point, a message is given to PVI that they successfully arrived at their destination. For the navigation system to work accurately, all the situations and conditions that PVI may face during navigation are add. An android smartphone (HTC desire 826) with 30 frames per second was used. It means that the camera takes 30 images every second and sends it to the application for processing. Most of the images sent in a second have nearly the same scene, if it misses detecting marker in one frame, it will likely successfully identify it in the next ones. If PVI finds another marker, there are two possibilities. If this marker is in the list of points to the destination, the system continues giving navigation commands from this marker to the destination point. However, if this marker is not on the list, the system searches for a new shortest path from that new marker to the destination point. If the PVI moves in a wrong direction, the camera will likely find another marker since markers cover most places inside the building. If the camera fails to detect any marker for some

time such as 30 seconds, the application gives feedback to PVI that they are walking in the wrong direction. The navigation system has been evaluated at university of Pannonia. HTC Desire 826 smartphone with 2 GB RAM, octa-core (4 x 1.7 GHz Cortex-A53 and 4 x 1.0 GHz Cortex-A53) was used. At first, PVI press on the screen to select the starting point then, the application opens the smartphone camera and guides PVI to search for any markers around them to be used as an initial point. After that, PVI select the destination like saying "lab 404" to be their destination using voice commands. Depending on the initial position and the destination, the right maps are loaded from the database then, the shortest path to the destination is calculated. Finally, it starts guiding the PVI to the next point using the voice navigation commands listed in Table 4-3.

Table 4-3. List of the input commands and the navigation feedbacks given by the prototype.

Command Type	Name	Description
PVI's voice commands	"Go to" + destination	The PVI order the prototype to lead them toward the predefined destinations.
	"Start"	The PVI order the prototype to go to the start activity to select the start point.
	"Exit"	The PVI order the prototype to exit.
Navigation instructions	"Incorrect destination, you should press on the screen and select it again"	The prototype informs the user that they should provide another destination.
	"Go straight" + number of steps	The prototype directs the user to go straight for a number of steps.
	"Turn left", "Turn right"	The prototype directs the user to turn left or right.
	"Use Elevator"	The prototype directs the user to use the elevator from one floor to another.
	"You have detected your next point, so, you should go straight to reach it"	The prototype informs the user that the next point is detected, and the user should move to it.
	"You have passed this point successfully"	The prototype informs the user that they passed this point successfully and have started navigating to the next point.
	"You have reached your destination so, go straight to it"	Once the user reaches the desired destination, the prototype informs them.

Figure 4-7 showed screenshots of the system. As shown in part (a), it asks the PVI to select the starting point by pressing on the screen. As shown in part (b), it launches the mobile camera and guides them to search for any markers to be used as a starting point.

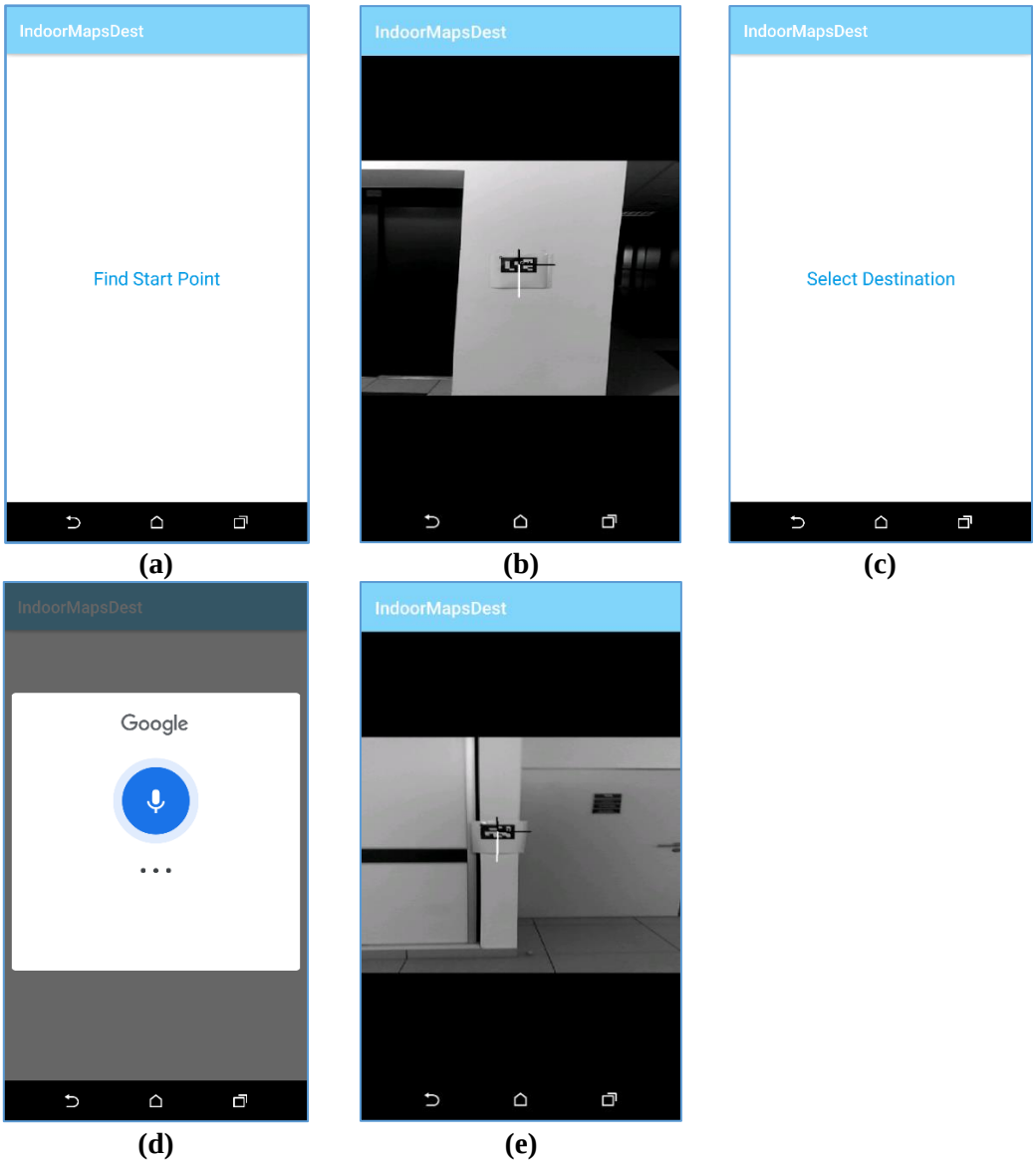


Figure 4-7. Screenshots of the prototype.

As shown in part (c) and (d), the third step is to select the destination using voice commands. For example, PVI say “lab 417” to be their destination. The system calculates the shortest path from the starting point to the destination using the Dijkstra algorithm. Then, it launches the smartphone camera and starts guiding the PVI to the next point using voice commands as shown in part (e). To show a simple test case to reach a destination on floor number four. Starting from the building entrance on the ground floor, PVI select laboratory number 404 to be the destination point which is stored on the map as node four in floor number four. So, the starting point and destination are on different floors. At first, the shortest path is calculated from the entrance on the ground floor to the ground floor elevator. After reaching the fourth floor successfully, the shortest path between the elevator and lab number 404 is calculated. To reach the destination point, the application asks PVI to start walking from the current position, which is node 1 to the next node, which is node 2. At node 2, it guides them to turn right and walk for 10 straight steps to reach the next point which is node 3. Finally, PVI walk for another ten steps to arrive at the destination which is node 4.

4.2.3 Test cases

Testing of the prototype was divided into two test cases. The first case was to test it with blindfolded people or PVI, collection of the feedback and updating it. The second case is to evaluate the prototype again after applying modifications using the PVI's comments. The prototype was tested on the corridor of the first and the fourth floor. In the beginning, a short introduction to the case study was provided to the participants. The users were trained for 30 min to know how to use the prototype for navigating from one place to another. The goal was to test whether the prototype was easy to use or not. It also tested if the users could effectively interpret the feedbacks or not. It was assumed that there were no objects or obstacles on the way to the destination. During navigation, the user held the smartphone in his hands roughly at chest level with the screen facing towards him. The smartphone was held in portrait orientation while slightly tilted at an angle nearly perpendicular to the horizontal plane. As shown in Figure 4-8, using this angle is enough for covering the walking area in front of the PVI and identifying markers. For a hands-free option, the smartphone may also be mounted on the user's chest. Audio feedback is provided to the user via headphones connected to the smartphone or by a smartphone's speaker.

(a) **(b)**

Figure 4-8. Screenshots of the prototype: (a) a blindfolded person; (b) PVI.

4.2.3.1 First Test Case

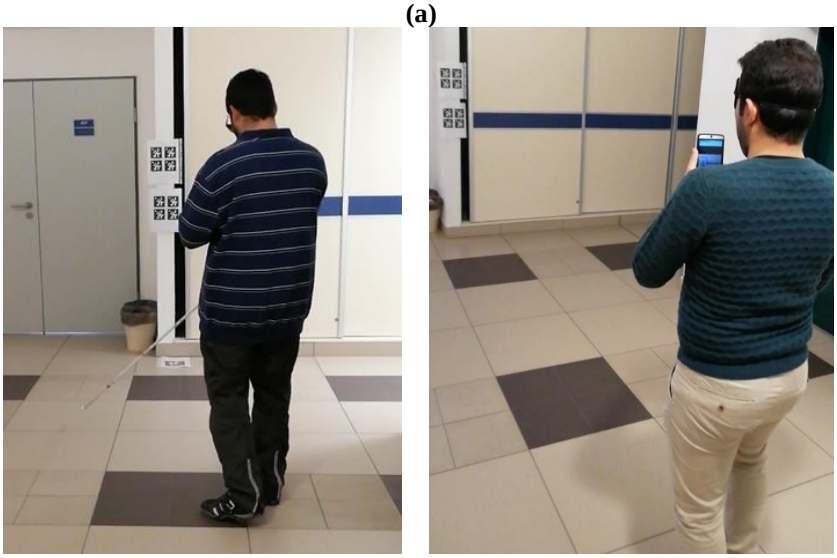
After learning how to use the prototype, they have tested it several times by selecting a start point and destination. The prototype assists them in moving from the starting point to the destination using navigation feedback. During the process, some problems were discovered: 1. Sometimes PVI failed to understand the feedbacks and as a result, the feedback needed to be improved. 2. PVI can hardly detect markers because they were placed higher than the view of the camera. So, installing markers at a lower position is necessary. 3. PVI move their hands rapidly during navigation which causes images to be captured with occlusion. 4. PVI cannot detect markers because they are moving their hands a lot and tags move out of the smartphone's camera view. 5. PVI take shorter steps compared to blindfolded participants, so I should calculate the number of steps based on PVI rather than the blindfolded individuals. 6. PVI occasionally create situations that cannot be managed by the prototype. For example, if their next point is node 7 and

they go in the wrong direction leading to another point. The prototype should check whether this point is node 6 or not. If it is node 6, the prototype should continue navigating, because node 6 in the graph is the next point to destination after the node. However, if it was another point, the prototype should ask the PVI to go back and search for node 7 again.

4.2.3.2 Second Test Case

This thesis tried to solve the problem occurring during the first test case. For the first problem, the feedback was improved based on the comments of the PVI. As shown in Figure 4-9, markers should be installed in a different style to solve the second problem. Instead of adding one marker at each interest point, eight markers are installed with the same ID. This implementation makes detection easier and solved the third and fourth problems. It also helps PVI of different heights to detect markers easily. The steps are counted in the same way as the PVI walk to solve the fifth problem. All situations and conditions raised during the testing phase of the prototype have been managed. Users have tested the prototype several times by selecting a starting point and destination. PVI found it easier to detect markers faster than before. Using this arrangement of markers can be detected easily while moving their hands rapidly. Finally, the audio feedback is satisfactory.

(a)



(b)

Figure 4-9. Screenshots of the testing environment.

Navigation Efficiency Index (NEI) [98] was included to evaluate the navigation performance of the systems. NEI is defined as the ratio of actual traveled path's distance to optimal path's distance between source and destination. The average NEI is calculated on sub-paths, i.e., a part of the path taken by the subject while walking from the beginning to the end of the path as follows:

$$\text{NEI} = \frac{1}{N}\sum_{i=1}^{N}\frac{L_A(S_i)}{L_O(S_i)} \qquad (4-1)$$

where N is the number of sub-paths, S_i is a sub-path, L_A is the actual length traveled, and L_O is the optimal length of S_i. The navigated paths were evaluated using NEI. In this case, main path was divided into 12 sub-paths. The results are given in Figure 4-10. The measured NEI score shows that the usability of this system is indeed acceptable in the tested indoor navigation scenarios. As shown, the low values happen when there are some turns to left or right and there are no markers in these turns. So, this will be improved it by adding check-point markers at these turns to improve navigation.

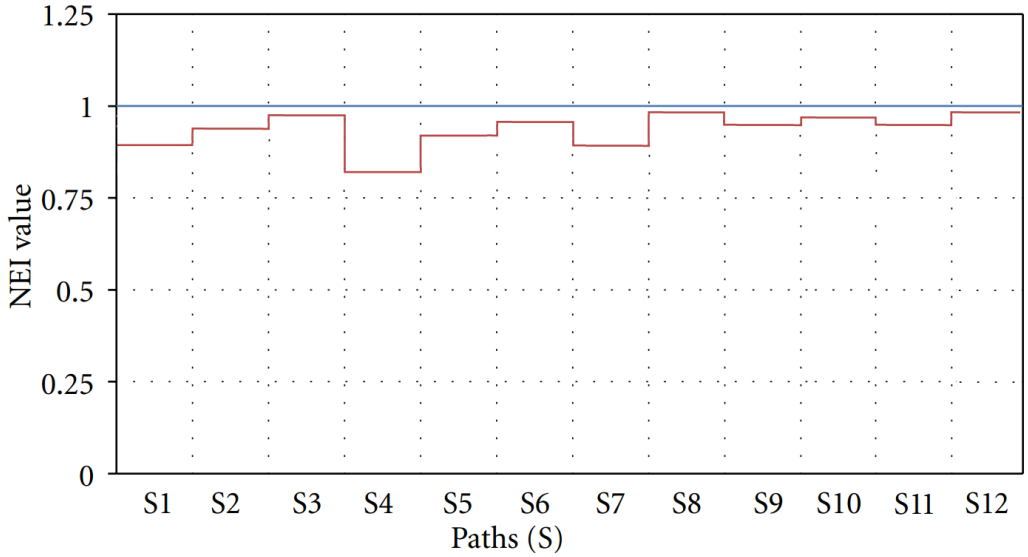


Figure 4-10. Mean navigation efficiency index (NEI) versus paths (S).

4.3 Objects Detection System Architecture

Precise and fast indoor object detection and recognition is an important task to helps the PVI interacting with the external world. DL models are useful and proved their big performances in object detection and recognition tasks. A system to help PVI avoiding objects using deep learning model is proposed. It starts by opening the camera and asking the PVI to move to reach their destination. While walking, a real stream of images from the smartphone camera is captured then, these images are converted to grayscale ones and sent to the deep learning model to detect objects. If any object is detected, feedback to PVI to avoid it is returned. However, if it fails to do so, it decides that no objects are available and continues processing the next image. Figure 4-11 shows the flowchart for this process.

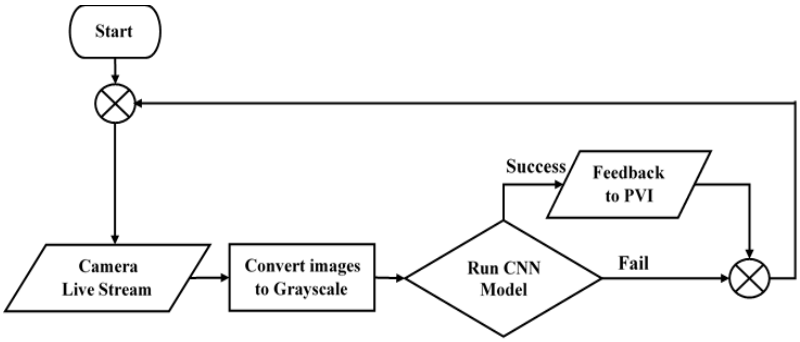


Figure 4-11. Flowchart of the detection process.

4.3.1 Dataset

Images have been collected for different five indoor objects for simplification. 2000 images for each class were used so, a total of 10000 images for training. For validation, 400 images per class were used which is a total of 2000. For testing, 100 images per class were used which is a total of 500 images. Table.4-4, shows examples of these images. All images are annotated manually and use text annotation files to store information about bounding boxes. Then, the final dataset was used to train the YOLOv3 and Tiny-YOLOv3 models and select the best-trained model.

Table 4-4. Samples from indoor objects collected from dataset.

Class Name	Chair	Desk
Class Image		
Class Name	Door	Stairs
Class Image		

4.3.2 Deep Learning Model

In recent years, ML algorithms have been used in the field of CV to improve object detection. The deep convolutional neural network increases the network levels, which makes the network have stronger detection capabilities. DL algorithms for object detection can be divided into two categories: two-stage and one-stage. Region-based CNNs (R-CNN)s [99], Fast R-CNNs [100], and Faster R-CNNs [101] are two-stage algorithms that exceed many other detection algorithms in terms of accuracy. The R-CNN predict object locations using region proposal algorithms. Features are extracted from each candidate region, fed into CNNs, and finally evaluated by Support Vector Machines (SVMs). R-CNN increases the target detection accuracy while the efficiency is very low. Faster R-CNN can detect the region of interest in the input image using the region proposal network. Then, it uses a classifier to classify these regions of interest which

are called bounding boxes. Such models reach the highest accuracy rates. However, they need more computational resources and processing time.

On the other side, single-stage detectors such as Single Shot Detector (SSD) [102] and YOLO [103] were proposed to improve the detection efficiency to be suitable for real-time applications using a simple regression. Such models are much faster than two-stage object detectors however they achieve lower accuracy rates [104]. YOLO uses a single CNN to predict object categories and find their locations. Several versions of the YOLO model were proposed to improve the accuracy without notable effect on speed. YOLOv2 [105] is an improvement of YOLO by using higher-resolution feature maps that help the network detect objects of different scales. It also has an added batch normalization on each convolution layer and bounding boxes are being predicted by using anchor boxes. YOLOv3 improves previous YOLO versions by using multi-scale detection, a more powerful feature extractor network, and some modifications in the loss function, which allows detecting big and small targets [106]. Moreover, YOLOv3 is more accurate than some of the two-stage detectors such as Faster RCNN and can detect small targets very well. The detection accuracy of YOLOv3 model is very high, but the execution time needs to be improved for real-time applications, especially on smartphones. Figure 4-12. shows the architecture of the YOLOv3 model using input images with dimension 416×416.

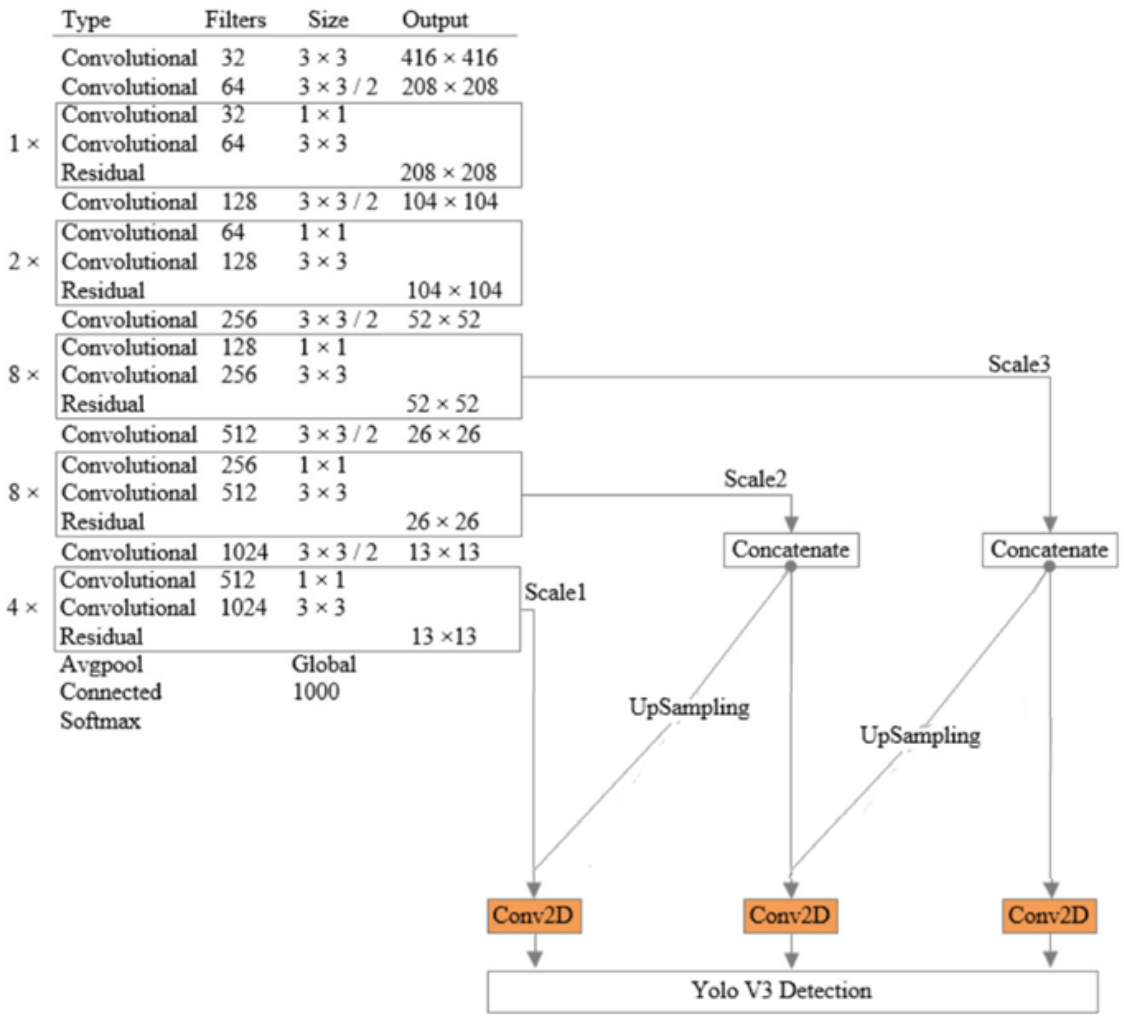


Figure 4-12. The architecture of YOLOv3 model.

Also, Tiny-YOLOv3 is a simplified version of YOLOv3 model that accepts images as an input. It consists of two main blocks: feature extractor and detector. The feature extractor uses the input images to extract features embedding at different scales. They feed these features into two detectors to get bounding boxes and class information. The feature extractor hierarchically extracts features from images of the input layer. It uses 3x3 filters and max-pooling layers to reduce the dimension size of the input. On the other hand, the detector used 1x1 convolutions structure to analyze the produced results to predict position and class of detected objects in the input image. Figure 4-13. shows the architecture of Tiny-YOLOv3 model using input images with dimension 416×416. Both models were used to compare and balance between the accuracy and the execution time. Given an image to this model, the final output is a list of bounding boxes along with the recognized classes. At first, the dimension of input images is reduced, and the features are extracted by going through several convolutional layers which makes the detection classifications. Finally, the output is given as a feature map which represents the network class prediction. This output is converted to bounding boxes with class id.

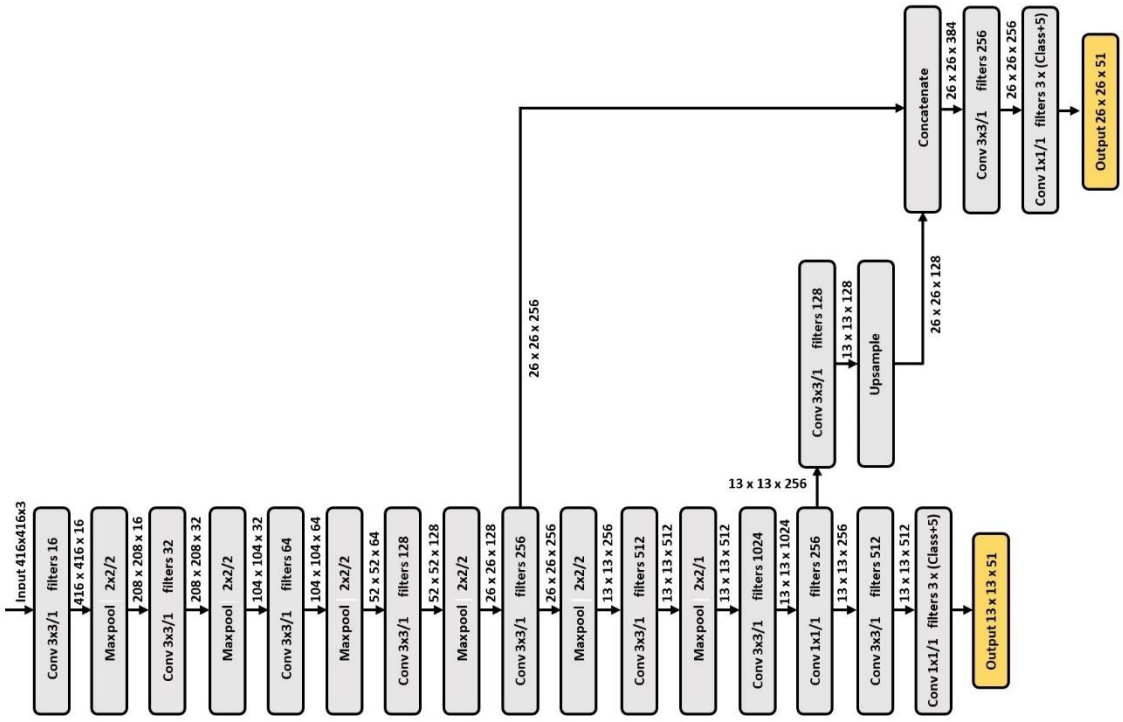


Figure 4-13. The architecture of Tiny-YOLOv3 model.

4.3.3 Experiments

Training proposed marker detection model requires a lot of resources. Therefore, Google Colab was used to leverages the power of free GPU for training the dataset quite easily. The models were evaluated using videos on a Dell Inspiron N5110 computer with Intel Core i7-2630 QM 2.00 GHz CPU, 6 MB cache, quad-core, and 8 GB RAM. The models were uploaded to the mobile device with the following specification: HTC Desire 826 smartphone with 2 GB RAM, octa-core CPU and Adreno 405 GPU. As discussed, the YOLOv3 and Tiny-YOLOv3 model were trained using the collected dataset. The models are trained with the dataset and initialized parameters as shown in Table 4-5.

Table 4-5. The initialization params of the YOLOv3 and Tiny-YOLOv3 models.

Model	input size	Batch size	Momentum	learning rate	Decay	Training steps
YOLOv3	416 × 416	16	0.9	0.001	0.0005	10000
Tiny-YOLOv3	416 × 416	16	0.9	0.001	0.0005	10000

The size of the input images is adjusted to 416 × 416 pixels. The training steps are set to 10000 to give better analysis, and 16 is used as batch size. After completing the training step, 100 test images for each class are used to verify the algorithm performance. Table. 4-6 shows the performance analysis of the YOLOv3 and Tiny-YOLOv3 models using the test set. As shown, an average accuracy of 94.6% is achieved for object detection using for YOLOv3 model while 52.6% for object detection using Tiny-YOLOv3 model. Also, 97.91% for recognition accuracy using YOLOv3 model while 95.12% for Tiny-YOLOv3 model. The results showed that YOLOv3 gives better accuracy for detecting and recognition. Furthermore, once the object is detected, the two models properly classify it among the list of object classes. From these results, YOLOv3 model gave better accuracy for detection.

Table 4-6. Performance analysis of the YOLOv3 and Tiny-YOLOv3 models.

Objects	Total Images	Correctly Detected		Correctly Recognized		Detection Accuracy (%)		Recognition Accuracy (%)	
		YOLOv3	Tiny YOLOv3	YOLOv3	Tiny YOLOv3	YOLOv3	Tiny YOLOv3	YOLOv3	Tiny YOLOv3
Chair	100	95	53	93	51	95	53	97.9	96.23
Table	100	94	51	92	49	94	51	97.9	96.08
Door	100	96	54	92	49	96	54	95.83	90.74
Person	100	91	50	91	49	91	50	100	98
Stairs	100	97	55	95	52	97	55	97.94	94.55
Average						**94.6**	**52.6**	**97.91**	**95.12**

The average detection time for both models is shown in Table 4-7.

Table 4-7. The average detection time of the two models in milliseconds.

Model	GPU	CPU
YOLOv3	9.8	2000
Tiny-YOLOv3	3.16	250

Results for detecting different object classes are shown below in Figure 4-14. YOLOv3 model can detect and recognize the object correctly.

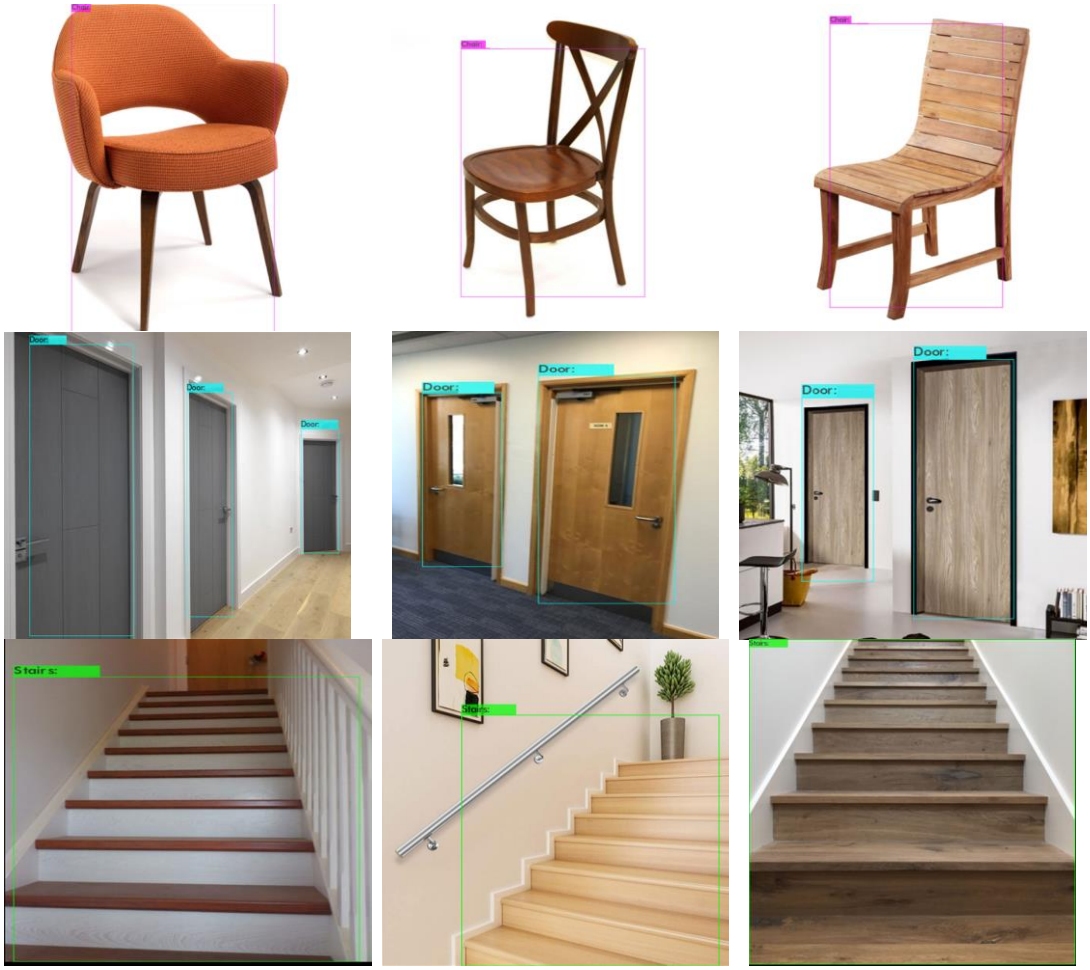


Figure 4-14. Results after object detection and recognition.

4.4 Conclusions

A navigation system has been built using CV which helped PVI navigate indoors easily. But, before using it, a map should be constructed for each floor in the building by sighted people. They should move around the building to find the points of interest such as labs and lecture rooms. Then, markers are printed and installed on the wall at those points. After that, they explore all the available paths to each interest point and calculate the number of steps between them. An internal map is created using a graph to save the interest points and the relation between them. Nodes in this graph represent the accurate position of the markers while edges are labelled with the number of steps and navigation instructions. All the situations and conditions that PVI may face during navigation were added. The system always tries to detect markers however, if it misses detecting markers in one frame, it will likely detect it successfully in one of the next frames. When any marker is detected, the system asks PVI to move towards it. So, in the sequence of images with the same marker when some of them are not recognized correctly and PVI still move towards it as mentioned before, then they will identify it correctly in the subsequent frames. If PVI find another marker and this marker is in the list of markers to the destination point, the system continues giving navigation commands from this marker to the destination point. However, if this marker is not on the list, the system starts to find the shortest path from that marker to the destination point. If the PVI move in a wrong direction such as going right instead of going left and find another marker, the system starts to find the shortest path from that marker

to the destination point. The number of steps that PVI take for moving from one marker to another have been counted which makes the system more accurate.

During testing, some problems have been discovered: sometimes PVI failed to understand the feedbacks so, the feedback was improved based on the comments of the PVI and found the audio feedback to be satisfactory. PVI move their hands rapidly during navigation which causes images to be captured with occlusion. Sometimes PVI cannot detect markers because they are moving their hands a lot and tags move out of the smartphone's camera view. Markers also can be captured with angles which cannot be detected correctly with the system. So, this was improved by installing eight markers with the same id at each interest point instead of adding only one as shown in Figure 4-8. This implementation makes detection easier and solved the problem of occlusion and decreased the chance for the markers to be outside of the camera view. It also helps PVI of different heights to detect markers easily. But markers cannot be detected from long distance and in challenging conditions.

An assistive system for PVI is proposed to detect and avoid objects independently where all feedback is provided to PVI in the form of audio. Images of objects are collected and manually labelled to create the dataset which is used to train the DL model. In this system, YOLOv3 and Tiny-YOLOv3 models were compared for detecting objects. The results showed that YOLOv3 gives better accuracy for detecting and recognition as the accuracy of 94.6% for object detection is achieved using YOLOv3 model while 52.6% is achieved for object detection for Tiny-YOLOv3 model. Also, 97.91% is achieved for recognition accuracy for YOLOv3 model while 95.12% for Tiny-YOLOv3 model. The results showed that Tiny-YOLOv3 model is faster than YOLOv3 model as it took an average of 3.16 milliseconds for recognition while YOLOv3 took 9.8 milliseconds using GPU. Tiny-YOLOv3 model took an average of 250 milliseconds for recognition while YOLOv3 took 2 seconds using CPU. So, YOLOv3 have been selected for the system. Future work will focus on minimizing recognition time for YOLOv3 and adding more objects in the dataset to make it more useful for PVI.

5 Detecting Markers from Longer Distances

Results from the previous chapter showed that Aruco markers were better than QR codes so, they are used as markers in the proposed system. An indoor navigation system has been built to help PVI in navigating safely inside buildings. But markers cannot be detected from long distances and in challenging conditions. So, this chapter aims to use a CNN model to solve this problem. The proposed system is compared with another one as a baseline where the results proved that it gives better accuracy.

5.1 System Architecture

In the previous chapter, an application has been built which works as the following way: At first, the application opens the camera to get a live stream of images. Then, it converts the image to grayscale and sends it to the desired library to detect and identify the marker. If it detects any markers, it gives feedback to the PVI using voice commands. However, this system failed to detect markers from long distance and in challenging conditions. The architecture of it has been modified by adding a CNN model as shown in Figure 5-1. The identification part of the problem has been converted to a classification one and CNN model has been used to identify the markers.

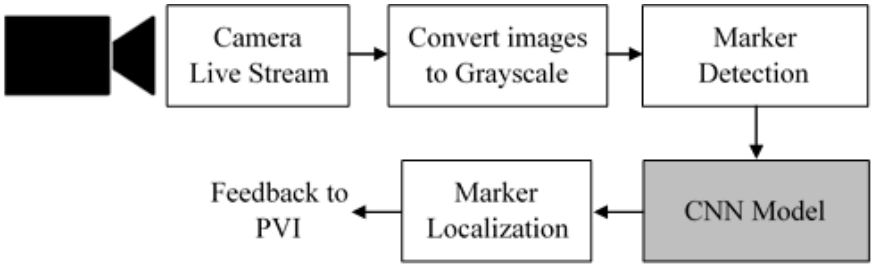


Figure 5-1. Main components of the application using CNN.

The architecture consists of two main units, the first unit marked in white is used to extract markers from captured frames, while the second one is marked in grey and used for classification. The most important part of the first unit is the marker detection and the main steps for it are depicted in Figure 5-2. Given the input image as shown in Figure 5-2.-a, after converting it to grayscale, several steps have been taken to prepare the input candidates for the CNN model:

- **Image segmentation** Aruco markers are used for the proposed system. They have black borders surrounded by white space. So, segmentation is used to separate these borders from the white background. A local adaptive method is used to do it, where the mean intensity value *m* is computed for each pixel. If the intensity is greater than the difference between *m* and a constant value *c*, the pixel intensity is set to zero. Figure 5-2.-b shows the results after applying image thresholding using an adaptive method.
- **Contour extraction** After making a segmentation, the next step is to extract the existing contours from the image thresholding using algorithms like Suzuki and Abe. The extracted contours are with different shapes where most of them are irrelevant background elements. Figure 5-2.-c shows the results after extracting contours [107].
- **Contour filtering:** The first task is to remove contours that are too small as they are not the candidate markers. The next step is to filter the remaining contours and select ones

that form a four-corner convex polygon. The last step is to send the selected candidates to the proposed model for classification. Figure 5-2.-d, shows the filters' contours which are sent to the proposed model [108].

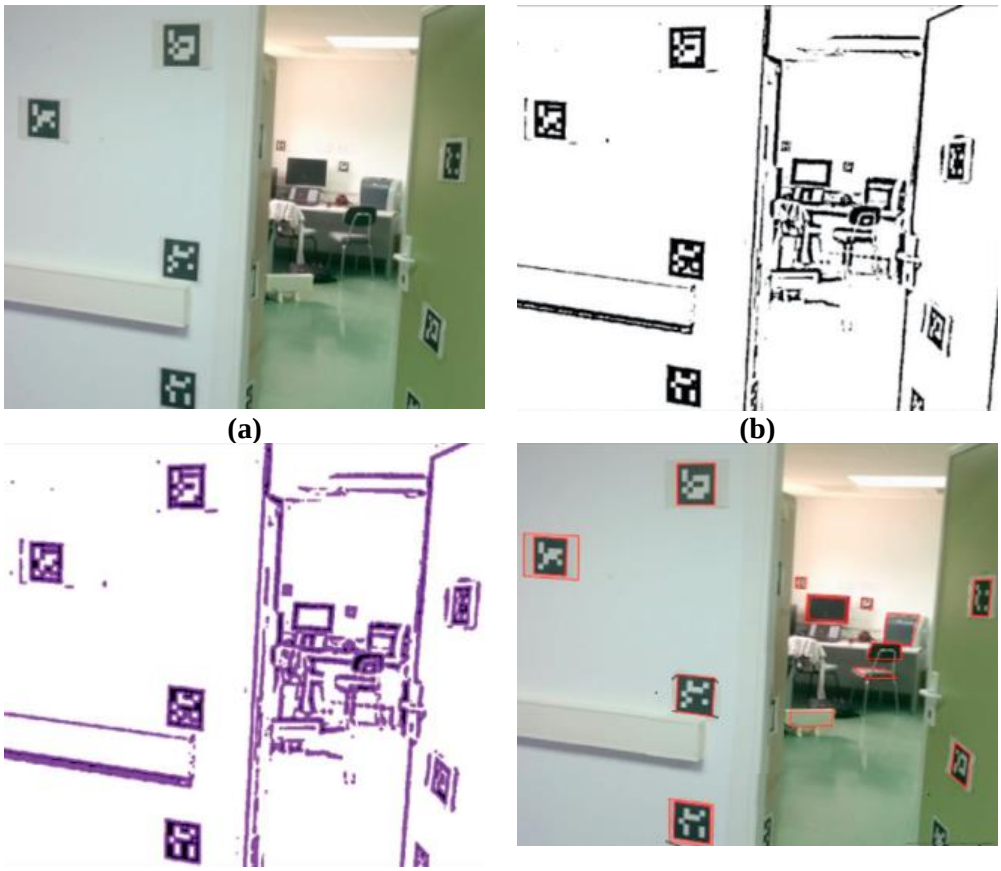


Figure 5-2. The process of detecting markers.

Following the image conversion to greyscale and detecting markers, a CNN model is used to identify them. This model returns the correct ID of detected markers or if it fails to do so, it determines that no marker is available and continue with the next image. So, Figure 5-3. Flowchart of the marker identification process in the navigation system which works the following way:

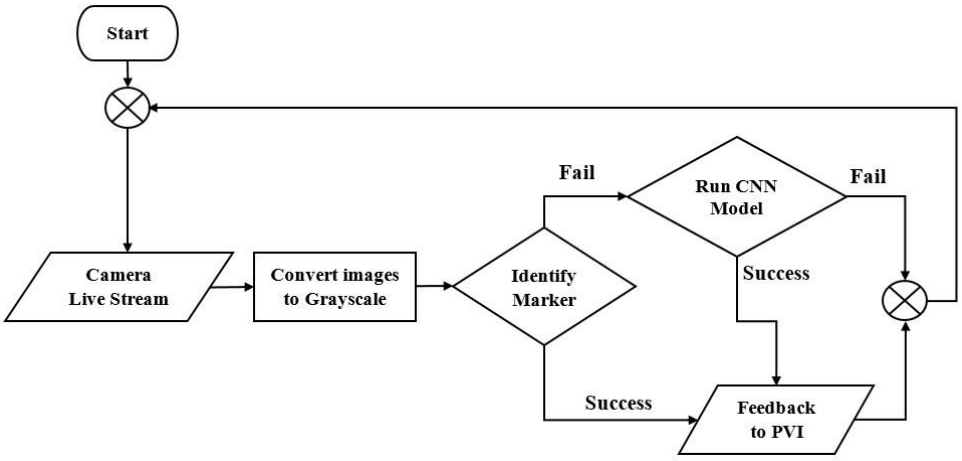


Figure 5-3. Flowchart of detecting markers and giving feedback to PVI.

While receiving a real-time stream of images from the smartphone camera, images are converted to grayscale ones and sent to the prototype for detecting and identifying markers. If any marker is identified, voice feedbacks are given to the PVI. If it fails to detect markers, the image is given to the CNN model to identify markers. This model processes the image and returns the correct id if any maker is detected. However, if it fails to do so, it decides that no marker is available and continues processing the next image. This process is repeated until the PVI reach their destination.

5.1.1 Dataset

In a typical marker-based application, image frames are affected by various noisy conditions such as blurring and distortion of markers, etc. Such noises affect the accuracy of marker identification. However, it is hard to get a large dataset of markers in several conditions from real images. So, transformations are applied to the original images to enhance and provide the dataset with several images in different conditions. These transformations are used to imitate real situations such as those shown in Figure 5-4. As result, a labeled dataset is created corresponding to each class by maintaining the challenging conditions. Seven markers are used only for evaluation to simplify the problem. So, the dataset has 29 classes (7 markers × 4 + 1), e.g., 4 different orientations (0°, 90°, 180°, 270°) with respect to 7 markers and an additional class to denote null or no marker class. For each class, 500 samples are generated. So, a total of 14,000 images were created for the 28 classes. To identify that no markers were visible in the image captured by the camera, the null class was assigned with images taken from a dataset [81][109]. 75% of the input samples from each class were used for training and the rest for validation.

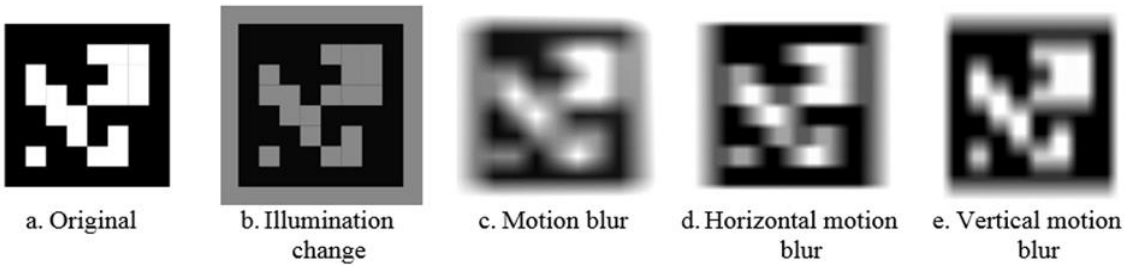


Figure 5-4. A few samples of makers with illumination change and motion blur.

5.1.2 Proposed CNN Model

CNN automatically learns the most efficiently from raw data instead of using traditional machine learning techniques. A typical CNN consists of two basic building blocks. The first block is the convolution block which consists of a repeated occurrence of a cascaded convolution layer and a pooling layer. This first block is called Feature Extraction Unit (FEU) and is used for feature extraction. The second block consists of a flattening layer and a fully connected neural network. The main role of this second part is for classification tasks using features from the convolutional part as shown in Figure 5-5.

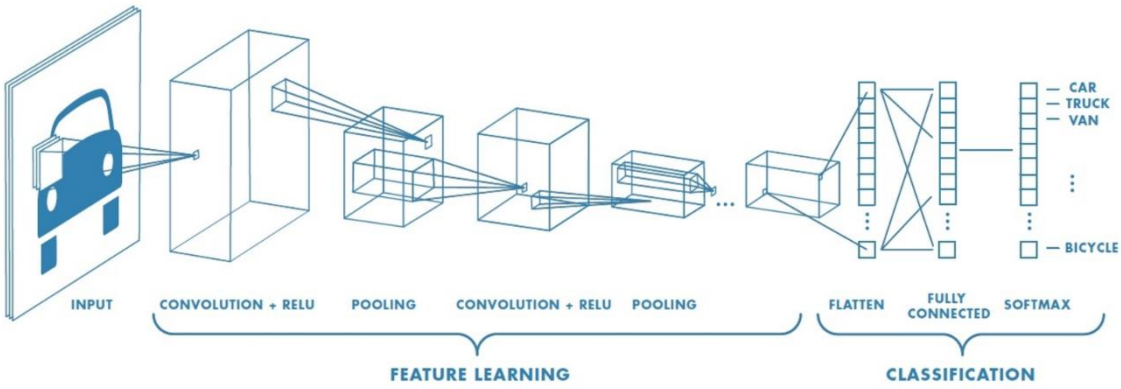


Figure 5-5. The basic layers of CNN

Convolutional Layer: contains a series of filters known as convolutional kernels. The filter is a matrix of integers that are used on a subset of the input pixel values, the same size as the kernel. Each pixel is multiplied by the corresponding value in the kernel, then the result is summed up for a single value for simplicity representing a grid cell, like a pixel, in the output channel/feature map. For simplicity, a greyscale image that has one channel and a 3x3 convolutional kernel are used. The kernel strides over the input matrix of numbers moving horizontally column by column, sliding/scanning over the first rows in the matrix containing the images pixel values. Then the kernel strides down vertically to subsequent rows as shown in Figure 5-6.

0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

Kernel

0	-1	0
-1	5	-1
0	-1	0

114	328	-26	470	158
53	266	-61	-30	344
403	116	-47	295	244
108	-135	256	-128	344
314	346	279	153	421

Figure 5-6. The convolution of a filter over a 2D image

Finally, a Rectified Linear Unit (RELU) is applied to all the cells of all the output-matrix. It used for transforming the summed weighted input from the node into the activation of the node or output for that input. The basic intuition to derive from here is that, after convolution, if a particular convolution function results in '0' or a negative value, it implies that the feature is not present there and denote it by '0' and keep the value for all the other cases. With all the operations and the functions applied on the input image, the first part of the convolutional block is formed.

Pooling Layer: performing the process of extracting a particular value from a set of values, usually the max value or the average value of all the values. This reduces the size of the output matrix. For example, for max-pooling, the max value among all the values of saying a 2 X 2 part of the matrix is taken. Thus, the values denoting the presence of a feature in that section of the image are taken. So, this way is getting rid of unwanted information regarding the presence of a

feature in a particular portion of the image and considering only what is required to know. It is common to insert a pooling layer in-between successive convolutional blocks in a CNN architecture. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network.

Flattening Layer: This layer forms the first part of the second block of the CNN architecture which is used for the classification task. After multiple convolution layers and down-sampling operations, the representation of the image is converted into a feature vector using this layer as shown in Figure 5-7.

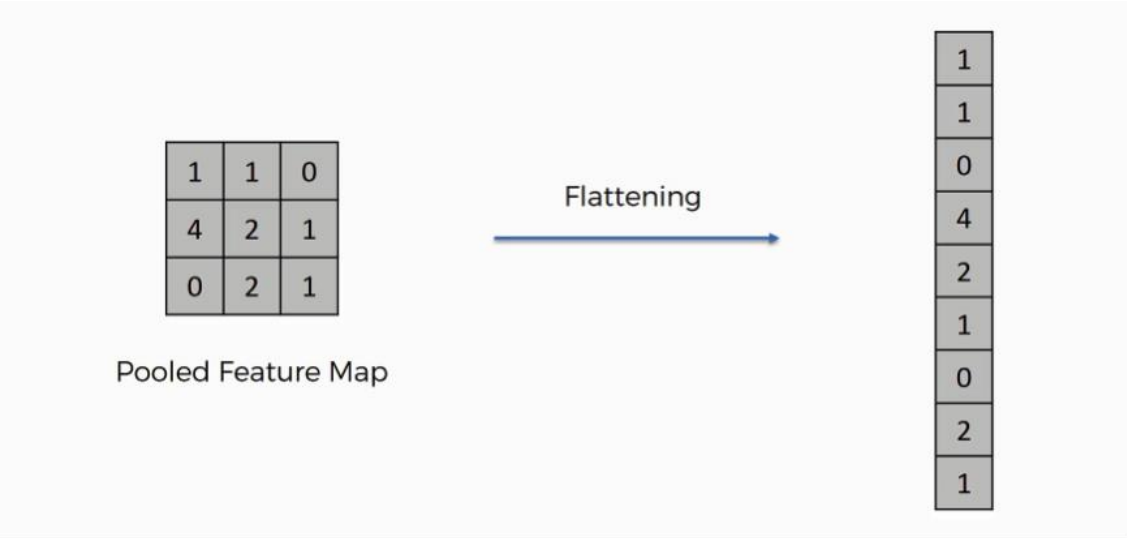


Figure 5-7. The flattening operation.

Fully Connected Layer: In this step, the flattened feature map is passed through a neural network. This step is made up of the input layer, the fully connected layer, and the output layer. The fully connected layer is like the hidden layer in ANNs but in this case, it's fully connected. The output layer is where to get the predicted classes. The information is passed through the network and the error of prediction is calculated. The error is then backpropagated through the system to improve the prediction. The final output produced by the dense layer neural network does not usually add up to one. However, these outputs must be brought down to numbers between zero and one, which represent the probability of each class. This is the role of the Softmax function. The output of this dense layer is therefore passed through the Softmax activation function, which maps all the final dense layer outputs to a vector whose elements sum up to one.

The way this fully connected layer works is that it looks at the output of the previous layer and determines which features most correlate to a particular class. For example, if the program is predicting that some image is a dog, it will have high values in the activation maps that represent high-level features like a paw or 4 legs, etc. Similarly, if the program is predicting that some image is a bird, it will have high values in the activation maps that represent high-level features like wings or a beak. A fully connected layer looks at what high-level features most strongly correlate to a particular class and has weights so that when you compute the products between the weights and the previous layer, you get the correct probabilities for the different classes. Figure 5-8 shows the structure of the CNN used in the proposed system. Three FEUs are used in the proposed model. In the first FEU, a convolutional layer with a dimension of $20 \times 5 \times 5$ and a sigmoid activation function are used. Then, max-pooling with a pooling filter size of 2 and stride

of 2 is used to down sample the outputs by a scale factor of 2. The second and the third FEU are of the same structure as the first one with a slight variation. A convolutional layer with a dimension of 50 × 5 × 5 in the second FEU is used, while a convolutional layer with a dimension of 200 × 5 × 5 is used in the third one.

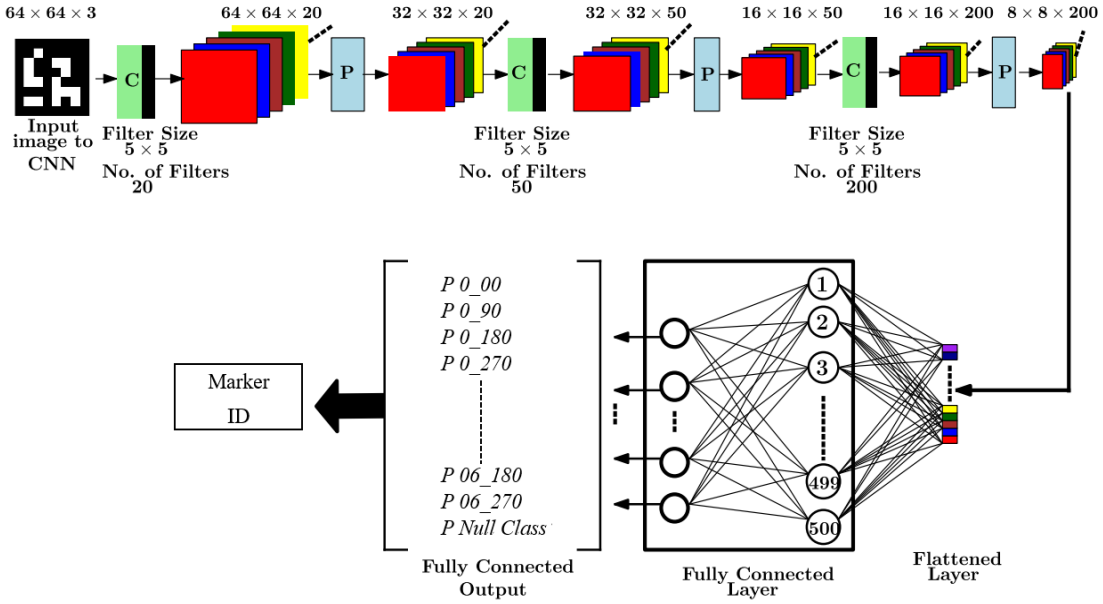


Figure 5-8. The proposed CNN architecture used in training of markers.

For classification, the output of the last feature extraction unit is converted into a vector through flattening and is provided as an input to the fully connected layer. The output of the fully connected layer is converted into probabilities corresponding to 29 classes of the classifier. During the training stage, the Adam optimizer with a learning rate of 0.001 has been used over the cost function to determine the optimum value of the weight parameters. The results showed that Aruco markers can be detected from longer distances than before. However, this model still failed to detect markers under occlusive conditions. Moreover, the time required for this model to detect markers should be minimized.

5.1.3 Simplified CNN Model

The results showed that Aruco markers can be detected in adverse conditions, albeit the execution time is not suitable for real-time usage. To minimize the response time, a simplified version of proposed CNN model is used with the same parameters for training and validation. As shown in Figure 5-9, two FEUs in the simplified model are used. In the first FEU, a convolutional layer with a dimension of 20 × 5 × 5 and a sigmoid activation function; then Max pooling was used with a pooling filter size of 2 and stride of 2 to downsample the layer's outputs by a scale factor of 2. In the second FEU, a convolutional layer with a dimension of 200 × 5 × 5 was used. Dropout layers are placed to prevent an overfitting problem. The simplified model can detect Aruco markers very well in adverse conditions and the execution time is minimized to be suitable for real-time identification of markers. However, this model still failed to detect markers under occlusive conditions.

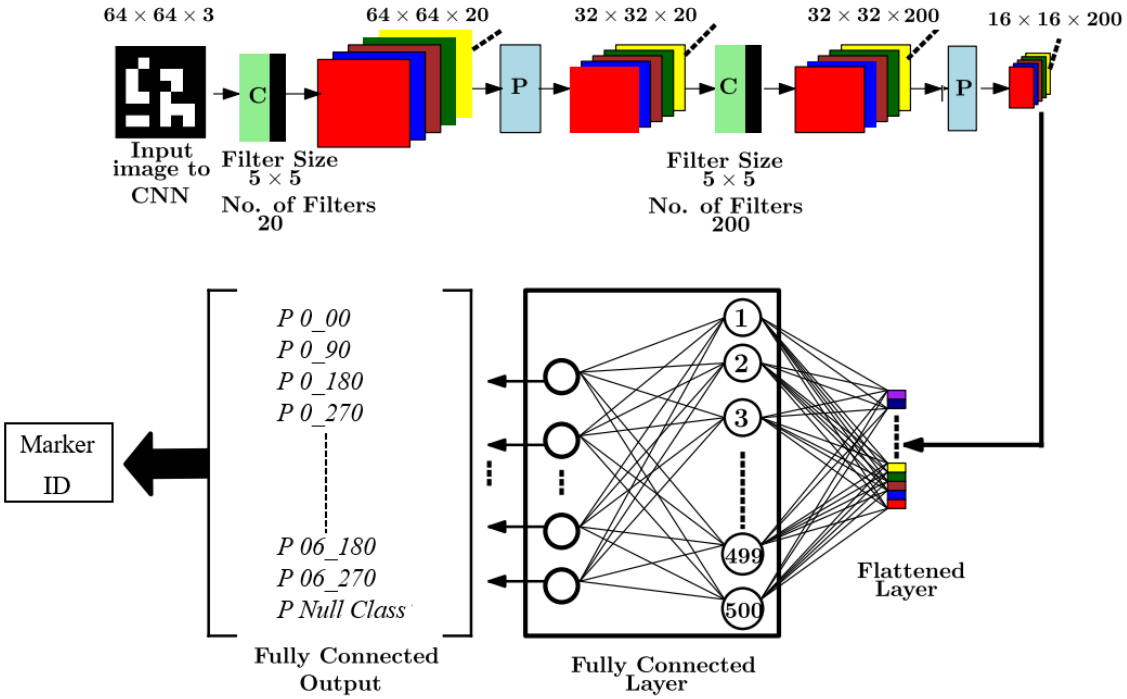
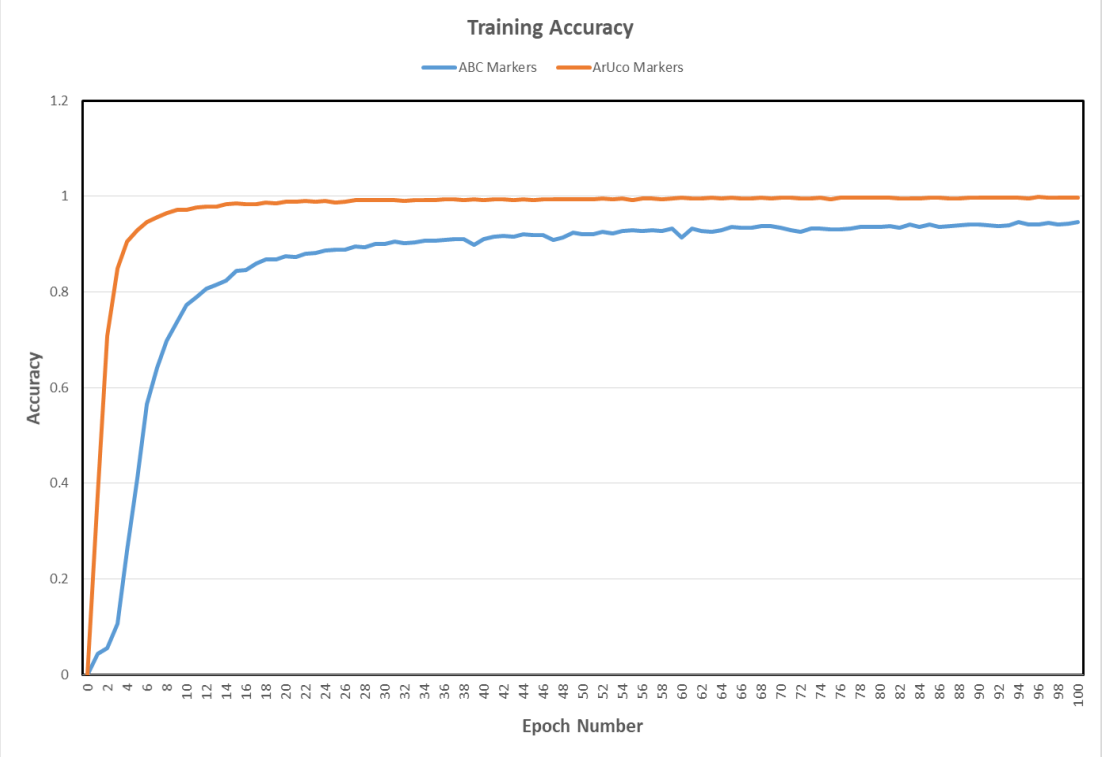


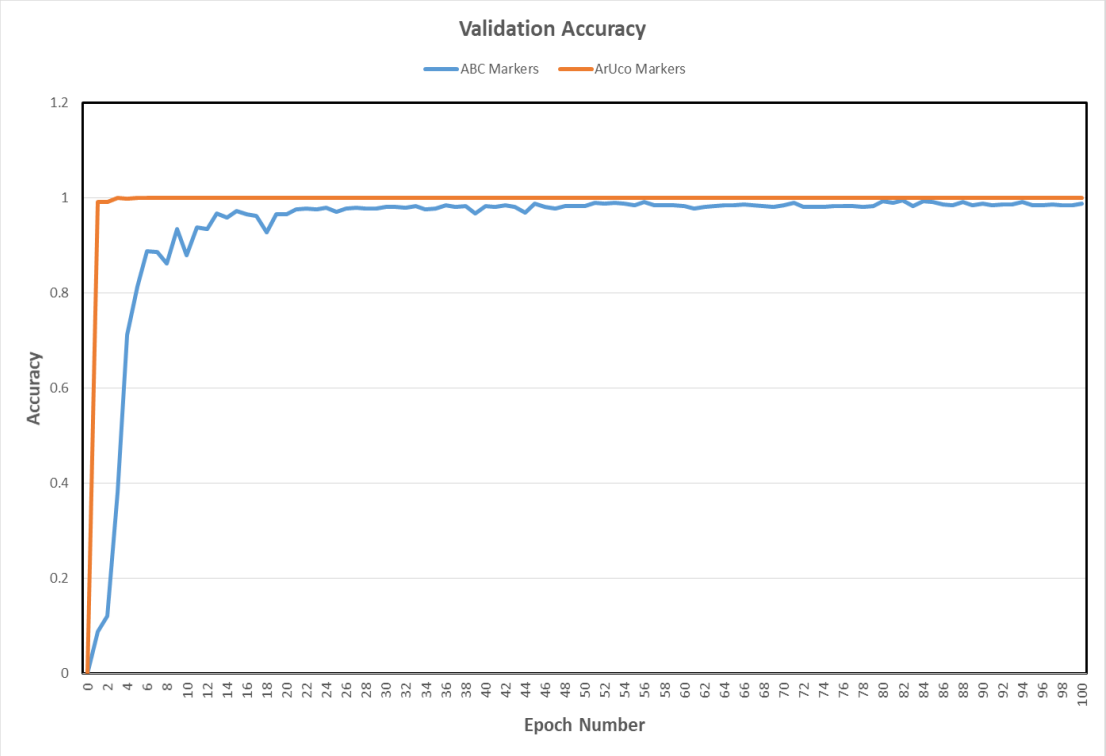
Figure 5-9. The proposed simplified CNN architecture.

5.2 Evaluation

As discussed, the CNN model is trained using the created dataset. The training was conducted in batches with size of 32. The Keras framework for deep learning was used for implementing the CNN model. The original size of the markers in the synthetic data set were 512 × 512. For training the CNN model, the original images were resized to 64 × 64 and were given as an input to the first convolution layer. The entire dataset was divided into two parts: 75% for training and 25% for validation. Training and validation datasets were used to determine the layer parameters during the training phase of the model. Additionally, the model was tested with new data that had not been encountered before. The test performance of the model was carried out on the trained model. The output was produced with 29 class values. For all the experimental results, the training phase of the CNN model was carried out for 100 epochs. Experimental results were obtained for all classes. The model was trained with the dataset proposed to find which marker produces better results. The training and validation plots for accuracy are shown in Figure 5-10 where the results of our model on our dataset is drawn in orange and the other dataset is drawn in blue: (a) results of the training accuracy of the two datasets; (b) results of the validation accuracy of the two datasets. The training and validation plots for loss are shown in Figure 5-11 where the results of our model on our dataset is drawn in red and the other dataset is drawn in blue: (a) results of the training loss of the two datasets; (b) results of the validation loss of the two datasets.

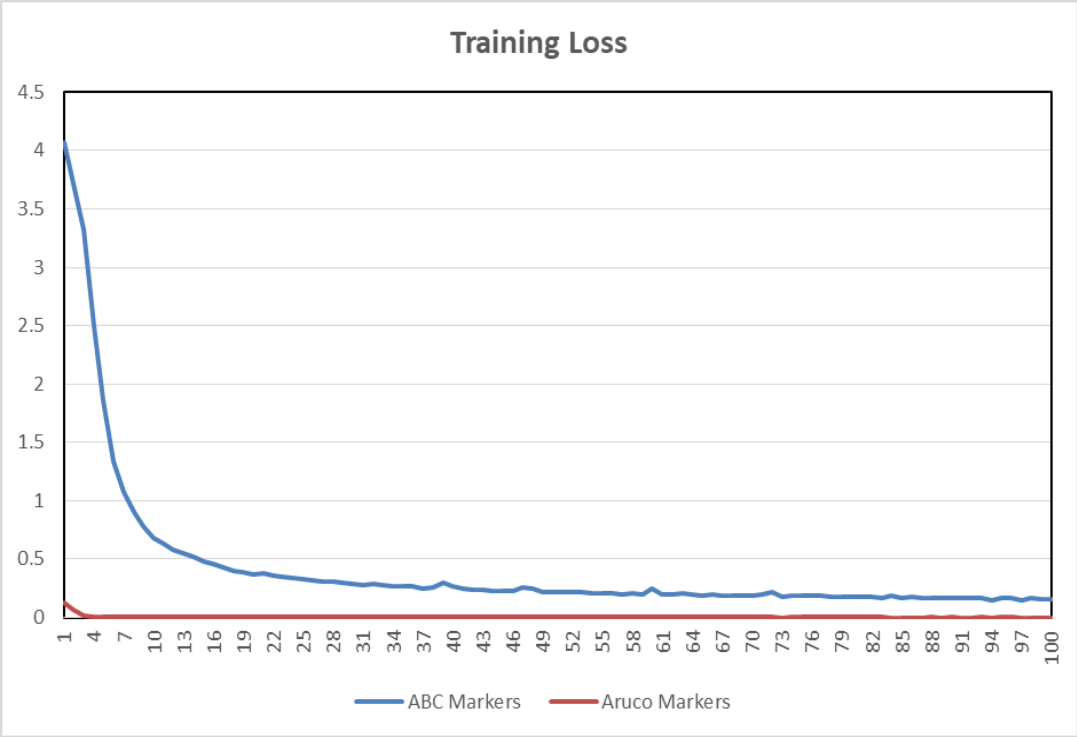


(a)

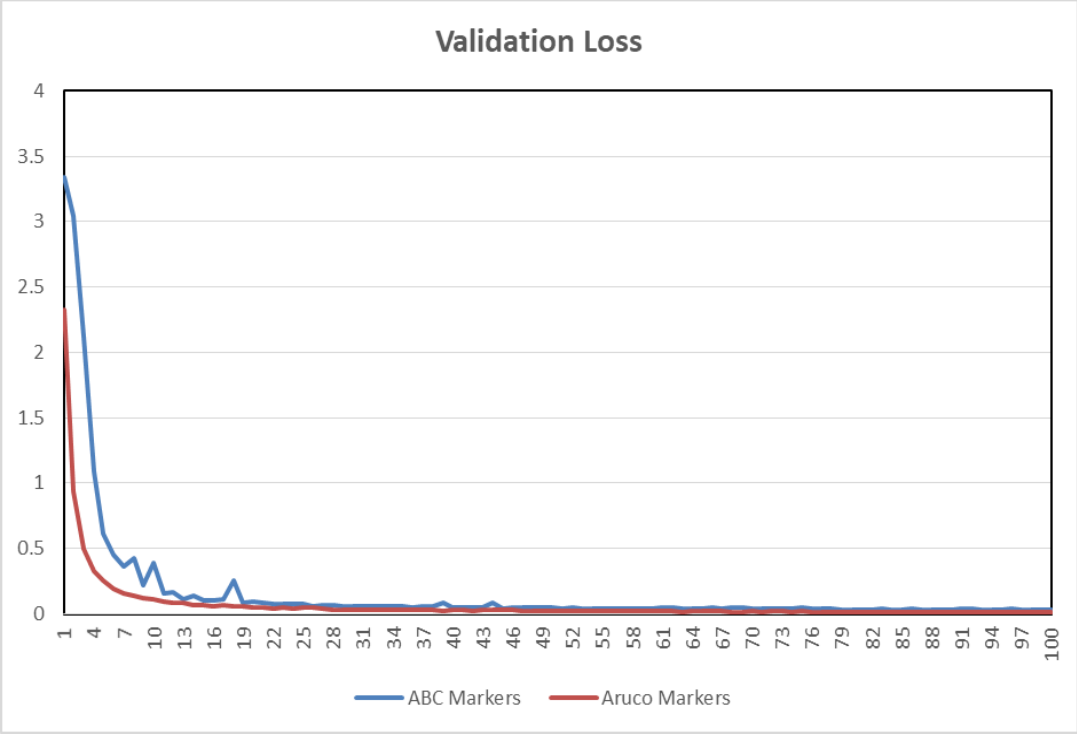


(b)

Figure 5-10. Comparative accuracy graphs after applying the model on two datasets.



(a)

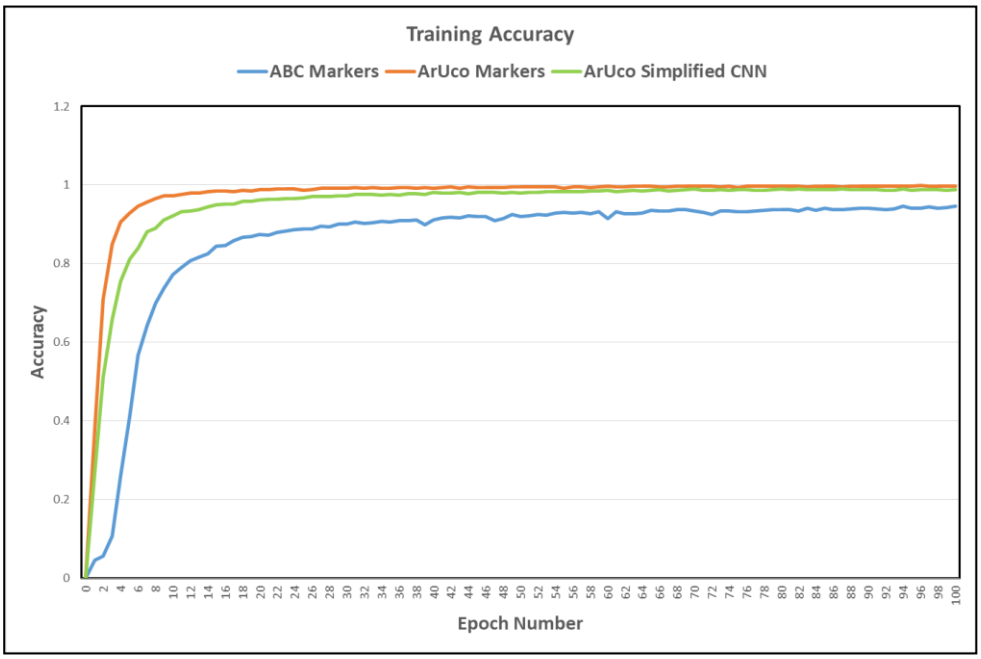


(b)

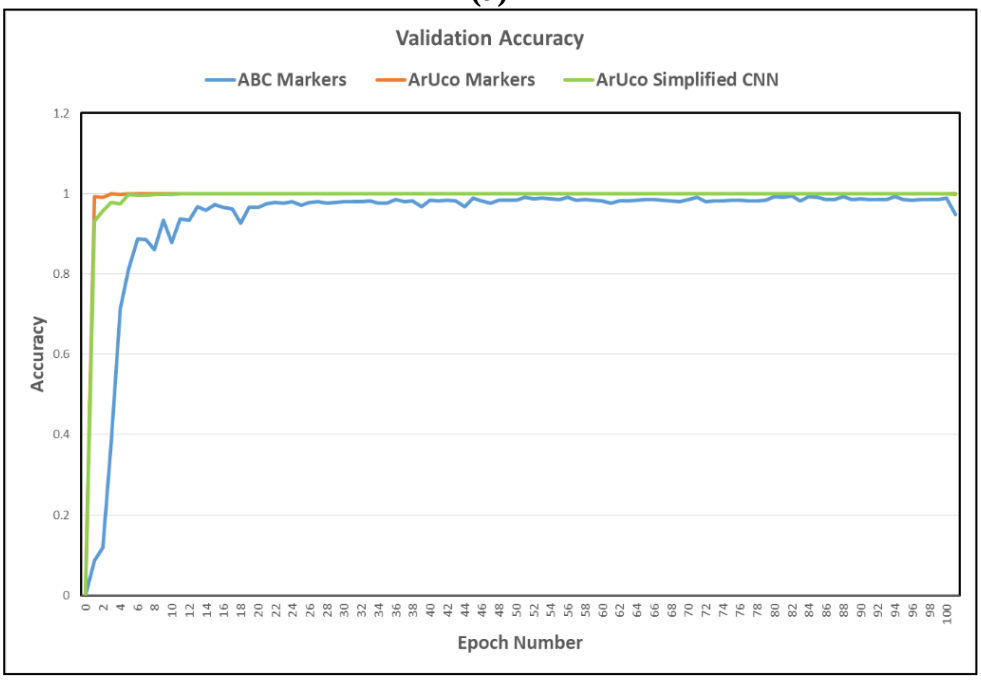
Figure 5-11. Comparative loss graphs after applying the model on two datasets.

The results show that the proposed model can detect Aruco markers very well in challenging conditions as it provides approximately 97% accuracy for training and 99.97% accuracy for testing. When applying it to the other dataset, it gave 86% for training accuracy and 94.74% for testing accuracy. From these results, the detection in challenging conditions is improved and the detection of markers from a long distance is also improved. However, this model still failed to detect markers under occlusive conditions. Moreover, the time required for this model to detect

markers should be minimized. The same parameters and data of the CNN model are used for training, validating, and testing. Figure 5-12 and Figure 5-13 show the training and validation accuracy graphs of the proposed model. Comparative accuracy and loss graphs after applying the first model to the dataset which is drawn in red and applying the simplified models to the two datasets where the dataset is drawn in green, and the other dataset is drawn in blue. As shown, The training and validation accuracy of the simplified model is better than the accuracy of the model proposed in [81]. It also shows that the training and validation curves of two proposed models are close to each other.

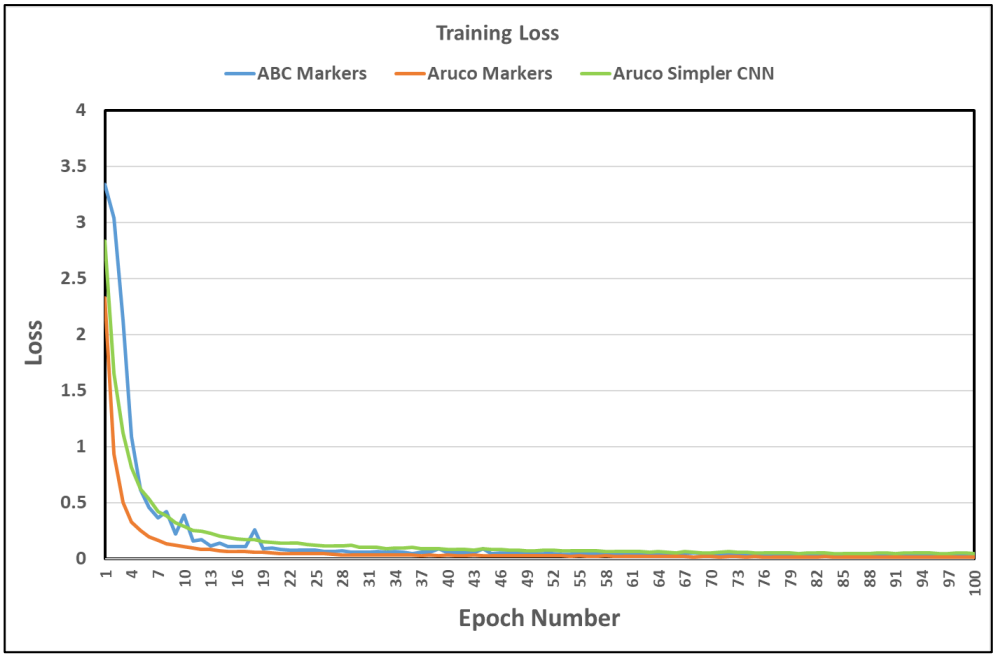


(a)

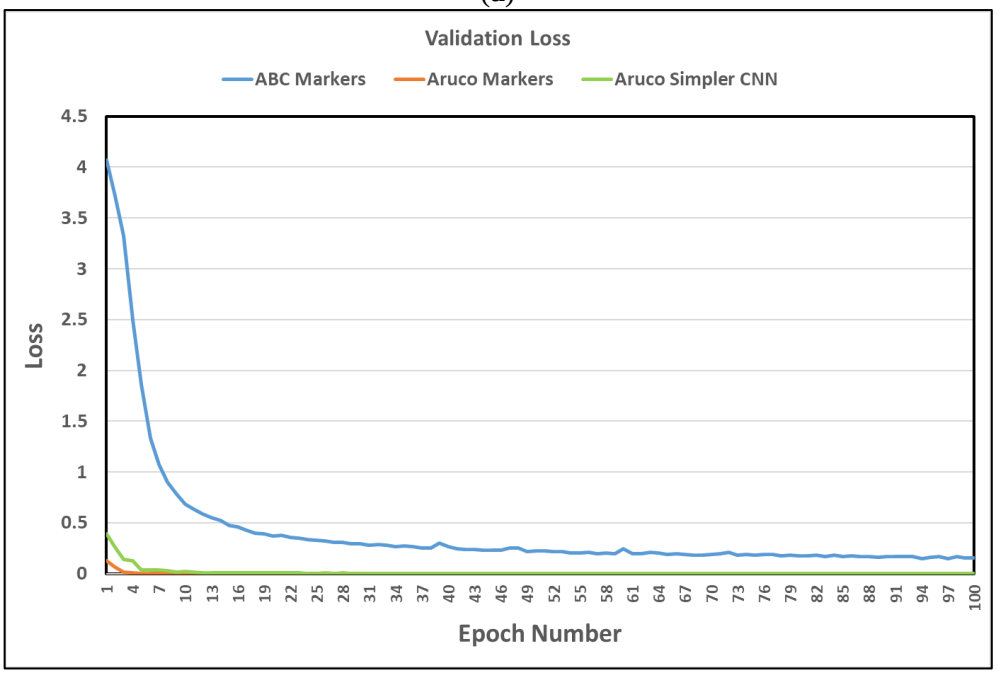


(b)

Figure 5-12. Comparative accuracy graphs for the three models.



(**a**)



(**b**)

Figure 5-13. Comparative loss graphs for the three models.

The execution time for detecting markers in the simplified model is better than the complex one. Thus, this model is suitable for real-time identification of markers. I have simplified the convolutional layers of the CNN model and used the same parameters for training and validation. The simplified model can detect Aruco markers very well in challenging conditions as it gives approximately 95.5% accuracy for training and 99.82% accuracy for testing. Furthermore, the training and testing accuracy of the simplified model is better than the accuracy of the other dataset. The training and testing curves of the proposed models are also close to each other. The execution time for detecting markers in the simplified model is better than the complex one. So, this simplified model minimized the response time.

5.3 Conclusions

The goal was to design a navigation system for PVI using makers. The evaluation showed that Aruco markers can be detected from longer distances and in adverse conditions using CNN and the time for identification can be minimized by simplifying the CNN. However, this work has only dealt with the identification steps, while detection has been done using a method based on image thresholding and rectangle extraction. CNN's models such as YOLO can be used to fully perform the detection and identification steps.

6 Detecting markers in challenging conditions using YOLOv3

In the previous chapter, a navigation system has been designed for PVI using makers. The system has been improved to detect markers from longer distances. However, this work has only dealt with the identification steps, while detection has been done using a method based on image thresholding and rectangle extraction. In this chapter, a YOLO DL model is used to detect markers in challenging condition and fully perform both detection and identification steps. Therefore, different modified versions of Tiny-YOLOv3 model are proposed to improve the detection accuracy of the original Tiny-YOLOv3 and reduce the execution time to make it suitable for real-time usage. The main contributions are the following:

- Proposal of a novel model to improve detecting markers in different challenging situations based on Tiny-YOLOv3.
- Several modified versions of the original model were implemented and compared to improve detection accuracy. These models were tested using real test cases. They provided high accuracy and very good performance in detecting markers.

6.1 Detecting markers using deep learning models

The proposed system failed to detect markers in challenging conditions. Therefore, a simplified model of YOLOv3 called Tiny-YOLOv3 is used to improve the real-time performance by reducing the number of the convolutional layers. So, the original Tiny-YOLOv3 model is used first to detect Aruco markers. Then, it has been modified to improve feature extraction and detection accuracy. So, detecting markers in the navigation system works the following way: While receiving a real-time stream of images from the smartphone camera, images are converted to grayscale ones and sent to the prototype for detecting and identifying markers. If any marker is detected, voice feedbacks are given to the PVI. If it fails to detect markers, the image is given to the deep learning model to detect markers. This chapter proposes a modified Tiny-YOLOv3 version, which processes the image and returns the correct id if any maker is detected. However, if it fails to do so, it decides that no marker is available and continues processing the next image. This process is repeated until the PVI reach their destination. Also, the following hypotheses are made:

H1: The modified versions of the original Tiny-YOLOv3 model will improve the detection accuracy.

H2: The modified versions of the original Tiny-YOLOv3 model will lower the execution time.

6.1.1 Original Tiny-YOLOv3 model

Tiny-YOLOv3 model is a CNN that accepts images as an input. It consists of two main blocks: feature extractor and detector. During training, the original Tiny-YOLOv3 model uses the same loss function used by YOLOv3, which consist of four parameters: (1) position of the prediction frame (x, y); (2) the prediction frame size (w, h); (3) the prediction class *class*; (4) the prediction confidence *con*.
The loss function of the original Tiny-YOLOv3 model is shown in equation (1):

$$loss = \frac{1}{n}\sum_{k=0}^{n} loss_{xy} + \frac{1}{n}\sum_{k=0}^{n} loss_{wh} + \frac{1}{n}\sum_{k=0}^{n} loss_{class} + \frac{1}{n}\sum_{k=0}^{n} loss_{con} \quad (6\text{-}1)$$

where *n* is the total number of the trained targets, and the loss function in this equation (6-1) is calculated for each parameter as shown in the following equations:

$$loss_{xy} = OM \times (2 - w \times h) \times BCE\left(true_{xy}, pred_{xy}\right) \quad (6\text{-}2)$$

$$loss_{wh} = OM \times (2 - w \times h) \times 0.5 \times square\left(true_{wh}, pred_{wh}\right) \quad (6\text{-}3)$$

$$loss_{class} = OM \times BCE\left(true_{class}, pred_{class}\right) \quad (6\text{-}4)$$

$$loss_{con} = OM \times BCE(OM, PM) + (1 - OM) \times BCE(OM, PM) \times IM \quad (6\text{-}5)$$

where *OM* is the point of the object; *w* and *h* represent the width and height of the prediction box respectively; *BCE* is a binary cross entropy function and *square* is a function of variance; $pred_{xy}$ is the predicted position while $true_{xy}$ is the actual target position; $pred_{wh}$ is the size of the prediction frame size while $true_{wh}$ is the size of actual ground truth box; $true_{class}$ and $pred_{class}$ are the actual target class and prediction class respectively; *PM* is the predicted object point; *IM* is related to Intersection Over Union (IOU) which is used for measuring the detection accuracy of corresponding objects in the dataset and is calculated using equation (6-6). If IOU is less than the specified threshold, *IM* is 0.

$$\text{IOU} = \frac{\text{TP}}{\text{FP+TP+FN}} \quad (6\text{-}6)$$

Where TP represents true positive, FP is the false positive and FN means false negative.

6.1.2 Modified Tiny-YOLOv3 models

To improve the accuracy of detection, there are two ways to change the original Tiny-YOLOv3. The first one is to change the feature extraction part by increasing or decreasing the depth of the network. The second one is to change the detection part by adding more extra branches of the detector, which generates the bounding boxes and class information. In this chapter, the original Tiny-YOLOv3 has been modified several times by changing the feature extraction and detection parts to improve the detection accuracy. The results of these modifications are three modified versions of the original Tiny-YOLOv3 model.

6.1.2.1 First version

In this version, model accuracy can be improved by changing the feature extraction part while the detection part was kept the same without any modification. They increased the depth of the network by adding residual network structures between the layers of the original model. The roles of the added layers are to extract more features from the target and reduce information loss. The residual network uses 1x1 and 3x3 convolutional layers to extract features. The feature map of the fourth convolution layer is concatenated to the feature map generated after adding the residual structure. Then, the output is transmitted to the fifth convolution layer to extract features. This structure is repeated as shown in Figure 6-1. The red parts are the added residual network structures to the original Tiny-YOLOv3 model.

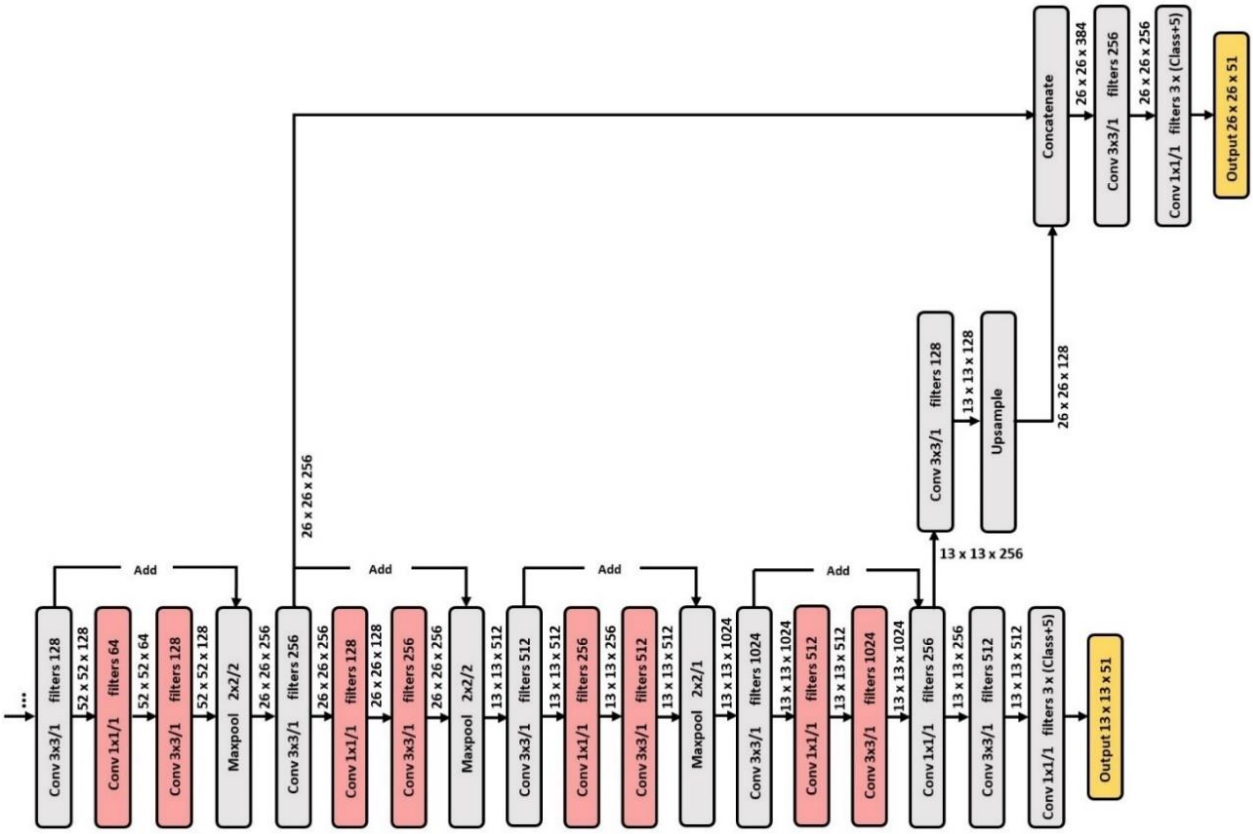


Figure 6-1. The architecture of the first modified version of Tiny-YOLOv3.

6.1.2.2 Second version

The input images are down-sampled by the Tiny-YOLOv3 original model until reaching the first detection layer, where the prediction is performed at the first scale with stride 32 and 13x13 scale. Then, the output of one of the layers is up-sampled by a factor of two and concatenated with the output from one of the previous layers. The up-sampling is a layer without any weights and is used to double the dimensions of input. Finally, the output is used for predication on the second scale with stride 16 and 26x26 scale. This concept is used to build the second modified version of the Tiny-YOLOv3 model with predictions across three different scales. The model makes detection at feature maps of three different sizes using strides 32, 16, 8 and detection are made on scales 13 × 13, 26 × 26, 52 × 52. Figure 6-2. shows the architecture of the second modified version of Tiny-YOLOv3 model. The output of the convolution layer is up-sampled and concatenated with the output from the fourth layer. Then, this output is used for the third prediction on a 52x52 scale with stride 8. This modification enriches high-level features that are important to detect small objects.

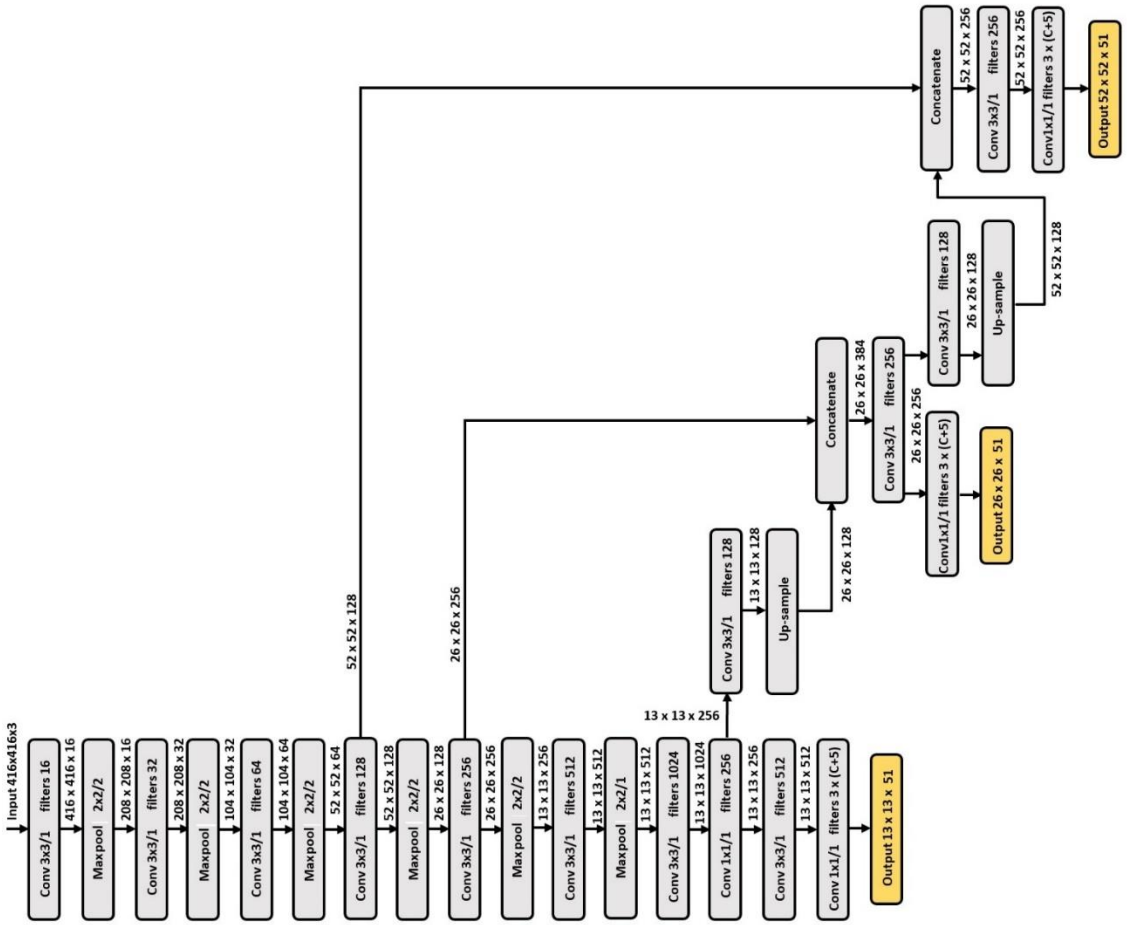


Figure 6-2. The second modified version of the original Tiny-YOLOv3.

6.1.2.3 Third version

As explained, the first modified Tiny-YOLOv3 version works on modifying the feature extraction part to improve accuracy. Also, the second modified version works on modifying the detection part by adding prediction at a third scale. The two architectures are combined to make the third modified Tiny-YOLOv3 version, as shown in Figure 6-3.

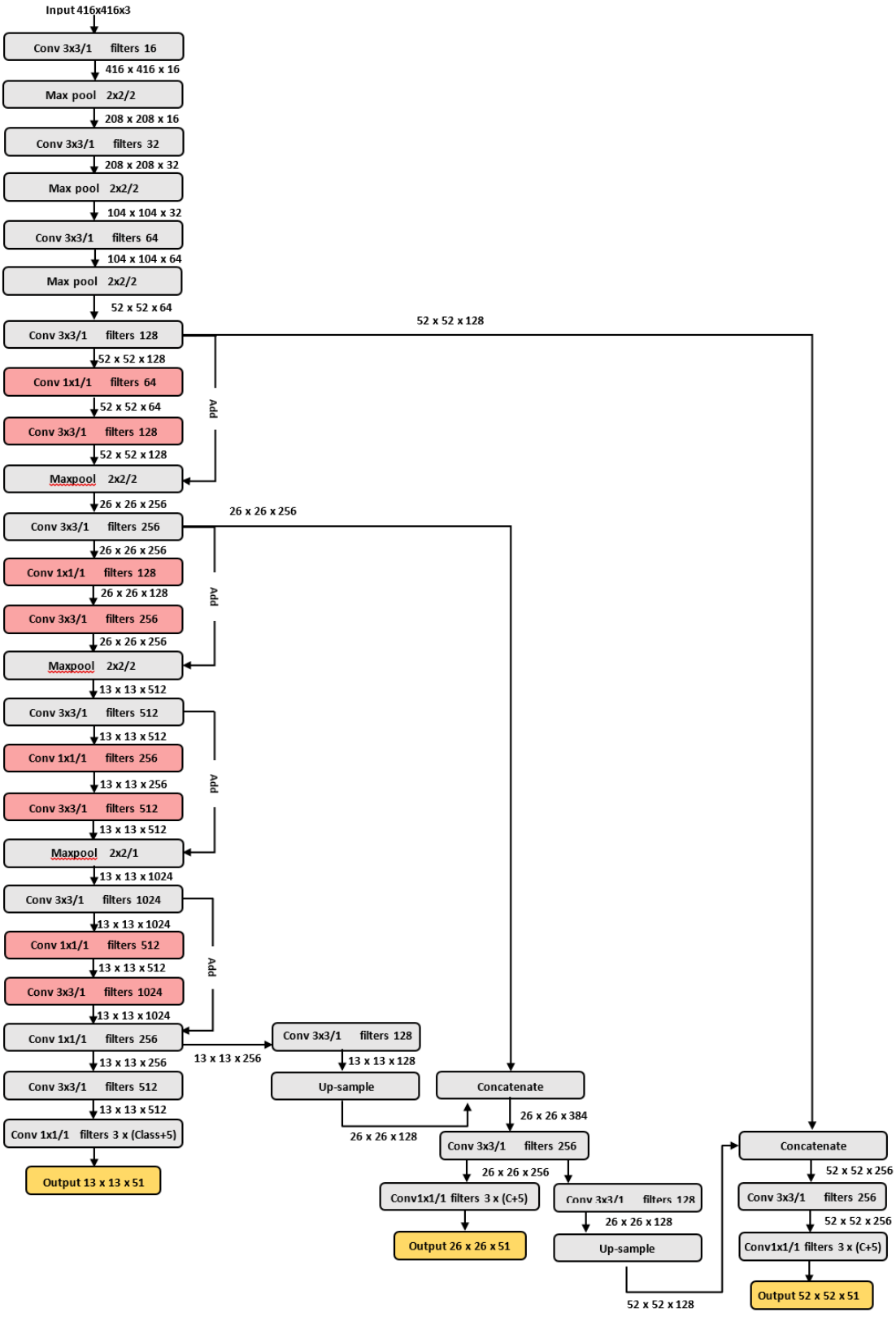


Figure 6-3. The network structure of the modified version 3.

6.2 Experiments

The system was evaluated in three steps: First, training the marker detection model requires a lot of resources. Therefore, Google Colab is used which leverages the power of free GPU for training the dataset quite easily. Second the performance of the proposed models was evaluated using videos on a DELL INSPIRON N5110 computer with Intel Core i7-2630 QM 2.00 GHz CPU, 6 MB cache, quad-core, and 8 GB RAM. Finally, the model was uploaded and evaluated using HTC Desire 826 smartphone with 2 GB RAM, octa-core CPU and Adreno 405 GPU.

6.2.1 Dataset

Images were collected from the testing environment using a smartphone camera. The dataset used twelve classes to represent twelve markers for the interest points on the map. For each marker, the authors used 600 images; 300 were captured from long distances between the camera and markers, while the other 300 were taken from short distances. Then, these 600 images were expanded to 7,200 using techniques such as rotation, blur, and lighting effects to improve the detection accuracy of the neural network. To achieve this, the original images were rotated by 90, 180, and 270 degrees. The reason for rotation is to represent holding mobile in different angles. After that, images were blurred to simulate real situations such as incorrect focus or camera movement, and finally, some lightning effects were applied to simulate corridors lighting which improves detection accuracy. The result is a total of 86,400 images for all markers; 57,600 images for training and validation while the remaining 28,800 images were used for testing. It is desirable to split the dataset into training, validation, and testing sets in a way that preserves the same proportions of examples in each class as observed in the original dataset. Training, validation, and testing sets are generally well selected to contain carefully sampled data that spans the various marker classes that the model would face when used in the real world. Finally, manual annotation was applied where bounding boxes were drawn, and categories were classified manually. Figure 6-4 showed examples from the dataset used under challenging conditions. (a) Ideal conditions. (b) Lighting conditions. (c) Motion blur. (d) Rotation with Motion blur.

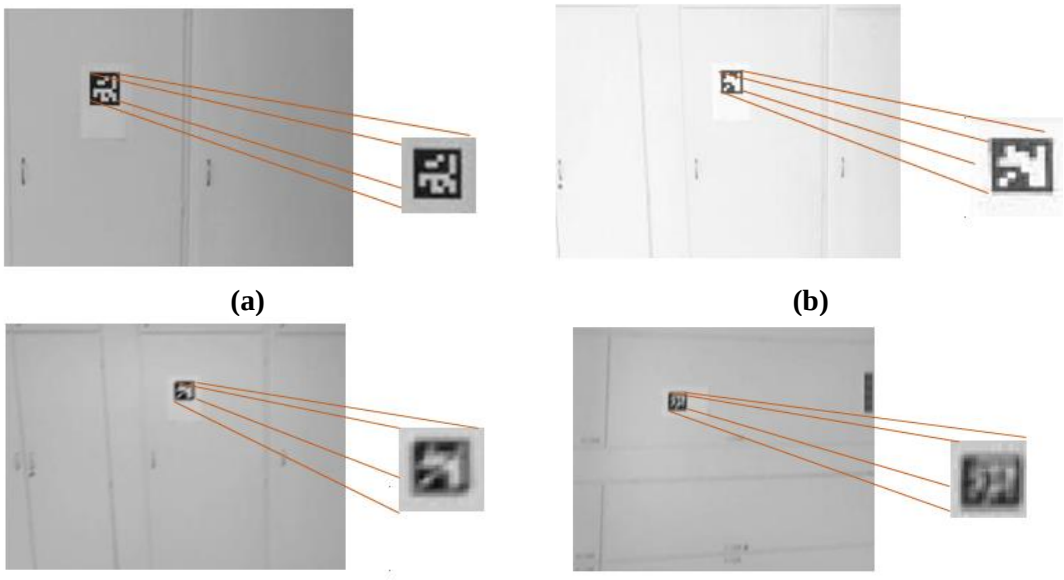


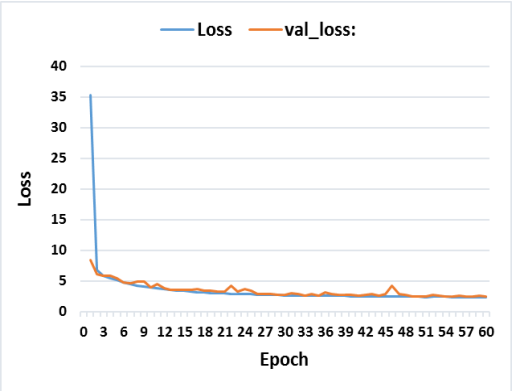
Figure 6-4. Marker images obtained under challenging conditions.

6.2.2 Evaluating Models

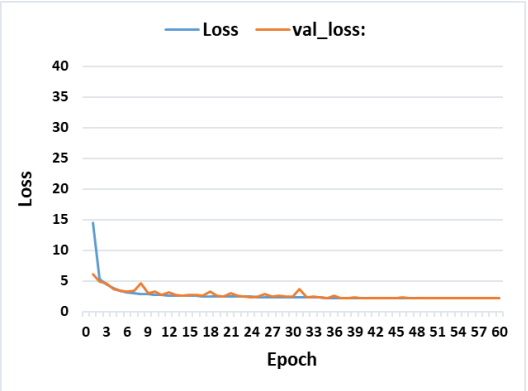
The Tiny-YOLOv3 model and the three modified models are trained using the created dataset on four steps to train and test the proposed models in different parts of the datasets. The first step (Far dataset) was to evaluate the four models using a part of the full dataset that contains images captured from long distances. Rotation is applied with different angles such as 90, 180, 270 degrees. The second step (Far Challenging) was to evaluate the four models using images captured from long distances with or without applying challenging conditions such as blur and lighting effects. The third step (Full dataset) is the same as the first step, but images captured from long and short distances are used. In the last step (Full Challenging), the full dataset is used which contains images from long and short distances, rotated images with different angles, and images after applying challenging conditions like step two. Different batch sizes and the number of epochs is used where the results proved that 60 epochs in batches with a size of 16 give the best results. A momentum of 0.9, the decay of 0.0005, Adam optimization, and learning rate was 0.001. From experts' experience, these values are the best to be used for this model. Models fine-tuning usually used a pre-trained model with a large dataset to get common weights and feature representation, then freeze some bottom part for further training on small or incremental data to improve and fasten the training process. So, the proposed models used a transfer training stage from ImageNet pre-trained backbone weights. Then, the model was Unfrozen after the first 20 epochs and continued training to fine-tune. In each epoch, 2550 iterations are used for training and 637 for validation. Furthermore, the training and validation losses are calculated. Every 5 epochs, precision, recall, F1 score, and mean Average Precision (mAP) are calculated to monitor the improvements during training. Python, Tensorflow, and Keras framework are used for implementation. To evaluate these models after training: precision, recall, F1 score, Average precision (AP), and mAP of the testing sets are calculated as shown in the following subsections.

6.2.2.1 Loss curves

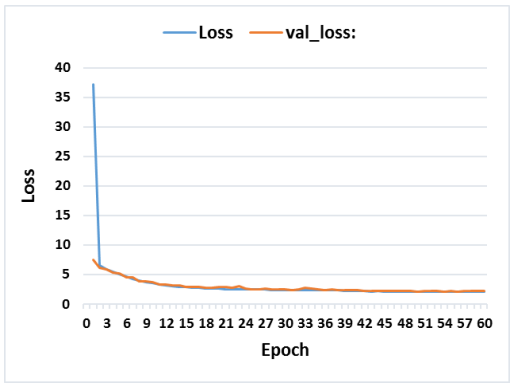
The loss is calculated using the number of examples that a model classifies wrong divided by the number of performed classifications. A good fit is identified by a training and validation loss that decreases to a point of stability with a minimal gap between the two final loss values. The loss of the model usually is lower on the training dataset than the validation dataset. Figure 6-5 shows the training and validation loss for the four models using the full dataset in normal and challenging conditions. The loss curves are smoothly going down, indicating that the proposed models fit better with training. The validation loss curves are slightly lower than the training loss, which indicates a good model fit. Based on this, the four models were trained and validated well using the dataset.



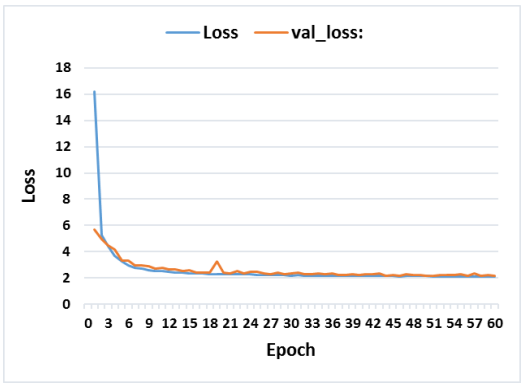
(a) Tiny-YOLOv3



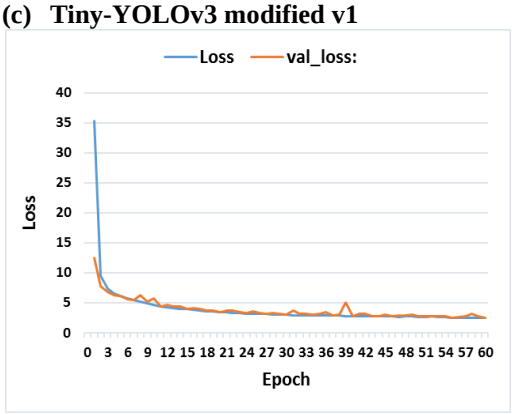
(b) Tiny-YOLOv3 challenging



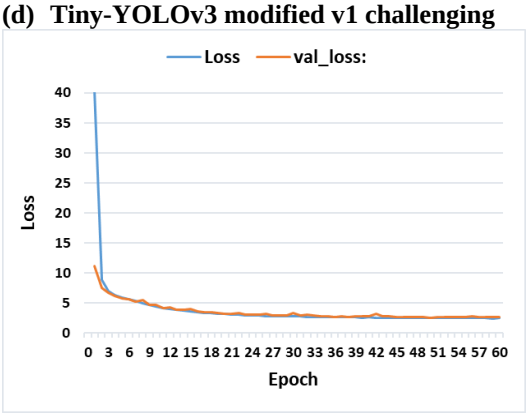
(c) Tiny-YOLOv3 modified v1



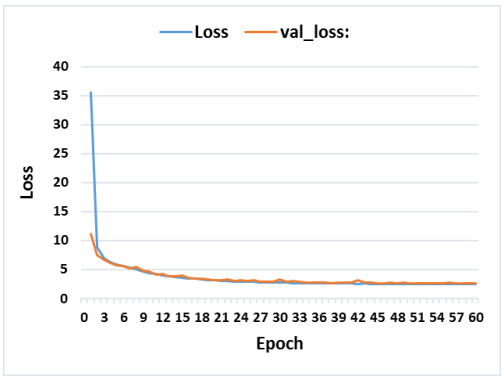
(d) Tiny-YOLOv3 modified v1 challenging



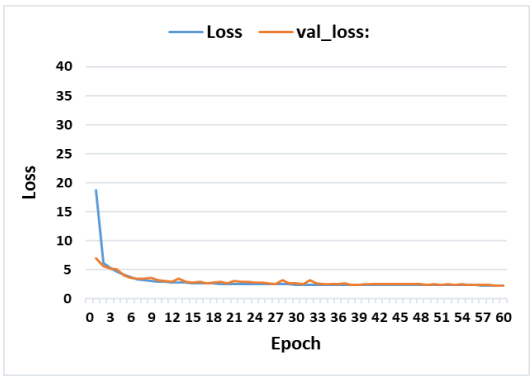
(e) Tiny-YOLOv3 modified v2



(f) Tiny-YOLOv3 modified v2 challenging



(g) Tiny-YOLOv3 modified v3

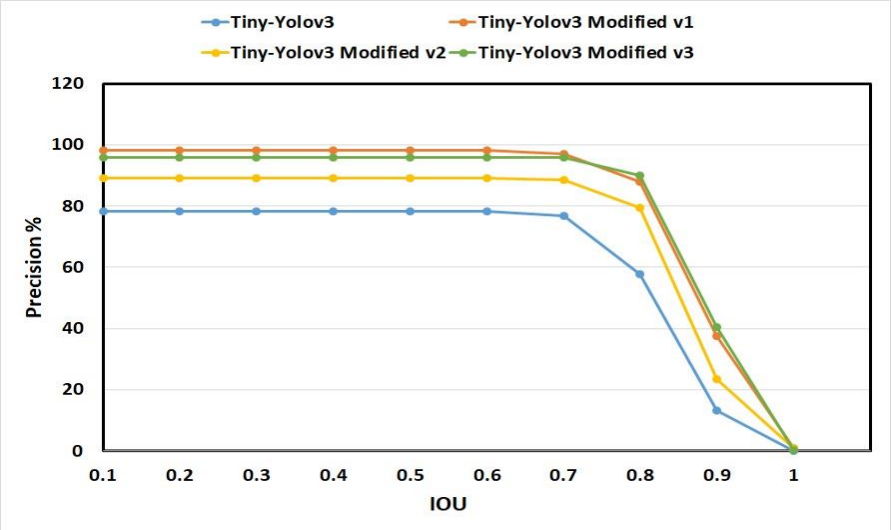


(h) Tiny-YOLOv3 modified v3 challenging

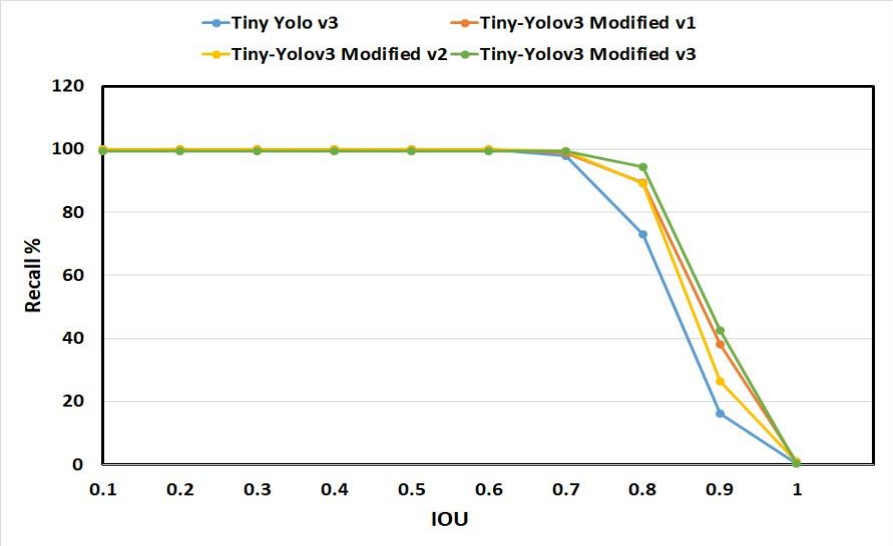
Figure 6-5. Training loss and validation loss versus epoch for the four models.

6.2.2.2 Precision, Recall, and F1 Score

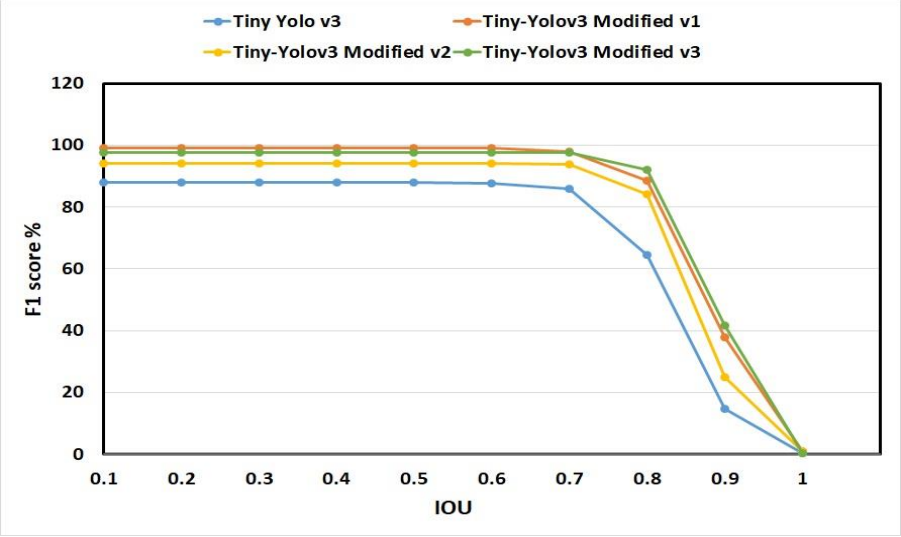
The analysis of precision, recall, and F1 score at different IOUs is a conventional method of evaluating object detection accuracy. If there is no detected box although there are markers in the image, the case is considered as FN. If the bounding box detected has an IOU value greater than or equal to the predefined threshold, there are two cases. In the first case, when the predicted markers are a correct one, the box is considered a TP. In the second case, the box is considered a FP, when the predicted class is not a correct marker. Precision presents the percentage of right predictions over the total number of predicted bounding boxes. Recall is the fraction of correctly detected markers over the total number of markers. Figure 6-6. shows the precision, recall and F1 score of various Tiny-YOLOv3 marker detectors at different IOU thresholds for the full dataset.



(a)



(b)



(c)

Figure 6-6. Graphs for (a) precision, (b) recall and (c) F1 score in normal conditions.

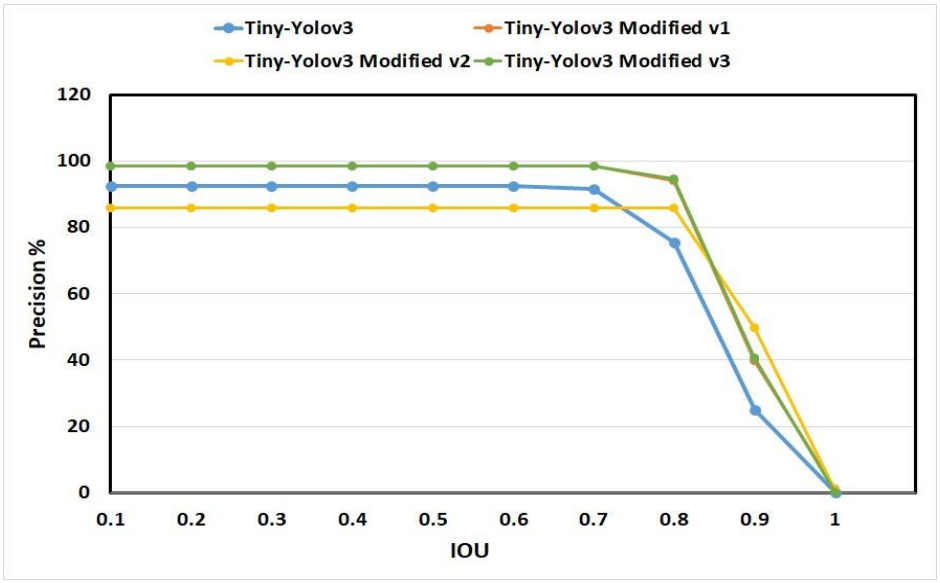
Precision and recall are used to evaluate the performance of any model.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{6-7}$$

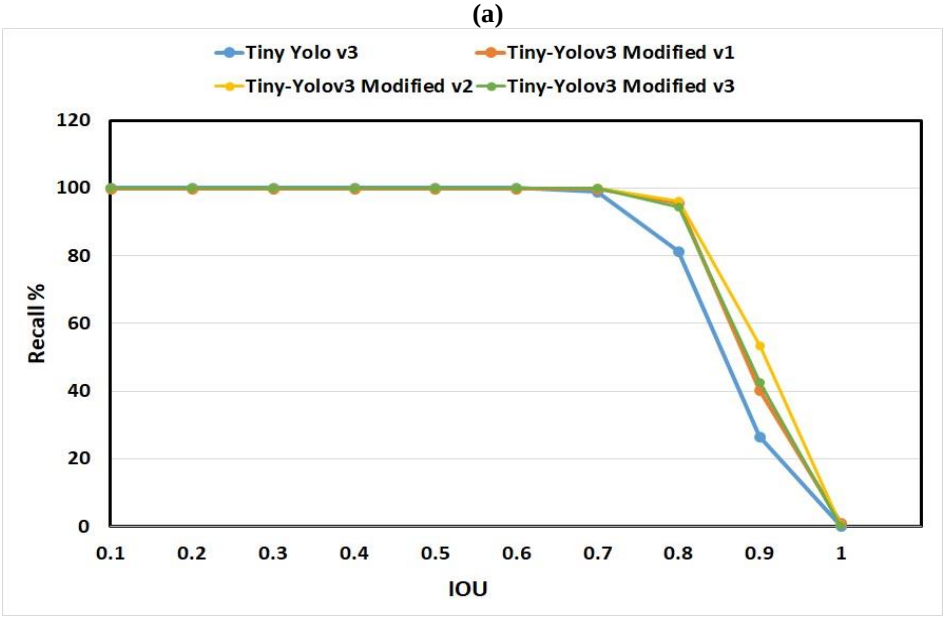
$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{6-8}$$

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision+Recall}} \tag{6-9}$$

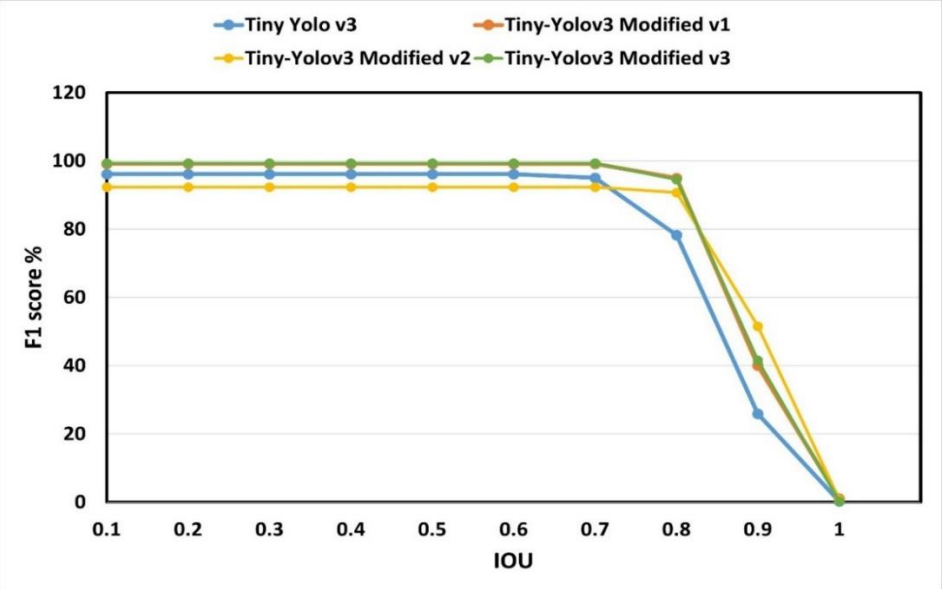
Although the graphs of a recall are nearly the same as shown in Figure 6-8. (b), the precision and F1 score curves of the modified Tiny-YOLOv3 version1 is the highest one as shown in (a) and (c) of Figure 6-8. It is also shown that the modified Tiny-YOLOv3 version 3 is better than modified Tiny-YOLOv3 version 2 and the original Tiny-YOLOv3. This means that the modified Tiny-YOLOv3 version 1 and 3 give better performance than the other two models. Figure 6-7 shows the results of the Tiny-YOLOv3 models when using the full dataset in challenging situations. It is shown that Tiny-YOLOv3 version3 and the Tiny-YOLOv3 version1 had the best precision and F1 score curves.



(a)



(b)



(c)

Figure 6-7. Graphs for (a) precision, (b) recall and (c) F1 score in challenging situation.

Table 6-1 shows the Precision (P), recall (R) and F1 score at IOU = 0.5 of the four models. As shown, the results for the first modified model gives better accuracy than the other models in non-challenging conditions as it gives 98.50% F1 score for the far dataset while the original model gives 97.88%. It gives 99.13% for the full dataset while the original model gives 87.84%. The modified version 3 is better than the original model as it gives 97.60% F1 score for the full dataset. For the dataset in challenging conditions, the modified version 3 is the best as it gives 98.40% for the far dataset and 99.31% for the full dataset while the original model gives 96.52% and 96.11% respectively. From Figure 6-6, Figure 6-7, and Table 6-1, it is seen that modified Tiny-YOLOv3 version3 is the best choice and gives the best accuracy when used for detecting markers in challenging conditions. However, modified Tiny-YOLOv3 version 1 is the best choice in normal conditions.

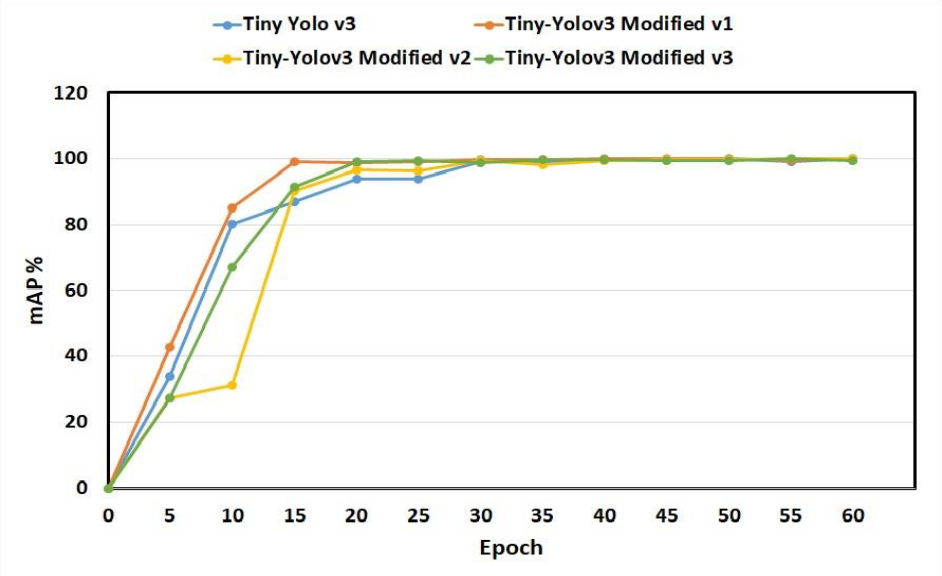
Table 6-1. Precision (P), recall (R) and F1 score at IOU = 0.5 of different models.

	Far dataset			**Far Challenging**			**Full dataset**			**Full Challenging**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
Tiny-YOLOv3	95.84	**100**	97.88	93.29	99.99	96.52	78.32	99.98	87.84	92.52	**100**	96.11
Modified version 1	**97.21**	99.83	**98.50**	94.43	99.90	97.09	**98.27**	**100**	**99.13**	98.43	99.96	99.19
Modified version 2	79.88	99.63	88.67	93.94	99.56	96.66	89.17	100	94.28	85.82	99.96	92.35
Modified version 3	90.86	99.29	94.89	**96.85**	**100**	**98.40**	95.81	99.47	97.60	**98.97**	99.97	**99.31**

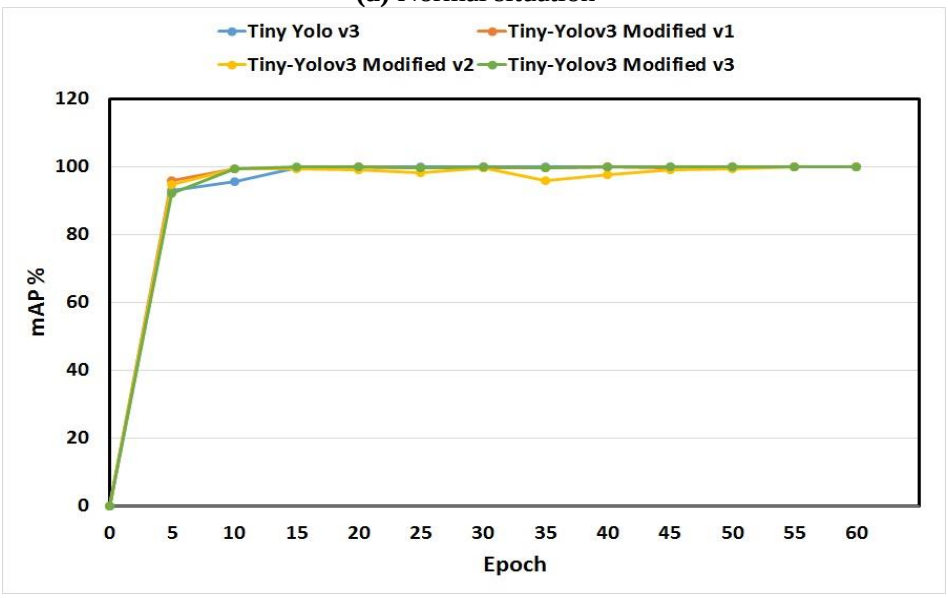
6.2.2.3 The mAP and AP

The mAP is an average AP value for several sets. mAP is used to measure detection accuracy and is calculated using the average value of the AP values. Figure 6-8 shows the mAP curves of the four models in normal and challenging situations. It is shown that mAP values for the four models are close to each other. So, mAP curves are not enough for evaluating these models. It is also seen that the curves became more stable in challenging conditions than in normal conditions

because in challenging conditions more images are used to represent situations such as rotation, blur, and lighting effects.



(a) Normal situation



(b) Challenging situation

Figure 6-8. Comparative graphs for different Tiny-YOLOv3 versions using mAP.

To prove the first hypothesis, fifty sub-datasets each with 408 images were sampled randomly from the test set. Each model was applied on the 50 sub-datasets where mAP, precision, recall, and F1 score was calculated. The p-values for each pair of methods were calculated using the t-test. The results are analyzed at a significance level of 0.05, i.e., the null hypothesis $\mathbf{H_{null}}$:" there is no significant difference between the two methods" is rejected if p-value ≤ 0.05. Table 6-2 shows the results of p-values when comparing the original model with each of the modified versions. One and two tailed t-tests are used. From the results, there is a significant difference between the original model and the first and the third modified versions. Also, there is no significant difference between the original version and the second modified version. Based on these results, null hypothesis is rejected and there is no significance difference between the first modified version and the third modified version.

Table 6-2. P-value for different t-tests of different modified versions and the original one.

	Modified version 1		Modified version 2		Modified version 3	
	One tail	**Two tails**	**One tail**	**Two tails**	**One tail**	**Two tails**
Original Version	2.86815E-09	5.7363E-09	0.378143864	0.756287728	8.53592E-09	1.70718E-08

6.2.2.4 Execution time

In addition to detection accuracy, an important performance indicator is processing time which is extremely important for some applications. The execution time is tested by running the models in about 15 videos and calculated the average execution time for the different algorithms. The mean processing time of the original Tiny-YOLOv3 model is 0.0351 while the average time for the first modified version is 0.0294 using the full dataset in challenging situations. Also, the average time for the second modified version is 0.0389 and 0.0323 for the third modified version. It is shown that the Tiny-YOLOv3 modified version 1 is the fastest model then, the Tiny-YOLOv3 modified version 3 is ranked as the second one. Also, the original Tiny-YOLOv3 is faster than the second modified version. Figure 6-9 shows the box diagram which represents the distribution of execution time for running the four models.

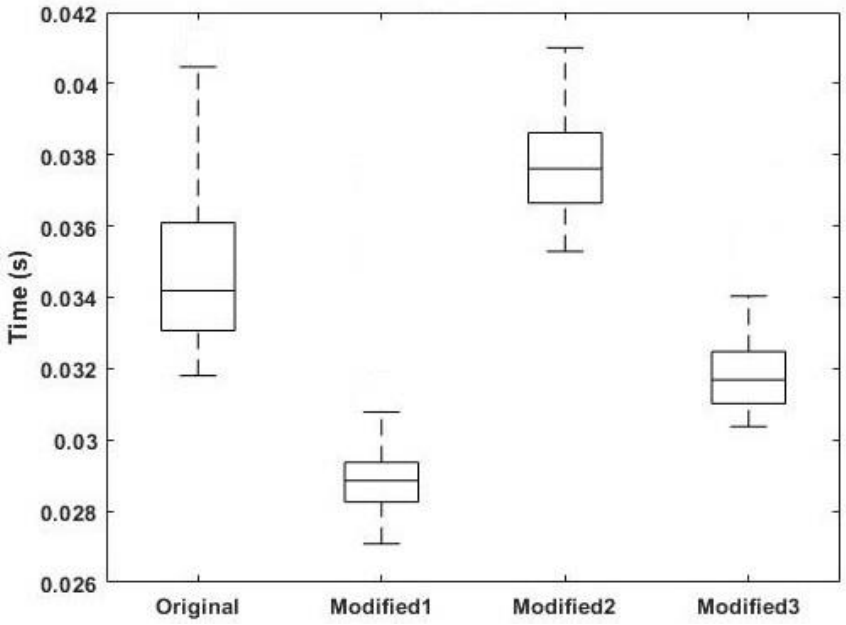


Figure 6-9. Box diagram representing the distribution of execution time of the four models.

As shown, the execution time for the first and the third modified versions are the best. A t-test for two samples has been made to compare the running time of the original model with every one of the modified versions and assumed that the two samples have equal variances for the first test. The second test for sample mean has been made and assumed alpha is 0.05. The null hypothesis is $\mathbf{H_{null}}$**:** the two models have the same running time. $\mathbf{H_{alt}}$**:** the two models have a different running time. The results are shown in Table 6-3. From these results, there is a difference between running time of the two models so, null hypothesis is rejected and found that the first or the third modified model are the best to be used in the navigation system. Figure 6-10 shows the marker detection examples obtained by the proposed models and the original Tiny-YOLOv3 model from different distances. The modified versions showed better results than those obtained by the original model where markers were successfully detected at both long and close distances in all cases.

Table 6-3. P-value for different t-tests of different modified versions.

		Modified version 1		Modified version 2		Modified version 3	
		One tail	Two tails	One tail	Two tails	One tail	Two tails
Original Version	Variance	3.10304E-42	6.20608E-42	1.73118E-07	3.46237E-07	4.0966E-15	8.19321E-15
	Mean	5.41361E-30	1.08272E-29	5.00728E-07	1.00146E-06	6.30346E-13	1.26069E-12

(a) Original version **(b)** Modified versions

Figure 6-10. Screenshots of detected markers from different distances.

To answer the research question: the original Tiny-YOLOv3 model and the modified version have been tested and evaluated using different evaluation metrics. The four models have been executed several times with different combinations of configurations. For example, the models

were executed for 60 or 100 epochs and batch size of 16 or 32. In each run, the loss, mean precision, recall and F1-score have been calculated in each epoch. From these experiments, the first modified version showed the best performance in normal situations while the third modified version showed the best performance in challenging situations. From these results, Figures 6-6 and Figure 6-7, and hypothesis testing, the first hypothesis H1 is accepted. So, the first or the third modified versions are used for the navigation system. To evaluate the second hypothesis, the original and the modified versions have been executed and the inference time is calculated. The results showed that the mean processing time of the original Tiny-YOLOv3 model is 0.0351s while the average time for the first modified version is 0.0294s using the full dataset in challenging situations. Furthermore, the average time for the second modified version is 0.0389 s and 0.0323s for the third modified version. It is shown that The Tiny-YOLOv3 modified version 1 is the fastest model. Also, the Tiny-YOLOv3 modified version 3 is faster than the original Tiny-YOLOv3 model and the second modified version as shown in Figure 6-9. From these results and from the hypothesis testing, the second hypothesis is accepted.

The proposed system has been compared with the others in the related work, as shown in Table 3-2. In the first criterion, most solutions used deep learning to detect objects and avoid obstacles. In the proposed system, deep learning models are used to detect markers in challenging conditions. The results give an F1 score of 99% which is evidence that the modified models are useful for this problem. For the second criterion, some solutions used laptops as a processing unit which is heavy to carry. A smartphone is used for proposed system as it is easy for PVI to carry, and most of them use it for daily tasks. For the third criterion, most solutions installed QR codes or markers in the environment. However, Aruco markers is used as they are more accurate than QR codes and, the proposed system can detect from longer distances. Other solutions did not use any markers and only described the scenes around PVI to avoid obstacles. An admin application is used to build the virtual map in the fourth criterion. The application constructs, and updates maps easily if needed. Some solutions used a manual map creation which has a problem in updating it if required. Other solutions did not use maps and depend on identifying the environment using computer vision techniques. In the fifth criterion, most solutions cannot detect markers in challenging conditions and from longer distances. However, there is a solution that supports identifying them in some challenging situations [43]. But it was used for kids, failed to detect markers from long distance, and developed as a desktop application. Furthermore, it used images processing techniques to select the candidate markers. These techniques take processing time that should be minimized to be suitable for real time usage. For the sixth criterion, the proposed system and some of the others concentrated only on navigation. They assumed that the environment is free of obstacles or can be avoided easily using a white cane. Several solutions are used to identify objects and avoid obstacles, while others combined both. Identifying objects and avoiding obstacles are serious problems that are associated with dangerous situations. Several solutions were tested by PVI and sighted people in the seventh criterion, while others were tested only by PVI. For the last criterion, most of the accuracies were calculated for the objects and obstacles' detection where the best one achieved 97%.

In summary:

- Some application installed markers on the ceiling of the building which is difficult for PVI to detect.
- In some applications, markers are installed on the floor which cannot be detected in a crowded environment.

- Using markers is better than QR codes which can be detected from a long distance.
- Most applications failed to detect markers from long distances and in challenging conditions such as motion blur or rapid walking speed.
- Some applications use images processing techniques to select candidate markers from images and send them to classification models which takes processing time that should be minimized.
- Using IMU sensors has an acceptable positioning accuracy only for a short distance since it suffers from drift error estimation over time.
- Some systems used obstacle detection sensors which is expensive and not available for common people.
- Some application needed Internet connection to download the graph of the building from the server.
- The use of Google Glasses is an additional burden for the user and not available for common people.
- Some systems were implemented as a logging system which is not suitable for real-time usage.
- The size and weight of the processing unit of some systems are cumbersome for PVI to carry for a long time.

6.3 Conclusion

The goal was to design a real-time marker detection system using the Tiny-YOLOv3 model. The original architecture has been modified several times to increase detection accuracy. It has been trained with a custom dataset for Aruco markers in different situations and from different distances. Also, experimental evaluations have been carried out to measure performances in terms of inference speed and accuracy. Experimentation results showed that from figure 6-6 and figure 6-7 how the F1 score of modified version 1 and version 3 is better than the original version. Also, Table 6-1 showed exactly the F1 score for the four models in different situations and from it you can see that the first modified version gives 99.13% F1 in normal situations and 99.19 % in challenging situations while the original version gives 87.84% F1 in normal situations and 96.11 % in challenging situations. The same for the modified version 3, it gives 97.60% F1 in normal situations and 99.31 % in challenging situations. It shown also that it minimizes the execution time, as the mean processing time of the original Tiny-YOLOv3 model is 0.0351 while the average time for the first modified version is 0.0294 using the full dataset in challenging situations. Also, the average time for the second modified version is 0.0389 and 0.0323 for the third modified version. It has been shown that the Tiny-YOLOv3 modified version 1 is the fastest model. Then, the Tiny-YOLOv3 modified version 3 is the second fastest one. Moreover, the proposed model can be installed on different embedded solutions such as edge computing to improve inference time.

7 Conclusion

In this thesis, a navigation system has been built using CV which helped PVI navigate indoors easily. But, before using it, a map should be constructed for each floor in the building by sighted people. They should move around the building to find the points of interest such as labs and lecture rooms. Then, markers are printed and installed on the wall at those points. After that, they explore all the available paths to each interest point and calculate the number of steps between them. An internal map is created using a graph to save the interest points and the relation between them. Nodes in this graph represent the accurate position of the markers while edges are labelled with the number of steps and navigation instructions. This graph is stored in a database to be used during navigation. The system starts by requesting the PVI to select their starting point based on the surrounding tags. When a marker is detected by the smartphone camera, the system will use this marker as a starting position. To start navigation, the system asks the PVI to choose their destination using voice commands. Then, it searches in the database for the shortest path from this point to the destination. This path is a list of checkpoints that the PVI should pass to arrive at their destination. During navigation, continuous guidance is given to them when moving from one point to the next. The system uses a voice recognition API to covert PVI commands to text. It also uses text to speech to provide audio feedback to the blind person.

The system always tries to detect markers however, if it misses detecting markers in one frame, it will likely detect it successfully in one of the next frames. If PVI finds another marker and this marker is in the list of markers to the destination point, the system continues giving navigation commands from this marker to the destination point. However, if this marker is not on the list, the system starts to find the shortest path from that marker to the destination point. If the PVI move in a wrong direction such as going right instead of going left and find another marker, the system starts to find the shortest path from that marker to the destination point. During testing, some problems were discovered: sometimes PVI failed to understand the feedbacks so, these feedbacks have been improved based on the comments of the PVI and found the audio feedback to be satisfactory. PVI move their hands rapidly during navigation which causes images to be captured with occlusion. Sometimes PVI cannot detect markers because they are moving their hands a lot and tags move out of the smartphone's camera view. Markers can also be captured with angles that cannot be detected correctly with the proposed system. So, it has been improved by installing eight markers with the same id at each interest point instead of adding only. This implementation makes detection easier and solved the problem of occlusion and decreased the chance for the markers to be outside of the camera view. It also helps PVI of different heights detect markers easily.

The proposed system has been improved to detect markers from a longer distance by using CNN and the system work as follow: While receiving a real-time stream of images from the smartphone camera, images are converted to grayscale ones and sent to the prototype for detecting and identifying markers. If any marker is identified, voice feedbacks are given to the PVI. If it fails to detect markers, the image is given to our CNN model to identify markers. This model processes the image and returns the correct id if any maker is detected. However, if it fails to do so, it decides that no marker is available and continues processing the next image. This process is repeated until the PVI reach their destination. The CNN model has also been improved by using the original Tiny-YOLOv3 model first to detect Aruco markers. Then, it has been modified to improve feature extraction and detection accuracy. So, detecting markers in the navigation system works the following way: While receiving a real-time stream of images from the smartphone camera, images are converted to grayscale ones and sent to the prototype for detecting and identifying markers. If any marker is detected, voice feedbacks are given to the PVI. If it fails to detect markers, the image is given to our deep learning model to detect markers.

An assistive system for PVI is proposed to detect and avoid objects independently where all feedback is provided to PVI in the form of audio. Images of objects are collected and manually labelled to create the dataset which is used to train the DL model. In this system, YOLOv3 and Tiny-YOLOv3 models were compared for detecting objects. The results showed that the YOLOv3 gives better accuracy for detecting and recognition. The results also showed that Tiny-YOLOv3 model is faster than the YOLOv3 model. So, YOLOv3 has been selected for the system. Future work will focus on minimizing recognition time for YOLOv3 and adding more objects for objects in the dataset to make it more useful for PVI.

7.1 New Scientific Results

7.1.1 Thesis I: Build an indoor navigation system to help PVI navigate and avoid objects during navigation using deep learning.

Different technologies have been compared using several criteria such as the cost of applying the technology to any solution, the equipment needed, the number of items able to be scanned at the same time, whether the PVI must be in the line of sight with the tag, and the storage capacity of each solution. After that CV tag-based systems were selected based on this comparison. Also, a comparison between Aruco markers and QR codes was done, and the results showed that Aruco markers were better as shown in Table 4-2. So, Aruco markers have been selected as the best ones for the navigation system.

A navigation system has been built using CV to help PVI navigate indoors easily using markers. All the situations and conditions that PVI may face during navigation were added. An admin application was developed to automate the map building process. So, markers were printed and installed on the wall at the interest points. Then, sighted people used this application to store the interest points and the relation between them. Finally, an internal map has been created using a graph to save the interest points and the relation between them.

The system can detect markers under occlusion. PVI moved their hands rapidly during navigation, causing the images to be captured with a part of it is occluded; sometimes, the PVI cannot detect markers because they move their hands a lot and markers move out of the smartphone's camera view; markers can be captured with angles which cannot be detected correctly with our current system. The proposed system solved this problem and improved the environment by installing eight markers with the same id at each interest point instead of adding only one. The experiments proved that this implementation made detection easier and solved the problem of occlusion and decreased the chance for the markers to be outside of the camera view. It also helped PVI of different heights to detect markers easily.

A protype of an assistive system application was proposed for PVI to detect and avoid objects independently. YOLOv3 and Tiny-YOLOv3 models were used to compare and balance between the accuracy and the execution time and YOLOv3 was selected based on the comparison. A system to help PVI avoid objects using YOLOv3 has been proposed. It started by opening the camera and asking the PVI to move to reach their destination. While walking, a real stream of images from the smartphone camera were captured and converted to grayscale ones. Then, they were sent to a deep learning model to detect objects. If any object is detected, feedback to was returned to PVI to avoid it. However, if it failed to do so, it decided that no objects were available and continued processing the next image.

7.1.2 Thesis II: Identify Markers from longer distances using CNN model.

The navigation system has been improved by adding a CNN model to detect markers. A CNN model was proposed to detect markers from longer distances. The proposed model is based on the Alexnet model. The Alexnet model has 5 convolutional layers and to simplify it, only 3 convolutional layers have been used and the output is the marker id if anything was detected. This model was compared with another model and the results showed that it gave better accuracy. The results showed that the proposed model can detect Aruco markers from longer distance as it provided approximately 97% accuracy for training and 99.97% accuracy for testing. When applying it to the other dataset, it gave 86% for training accuracy and 94.74% for testing accuracy. From these results, the detection of markers from a long distance has been improved. However, the time required for this model to detect markers should be minimized.

This CNN model has been simplified by using two layers instead of three to improve the execution time. I have simplified the convolutional layers of the CNN model and used the same parameters for training and validation. This simplified model was compared with the basic one and the simplified model gave 95.5% accuracy for training and 99.82% accuracy for testing. So, the training and testing curves of the proposed models are close to each other. However, the execution time for detecting markers in the simplified model were better than the basic one. As, the execution time for the basic model was 0.59 seconds while, the execution time for the simplified model was 0.19 seconds. Also, this simplified model was compared with the baseline and the results showed that it gave better accuracy as the baseline gave 86% for training accuracy and 94.74% for testing accuracy.

7.1.3 Thesis III: Build a novel marker detection system for PVI using the improved Tiny-YOLOv3 model.

Using the proposed CNN to identify markers has only dealt with the identification steps, while detection has been done using a method based on image thresholding and rectangle extraction. This model has been improved by using YOLO to fully perform the detection and identification steps. The goal was to design a real-time marker detection system using the Tiny-YOLOv3 model. The original architecture has been modified several times to increase detection accuracy. It has been trained with a custom dataset for Aruco markers in different situations and from different distances. Also, experimental evaluations have been carried out to measure performances in terms of inference speed and accuracy.

Experimentation results showed that the F1 score of modified version 1 and version 3 was better than the original version. As the first modified version gave 99.13% F1 in normal situations and 99.19 % in challenging situations while the original version gave 87.84% F1 in normal situations and 96.11 % in challenging situations. The same for the modified version 3, it gave 97.60% F1 in normal situations and 99.31 % in challenging situations. It has been shown that it minimized the execution time, as the mean processing time of the original Tiny-YOLOv3 model was 0.0351 while the average time for the first modified version was 0.0294 using the full dataset in challenging situations. Also, the average time for the second modified version was 0.0389 and 0.0323 for the third modified version. It has been shown that the Tiny-YOLOv3 modified version 1 is the fastest model. Also, the Tiny-YOLOv3 modified version 3 is the second fastest one.

7.2 Future plans

Naturally, the results that were presented and the theses that were formed are not the end of this scientific work. Thus, it can be continued in the following ways:

- **Object detection:** System can detect objects, but can't calculate the distance, so depth camera or ultrasonic sensor can be used. Recognition time should be minimized. I am working on training different classification algorithms such as VGG16, Resnet50, inception V3, Mobilenet and comparing them. Then I'll modify the best one to improve accuracy. Also, I will install the final model on Raspberry pi, Intel NCS to minimize the execution time.
- **Occlusion problem:** detecting markers under occlusion can be improved using deep learning models.
- **Improve feedback:** integrate orientation sensors to quickly warn PVI if they turn in the wrong direction.

7.3 Publications

The main results of this PhD dissertation were published in multiple international journals, and some were presented at national and international conferences. The number of the respective thesis was shown in parentheses. The publications were sorted by year in a decreasing order.

7.3.1 Publications related to this Thesis.

1. **Mostafa Elgendy**, Cecilia Sik Lanyi and Arpad Kelemen. A Novel Marker Detection System for People with Visual Impairment Using the improved Tiny-YOLOv3 model. computer methods and programs in biomedicine, Elsevier, Volume 205, 2021, 106112, ISSN 0169-2607. **IF: 5.428, (Thesis III).**
2. **Mostafa Elgendy** and Cecilia Sik Lanyi. Helping People with Visual Impairments to Avoid Obstacles Using Deep Learning. In Proc. of the 6th International Congress on Information and Communication Technology (ICICT), (2021). Springer, **(Thesis I).**
3. **Mostafa Elgendy**, Tibor Guzsvinecz and Cecilia Sik Lanyi. Indoor Navigation for People with Visual Impairment using Augmented Reality Markers. In Proc. of the 10th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), (2019), pp. 425-429. IEEE**, (Thesis III).**
4. **Mostafa Elgendy** and Cecilia Sik Lanyi. Comparing QR Codes with Aruco Markers Using Indoor Navigation Prototype for People with Visual Impairment. In Proc. of the IEEE 31th Neumann Colloquium (NC), (2019), pp. 19-24**, (Thesis I).**
5. **Mostafa Elgendy**, Tibor Guzsvinecz and Cecilia Sik Lanyi. Identification of Markers in Challenging Conditions for People with Visual Impairment Using Convolutional Neural Network. Applied Sciences Basel, 9(23): Article Number: 5110, (2019). **IF: 2.679, (Thesis II).**
6. **Mostafa Elgendy**, Viktor Földing, Miklós Herperger and Cecilia Sik Lanyi. "Indoor Navigation for People with Visual Impairment." In Proc. of the Association for the Advancement of Assistive Technology in Europe Conference (AAATE), (2019)**, (Thesis I).**
7. **Mostafa Elgendy**, Cecilia Sik Lanyi and Arpad Kelemen. Making Shopping Easy for People with Visual Impairment Using Mobile Assistive Technologies. Applied Sciences Basel, 9(6): 1061-1076, Article Number: 1061, (2019). **IF: 2.679, (Thesis I).**

8. **Mostafa Elgendy** and Cecilia Sik Lanyi. Review on Smart Solutions for People with Visual Impairment. In Proc. of the International Conference on Computers Helping People with Special Needs (ICCHP), K. Miesenberger and G. Kouroupetroglou (Eds.) Part I, LNCS 10896, pp. 81-84, Springer International Publishing Switzerland, (2018)**, (Thesis I).**

7.3.2 Publications not related to this Thesis

1. Veronika Szűcs and **Mostafa Elgendy**. Assistive method for people with hearing disability-music visualisation. IEEE 30th Neumann Colloquium (NC), pp. 117-118. IEEE, (2017)

Bibliography

[1] "WHO | Global trends in the magnitude of blindness and visual impairment." [Online]. Available: https://www.who.int/blindness/causes/trends/en/. [Accessed: 06-Oct-2020].

[2] R. R. A. Bourne *et al.*, "Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment: a systematic review and meta-analysis," *Lancet Glob. Heal.*, vol. 5, no. 9, pp. e888–e897, Sep. 2017.

[3] P. Ackland, S. Resnikoff, and R. Bourne, "World blindness and visual impairment: Despite many successes, the problem is growing," *Community Eye Health Journal*, vol. 30, no. 100. pp. 71–73, 2018.

[4] N. A. Giudice, "Navigating without vision: principles of blind spatial cognition," in *Handbook of Behavioral and Cognitive Geography*, no. January, Edward Elgar Publishing, 2018, pp. 260–288.

[5] E. Kostyra, S. Żakowska-Biemans, K. Śniegocka, and A. Piotrowska, "Food shopping, sensory determinants of food choice and meal preparation by visually impaired people. Obstacles and expectations in daily food experiences," *Appetite*, vol. 113, pp. 14–22, 2017.

[6] A. Bhowmick and S. M. Hazarika, "An insight into assistive technology for the visually impaired and blind people: state-of-the-art and future trends," *J. Multimodal User Interfaces*, vol. 11, no. 2, pp. 149–172, Jun. 2017.

[7] M. A. Elgendy, A. Shawish, and M. I. Moussa, "MCACC: New approach for augmenting the computing capabilities of mobile devices with Cloud Computing," in *2014 Science and Information Conference*, 2014, pp. 79–86.

[8] P. Angin, B. B.-I. J. of Next-Generation, and U. 2011, "Real-time mobile-cloud computing for context-aware blind navigation," *cs.purdue.edu*.

[9] K. Manjari, M. Verma, and G. Singal, "A survey on Assistive Technology for visually impaired," *Internet of Things*, vol. 11, p. 100188, Sep. 2020.

[10] "What is visual impairment?" [Online]. Available: https://www.news-medical.net/health/What-is-visual-impairment.aspx. [Accessed: 07-Oct-2020].

[11] "Types of visual impairment." [Online]. Available: https://www.news-medical.net/health/Types-of-visual-impairment.aspx. [Accessed: 07-Oct-2020].

[12] W. H. O.-G. W. H. Organization and undefined 2001, "World Health Organization International Classification of Functioning, Disability and Health."

[13] N. Kostanjsek, "Use of the International Classification of Functioning, Disability and Health (ICF) as a conceptual framework and common language for disability statistics and health information systems," in *BMC Public Health*, 2011, vol. 11, no. SUPPL. 4.

[14] G. Whiteneck, C. Harrison-Felix, … D. M.-A. of physical, and undefined 2004, "Quantifying environmental factors: a measure of physical, attitudinal, service, productivity, and policy barriers," *Elsevier*.

[15] L. Hakobyan, J. Lumsden, D. O'Sullivan, and H. Bartlett, "Mobile assistive technologies for the visually impaired," *Surv. Ophthalmol.*, vol. 58, no. 6, pp. 513–528, 2013.

[16] T. B. Moeslund and E. Granum, "A survey of computer vision-based human motion capture," *Comput. Vis. Image Underst.*, vol. 81, no. 3, pp. 231–268, Mar. 2001.

[17] R. Szeliski, *Computer vision: algorithms and applications*. 2010.

[18] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International Journal of Remote Sensing*, vol. 28, no. 5. Taylor and Francis Ltd., pp. 823–870, 2007.

[19] A. Morar *et al.*, "A Comprehensive Survey of Indoor Localization Methods Based on Computer Vision," *Sensors*, vol. 20, no. 9, p. 2641, May 2020.

[20] L. Liu *et al.*, "Deep Learning for Generic Object Detection: A Survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, Feb. 2020.

[21] A. Hub, J. Diepstraten, and T. Ertl, "Design and development of an indoor navigation and object identification system for the blind," *ACM SIGACCESS Access. Comput.*, no. 77–78, pp. 147–152, Sep. 2003.

[22] R. Yao, G. Lin, S. Xia, J. Zhao, and Y. Zhou, "Video Object Segmentation and Tracking: A Survey," *arXiv*, vol. 39, Apr. 2019.

[23] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4. p. 45, 25-Dec-2006.

[24] F. Hutter, L. Kotthoff, and J. Vanschoren, "The Springer Series on Challenges in Machine Learning Automated Machine Learning Methods, Systems, Challenges," 2019.

[25] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. 2015.

[26] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, "Data Cleaning: Overview and Emerging Challenges," *dl.acm.org*, vol. 26-June-2016, pp. 2201–2206, Jun. 2016.

[27] X. L. Dong and T. Rekatsinas, "Data Integration and Machine Learning: A Natural Synergy," *dl.acm.org*, pp. 1645–1650, May 2018.

[28] J. Han, "Data Mining: Concepts and Techniques-Chapter 6," 2015.

[29] H. Wang, S. Wang, H. Wang, and S. Wang, "Mining incomplete survey data through classification Linear discriminant analysis MI Multiple imputation MSA Metropolitan Statistical Area," *Knowl Inf Syst*, vol. 24, no. 2, pp. 221–233, 2010.

[30] D. Charte, F. Charte, S. García, and F. Herrera, "A snapshot on nonstandard supervised learning problems: taxonomy, relationships, problem transformations and algorithm adaptations," *Progress in Artificial Intelligence*, vol. 8, no. 1. Springer Verlag, 01-Apr-2019.

[31] R. Saravanan and P. Sujatha, "A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification," in *Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, ICICCS 2018*, 2019, pp. 945–949.

[32] S. Raschka, "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning," Nov. 2018.

[33] M. Stone, "Cross-Validatory Choice and Assessment of Statistical Predictions," *J. R. Stat. Soc. Ser. B*, vol. 36, no. 2, pp. 111–133, Jan. 1974.

[34] Q. Yao *et al.*, "Taking the Human out of Learning Applications: A Survey on Automated Machine Learning."

[35] B. Yegnanarayana, *Artificial neural networks*. 2009.

[36] S. Shanmuganathan, “Artificial neural network modelling: An introduction,” in *Studies in Computational Intelligence*, vol. 628, Springer Verlag, 2016, pp. 1–14.

[37] J. S.-N. networks and undefined 2015, “Deep learning in neural networks: An overview,” *Elsevier*.

[38] N. Aloysius, M. G.-2017 I. C. on, and undefined 2017, “A review on deep convolutional neural networks,” *ieeexplore.ieee.org*.

[39] W. Rawat and Z. Wang, “Deep convolutional neural networks for image classification: A comprehensive review,” *Neural Computation*, vol. 29, no. 9. MIT Press Journals, pp. 2352–2449, 01-Sep-2017.

[40] P. A. Zientara *et al.*, “Third Eye: A Shopping Assistant for the Visually Impaired,” *Computer (Long. Beach. Calif).*, vol. 50, no. 2, pp. 16–24, Feb. 2017.

[41] S. Kesh and others, “Shopping By Blind People: Detection of Interactions in Ambient Assisted Living Environments using RFID,” *Int. J.*, vol. 6, no. 2, 2017.

[42] E. J. Wong, K. M. Yap, J. Alexander, and A. Karnik, “HABOS: Towards a platform of haptic-audio based online shopping for the visually impaired,” in *2015 IEEE Conference on Open Systems (ICOS)*, 2015, pp. 62–67.

[43] A. Aziz, M. H. A. Wahab, A. Mustapha, and M. F. M. Mohsin, “Design and development of smart home security system for disabled and elderly people,” *J. Telecommun. Electron. Comput. Eng.*, vol. 9, no. 3–7, pp. 135–138, 2017.

[44] S. Solaimani, W. Keijzer-Broers, and H. Bouwman, “What we do – and don’t – know about the Smart Home: An analysis of the Smart Home literature,” *Indoor Built Environ.*, vol. 24, no. 3, pp. 370–383, May 2015.

[45] B. Busatlic, N. Dogru, I. Lera, and E. Sukic, “Smart Homes with Voice Activated Systems for Disabled People,” *Tem Journal-Technology Educ. Manag. Informatics*, vol. 6, no. 1, pp. 103–107, 2017.

[46] A. Saad Al-Sumaiti, M. H. Ahmed, and M. M. A. Salama, “Smart home activities: A literature review,” *Electric Power Components and Systems*, vol. 42, no. 3–4. pp. 294–305, 12-Mar-2014.

[47] D. López-De-Ipiña, T. Lorido, and U. López, “Indoor navigation and product recognition for blind people assisted shopping,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 6693 LNCS, pp. 33–40.

[48] M. Kang, S. Chae, J. Sun, … J. Y.-I. T. on, and undefined 2015, “A novel obstacle detection method based on deformable grid for the visually impaired,” *ieeexplore.ieee.org*.

[49] R. Jafri, S. A. Ali, H. R. Arabnia, and S. Fatima, “Computer vision-based object recognition for the visually impaired in an indoors environment: a survey,” *Vis. Comput.*, vol. 30, no. 11, pp. 1197–1222, Nov. 2014.

[50] B. F. G. Katz *et al.*, “NAVIG: augmented reality guidance system for the visually impaired,” *Virtual Real.*, vol. 16, no. 4, pp. 253–269, Nov. 2012.

[51] S. A. Paneels, D. Varenne, J. R. Blum, J. R. Cooperstock, S. A. Panëels, and R. Blum, “the Walking Straight Mobile Application : Impaired the Avoid Visually Veering Helping Impaired Avoid Veering,” pp. 25–32, 2013.

[52] M. C. Rodriguez-Sanchez, M. A. Moreno-Alvarez, E. Martin, S. Borromeo, and J. A. Hernandez-Tamames, “Accessible smartphones for blind users: A case study for a

wayfinding system," *Expert Syst. Appl.*, vol. 41, no. 16, pp. 7210–7222, 2014.

[53] J. Cheng, L. Yang, Y. Li, and W. Zhang, "Seamless outdoor/indoor navigation with WIFI/GPS aided low cost Inertial Navigation System," *Phys. Commun.*, vol. 13, no. PA, pp. 31–43, 2014.

[54] S. Jonas *et al.*, "IMAGO: Image-guided navigation for visually impaired people," *Artic. J. Ambient Intell. Smart Environ.*, 2015.

[55] S. A. De Silva and D. Dias, "A sensor platform for the visually impaired to walk straight avoiding obstacles," *Proc. Int. Conf. Sens. Technol. ICST*, vol. 2016-March, pp. 838–843, 2016.

[56] A. Wachaja, P. Agarwal, M. Zink, M. R. Adame, K. Möller, and W. Burgard, "Navigating blind people with walking impairments using a smart walker," *Auton. Robots*, vol. 41, no. 3, pp. 555–573, Mar. 2017.

[57] M. L. Mekhalfi, F. Melgani, A. Zeggada, F. G. B. De Natale, M. A.-M. Salem, and A. Khamis, "Recovering the sight to blind people in indoor environments with smart technologies," *Expert Syst. Appl.*, vol. 46, pp. 129–138, Mar. 2016.

[58] A. Aladren, G. Lopez-Nicolas, L. Puig, and J. J. Guerrero, "Navigation Assistance for the Visually Impaired Using RGB-D Sensor With Range Expansion," *IEEE Syst. J.*, vol. 10, no. 3, pp. 922–932, Sep. 2016.

[59] C.-F. Liao, "An Integrated Assistive System to Support Wayfinding and Situation Awareness for People with Vision Impairment," *ProQuest Diss. Theses*, no. May, p. 291, 2016.

[60] M. C. Kang, S. H. Chae, J. Y. Sun, S. H. Lee, and S. J. Ko, "An enhanced obstacle avoidance method for the visually impaired using deformable grid," *IEEE Trans. Consum. Electron.*, vol. 63, no. 2, pp. 169–177, 2017.

[61] R. Jafri, R. L. Campos, S. A. Ali, and H. R. Arabnia, "Visual and Infrared Sensor Data-Based Obstacle Detection for the Visually Impaired Using the Google Project Tango Tablet Development Kit and the Unity Engine," *IEEE Access*, vol. 6, pp. 443–454, 2017.

[62] V. N. Hoang, T. H. Nguyen, T. L. Le, T. T. H. Tran, T. P. Vuong, and N. Vuillerme, "Obstacle detection and warning for visually impaired people based on electrode matrix and mobile Kinect," in *Proceedings of 2015 2nd National Foundation for Science and Technology Development Conference on Information and Computer Science, NICS 2015*, 2015, pp. 54–59.

[63] J. R. Rizzo, Y. Pan, T. Hudson, E. K. Wong, and Y. Fang, "Sensor fusion for ecologically valid obstacle identification: Building a comprehensive assistive technology platform for the visually impaired," in *2017 7th International Conference on Modeling, Simulation, and Applied Optimization, ICMSAO 2017*, 2017.

[64] E. Valero, A. Adán, and C. Cerrada, "Evolution of RFID Applications in Construction: A Literature Review," *Sensors*, vol. 15, no. 7, pp. 15988–16008, Jul. 2015.

[65] W. E. Sakpere, N. B. W. Mlitwa, and M. A. Oshin, "Towards an efficient indoor navigation system: a near field communication approach," *J. Eng. Des. Technol.*, vol. 15, no. 4, pp. 505–527, Aug. 2017.

[66] Z. Farid, R. Nordin, and M. Ismail, "Recent advances in wireless indoor localization techniques and system," *Journal of Computer Networks and Communications*, vol. 2013. Hindawi, pp. 1–12, 22-Sep-2013.

[67] W. Sakpere, M. Adeyeye Oshin, and N. B. Mlitwa, "A State-of-the-Art Survey of Indoor

Positioning and Navigation Systems and Technologies," *South African Comput. J.*, vol. 29, no. 3, pp. 145–197, Dec. 2017.

[68] J. Cecílio, K. Duarte, and P. Furtado, "BlindeDroid: An information tracking system for real-time guiding of blind people," *Procedia Comput. Sci.*, vol. 52, no. 1, pp. 113–120, 2015.

[69] S. K.-I. Journal and U. 2017, "Shopping By Blind People: Detection of Interactions in Ambient Assisted Living Environments using RFID," *d.researchbib.com*, vol. 6, no. 2, 2017.

[70] D. López-De-Ipiña, T. Lorido, and U. López, "BlindShopping: Enabling accessible shopping for visually impaired people through mobile technologies," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 6719 LNCS, pp. 266–270.

[71] M. Alnfiai, S. Sampalli, and B. Mackay, "VirtualEyez: Developing NFC Technology to Enable the Visually Impaired to Shop Independently," *Int. J. Electr. Comput. Syst.*, Jul. 2016.

[72] B. Ozdenizci, V. Coskun, and K. Ok, "NFC Internal: An Indoor Navigation System," *Sensors*, vol. 15, no. 4, pp. 7571–7595, Mar. 2015.

[73] P. A. Zientara *et al.*, "Third Eye: A Shopping Assistant for the Visually Impaired," *Computer (Long. Beach. Calif).*, vol. 50, no. 2, pp. 16–24, Feb. 2017.

[74] R. Kumar and S. Meher, "A Novel method for visually impaired using object recognition," in *2015 International Conference on Communications and Signal Processing (ICCSP)*, 2015, pp. 0772–0776.

[75] V.-N. Hoang, T.-H. Nguyen, T.-L. Le, T.-T. H. Tran, T.-P. Vuong, and N. Vuillerme, "Obstacle detection and warning for visually impaired people based on electrode matrix and mobile Kinect," in *2015 2nd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS)*, 2015, vol. 4, no. 2, pp. 54–59.

[76] S. Al-Khalifa and M. Al-Razgan, "Ebsar: Indoor guidance for the visually impaired," *Comput. Electr. Eng.*, vol. 54, pp. 26–39, Aug. 2016.

[77] J. C. Torrado, G. Montoro, and J. Gomez, "Easing the integration: A feasible indoor wayfinding system for cognitive impaired people," *Pervasive Mob. Comput.*, vol. 31, pp. 137–146, Sep. 2016.

[78] H. Zhang, C. Zhang, W. Yang, and C.-Y. Chen, "Localization and navigation using QR code for mobile robot in indoor environment," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2015, pp. 2501–2506.

[79] E. Ko and E. Y. Kim, "A Vision-Based Wayfinding System for Visually Impaired People Using Situation Awareness and Activity-Based Instructions," *Sensors*, vol. 17, no. 8, p. 1882, Aug. 2017.

[80] A. Idrees, Z. Iqbal, and M. Ishfaq, "An efficient indoor navigation technique to find optimal route for blinds using QR codes," in *2015 IEEE 10th Conference on Industrial Electronics and Applications (ICIEA)*, 2015, pp. 690–695.

[81] A. K. Dash, S. K. Behera, D. P. Dogra, and P. P. Roy, "Designing of marker-based augmented reality learning environment for kids using convolutional neural network architecture," *Displays*, vol. 55, pp. 46–54, Dec. 2018.

[82] G. C. La Delfa, V. Catania, S. Monteleone, J. F. De Paz, and J. Bajo, "Computer Vision

Based Indoor Navigation: A Visual Markers Evaluation," in *Advances in Intelligent Systems and Computing*, vol. 376, 2015, pp. 165–173.

[83] J. Bacik, F. Durovsky, P. Fedor, and D. Perdukova, "Autonomous flying with quadrocopter using fuzzy control and ArUco markers," *Intell. Serv. Robot.*, vol. 10, no. 3, pp. 185–194, Jul. 2017.

[84] S. Kayukawa *et al.*, "BBeep: A Sonic Collision Avoidance System for Blind Travellers and Nearby Pedestrians," *Proc.*, May 2019.

[85] M. Liu, X. Wang, A. Zhou, X. Fu, Y. Ma, and C. Piao, "UAV-YOLO: Small Object Detection on Unmanned Aerial Vehicle Perspective," *Sensors*, vol. 20, no. 8, p. 2238, Apr. 2020.

[86] R. Tapu, B. Mocanu, and T. Zaharia, "DEEP-SEE: Joint Object Detection, Tracking and Recognition with Application to Visually Impaired Navigational Assistance," *Sensors*, vol. 17, no. 11, p. 2473, Oct. 2017.

[87] Y. Tian, G. Yang, Z. Wang, H. Wang, E. Li, and Z. Liang, "Apple detection during different growth stages in orchards using the improved YOLO-V3 model," *Comput. Electron. Agric.*, vol. 157, pp. 417–426, Feb. 2019.

[88] Y. Bazi, H. Alhichri, N. Alajlan, and F. Melgani, "Scene Description for Visually Impaired People with Multi-Label Convolutional SVM Networks," *Appl. Sci.*, vol. 9, no. 23, p. 5062, Nov. 2019.

[89] T. L. McDaniel, K. Kahol, D. Villanueva, and S. Panchanathan, "Integration of RFID and computer vision for remote object perception for individuals who are blind," *Proc. HAS 2008*, p. Article 7, 2008.

[90] H. Fernandes, P. Costa, H. Paredes, V. Filipe, and J. Barroso, "Integrating Computer Vision Object Recognition with Location Based Services for the Blind," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8515 LNCS, no. PART 3, 2014, pp. 493–500.

[91] D. Khan, S. Ullah, and S. Nabi, "A Generic Approach toward Indoor Navigation and Pathfinding with Robust Marker Tracking," *mdpi.com*, 2019.

[92] G. Lee and H. Kim, "A Hybrid Marker-Based Indoor Positioning System for Pedestrian Tracking in Subway Stations," *Appl. Sci.*, vol. 10, no. 21, p. 7421, Oct. 2020.

[93] G. Fusco and J. M. Coughlan, "Indoor localization using computer vision and visual-inertial odometry," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 10897 LNCS, pp. 86–93.

[94] M. Elgendy, T. Guzsvinecz, and C. Sik-Lanyi, "Identification of Markers in Challenging Conditions for People with Visual Impairment Using Convolutional Neural Network," *Appl. Sci.*, vol. 9, no. 23, p. 5110, Nov. 2019.

[95] G. López, L. Quesada, and L. A. Guerrero, "Alexa vs. Siri vs. Cortana vs. Google Assistant: A Comparison of Speech-Based Natural User Interfaces," in *Advances in Intelligent Systems and Computing*, 2018, vol. 592, pp. 241–250.

[96] "ArUco: a minimal library for Augmented Reality applications based on OpenCV | Aplicaciones de la Visión Artificial." [Online]. Available: http://www.uco.es/investiga/grupos/ava/node/26. [Accessed: 23-Dec-2020].

[97] D. B. Johnson, "A Note on Dijkstra's Shortest Path Algorithm," *J. ACM*, vol. 20, no. 3, pp. 385–388, Jul. 1973.

[98] A. Ganz, J. Schafer, S. Gandhi, … E. P.-I. journal of, and undefined 2012, "PERCEPT indoor navigation system for the blind and visually impaired: architecture and experimentation," *hindawi.com*.

[99] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," in *IEEE transactions on pattern*, 2014, pp. 346–361.

[100] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[101] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017.

[102] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, vol. 9905 LNCS, pp. 21–37.

[103] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[104] P. Soviany and R. T. Ionescu, "Optimizing the Trade-Off between Single-Stage and Two-Stage Deep Object Detectors using Image Difficulty Prediction," in *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, 2018, pp. 209–214.

[105] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517–6525.

[106] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018.

[107] S. S.-C. vision, undefined graphics, and image processing, and undefined 1985, "Topological structural analysis of digitized binary images by border following," *Elsevier*.

[108] D. H. DOUGLAS and T. K. PEUCKER, "ALGORITHMS FOR THE REDUCTION OF THE NUMBER OF POINTS REQUIRED TO REPRESENT A DIGITIZED LINE OR ITS CARICATURE," *Cartogr. Int. J. Geogr. Inf. Geovisualization*, vol. 10, no. 2, pp. 112–122, Dec. 1973.

[109] M. J. Huiskes and M. S. Lew, "The MIR Flickr retrieval evaluation," in *Proceedings of the 1st International ACM Conference on Multimedia Information Retrieval, MIR2008, Co-located with the 2008 ACM International Conference on Multimedia, MM'08*, 2008, pp. 39–43.